EXHIBIT G

Exhibit 1036
Microsoft v. Proxense
IPR2024-00573

U.S. Patent Number US 8,646,042– Final Infringement Contentions[1]

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[2]		**Accused Instrumentality And Where Each Claim Element Is Found[3]**
10	A method comprising[4]:	This preamble is not limiting. Microsoft Entra ID, as well as its subscribers (which Microsoft directs and controls with online instructions and direct on-screen prompts to action to receive the benefits of not having to maintain an authentication server and/or permit the use of a Microsoft Account to sign on users) perform the claimed method, literally or by the doctrine of equivalents, for at least the reasons set forth below. Non-limiting examples include the use of Passkeys and/or device bound FIDO Credentials created at the direction of Microsoft and stored on Android and iOS devices as directed by respective OS, for purposes of accessing application and services, such as 365 services, including Outlook, and Word, and services and on-premise applications provided by subscribers to Microsoft Entra ID.

[1] The Final Infringement Contentions (FICS) 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 PICS 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 ¶2 of the Order Governing Proceedings – Patent Cases under Hon. Alan D. Albright.

[2] All FICS set forth herein for any independent patent claims are hereby incorporated by reference into the FICS 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.

	creating a first wireless link between an integrated receiver-decoder circuit (RDC) of a hybrid device and an external personal digital key (PDK), the hybrid device including an integrated PDK and the integrated RDC;	creating a first wireless link between an integrated receiver-decoder circuit (RDC) of a hybrid device and an external personal digital key (PDK) "Authenticator is a free passkey solution that lets users do passwordless phishing-resistant authentications from their own phones." PROX_MSFT_003570, Microsoft Authenticator authentication method - Microsoft Entra ID | Microsoft Learn. Android and iOS devices utilizing Microsoft's Authenticator app communicate passkey signatures over an encrypted connection, through the internet, via Wi-Fi, and/or cellular protocols. Android and iOS devices include an RDC enabling encrypted communications. Establishing the connection between an Android or iOS device and an external device the requires a first wireless connection between the two devices. The CTAP specification details a first wireless signal using a hybrid BLE protocol to perform a handshake via a secure tunnel server to establish a secure connection and to prove proximity. https://fidoalliance.org/specs/fido-v2.2-rd-20230321/fido-client-to-authenticator-protocol-v2.2-rd-20230321.html#ble; https://learn.microsoft.com/en-us/windows/security/identity-protection/passkeys/?tabs=windows%2Cintune ("For passkey cross-device authentication scenarios, both the Windows device and the mobile device must have Bluetooth enabled and connected to the Internet."); MSFT_PROXE000198405 at 198504, 198511-198523. "The client platform speaks first to prove possession of the BLE advert. The authenticator thus needs only to receive the client platform's handshake message and send a reply in order to complete the handshake. . . . This transport requires a proof of proximity to help prevent attacks, thus notification of the connection attempt comes in the form of a BLE advertisement . . . The connection nonce is the value that demonstrates possession of the BLE advert, and thus proximity to the authenticator. The authenticator needs two values to start communicating on the tunnel: the link ID so that it knows which client platform is contacting it (and thus which keys to use), and a nonce from the client platform." the hybrid device including an integrated PDK "Authenticator is a free passkey solution that lets users do passwordless phishing-resistant authentications from their own phones." PROX_MSFT_003570, Microsoft Authenticator authentication method - Microsoft Entra ID | Microsoft Learn.

		Providing a “free passkey solution”, the Microsoft Authenticator app creates a PDK on iOS and Android devices, to the extent one is not natively present. The 042 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”. 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 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, 6:23-39 ("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 placed within memory information that can only be accessed by a corresponding access key provided by an external application. Such memory is created on Android and iOS devices by the Microsoft Authenticator App. The FIDO CTAP specification incorporates the WebAuthn Specification. PROX_MSFT_002849, Client to Authenticator Protocol (CTAP) (fidoalliance.org), § 1.1 ("This specification is part of the FIDO2 project, which includes this specification and is related to the W3C [WebAuthn] specification."). Under the WebAuthn specification, "compliant authenticators protect public key credentials." PROX_MSFT_003098, 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. PROX_MSFT_003098, 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. *See* PROX_MSFT_003098, 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". PROX_MSFT_003098, 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. PROX_MSFT_003098, Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 4 (Defining a "credential key pair".). Every FIDO compliant authenticator, 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 a compliant authenticator 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." PROX_MSFT_003098, 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 FIDO server. PROX_MSFT_002849, 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. During a WebAuