UNITED STATES DISTRICT COURT FOR THE WESTERN DISTRICT OF TEXAS

PROXENSE, LLC, *Plaintiffs*, v. APPLE, INC, *Defendants*.	Civil Action No. 6:24-cv-00143-ADA

PLAINTIFF'S FINAL INFRINGMENT CONTENTIONS

Pursuant to this Court's Standing Order Governing Proceedings in Patent Cases, Plaintiff Proxense, LLC ("Proxense" or "Plaintiff"), hereby provides its final infringement contentions concerning Defendant Apple Inc, ("Apple" or "Defendant").

Proxense hereby certifies that it undertook reasonable efforts to prepare its final infringement contentions and reserves the right to amend them based on material identified after the final invalidity contentions are served and during discovery.

These infringement contentions are based on Plaintiff's current knowledge, understanding, and belief as to the facts and information available to it as of the date of these disclosures. Proxense has not yet completed its investigation, collection of information, discovery, or analysis related to this action. The evidence cited in support of the disclosures herein is necessarily exemplary and is therefore illustrative and not exhaustive. For example, Proxense anticipates that relevant facts and evidence are uniquely within the possession, custody, or control of Defendants. As such, Proxense reserves the right to supplement, amend, or modify the information contained herein and to use and introduce at trial such information and any subsequently identified or discovered information.

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

These final infringement contentions are based at least in part upon Proxense's present understanding of U.S. Patent No. 8,352,730 (the "730 patent"), U.S. Patent No. 8,886,954 (the "954 Patent"), U.S. Patent No. 8,646,042 (the "042 Patent"), and U.S. Patent No. 9,049,188 (the "188 Patent") and the accused products. The 730, 954, 042, and 188 Patents are collectively referred to herein as the "Proxense Patents" or the "Patents- in-Suit."

Proxense reserves the right to amend its infringement contentions and asserted claims based on information Proxense obtains through discovery and otherwise as this case progresses. Proxense further reserves the right to amend its final infringement contentions and asserted claims based on any proceedings before the United States Patent and Trademark Office regarding the Proxense Patents.

I. IDENTIFICATION OF INFRINGED CLAIMS

The asserted claims of the Patents-in-Suit are as follows:

Patent	Claims
730 Patent	1,2, 5, and 6
954 Patent	1, 2, 5, 6, and 7
042 Patent	1 and 6
188 Patent	1, 3, 4, and 7

Proxense identifies these asserted claims based on its current. understanding and reserves the right to supplement and/or change its identification as discovery proceeds, including identifying additional claims and withdrawing claims.

II. IDENTIFICATION OF ACCUSED PRODUCTS, SYSTEMS, METHODS AND SERVICES

Based on the information currently available, Proxense identifies the following accused products, systems, and methods, including all reasonably similar products, systems, methods, and their variants for each of the Patents-in-Suit.

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

Hereafter, the term "Accused Instrumentalities" or "Accused Products" refers to the products listed below, (1) Apple's Universal Passwordless Architecture, which includes (a) Apple's MacOS, iPhone OS, and iPad OS operating systems, (b) Safari browser, and (c) Apple Security Platform (2) Apple Pay services (also known as Apple Wallet) and all Apple devices preloaded with Apple Pay/Wallet.

III. CHARTS SETTING FORTH WHERE IN THE ACCUSED PRODUCT(S) EACH ELEMENT OF THE ASSERTED CLAIM(S) ARE FOUND

Subject to ongoing discovery and investigation, Proxense provides claim charts pertaining to the Patents-in-Suit attached hereto as Exhibit A for the 730 patent, Exhibit D for the 042 patent, Exhibit E for the 188 patent and Exhibit F also for the 954 patent. These infringement contentions serve a notice function and are not required to—and therefore do not—present every possible permutation or theory of Proxense's case.

IV. LITERAL INFRINGEMENT AND INFRINGEMENT UNDER THE DOCTRINE OF EQUIVALENTS

Proxense contends that Apple's products literally infringe the asserted claims of the Patents-in-Suit as more specifically explained in the attached claim charts. Proxense reserves the right to assert infringement under the doctrine of equivalents to the extent that the difference between any component of any product or system, and any step of any service, method, and/or claim element is insubstantial.

V. PRIORITY DATES AND ALL DOCUMENTS EVIDENCING CONCEPTION AND REDUCTION TO PRACTICE FOR EACH CLAIMED INVENTION

The 730 patent matured from application number 11/314,199 ("the 199 application"), which was filed on December 20, 2005. The 199 application claims priority from provisional patent application 60/637,538 which was filed on December 20, 2004. Documents evidencing conception and/or reduction to practice for the claimed inventions of

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

the 730 patent are being collected and will be produced shortly.

The 954 patent matured from application number 13/710,109 ("the 109 application"), which was filed on December 10, 2012. The 109 application claims priority from provisional patent application 60/637,538 which was filed on December 20, 2004. Documents evidencing conception and/or reduction to practice for the claimed inventions of the 954 patent are being collected and will be produced shortly.

The 042 patent matured from application number 13/445,825 ("the 825 application"), which was filed on April 12, 2012. The 825 application claims priority from provisional patent application 60/992,953 which was filed on December 6, 2007. Documents evidencing conception and/or reduction to practice for the claimed inventions of the 042 patent are being collected and will be produced shortly.

The 188 patent matured from application number 14/171,705 ("the 705 application"), which was filed on February 3, 2014. The 705 application claims priority from provisional patent application 16/048,044 which was filed on July 27, 2018. Documents evidencing conception and/or reduction to practice for the claimed inventions of the 188 patent are being collected and will be produced shortly.

VI. A COPY OF THE FILE HISTORY FOR EACH PATENT IN SUIT

A copy of the file history for each of the Patents-in-Suit has been produced in this matter. The file history of the 730 patent bears production number PROX_APPLE-00000001. The file history of the 042 patent bears production number PROX_APPLE-00002580. The file history of the 188 patent bears production number PROX_APPLE-00003077. The file history of the 954 patent bears production number PROX_APPLE-00003611.

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

Dated: March 4, 2025

Respectfully submitted,

/s/ David L. Hecht

David L. Hecht (**Co-Lead Counsel**)
dhecht@hechtpartners.com

Maxim Price (*pro hac vice*)
mprice@hechtpartners.com

Yi Wen Wu (*pro hac vice*)
wwu@hechtpartners.com

HECHT PARTNERS LLP
125 Park Avenue, 25th Floor

New York, New York
10017 Telephone: (212)
851-6821

Brian D. Melton (**Co-Lead Counsel**)
bmelton@susmangodfrey.com
SUSMAN GODFREY L.L.P.
1000 Louisiana Street, Suite 5100
Houston, Texas 77002-5096
Telephone: (713) 653-7807
Facsimile: (713) 654-6666

Lear Jiang
ljiang@susmangodfrey.com
SUSMAN GODFREY L.L.P.
1900 Avenue of the Stars, Suite 1400 Los Angeles, California 90067-6029 Telephone: (310) 789-3100
Facsimile: (310) 789-3150

Counsel for Plaintiff Proxense, LLC

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

CERTIFICATE OF SERVICE

I hereby certify that on March 4, 2025, the foregoing was served on counsel for Defendant Apple, LLC.

/s/ David L. Hecht
David L. Hecht

EXHIBIT A

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

U.S. Patent Number . 8,352,730 – Apple Pay[1]

Assignee:	Proxense, LLC
Title:	Biometric personal data key (PDK) authentication
Filing Date:	2014-10-23
Publication Date:	2016-03-29
Inventor:	Giobbi, John J.

'730 Patent Claim[2]		**Accused Instrumentality And Where Each Claim Element Is Found[3]**
1	A method for verifying a user during authentication of an integrated device, comprising the steps of:[4]	Apple, using its proprietary software on Apple iPhones with Apple Pay pre-installed, alone and as part of a joint enterprise with various issuing banks and/or via direction and control the activities of various issuing banks, carry out the claimed method, literally or by the doctrine of equivalents, for at least the reasons set forth below.

[1] The Infringement Contentions provided herein are based on information obtained to date and may not be exhaustive. Plaintiff's investigation of Defendants' infringement is ongoing. Plaintiff reserves the right to supplement and/or amend these contentions to identify additional instrumentalities and to further identify where each element of each claim is found in each accused instrumentality, including on the basis of discovery obtained from Defendants, and from third parties during the course of this litigation, pursuant to the Order Governing Proceedings – Patent Cases under Hon. Alan D. Albright. Proxense incorporates by reference Apple's responses to Proxense's first set of RFAs.

[2] All FICS set forth herein for any independent patent claims are hereby incorporated by reference into the contentions alleged for any dependent patent claims that depend on such independent claims, as if fully set forth therein.

[3] The Accused Instrumentalities and associated exhibits discussed and/or cited for any claim herein are representative in all material respects of all other accused instrumentalities identified for that claim (e.g., a specified device or service may be used as a representative example in these charts since the other accused instrumentalities have immaterial differences in their hardware and/or software configuration, the cited references are believed to be illustrative of all such accused devices).

[4] Plaintiff's inclusion of any claim preamble in this claim chart should not be interpreted as an admission that the preamble is limiting. Plaintiff reserves the right to take the position that the claim preambles are limiting or not limiting on a claim-by-claim basis.

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

	persistently storing biometric data of the user and a plurality of codes and other data values comprising a device ID code uniquely identifying the integrated device and a secret decryption value in a tamper proof format written to a storage element on the integrated device that is unable to be subsequently altered;	**persistently storing biometric data of the user … in a tamper proof format written to a storage element on the integrated device that is unable to be subsequently altered** Utilizing Apple's Secure Enclave, iPhones persistently store biometric data of a user written to the storage element on the integrated device that is unable to be subsequently altered. "The Secure Enclave is a dedicated secure subsystem … isolated from the main processor to provide an extra layer of security designed to keep sensitive data secure even when the Application Processor kernel becomes compromised." Apple Platform Security, page 9. Protecting data even if a hack or malware compromises the Application Processor, the Secure Enclave provides a tamper proof format for sensitive data. The sensitive data protected by Apple's Secure Enclave includes the biometric data. "During enrollment, the Secure Enclave processes, encrypts, and stores the corresponding Touch ID and Face ID template data." Apple Platform Security, , , page 19. **persistent storing … a plurality of codes and other data values comprising a device ID code uniquely identifying the integrated device … in a tamper proof format written to a storage element on the integrated device that is unable to be subsequently altered** In addition to registering user biometrics, Apply Pay requires registering credit and debit cards. To safeguard registered card information, Apply Pay utilizes EMV payment tokens stored within a secure element as a unique device account number. "Apple [was] among the first to implement EMV payment tokens in digital wallets that hold credential for several payments use cases." EMV Payment Tokenization Primer and Lessons Learned, , page 12. "EMV payment tokens are open-loop tokens provisioned by a [Token Service Provider] and, like other tokens, are used to replace the actual payment credential (e.g., [Primary Account Number]) with another numeric value." *Id.* Accordingly, when a card is added to Apple Pay, a payment token is issued to the iPhone in exchange for the

		for a credit card number by TSP, such as Visa, MasterCard, Discover, and American Express. *Id*, at 23 (Figure 5). Adding a card to Apple Pay on an iPhone causes "a unique Device Account Number [to be] created, encrypted, and then stored in the Secure Element." Apple Platform Security, , page 142. Secure elements are recognized as "a dynamic environment to store data securely, process data securely and perform communication with external entities securely," that "will not allow unauthorized access." EMV Payment Tokenization Primer and Lessons Learned, , page 41. Securely storing a EMV payment token as an unique Device Account Number on an element not allowing unauthorized access, iPhones with Apple Pay persistently store a device ID code uniquely identifying the smartphone in a tamper proof format, written to a storage element on the integrated device, that is unable to be subsequently altered. Adding a card to Apple Pay requires the approval of the bank issuing the card (issuing bank). When adding a card to Apple Wallet, "Apple securely sends the card information, along with other information about user's account and device, to the card issuer or card issuer's authorized service provider (usually the payment network). Using this information, the card issuer (or its service provider) determines whether to approve the user's request to add the card to Apple Wallet." Apple Platform Security, 179. Thus, the issuing banks determine whether to allow a card to be added to Apple Wallet on an iPhone. To provide the information necessary for the issuing banks to decide if a card will be added to Apple Wallet, "Apple Pay uses three server-side calls to send and receive communication with the card issuer or payment network." Apple Platform Security, 179. The information transmitted via these calls "enable the card issuer to verify, approve, and add cards to Apple Wallet." *Id.* The issuing banks, accordingly, make determinations under the direction and control of Apple. Furthermore, the final determination provided by the issuing banks gives each of the issuing banks equal right of control as they can refuse to allow their cards to be added to Apple Wallet. The result of adding a card to Apple is the creation of a device ID code uniquely identifying the integrated device. Card numbers are not stored on the device or by Apple's servers. "Instead, a unique Device Account Number is created by the card issuer, sent encrypted to Apple, and then stored in the Secure Element." Apple Platform Security, 179. Thus, cards are added to Apple Wallet as "***unique Device Account Numbers***," The unique Device Account "are never stored on

		Apple Pay servers or backed up to iCloud, and it's isolated from: Devices that use biometric authentication[;] Apple Watch[; and] Mac computers with Apple silicon that use the Magic Keyboard with Touch ID." Apple Platform Security, 179. As the Device Account Numbers are isolated from other devices and not synced, each remains unique to a specific device. **persistently storing … a secret decryption value in a tamper proof format written to a storage element on the integrated device that is unable to be subsequently altered;** An iPhone with Apple Pay persistently stores a secret decryption value as a pairing key shared by the Secure Enclave and the Secure Element. "Communication between the Secure Enclave and the Secure Element takes place over a serial interface…Though not directly connected, the Secure Enclave and Secure Element can communicate securely using a shared pairing key that provisioned during the manufacturing process." Apple Platform Security, , page 144. For the shared key to persist, it must be stored by both the Secure Element and the Secure Enclave. "The Secure Enclave is a dedicated secure subsystem … isolated from the main processor to provide an extra layer of security designed to keep sensitive data secure even when the Application Processor kernel becomes compromised." Apple Platform Security, , page 9. Protecting data even when a hack or malware comprises the Application Processor, the Secure Enclave provides a tamper proof format for sensitive data, such as shared pairing key with the Secure Element. Furthermore, secure elements are recognized as "a dynamic environment to store data securely, process data securely and perform

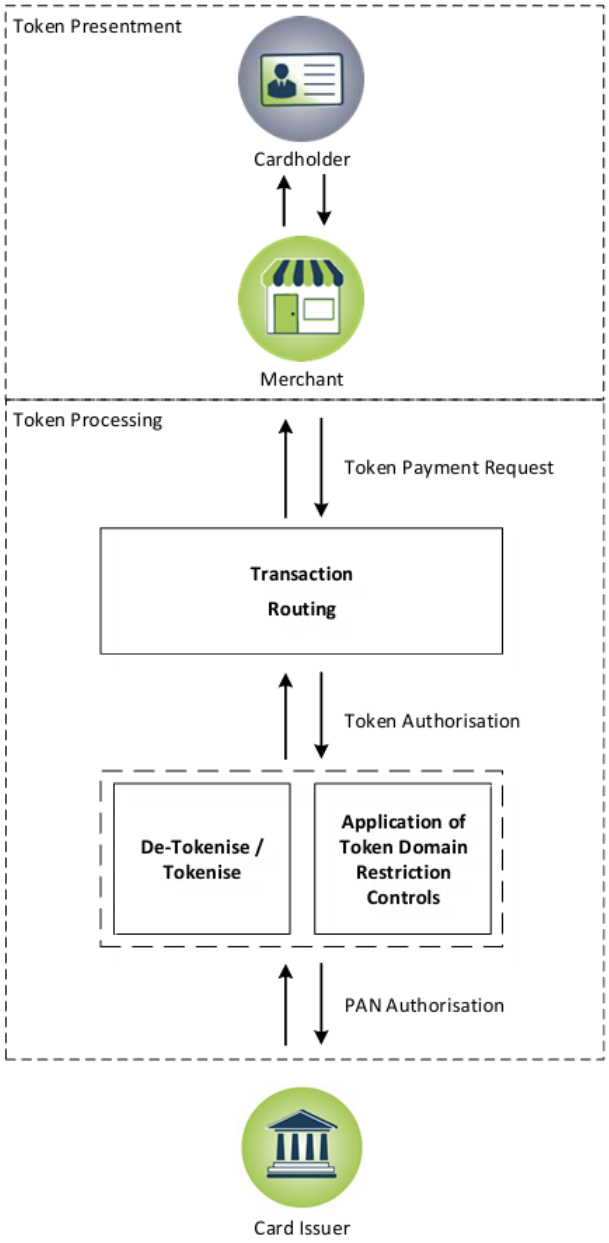
		communication with external entities securely," that "will not allow unauthorized access." EMV Payment Tokenization Primer and Lessons Learned, page 41. Additionally, Apple has described the Secure Enclave thusly: "When you store a private key in the Secure Enclave, you never actually handle the key, making it difficult for the key to become compromised. Instead, you instruct the Secure Enclave to create the key, securely store it, and perform operations with it. You receive only the output of these operations, such as encrypted data or a cryptographic signature verification outcome." Protecting Keys with the Secure Enclave Both the Secure Enclave and Secure Element, thus, store a secret decryption value in a tamper proof format written to a storage element on the integrated device that is unable to be subsequently altered.
	wherein the biometric data is selected from a group consisting of a palm print, a retinal scan, an iris scan, a hand geometry, a facial recognition, a signature recognition and a voice recognition;	The sensitive data protected by Apple's Secure Enclave includes the biometric data. "During enrollment, the Secure Enclave processes, encrypts, and stores the corresponding Touch ID and Face ID template data." Apple Platform Security, , page 19.

	responsive to receiving a request for a biometric verification of the user, receiving scan data from a biometric scan;	"[B]efore information is transmitted, the user must authenticate using Touch ID, Face ID or their passcode… No payment information is sent without user authentication." Apple Platform Security, , page 145. This requirement for authentication necessitates the Secure Enclave receiving a request for biometric verification. Completing this verification requires the Secure Element receive biometric scan data from either the TrueDepth camera or the fingerprint sensor. To biometrically verify themselves using Touch ID or Face ID, the user places their finger on the fingerprint sensor or looks at their device. "The sensor captures the biometric image and securely transmits it to the Secure Enclave." Apple Platform Security, , page 19. The Secure Enclave, accordingly, receives scan data from a biometric scan by the fingerprint sensor. When using Face ID, the user simply looks at their device. "After the TrueDepth camera confirms the presence of an attentive face, it protects and reads thousands of infrared dots to form a depth of the face along with a 2D image… A portion of the Secure Neural Engine – protected within the Secure Enclave – transforms this data into mathematical representation and compares that representation to the enrolled facial data." Apple Platform Security, , page 20. As the Secure Neural Network is within the Secure Enclave, the Secure Enclave must receive scan data from the TrueDepth camera.
	comparing the scan data to the biometric data to determine whether the scan data matches the biometric data;	"[B]efore information is transmitted, the user must authenticate using Touch ID, Face ID or their passcode… No payment information is sent without user authentication." Apple Platform Security, , page 145. This requirement for authentication necessitates the Secure Enclave match the received scan data to the stored biometric data. "During matching, the Secure Enclave compares incoming data from the biometric sensor against the stored templates to determine whether to unlock the device or respond to that a match is valid (for Apple Pay, in-app, and other uses of Touch ID and Face ID)." Apple Platform Security, , page 19.

	responsive to a determination that the scan data matches the biometric data, wirelessly sending one or more codes from the plurality of codes and the other data values for authentication by an agent that is a third-party trusted authority possessing a list of device ID codes uniquely identifying legitimate integrated devices, wherein the one or more codes and other data values includes the device ID code; and	**responsive to a determination that the scan data matches the biometric data, wirelessly sending one or more codes from the plurality of codes and other data values … wherein the one or more codes and other data values includes the device ID code** "[B]efore information is transmitted, the user must authenticate using Touch ID, Face ID or their passcode… No payment information is sent without user authentication." Apple Platform Security, , page 145. This requirement for authentication necessitates the "unique Device Account Number" stored in the Secure Element and used for payment is only sent responsive to a determination that the scan data matches the biometric data. As previously noted, "[d]uring matching, the Secure Enclave compares incoming data from the biometric sensor against the stored templates to determine whether to unlock the device or respond to that a match is valid (for Apple Pay, in-app, and other uses of Touch ID and Face ID)." Apple Platform Security, , page 19. Apple Platform Security, , page 19. If the scan matches the template, "the Secure Enclave then sends signed data about the type of authentication and details about the transaction to the (contactless or within apps) to the Secure Element… It's securely delivered to the Secure Element by leveraging the paring key." Apple Platform Security, , page 144. After receiving confirmation of authentication, "contactless responses prepared by the payment applets within the Secure Element are exclusively routed by the Controller to the NFC field." Apple Platform Security, , page 141. Thus, the Device Account Number (i.e., payment token) is only released from the Secure Element and transmitted to the payment terminal wirelessly by the NFC controller after biometric verification of the user. **for authentication by an agent that is a third-party trusted authority possessing a list of device ID codes uniquely identifying legitimate integrated devices**

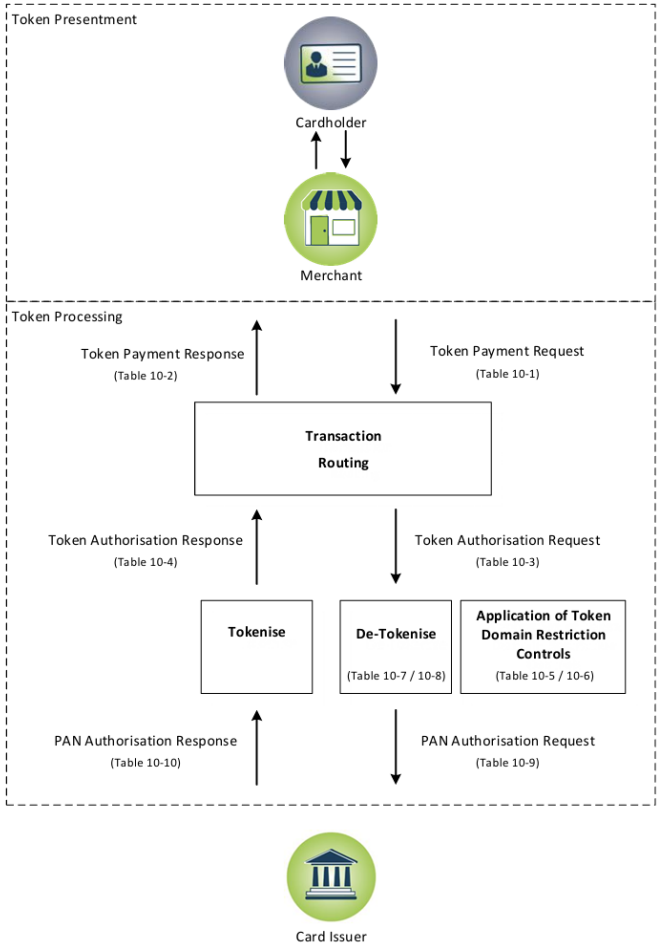
		After being received by the payment terminal, the unique Device Account Number is transmitted through the payment network as Token Payment Request. *See* EMV Payment Tokenisation Specification: Technical Framework, v2.2 (2020), Figure 10.1 and page 81 ("The basic authorisation flow is shown in Figure 10.1.") As it travels through the payment network to the token service provider, the Token Payment Request, containing the payment token wirelessly transmitted from an iPhone, is converted to a Token Authorization Request received by the Token Service Provider. *Id*.

Figure 3.2: Payment Token Transaction Overview



APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

Figure 10.1: Illustrative Payment Token Processing Flow for Authorisations



"The Token Payment Request is the first leg … of any transaction, starting with the interaction between the Cardholder and the Merchant." EMV Payment Tokenisation Specification: Technical Framework, v2.2 (2020), 84. The transaction begins with the cardholder interacting with a merchant's terminal. "The Cardholder interacting with a Terminal, website or application will result in Token Processing related data being passed to the Merchant." *Id.* As detailed above, the data is passed from the iPhone via wireless transmission. The passed data includes Payment Token (Device Account Number). *See* EMV Payment Tokenisation Specification: Technical Framework, v2.2 (2020), Table 10.1

Table 10.1: Fields Included in Token Payment Requests

Field Content	Field Name	R/C/O	Condition/Option	Comment
Payment Token	PAN	R		
Token Expiry Date	PAN Expiry Date	R		

Field Content	Field Name	R/C/O	Condition/Option	Comment
Token Presentment Mode	POS Entry Mode	R		Token Control Data
Token Cryptogram	Payment Network Specific	C	Required for EMV based and application based commerce in Cardholder Initiated Transactions	Generated and passed in appropriate cryptogram field. Token Control Data
Token Requestor ID	Payment Network Specific	O	Present if received by the Terminal or sourced by the Merchant directly from Token Requestor related interactions	
Merchant Identifiers	Payment Network Specific	O	May be present for e-commerce transactions	Token Control Data
Payment Account Reference	Payment Account Reference	O	Present if received by the Terminal or sourced by the Merchant directly from Token Requestor related interactions	

R – Required, C – Conditional, O – Optional

Accordingly, the Payment Token (Device Account Number) received from Apple Pay on iPhones is part of the Token Payment Request as the entry in the PAN field and thus takes the place of the account number on the card the issuing bank allowed to be added to Apple Wallet.

During transaction routing, the Token Payment Request is transformed to Token Authorisation Request. As with the Token Payment Request, the Token Authorisation Requests also includes the Payment Token (Device Account Number) the issuing bank allowed to be added to Apple Wallet in the PAN field. As such, the Payment Token (Device Account Number) takes the place of a card's account number. *See* EMV Payment Tokenisation Specification: Technical Framework, v2.2 (2020), Table 10.3

Table 10.3: Fields Included in Token Authorisation Requests

Field Content	Field Name	R/C/O	Condition/Option	Comment
Payment Token	PAN	R		

Field Content	Field Name	R/C/O	Condition/Option	Comment
Token Expiry Date	PAN Expiry Date	R		
Token Presentment Mode	POS Entry Mode	R		Inserted by Acquirer. Token Control Data
Token Cryptogram	Payment Network Specific	C	Required for EMV based and application based commerce in Cardholder Initiated Transactions	Token Control Data
Token Requestor ID	Payment Network Specific	O	May be present if read from Terminal or sourced directly from Token Requestor and passed to the Acquirer	
Merchant Identifiers	Payment Network Specific	O	May be present for e-commerce transactions	Token Control Data
Payment Account Reference	Payment Account Reference	O	Present if received by the Terminal or sourced by the Merchant directly from Token Requestor related interactions	

R – Required, C – Conditional, O – Optional

"The Token Authorisation request process continues until De-Tokenisation has been completed." EMV Payment Tokenisation Specification: Technical Framework, v2.2 (2020), 86. De-Tokenisation is "the process of converting a Payment Token and Token Expiry Date to its underlying PAN and PAN Expiry Date based on the Payment Token / Token Expiry Date mapping to the underlying PAN / PAN Expiry Date stored in the Token Vault." Id, at 6. "Token Service Providers are responsible for a number of discrete functions which may include, but are not limited to: Maintenance and operation of a Token Vault … [and] De-Tokenisation." *Id*. at 19.

Maintaining the token vault and performing de-tokenization, a token service provider receives the token authorization request. Upon receiving the request, "the

		Payment Token SHALL be de-tokenised to the underlying PAN in the incoming Token Authorisation prior to sending the PAN Authorisation to the Card Issuer." *Id*. at 91. As shown in Figure 10.1, "Once a Payment Token has been de-tokenised, the final request step is to initiate a PAN Authorisation, destined for the Card Issuer's authorisation system." *Id*. at 92. The PAN authorization request sent to the Card Issuer, contains the "underlying PAN from token vault." *Id*. Table 10.9, first row, at 93.

Table 10.9: Fields Included in PAN Authorisation Requests

Field Content	Field Name	R/C/O	Condition/Option	Comment
PAN	PAN	R		Underlying PAN from Token Vault
PAN Expiry Date	PAN Expiry Date	R		Expiry date associated with the underlying PAN
Token Presentment Mode	POS Entry Mode	R		Payment Token related data
Token Requestor ID	Payment Network Specific	R		Payment Token related data
Payment Token	Payment Network Specific	R		
Token Expiry Date	Payment Network Specific	O		Payment Token related data
Token Assurance Method	Payment Network Specific	C	Required if provided by Payment Network	Payment Token related data
Token Assurance Data	Payment Network Specific	O		Payment Token related data
Payment Account Reference	Payment Account Reference	O	Present if available to the Token Service Provider	The Payment Account Reference associated with the PAN

R – Required, C – Conditional, O – Optional

Obtaining the underlying primary account number from the token vault, as to prepare the PAN authorization request, requires gaining access to the file containing the account number. Converting the payment token to its underlying account number based on the Payment Token / Token Expiry Date mapping to the underlying PAN / PAN Expiry Date stored in the token vault, as to determine

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

		which file access, the token service provider necessarily authenticates the payment token and the other data values received from the iPhone. Opening the token vault to perform de-tokenisation occurs in response to receiving the token authorization request containing the payment token. De-Tokenisation is "the process of converting a Payment Token and Token Expiry Date to its underlying PAN and PAN Expiry Date based on the Payment Token / Token Expiry Date mapping to the underlying PAN / PAN Expiry Date stored in the Token Vault." Id, at 6. "The Payment Token SHALL be de -tokenised to the underlying PAN in the incoming Token Authorisation prior to sending the PAN Authorisation to the Card Issuer." Id., at 91. Detokenizing in response to an incoming token authorization requests requires the token service provider be sent the token authorization. As noted *supra*, the token authorization sent to the token service provider contains the payment token wirelessly sent from an iPhone after successful biometric authentication. Detokenizing in response to an incoming token authorization request containing a payment token provided by an iPhone following successful biometric authentication, therefore, requires the token service provider is wirelessly sent one or more codes from the plurality of codes and the other data values for authentication responsive to a determination that the scan data matches the biometric data.

	responsive to authentication of the one or more codes and the other data values by the agent, receiving an access message from the agent allowing the user access to an application, wherein the application is selected from a group consisting of a casino machine, a keyless lock, a garage door opener, an ATM machine, a hard drive, computer software, a web site and a file.	**responsive to authentication of the one or more codes and other data values by the agent receiving an access message from the agent allowing the user to access an application, wherein the application is selected from the group consisting of … computer software.** "Once a Payment Token has been de-tokenised, the final request step is to initiate a PAN Authorisation, destined for the Card Issuer's authorisation system." EMV Payment Tokenisation Specification: Technical Framework, v2.2 (2020), 92. The PAN Authorisation contains the actually account number for the card the issuing bank allowed to be added to Apple Wallet. *See* EMV Payment Tokenisation Specification: Technical Framework, v2.2 (2020), Table 10.9.

Table 10.9: Fields Included in PAN Authorisation Requests

Field Content	Field Name	R/C/O	Condition/Option	Comment
PAN	PAN	R		Underlying PAN from Token Vault
PAN Expiry Date	PAN Expiry Date	R		Expiry date associated with the underlying PAN
Token Presentment Mode	POS Entry Mode	R		Payment Token related data
Token Requestor ID	Payment Network Specific	R		Payment Token related data
Payment Token	Payment Network Specific	R		
Token Expiry Date	Payment Network Specific	O		Payment Token related data
Token Assurance Method	Payment Network Specific	C	Required if provided by Payment Network	Payment Token related data
Token Assurance Data	Payment Network Specific	O		Payment Token related data
Payment Account Reference	Payment Account Reference	O	Present if available to the Token Service Provider	The Payment Account Reference associated with the PAN

R – Required, C – Conditional, O – Optional

Accordingly, following authentication of the Payment Token (Device Account Number) an Authorization Request containing the PAN received from the token vault is received by the issuing bank's "authorisation system" so that the user may have the requested payment authorized by the issuing bank. Therefore, the authorization request containing the PAN in place of the unique Device Account Number allows the user access to the issuer's computer software necessary to process and authorize the payments and is thus an access message from the Token Service Provider received by the issuing Bank's computer software in response to

		successful authentication of the Device Account Number.
2	The method of claim 1, wherein the one or more codes and the other data values are transmitted to the agent over a network.	The terminal, TSP, and card issuer's server are components of a payment network.

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

5	The method of claim 1, wherein the biometric data and the scan data are both based on a fingerprint scan by the user.	As previously noted with respect to claim 1, "before information is transmitted, the user must authenticate using Touch ID, Face ID or their passcode… No payment information is sent without user authentication." Apple Platform Security, , page 145. "Touch ID is the fingerprint sensing system that makes secure access to supported Apple devices faster and easier." Apple Platform Security, , page 19. Incorporating a fingerprint sensing system permitting authentication and secure access, iPhones utilize a method in which the stored biometric data and scan data are both based on a fingerprint scan by the user.
6	The method of claim 1, further comprising:	
	establishing a secure communication channel prior to sending the one or more codes and the other data values for authentication.	When presenting EMV token payment requests and token authorization requests, Token cryptograms are "required for EMV based and application based commerce in Cardholder Initiated Transactions." EMV Payment Tokenisaton Specification: Technical Framework v2.2 (2020), pages 85 and 87. "Apple [was] among the first to implement EMV payment tokens in digital wallets that hold credential for several payments use cases." EMV Payment Tokenization Primer and Lessons Learned, , U.S. Payments Forum (2019), page 12. "Limited use token keys generate cryptograms that are passed with the EMV payment token for each transaction." *Id* at 11. As noted in Col. 4, lines 53-55 of the '730 Patent, "encryption/decryption keys [can be] utilized to establish secure communications links." Generating cryptograms with encryption/decryption keys to be passed in payment requests and token authorization requests, Apple Pay preloaded smartphones establish an embodiment of a secure communication channel specifically identified in the '730 patent. Additionally, Apple has described the Secure Enclave thusly: "When you store a private key in the Secure Enclave, you never actually handle the key, making it difficult for the key to become compromised. Instead, you instruct the Secure Enclave to create the key, securely store it, and perform operations with it. You receive only the output of these operations, such as encrypted data or a cryptographic signature verification outcome." Protecting Keys with the Secure Enclave

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

EXHIBIT D

CLAIM CHART – U.S. PATENT NO. 8,646,042
APPLE PASSKEYS

Assignee:	Proxense, LLC
Title:	Hybrid device having a personal digital key and receiver-decoder circuit and methods of use
Filing Date:	2012-04-12
Publication Date:	2014-02-04
Inventor:	Brown, David L.

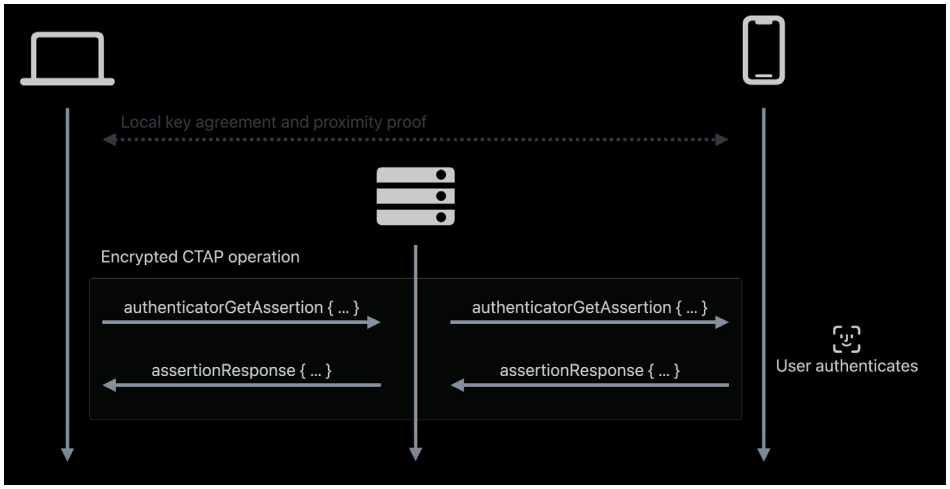
042 **Patent Claim**[1]		**Accused Instrumentality and Where Each Claim Element Is Found**[2]
1	A hybrid device comprising[3]:	This preamble is not limiting. Apple manufactures, imports, offers for sale, and sells in the United States mobile phones with the iOS 15+ operating system that include an integrated personal digital key and an integrated receiver decoder circuit. ("In macOS Monterey and iOS 15, we announced a developer preview of the solution -- passkeys -- and got so much great feedback. In macOS Ventura and iOS 16, we're excited to make passkeys available to everyone. Now is the time to adopt them." Meet passkeys - WWDC22 - Videos - Apple Developer).
	an integrated personal digital key (PDK) for storing information and capable of communicating wirelessly with	<u>an integrated personal digital key (PDK)</u> Apple iPhones with iOS 15 and later operating systems enable the use of passkeys by utilizing an integrated personal digital key (PDK).

[1] All PICS set forth herein for any independent patent claims are hereby incorporated by reference into the PICS alleged for any dependent patent claims that depend on such independent claims, as if fully set forth therein.
[2] The Accused Instrumentalities and associated exhibits discussed and/or cited for any claim herein are representative in all material respects of all other accused instrumentalities identified for that claim (e.g., a specified device or service may be used as a representative example in these charts since the other accused instrumentalities have immaterial differences in their hardware and/or software configuration, the cited references are believed to be illustrative of all such accused devices).
[3] Plaintiff's inclusion of any claim preamble in this claim chart should not be interpreted as an admission that the preamble is limiting. Plaintiff reserves the right to take the position that the claim preambles are limiting or not limiting on a claim-by-claim basis.

	at least one external receiver-decoder circuit (RDC); and	The 042 Patent defines a PDK as including "an antenna and a transceiver for communicating with an R[eceiver] D[ecoder] C[ircuit] (not shown) and a controller and memory for storing information particular to a user." 042 Patent, 13: 46-49. A component of the integrated PDK of the hybrid device, accordingly, is "a controller and memory for storing information particular to a user." The 042 Patent defines the operation of the controller and memory for storing information particular to the user as entailing the receipt of an access key from an external application that is used to access a specific service block. 042 Patent, 6:23-27 ("Regardless of how created, once created, external applications (such as applications 120 in FIG. 1) can gain access to specific service block 112 by providing the corresponding access key 118. In FIG. 2., this is shown conceptually by control logic 250.") The integrated PDK of the hybrid device, therefore, includes a controller placing within memory information that can only be accessed by a corresponding access key provided by an external application. Such memory is present in Apple iPhones with iOS 15, and later, because it is required by the standard. "Passkeys are built on the WebAuthentication -- or WebAuthn standard -- and use public-key cryptography." Meet passkeys - WWDC22 - Videos - Apple Developer. Under the WebAuthn specification, "compliant authenticators protect public key credentials." Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 1. A public key credential refers to a public key credential source, which includes a credential ID. Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 4 (Defining "public key credential" and "public key credential source."). The credential ID uniquely identifies its public key credential source. Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 4 (Defining a credential ID as "A probabilistically-unique byte sequence identifying a public key credential source and its authentication assertions."). In addition to the credential ID, each public key credential source contains a "credential private key." Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 4, (Defining "public key credential source" as data structure including the credential private key and the credential ID.). "The credential private key is bound to a particular authenticator" and part of an asymmetric key pair containing a public key

		returned to a relying party. Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 4 (Defining a "credential key pair."). Every Apple iPhone with iOS 15 and later, therefore, will store within memory a credential comprising a private key of an asymmetric key pair and a credential ID uniquely identifying the private/public key pair to which the private key belongs. The credentials stored within Apple iPhones with iOS 15 and later can only be accessed with the appropriate access key. "A public key credential can only be used for authentication with the same entity (as identified by the RP ID) it was registered with." Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 4 ("A public key credential can only be used for authentication with the same entity (as identified by RP ID) it was registered with."). When generating a response, therefore, the authenticator will only retrieve credentials corresponding to the RP ID provided to it by the external server. Client to Authenticator Protocol (CTAP) (fidoalliance.org), § 6.2.2 ("7.1 If the allowList parameter is present and is non-empty, locate all denoted credentials created by this authenticator and bound to the specified rpId. 7.2 If an allowList is not present, locate all discoverable credentials that are created by this authenticator and bound to the specified rpId."). As only credentials corresponding to the RP ID will be retrieved, the RP ID is an access key. As Apple states it, "the system will take care of only letting me use it in the correct app or website, with strong built-in phishing resistance." Meet passkeys - WWDC22 - Videos - Apple Developer. As Apple iPhones will only return credentials corresponding to the RP ID access key provided by the external relying party, Apple iPhones with iOS 15 and later have the controller and memory necessary for a minimal embodiment of a PDK. In addition to the controller and memory, a minimal embodiment of PDK is defined by the 042 Patent as including "an antenna and a transceiver for communication with an RDC." 042 Patent, 13:46-48. With an Apple iPhone with iOS 15 or later, "Passkeys can also be used to sign in across devices in a secure, phishing-resistant manner. Here's how that works. There are two devices here. The client, which is the device or web browser where I'm signing in, and the authenticator, which is the device which has my passkey. First, the client shows a QR code, which the authenticator scans. This QR

code contains a URL that encodes a pair of single-use encryption keys. Then, the authenticator produces a Bluetooth advertisement containing routing information for a network relay server. Once the local exchange and key agreement have happened, the two devices connect to a relay server picked by the phone. From there, they perform a standard FIDO CTAP operation, which is encrypted using the keys from earlier, so the relay server can't see anything that's going on. This whole process is performed by the device and the web browser. The website is not involved at any point in the cross-device communication." Meet passkeys - WWDC22 - Videos - Apple Developer. The process is shown in the below figure.



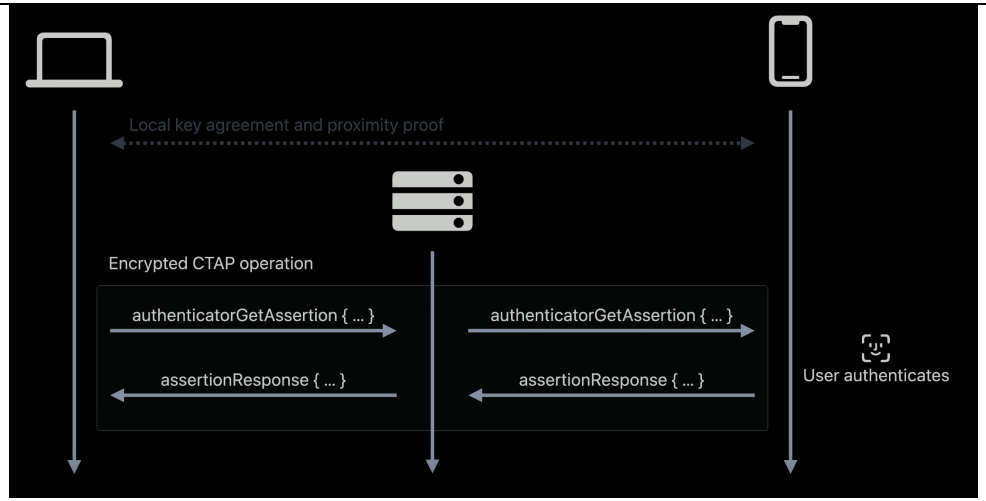
Meet passkeys - WWDC22 - Videos - Apple Developer.

Establishing the connection over the internet requires utilizing either the device's Wi-Fi or cellular capabilities – both of which are wireless protocols. Accordingly, iPhones with iOS 15 and later have the antenna and transceiver necessary to implement the wireless protocols enabling transmission over the internet.

Having each of the elements of a minimal embodiment of a PDK, Apple iPhones with iOS 15 and later include an integrated PDK.

		storing information Apple iPhones with iOS 15 later store information for authenticating users. "Passkeys are a replacement for passwords. They are faster and easier to sign in. Just use Touch ID or Face ID to authenticate and you're done." Deploy passkeys at work - WWDC23 - Videos - Apple Developer. "Apple platforms will always require UV for passkeys when biometrics are available, so you don't have to worry about that." Meet passkeys - WWDC22 - Videos - Apple Developer. Accordingly, Apple iPhones with iOS 15 and later locally store biometric information for authenticating user. Apple iPhones utilize the Secure Enclave to securely store the biometric information for authentication. "During enrollment, the Secure Enclave processes, encrypts, and stores the corresponding Touch ID and Face ID template data." Apple Platform Security, page 19. Apple has described the Secure Enclave thusly: "When you store a private key in the Secure Enclave, you never actually handle the key, making it difficult for the key to become compromised. Instead, you instruct the Secure Enclave to create the key, securely store it, and perform operations with it. You receive only the output of these operations, such as encrypted data or a cryptographic signature verification outcome." Protecting Keys with the Secure Enclave, https://developer.apple.com/documentation/security/certificate_key_and_trust_services/keys/protecting_keys_with_the_secure_enclave "The Secure Enclave is a dedicated secure subsystem … isolated from the main processor to provide an extra layer of security designed to keep sensitive data secure even when the Application Processor kernel becomes compromised." Apple Platform Security, page 9. Protected even when a hack or malware comprises the Application Processor, the biometric data for Touch ID and Face ID are securely stored. capable of communicating wirelessly with at least one external receiver-decoder circuit (RDC)

		As detailed above, Apple iPhones with iOS 15 or later enable the use of passkeys to sign in across devices in a secure, phishing-resistant manner by utilizing either the device's Wi-Fi or cellular capabilities – both of which are wireless protocols. Consequently, Apple iPhones with iOS 15 and later are capable of communicating wirelessly with an external receiver decoder circuit.
	an integrated RDC for communicating wirelessly with at least one external PDK within a proximity zone, the integrated RDC coupled to the integrated PDK by a first signal line for communication, the integrated RDC coupled to at least one other component of the hybrid device by a second signal line, one or more of the integrated RDC and integrated PDK enabling one or more of an application, a function, and a service on one or more of the hybrid device and a device associated with the external RDC.	<u>an integrated RDC for communicating wirelessly with at least one external PDK</u> As detailed above, Apple iPhones with iOS 15 and later communicate passkey signatures over an encrypted connection, through the internet, via Wi-Fi, and/or cellular protocols. Accordingly, Apple iPhones with iOS 15 and later include an RDC enabling wireless communications with at least one external device, such as a device running macOS Monterey or later requires. <u>the integrated RDC coupled to the integrated PDK by a first signal line for communication</u> Enabling the use of passkeys across devices with a QR code, Apple iPhones with iOS 15 or later include a signal line for communication that couples the integrated RDC to the integrated PDK. As noted above, using a passkey on Apple iPhone to sign into a website on external device begins by scanning a QR code. "First, the client shows a QR code, which the authenticator scans. This QR code contains a URL that encodes a pair of single-use encryption keys. Then, the authenticator produces a Bluetooth advertisement containing routing information for a network relay server. Once the local exchange and key agreement have happened, the two devices connect to a relay server picked by the phone. From there, they perform a standard FIDO CTAP operation, which is encrypted using the keys from earlier, so the relay server can't see anything that's going on. This whole process is performed by the device and the web browser. The website is not involved at any point in the cross-device communication." Meet passkeys - WWDC22 - Videos - Apple Developer. The process is shown in the below figure.



Meet passkeys - WWDC22 - Videos - Apple Developer.

As shown in the above figure, an "authenticatorGetAssertion" is forward to the phone, which is request to provide cryptographic proof of user authentication. Client to Authenticator Protocol (CTAP) (fidoalliance.org), § 6.2 (Defining authenticatorGetAssertion as the method "used by a host to request cryptographic proof of user authentication as well as user consent to a given transaction, using a previously generated credential that is bound to the authenticator and relying party identifier."). The authenticatorGetAssertion request contains a relying party identifier (RP ID) access key. Client to Authenticator Protocol (CTAP) (fidoalliance.org), § 6.2 (Defining the input parameters of the authenticatorGetAssertion as including a required relying party identifier.). A passkey, however, can only be used for authentication with the same entity (as identified by the RP ID) it was registered with." Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 4 ("A public key credential can only be used for authentication with the same entity (as identified by RP ID) it was registered with."). As such, the PDK on the phone must receive the RP ID access key, unlock the necessary passkey to generate cryptographic proof, and send the proof back to the external device via the relay server. The PDK within the iPhone, however, could only do so if the integrated RDC receiving the authenticatorGetAssertion and returning the cryptographic proof was communicatively

		coupled to the PDK. Apple iPhones, therefore, necessarily have the integrated RDC coupled to the integrated PDK by a first signal line for communication. <u>the integrated RDC coupled to at least one other component of the hybrid device by a second signal line</u> Apple iPhones with iOS 15 or later necessarily have a second signal line coupling the RDC to at least one other component. To function, the RDC must receive power from a battery and/or be coupled to at least one application processor or similar processing unit and/or one or more antenna. Accordingly, the 042 Patent states in Col. 14, ll. 12-14, "the cell phone components and a battery 2004 are coupled to the RDC 304a by signal line 1106." <u>one or more of the integrated RDC and integrated PDK enabling one or more of an application, a function, and a service on one or more of the hybrid device and a device associated with the external RDC</u> When using passkeys to sign in across devices, the PDK of an iPhone with iOS 15 or later enables an authentication service. Authentication is a service provided by the relying party, and the credential ID is necessary for the relying party to perform the authentication function. Upon receiving the response, the relying party will use the credential ID to locate the appropriate public key to verify a signature generated with the private key held by the authenticator. Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 7.2 ("7. Using credential.id (or credential.rawId, if base64url encoding is inappropriate for your use case), look up the corresponding credential public key and let credentialPublicKey be that credential public key… 20. Using credentialPublicKey, verify that sig is a valid signature over the binary concatenation of authData and hash… 22. If all the above steps are successful, continue with the authentication ceremony as appropriate. Otherwise, fail the authentication ceremony.") "[I]f an authenticator returns the wrong credential ID, or if an attacker intercepts and manipulates the credential ID, is that the WebAuthn Relying Party would not look up

		the correct credential public key with which to verify the returned signed authenticator data (a.k.a., assertion), and thus the interaction would end in an error." Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 13.1. As the proper credential ID is needed for relying party to authenticate a user, and the credential ID held within the PDK of the iPhone is included within a response to the authenticatorGetAssertion request generated by the iPhone, the PDK of the iPhone is enabling authentication by relying party. The PDK of the iPhone, accordingly, enables one or more of an application, a function, and a service.
6	The hybrid device of claim 1, wherein the hybrid device is a cell phone and the hybrid device is enabled to provide cell phone service subsequent to the external PDK entering the proximity zone of the integrated RDC and based on the information stored by one or more of the external PDK and the integrated PDK.	Apple iPhones with iOS 15 or later are cell phones.

EXHIBIT E

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

CLAIM CHART – U.S. PATENT NO. 9,049,188
APPLE PASSKEYS

Assignee:	Proxense, LLC
Title:	Hybrid device having a personal digital key and receiver-decoder circuit and methods of use
Filing Date:	2014-02-03
Publication Date:	2015-06-02
Inventor:	Brown, David L.

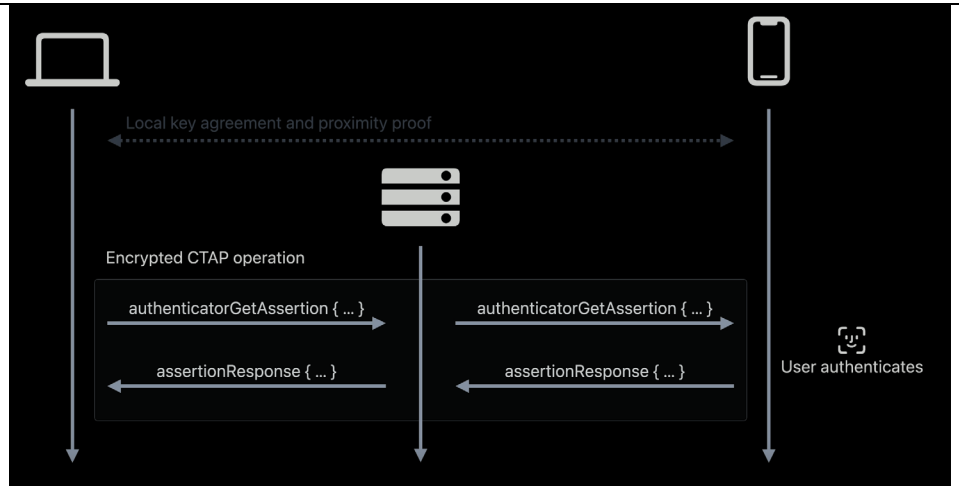
188 Patent Claim[1]		**Accused Instrumentality and Where Each Claim Element Is Found**[2]
1	A hybrid device comprising[3]:	This preamble is not limiting. Apple manufactures, imports, offers for sale, and sells in the United States mobile phones with the iOS 15+ operating system that include an integrated personal digital key and an integrated receiver decoder circuit. ("In macOS Monterey and iOS 15, we announced a developer preview of the solution -- passkeys -- and got so much great feedback. In macOS Ventura and iOS 16, we're excited to make passkeys available to everyone. Now is the time to adopt them." Meet passkeys - WWDC22 - Videos - Apple Developer).
	an integrated personal digital key (PDK) for storing local, secured biometric information for	<u>an integrated personal digital key (PDK)</u>

[1] All PICS set forth herein for any independent patent claims are hereby incorporated by reference into the PICS alleged for any dependent patent claims that depend on such independent claims, as if fully set forth therein.
[2] The Accused Instrumentalities and associated exhibits discussed and/or cited for any claim herein are representative in all material respects of all other accused instrumentalities identified for that claim (e.g., a specified device or service may be used as a representative example in these charts since the other accused instrumentalities have immaterial differences in their hardware and/or software configuration, the cited references are believed to be illustrative of all such accused devices).
[3] Plaintiff's inclusion of any claim preamble in this claim chart should not be interpreted as an admission that the preamble is limiting. Plaintiff reserves the right to take the position that the claim preambles are limiting or not limiting on a claim-by-claim basis.

	authenticating a user and capable of communicating wirelessly with an external receiver-decoder circuit (RDC); and	Apple iPhones with iOS 15 and later operating system enable the use of passkeys by utilizing an integrated personal digital key (PDK). The 188 Patent defines a PDK as including "an antenna and a transceiver for communicating with an RDC (not shown) and a controller and memory for storing information particular to a user." 188 Patent, 13: 46-49. A component of the integrated PDK of the hybrid device, accordingly, is "a controller and memory for storing information particular to a user." The 188 defines the operation of the controller and memory for storing information particular to the user as entailing the receipt of an access key from an external application that is used to access a specific service block. 188 Patent, 6:23-27 ("Regardless of how created, once created, external applications (such as applications 120 in FIG. 1) can gain access to specific service block 112 by providing the corresponding access key 118. In FIG. 2., this is shown conceptually by control logic 250.") The integrated PDK of the hybrid device, therefore, includes a controller placing within memory information that can only be accessed by a corresponding access key provided by an external application. Such memory is present in Apple iPhones with iOS 15, and later, because it is required by the standard. "Passkeys are built on the WebAuthentication -- or WebAuthn standard -- and use public-key cryptography." Meet passkeys - WWDC22 - Videos - Apple Developer. Under the WebAuthn specification, "compliant authenticators protect public key credentials." Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 1. A public key credential refers to a public key credential source, which includes a credential ID. Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 4 (Defining "public key credential" and "public key credential source."). The credential ID uniquely identifies its public key credential source. Web Authentication: An API for accessing

		Public Key Credentials - Level 2 (w3.org), § 4 (Defining a credential ID as "A probabilistically-unique byte sequence identifying a public key credential source and its authentication assertions."). In addition to the credential ID, each public key credential source contains a "credential private key." Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 4, (Defining "public key credential source" as data structure including the credential private key and the credential ID.). "The credential private key is bound to a particular authenticator" and part of an asymmetric key pair containing a public key returned to a relying party. Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 4 (Defining a "credential key pair."). Every Apple iPhone with iOS 15 and later, therefore, will store within memory a credential comprising a private key of an asymmetric key pair and a credential ID uniquely identifying the private/public key pair to which the private key belongs. The credentials stored within Apple iPhones with iOS 15 and later can only be accessed with the appropriate access key. "A public key credential can only be used for authentication with the same entity (as identified by the RP ID) it was registered with." Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 4 ("A public key credential can only be used for authentication with the same entity (as identified by RP ID) it was registered with."). When generating a response, therefore, the authenticator will only retrieve credentials corresponding to the RP ID provided to it by the external server. Client to Authenticator Protocol (CTAP) (fidoalliance.org), § 6.2.2 ("7.1 If the allowList parameter is present and is non-empty, locate all denoted credentials created by this authenticator and bound to the specified rpId. 7.2 If an allowList is not present, locate all discoverable credentials that are created by this authenticator and bound to the specified rpId."). As only credentials corresponding to the RP ID will be retrieved, the RP ID is an access key. As Apple states it, "And the system will take care of only letting me use it in the correct

		app or website, with strong built-in phishing resistance." Meet passkeys - WWDC22 - Videos - Apple Developer. As Apple iPhones will only return credentials corresponding to the RP ID access key provided by the external relying party, Apple iPhones with iOS 15 and later have the controller and memory necessary for a minimal embodiment of a PDK. In addition to the controller and memory, a minimal embodiment of PDK is defined by the 188 Patent as including "an antenna and a transceiver for communication with an RDC." 188 Patent, 13:46-48. With an Apple iPhone with iOS 15 or later, "Passkeys can also be used to sign in across devices in a secure, phishing-resistant manner. Here's how that works. There are two devices here. The client, which is the device or web browser where I'm signing in, and the authenticator, which is the device which has my passkey. First, the client shows a QR code, which the authenticator scans. This QR code contains a URL that encodes a pair of single-use encryption keys. Then, the authenticator produces a Bluetooth advertisement containing routing information for a network relay server. Once the local exchange and key agreement have happened, the two devices connect to a relay server picked by the phone. From there, they perform a standard FIDO CTAP operation, which is encrypted using the keys from earlier, so the relay server can't see anything that's going on. This whole process is performed by the device and the web browser. The website is not involved at any point in the cross-device communication." Meet passkeys - WWDC22 - Videos - Apple Developer. The process is shown in the below figure.



Meet passkeys - WWDC22 - Videos - Apple Developer.

Establishing the connection over the internet requires utilizing either the device's Wi-Fi or cellular capabilities – both of which are wireless protocols. Accordingly, iPhones with iOS 15 and later have the antenna and transceiver necessary to implement the wireless protocols enabling transmission over the internet.

Having each of the elements of a minimal embodiment of a PDK, Apple iPhones with iOS 15 and later include an integrated PDK.

storing local, secured biometric information for authenticating a user

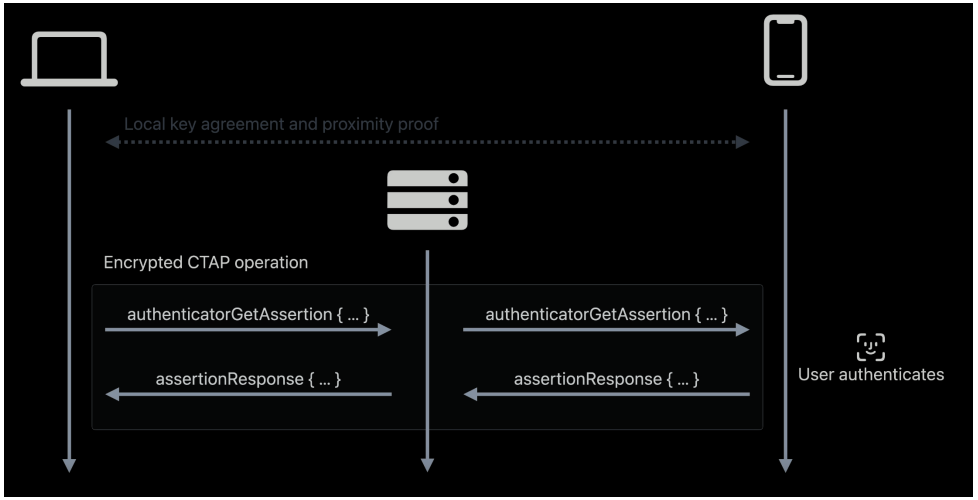
Apple iPhones with iOS 15 later store local, secure biometric information for authenticating user.

"Passkeys are a replacement for passwords. They are faster and easier to sign in. Just use Touch ID or Face ID to authenticate and you're done." Deploy passkeys at work - WWDC23 - Videos - Apple

		Developer. "Apple platforms will always require UV for passkeys when biometrics are available, so you don't have to worry about that." Meet passkeys - WWDC22 - Videos - Apple Developer. Accordingly, Apple iPhones with iOS 15 and later locally store biometric information for authenticating user. Apple iPhones utilize the Secure Enclave to securely store the biometric information for authentication. "During enrollment, the Secure Enclave processes, encrypts, and stores the corresponding Touch ID and Face ID template data." Apple Platform Security, page 19. Apple has described the Secure Enclave thusly: "When you store a private key in the Secure Enclave, you never actually handle the key, making it difficult for the key to become compromised. Instead, you instruct the Secure Enclave to create the key, securely store it, and perform operations with it. You receive only the output of these operations, such as encrypted data or a cryptographic signature verification outcome." Protecting Keys with the Secure Enclave "The Secure Enclave is a dedicated secure subsystem … isolated from the main processor to provide an extra layer of security designed to keep sensitive data secure even when the Application Processor kernel becomes compromised." Apple Platform Security, page 9. Protected even when a hack or malware comprises the Application Processor, the biometric data for Touch ID and Face ID are securely stored. Apple iPhones with iOS 15 and later, therefore, locally store secured biometric data for authenticating a user. capable of communicating wirelessly with an external receiver-decoder circuit (RDC)

		As detailed above, Apple iPhones with iOS 15 or later enable the use of passkeys to sign in across devices in a secure, phishing-resistant manner by utilizing either the device's Wi-Fi or cellular capabilities – both of which are wireless protocols. Consequently, Apple iPhones with iOS 15 and later are capable of communicating wirelessly with an external receiver decoder circuit.
	an integrated RDC for communicating wirelessly with at least one external PDK within a proximity zone, the integrated RDC coupled to the integrated PDK by a first signal line for communication, the integrated RDC coupled to at least one other component of the hybrid device by a second signal line, one or more of the integrated RDC and integrated PDK enabling one or more of an application, a function, and a service.	<u>an integrated RDC for communicating wirelessly with at least one external PDK</u> As detailed above, Apple iPhones with iOS 15 and later communicate passkey signatures over an encrypted connection, through the internet, via Wi-Fi, and/or cellular protocols. Accordingly, Apple iPhones with iOS 15 and later include an RDC enabling wireless communications with at least one external device, such as a device running macOS Monterey or later requires. <u>the integrated RDC coupled to the integrated PDK by a first signal line for communication</u> Enabling the use of passkeys across devices with a QR code, Apple iPhones with iOS 15 or later include a signal line for communication that couples the integrated RDC to the integrated PDK. As noted above, using a passkey on Apple iPhone to sign into a website on external device begins by scanning a QR code. "First, the client shows a QR code, which the authenticator scans. This QR code contains a URL that encodes a pair of single-use encryption keys. Then, the authenticator produces a Bluetooth advertisement containing routing information for a network relay server. Once the local exchange and key agreement have happened, the two devices connect to a relay server picked by the phone. From there, they perform a standard FIDO CTAP operation, which is encrypted using the keys from earlier, so the

relay server can't see anything that's going on. This whole process is performed by the device and the web browser. The website is not involved at any point in the cross-device communication." Meet passkeys - WWDC22 - Videos - Apple Developer. The process is shown in the below figure.



Meet passkeys - WWDC22 - Videos - Apple Developer.

As shown in the above figure, an "authenticatorGetAssertion" is forward to the phone, which is request to provide cryptographic proof of user authentication. Client to Authenticator Protocol (CTAP) (fidoalliance.org), § 6.2 (Defining authenticatorGetAssertion as the method "used by a host to request cryptographic proof of user authentication as well as user consent to a given transaction, using a previously generated credential that is bound to the authenticator and relying party identifier."). The authenticatorGetAssertion request contains a relying party identifier (RP ID) access key. Client to Authenticator Protocol (CTAP) (fidoalliance.org), § 6.2 (Defining the input parameters of the authenticatorGetAssertion as including a required relying party identifier.). A passkey, however, can only be

		used for authentication with the same entity (as identified by the RP ID) it was registered with." Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 4 ("A public key credential can only be used for authentication with the same entity (as identified by RP ID) it was registered with."). As such, the PDK on the phone must receive the RP ID access key, unlock the necessary passkey to generate cryptographic proof, and send the proof back to the external device via the relay server. The PDK within the iPhone, however, could only do so if the integrated RDC receiving the authenticatorGetAssertion and returning the cryptographic proof was communicatively coupled to the PDK. Apple iPhones, therefore, necessarily have the integrated RDC coupled to the integrated PDK by a first signal line for communication. <u>the integrated RDC coupled to at least one other component of the hybrid device by a second signal line</u> Apple iPhones with iOS 15 or later necessarily have a second signal line coupling the RDC to at least one other component. To function, the RDC must receive power from a battery and/or be coupled to at least one application processor or similar processing unit and/or one or more antenna. Accordingly, '188 Patent states in Col. 14, ll. 12-14, "the cell phone components and a battery 2004 are coupled to the RDC 304a by signal line 1106." <u>one or more of the integrated RDC and integrated PDK enabling one or more of an application, a function, and a service</u> When using passkeys to sign in across devices, the PDK of an iPhone with iOS 15 or later enables an authentication service.

		Authentication is a service provided by the relying party, and the credential ID is necessary for the relying party to perform the authentication function. Upon receiving the response, the relying party will use the credential ID to locate the appropriate public key to verify a signature generated with the private key held by the authenticator. Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 7.2 ("7. Using credential.id (or credential.rawId, if base64url encoding is inappropriate for your use case), look up the corresponding credential public key and let credentialPublicKey be that credential public key… 20. Using credentialPublicKey, verify that sig is a valid signature over the binary concatenation of authData and hash… 22. If all the above steps are successful, continue with the authentication ceremony as appropriate. Otherwise, fail the authentication ceremony.") "[I]f an authenticator returns the wrong credential ID, or if an attacker intercepts and manipulates the credential ID, is that the WebAuthn Relying Party would not look up the correct credential public key with which to verify the returned signed authenticator data (a.k.a., assertion), and thus the interaction would end in an error." Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 13.1. As the proper credential ID is needed for relying party to authenticate a user, and the credential ID held within the PDK of the iPhone is included within a response to the authenticatorGetAssertion request generated by the iPhone, the PDK of the iPhone is enabling authentication by relying party. The PDK of the iPhone, accordingly, enables one or more of an application, a function, and a service.
3	The hybrid device of claim 1, wherein at least one of the one or more of the application, the function, and the service are enabled at least in part on a device external to the hybrid	As detailed above, the PDK of an iPhone with iOS 15 or later is enabling an authentication service by a relying party that is external to the iPhone. The relying party must be communicatively coupled to the RDC of the external device to receive the iPhones response during the authentication ceremony. Accordingly, the application, the function, and the service are enabled at least

	device and communicatively coupled to the external RDC.	in part on a device external to the hybrid device and communicatively coupled to the external RDC.
4	The hybrid device of claim 1, wherein the local, secured biometric information for authenticating a user is based on one or more of a fingerprint, palm print, a retinal scan, an iris Scan, a photograph, a signature, a voice sample, DNA and RNA.	"Passkeys are a replacement for passwords. They are faster and easier to sign in. Just use Touch ID or Face ID to authenticate and you're done." Deploy passkeys at work - WWDC23 - Videos - Apple Developer. Accordingly, the secured biometric information for authenticating a user is based on a fingerprint or a photograph.
7	The hybrid device of claim 1, wherein the hybrid device is a cell phone.	Apple iPhones with iOS 15 or later are cell phones.

EXHIBIT F

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

U.S. Patent Number US 8,886,954 – Apple Pay[1]

Assignee:	Proxense, LLC
Title:	Biometric personal data key (PDK) authentication
Filing Date:	2012-12-10
Publication Date:	2014-11-11
Inventor:	Giobbi, John J.

'954 Patent Claim[1]		**Accused Instrumentality And Where Each Claim Element Is Found[2]**
1	A method comprising[3]:	This preamble is not limiting. Apple, using its proprietary software on Apple iPhones with Apple Pay pre-installed, alone or as part of a joint enterprise with various issuing banks and/or via the direction and control the activities of various issuing banks, carry out the claimed method, literally or by the doctrine of equivalents, for at least the reasons set forth below.
	persistently storing biometric data of a user and a plurality of codes and other data values comprising a device ID code uniquely identifying an integrated device and a secret decryption value in a tamper proof format written to a storage element on the integrated device that is not capable of being subsequently altered;	<u>Persistently storing biometric data of a user … in a tamper proof format written to a storage element on the integrated device that is not capable of being subsequently altered</u> Utilizing Apple's Secure Enclave, iPhones persistently store biometric data of a user written to storage element on the integrated device that is unable to be subsequently altered. "The Secure Enclave is a dedicated secure subsystem … isolated from the main processor to provide an extra layer of security designed to keep sensitive data secure even when the Application Processor kernel becomes compromised." Apple Platform Security , page 9. Protecting data even when a hack or malware comprises the Application Processor, the Secure Enclave provides a tamper proof format for sensitive data.

[1] All contentions set forth herein for any independent patent claims are hereby incorporated by reference into the contentions alleged for any dependent patent claims that depend on such independent claims, as if fully set forth therein.

[2] The Accused Instrumentalities and associated exhibits discussed and/or cited for any claim herein are representative in all material respects of all other accused instrumentalities identified for that claim (e.g., a specified device or service may be used as a representative example in these charts since the other accused instrumentalities have immaterial differences in their hardware and/or software configuration, the cited references are believed to be illustrative of all such accused devices).

[3] Plaintiff's inclusion of any claim preamble in this claim chart should not be interpreted as an admission that the preamble is limiting. Plaintiff reserves the right to take the position that the claim preambles are limiting or not limiting on a claim-by-claim basis.

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

The sensitive data protected by Apple's Secure Enclave includes the biometric data. "During enrollment, the Secure Enclave processes, encrypts, and stores the corresponding Touch ID and Face ID template data." Apple Platform Security , page 19.

persistent storing … a plurality of codes and other data values comprising a device ID code uniquely identifying the integrated device … in a tamper proof format written to a storage element on the integrated device that is not capable of being altered

In addition to registering user biometrics, Apply Pay requires registering credit and debit cards. To safeguard registered card information, Apply Pay utilizes EMV payment tokens stored within a secure element as a unique device account number. "Apple [was] among the first to implement EMV payment tokens in digital wallets that hold credential for several payments use cases." EMV Payment Tokenization Primer and Lessons Learned, page 12. "EMV payment tokens are open-loop tokens provisioned ty a [Token Service Provider] and, like other tokens, are used to replace the actual payment credential (e.g., [Primary Account Number]) with another numeric value." *Id.* Accordingly, when a card is added to Apple Pay, a payment token is issued to the iPhone in exchange for the for a credit card number by TSP, such as Visa, MasterCard, Discover, and American Express. *Id*, at 23 (Figure 5). Adding a card to Apple Pay on an iPhone causes "a unique Device Account Number [to be] created, encrypted, and then stored in the Secure Element." Apple Platform Security , page 142.

Secure elements are recognized as "a dynamic environment to store data securely, process data securely and perform communication with external entities securely," that "will not allow unauthorized access." EMV Payment Tokenization Primer and Lessons Learned, page 41. Securely storing a EMV payment token as an unique Device Account Number on an element not allowing unauthorized access, iPhones with Apple Pay persistently store a device ID code uniquely identifying the smartphone in a tamper proof format, written to a storage element on the integrated device, that is unable to be subsequently altered.

Adding a card to Apple Pay requires the approval of the bank issuing the card (issuing bank). When adding a card to Apple Wallet, "Apple securely sends the card information, along with other information about user's account and device, to the card issuer or card issuer's authorized service provider (usually the payment network). Using this information, the card issuer (or its service provider) determines whether to approve the user's request to add the card to Apple Wallet." Apple Platform Security, 179. Thus, the issuing banks determine whether to allow a card to be added to Apple Wallet on an iPhone.

To provide the information necessary for the issuing banks to decide if a card will be added to Apple Wallet, "Apple Pay uses three server-side calls to send and receive communication with the card

issuer or payment network." Apple Platform Security, 179. The information transmitted via these calls "enable the card issuer to verify, approve, and add cards to Apple Wallet." *Id.* The issuing banks, accordingly, make determinations under the direction and control of Apple. Furthermore, the final determination provided by the issuing banks gives each of the issuing banks equal right of control as they can refuse to allow their cards to be added to Apple Wallet.

The result of adding a card to Apple is the creation of a device ID code uniquely identifying the integrated device. Card numbers are not stored on the device or by Apple's servers. "Instead, a unique Device Account Number is created by the card issuer, sent encrypted to Apple, and then stored in the Secure Element." Apple Platform Security, 179. Thus, cards are added to Apple Wallet as "***unique Device Account Numbers***," The unique Device Account "are never stored on Apple Pay servers or backed up to iCloud, and it's isolated from: Devices that use biometric authentication[;] Apple Watch[; and] Mac computers with Apple silicon that use the Magic Keyboard with Touch ID." Apple Platform Security, 179. As the Device Account Numbers are isolated from other devices and not synced, each remains unique to a specific device.

<u>persistently storing … a secret decryption value in a tamper proof format written to a storage element on the integrated device that is not capable of being subsequently altered;</u>

An iPhone with Apple Pay persistently stores a secret decryption value as a pairing key shared by the Secure Enclave and the Secure Element. "Communication between the Secure Enclave and the Secure Element takes place over a serial interface…Though not directly connected, the Secure Enclave and Secure Element can communicate securely using a shared pairing key that provisioned

		during the manufacturing process." Apple Platform Security , page 144. For the shared key to persist, it must be stored by both the Secure Element and the Secure Enclave. "The Secure Enclave is a dedicated secure subsystem … isolated from the main processor to provide an extra layer of security designed to keep sensitive data secure even when the Application Processor kernel becomes compromised." Apple Platform Security , page 9. Protecting data even when a hack or malware comprises the Application Processor, the Secure Enclave provides a tamper proof format for sensitive data, such as shared pairing key with the Secure Element. Furthermore, secure elements are recognized as "a dynamic environment to store data securely, process data securely and perform communication with external entities securely," that "will not allow unauthorized access." EMV Payment Tokenization Primer and Lessons Learned, page 41. Additionally, Apple has described the Secure Enclave thusly: "When you store a private key in the Secure Enclave, you never actually handle the key, making it difficult for the key to become compromised. Instead, you instruct the Secure Enclave to create the key, securely store it, and perform operations with it. You receive only the output of these operations, such as encrypted data or a cryptographic signature verification outcome." Protecting Keys with the Secure Enclave Both the Secure Enclave and Secure Element, thus, store a secret decryption value in a tamper proof format written to a storage element on the integrated device that is unable to be subsequently altered. .
	responsive to receiving a request for a biometric verification of the user, receiving scan data from a biometric scan	"[B]efore information is transmitted, the user must authenticate using Touch ID, Face ID or their passcode… No payment information is sent without user authentication." Apple Platform Security , page 145. This requirement for authentication necessitates the Secure Enclave receiving a request for biometric verification. Completing this verification requires the Secure Element receive biometric scan data from either the TrueDepth camera or the fingerprint sensor. To biometrically verify themselves using Touch ID, the user places their finger on the fingerprint sensor or looks at their device. "The sensor captures the biometric image and securely transmits it to the Secure Enclave." Apple Platform Security , page 19. The Secure Enclave, accordingly, receives scan data from a biometric scan by the fingerprint sensor. When using Face ID, the user simply looks at their device. "After the TrueDepth camera confirms the presence of an attentive face, it protects and reads thousands of infrared dots to form a depth of the face along with a 2D image… A portion of the Secure Neural Engine – protected within the Secure Enclave – transforms this data into mathematical representation and compares that representation to the enrolled facial data." Apple Platform Security , page 20. As the Secure Neural Network is within the Secure Enclave, the Secure Enclave must receive scan data from the TrueDepth camera.

	comparing the scan data to the biometric data to determine whether the scan data matches the biometric data	"[B]efore information is transmitted, the user must authenticate using Touch ID, Face ID or their passcode… No payment information is sent without user authentication." Apple Platform Security , page 145. This requirement for authentication necessitates the Secure Enclave match the received scan data to the stored biometric data. "During matching, the Secure Enclave compares incoming data from the biometric sensor against the stored templates to determine whether to unlock the device or respond to that a match is valid (for Apple Pay, in-app, and other uses of Touch ID and Face ID)." Apple Platform Security , page 19.
	responsive to a determination that the scan data matches the biometric data, wirelessly sending one or more codes and other values from the plurality of codes and other data values for authentication to a third party that operates a trusted authority, wherein the one or more codes and other data values includes the device ID code; and	<u>responsive to a determination that the scan data matches the biometric data, wirelessly sending one or more codes from the plurality of codes and the other data values</u> "[B]efore information is transmitted, the user must authenticate using Touch ID, Face ID or their passcode… No payment information is sent without user authentication." Apple Platform Security , page 145. This requirement for authentication necessitates the "unique Device Account Number" stored in the Secure Element and used for payment is only sent responsive to a determination that the scan data matches the biometric data. As previously noted, "[d]uring matching, the Secure Enclave compares incoming data from the biometric sensor against the stored templates to determine whether to unlock the device or respond to that a match is valid (for Apple Pay, in-app, and other uses of Touch ID and Face ID)." Apple Platform Security , page 19. Apple Platform Security , page 19. If the scan matches the template, "the Secure Enclave then sends signed data about the type of authentication and details about the transaction to the (contactless or within apps) to the Secure Element… It's securely delivered to the Secure Element by leveraging the paring key." Apple Platform Security , page 144. After receiving confirmation of authentication, "contactless responses prepared by the payment applets within the Secure Element are exclusively routed by the Controller to the NFC field." Apple Platform Security , page 141. Thus, the Device Account Number (i.e., payment token) is only released from the Secure Element and transmitted to the payment terminal wirelessly by the NFC controller after biometric verification of the user. <u>responsive to a determination that the scan data match the biometric data, … sending one or more codes and other values from the plurality of codes and other data value for authentication … wherein the one or more codes includes the device ID code</u> "[B]efore information is transmitted, the user must authenticate using Touch ID, Face ID or their passcode… No payment information is sent without user authentication." Apple Platform Security , page 145. This requirement for authentication necessitates the "unique Device Account Number" stored in the Secure Element and used for payment is only sent responsive to a determination that the scan data matches the biometric data. As previously noted, "[d]uring matching, the Secure

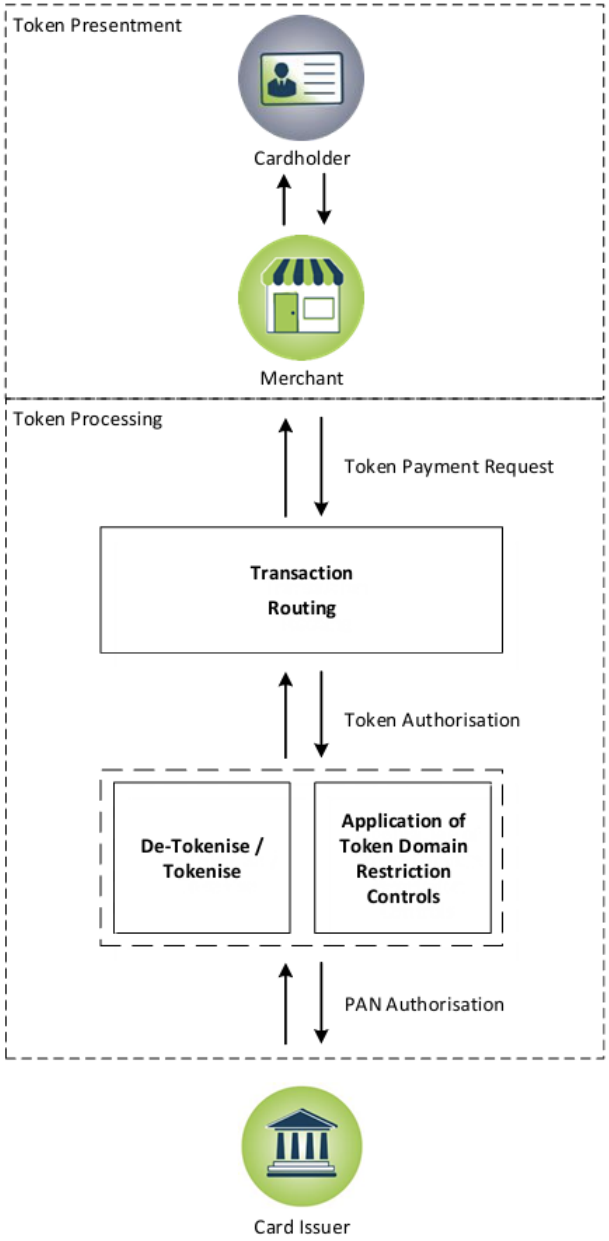
Enclave compares incoming data from the biometric sensor against the stored templates to determine whether to unlock the device or respond to that a match is valid (for Apple Pay, in-app, and other uses of Touch ID and Face ID)." Apple Platform Security , page 19. Apple Platform Security , page 19. If the scan matches the template, "the Secure Enclave then sends signed data about the type of authentication and details about the transaction to the (contactless or within apps) to the Secure Element… It's securely delivered to the Secure Element by leveraging the paring key." Apple Platform Security , page 144.

After receiving confirmation of authentication, "contactless responses prepared by the payment applets within the Secure Element are exclusively routed by the Controller to the NFC field." Apple Platform Security , page 141. Thus, the Device Account Number (i.e., payment token) is only released from the Secure Element and transmitted to the payment terminal wirelessly by the NFC controller after biometric verification of the user.

<u>responsive to a determination that the scan data matches the biometric data, wirelessly sending … for authentication to a third party that operates a trusted authority</u>

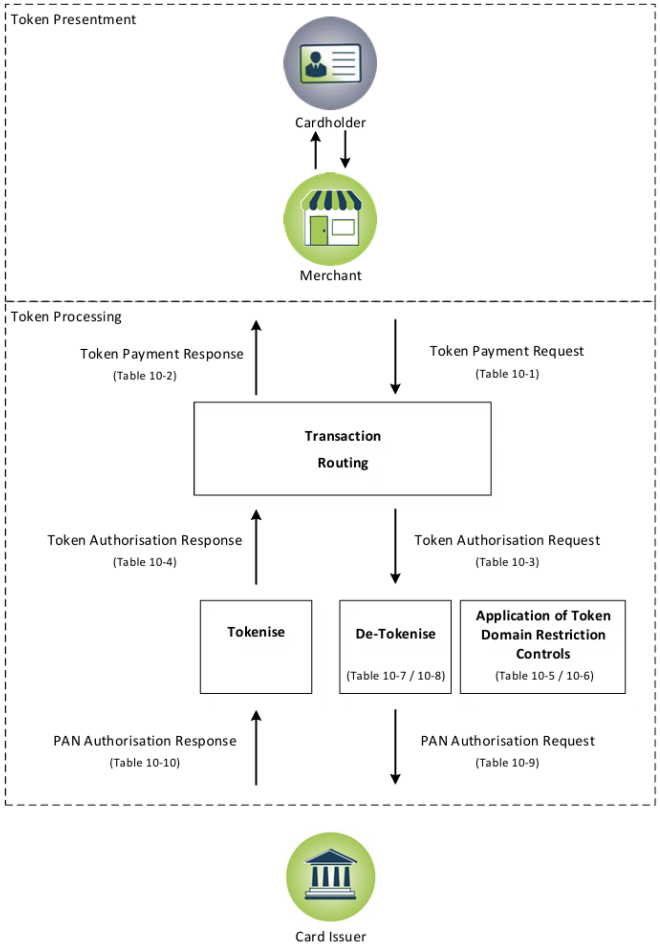
After being received by the payment terminal, the unique Device Account Number is transmitted through the payment network as Token Payment Request. *See* EMV Payment Tokenisation Specification: Technical Framework, v2.2 (2020), Figure 10.1 and page 81 ("The basic authorisation flow is shown in Figure 10.1."). As it travels through the payment network to the token service provider, the Token Payment Request, containing the payment token wirelessly transmitted from an iPhone, is converted to a Token Authorisation Request received by the Token Service Provider. *Id.*

Figure 3.2: Payment Token Transaction Overview



APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

Figure 10.1: Illustrative Payment Token Processing Flow for Authorisations



"The Token Payment Request is the first leg … of any transaction, starting with the interaction between the Cardholder and the Merchant." EMV Payment Tokenisation Specification: Technical Framework, v2.2 (2020), 84. The transaction begins with the cardholder interacting with a merchant's terminal. "The Cardholder interacting with a Terminal, website or application will result in Token Processing related data being passed to the Merchant." *Id.* As detailed above, the data is passed from the iPhone via wireless transmission. The passed data includes Payment Token (Device Account Number). *See* EMV Payment Tokenisation Specification: Technical Framework, v2.2 (2020), Table 10.1

Table 10.1: Fields Included in Token Payment Requests

Field Content	Field Name	R/C/O	Condition/Option	Comment
Payment Token	PAN	R		
Token Expiry Date	PAN Expiry Date	R		

Field Content	Field Name	R/C/O	Condition/Option	Comment
Token Presentment Mode	POS Entry Mode	R		Token Control Data
Token Cryptogram	Payment Network Specific	C	Required for EMV based and application based commerce in Cardholder Initiated Transactions	Generated and passed in appropriate cryptogram field. Token Control Data
Token Requestor ID	Payment Network Specific	O	Present if received by the Terminal or sourced by the Merchant directly from Token Requestor related interactions	
Merchant Identifiers	Payment Network Specific	O	May be present for e-commerce transactions	Token Control Data
Payment Account Reference	Payment Account Reference	O	Present if received by the Terminal or sourced by the Merchant directly from Token Requestor related interactions	

R – Required, C – Conditional, O – Optional

Accordingly, the Payment Token (Device Account Number) received from Apple Pay on iPhones is part of the Token Payment Request as the entry in the PAN field and thus takes the place of the account number on the card the issuing bank allowed to be added to Apple Wallet.

During transaction routing, the Token Payment Request is transformed to Token Authorisation Request. As with the Token Payment Request, the Token Authorisation Requests also includes the Payment Token (Device Account Number) the issuing bank allowed to be added to Apple Wallet in the PAN field. As such, the Payment Token (Device Account Number) takes the place of a card's

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

account number. *See* EMV Payment Tokenisation Specification: Technical Framework, v2.2 (2020), Table 10.3

Table 10.3: Fields Included in Token Authorisation Requests

Field Content	Field Name	R/C/O	Condition/Option	Comment
Payment Token	PAN	R		

Field Content	Field Name	R/C/O	Condition/Option	Comment
Token Expiry Date	PAN Expiry Date	R		
Token Presentment Mode	POS Entry Mode	R		Inserted by Acquirer. Token Control Data
Token Cryptogram	Payment Network Specific	C	Required for EMV based and application based commerce in Cardholder Initiated Transactions	Token Control Data
Token Requestor ID	Payment Network Specific	O	May be present if read from Terminal or sourced directly from Token Requestor and passed to the Acquirer	
Merchant Identifiers	Payment Network Specific	O	May be present for e-commerce transactions	Token Control Data
Payment Account Reference	Payment Account Reference	O	Present if received by the Terminal or sourced by the Merchant directly from Token Requestor related interactions	

R – Required, C – Conditional, O – Optional

"The Token Authorisation request process continues until De-Tokenisation has been completed." EMV Payment Tokenisation Specification: Technical Framework, v2.2 (2020), 86. De-Tokenisation is "the process of converting a Payment Token and Token Expiry Date to its underlying PAN and PAN Expiry Date based on the Payment Token / Token Expiry Date mapping to the underlying PAN / PAN Expiry Date stored in the Token Vault." Id, at 6. "Token Service

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

		Providers are responsible for a number of discrete functions which may include, but are not limited to: Maintenance and operation of a Token Vault … [and] De-Tokenisation." *Id*, at 19. Maintaining the token vault and performing de-tokenization, a token service provider receives token authorization request. Upon receiving the request, "the Payment Token SHALL be de-tokenised to the underlying PAN in the incoming Token Authorisation prior to sending the PAN Authorisation to the Card Issuer." *Id*., at 91. As shown in Figure 10.1, "Once a Payment Token has been de-tokenised, the final request step is to initiate a PAN Authorisation, destined for the Card Issuer's authorisation system." *Id*. at 92. The PAN authorization request sent to the Card Issuer, contains the "underlying PAN from token vault." *Id*. Table 10.9, first row, at 93.

Table 10.9: Fields Included in PAN Authorisation Requests

Field Content	Field Name	R/C/O	Condition/Option	Comment
PAN	PAN	R		Underlying PAN from Token Vault
PAN Expiry Date	PAN Expiry Date	R		Expiry date associated with the underlying PAN
Token Presentment Mode	POS Entry Mode	R		Payment Token related data
Token Requestor ID	Payment Network Specific	R		Payment Token related data
Payment Token	Payment Network Specific	R		
Token Expiry Date	Payment Network Specific	O		Payment Token related data
Token Assurance Method	Payment Network Specific	C	Required if provided by Payment Network	Payment Token related data
Token Assurance Data	Payment Network Specific	O		Payment Token related data
Payment Account Reference	Payment Account Reference	O	Present if available to the Token Service Provider	The Payment Account Reference associated with the PAN

R – Required, C – Conditional, O – Optional

Obtaining the underlying primary account number from the token vault, as to prepare the PAN authorization request, requires gaining access to the file containing the account number. Converting the payment token to its underlying account number based on the Payment Token / Token Expiry Date mapping to the

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

		underlying PAN / PAN Expiry Date stored in the token vault, as to determine which file access, the token service provider necessarily authenticates the payment token and the other data values received from the iPhone. Opening the token vault to perform de-tokenisation occurs in response to receiving the token authorization request containing the payment token. De-Tokenisation is "the process of converting a Payment Token and Token Expiry Date to its underlying PAN and PAN Expiry Date based on the Payment Token / Token Expiry Date mapping to the underlying PAN / PAN Expiry Date stored in the Token Vault." Id, at 6. "The Payment Token SHALL be de -tokenised to the underlying PAN in the incoming Token Authorisation prior to sending the PAN Authorisation to the Card Issuer." Id., at 91. Detokenizing in response to an incoming token authorization requests requires the token service provider be sent the token authorization. As noted supra, the token authorization sent to the token service provider contains the payment token wirelessly sent from an iPhone after successful biometric authentication. Detokenizing in response to an incoming token authorization request containing a payment token provided by an iPhone following successful biometric authentication, therefore, requires the token service provider is wirelessly sent one or more codes from the plurality of codes and the other data values for authentication responsive to a determination that the scan data matches the biometric data.
	receiving, at an application, an access message from the trusted authority indicating that the trusted authority successfully authenticated the one or more codes and other data values sent to the third party and allowing the user access to the application.	"Once a Payment Token has been de-tokenised, the final request step is to initiate a PAN Authorisation, destined for the Card Issuer's authorisation system." EMV Payment Tokenisation Specification: Technical Framework, v2.2 (2020), 92. The PAN Authorisation contains the actually account number for the card the issuing bank allowed to be added to Apple Wallet. *See* EMV Payment Tokenisation Specification: Technical Framework, v2.2 (2020), Table 10.9.

Table 10.9: Fields Included in PAN Authorisation Requests

Field Content	Field Name	R/C/O	Condition/Option	Comment
PAN	PAN	R		Underlying PAN from Token Vault
PAN Expiry Date	PAN Expiry Date	R		Expiry date associated with the underlying PAN
Token Presentment Mode	POS Entry Mode	R		Payment Token related data
Token Requestor ID	Payment Network Specific	R		Payment Token related data
Payment Token	Payment Network Specific	R		
Token Expiry Date	Payment Network Specific	O		Payment Token related data
Token Assurance Method	Payment Network Specific	C	Required if provided by Payment Network	Payment Token related data
Token Assurance Data	Payment Network Specific	O		Payment Token related data
Payment Account Reference	Payment Account Reference	O	Present if available to the Token Service Provider	The Payment Account Reference associated with the PAN

R – Required, C – Conditional, O – Optional

Accordingly, following authentication of the Payment Token (Device Account Number) an Authorization Request containing the PAN received from the token vault is received by the issuing bank's "authorisation system" so that the user may have the requested payment authorized by the issuing bank. Therefore, the authorization request containing the PAN in place of the unique Device Account Number allows the user access to the issuer's computer software necessary to process and authorize the payments and is thus an access message from the Token Service Provider received by the issuing Bank's computer software in response to successful authentication of the Device Account Number.

APPLE EXHIBIT 1052
APPLE v. PROXENSE
IPR2024-00783

2	The method of claim 1, wherein the one or more codes and other data values are transmitted to the trusted authority over a network.	The payment token (i.e., Device ID Code of the one or more codes and other data values) will be transmitted to the token service provider acting as third party trusted authority (i.e., agent) over the payment network.

5	The method of claim 1, wherein the biometric data includes one or more of a fingerprint, palm print, a retinal scan, an iris scan, a hand geometry, a facial recognition, a signature recognition and a voice recognition.	The sensitive data protected by Apple's Secure Enclave includes the biometric data. "During enrollment, the Secure Enclave processes, encrypts, and stores the corresponding Touch ID and Face ID template data." Apple Platform Security , page 19.
6	The method of claim 1, wherein the integrated device comprises one or more of a mobile phone, tablet, laptop, mp3 player, mobile gaming device, watch and a key fob.	iPhones are mobile phones.
7	The method of claim 1, wherein the application includes one or more of a casino machine, a keyless lock, a garage door opener, an ATM machine, a hard drive, computer software, a web site, a file and a financial account.	As noted above, receipt of the PAN Authorisation Request gives access to the issuer's computer software used validate and authorize a transaction.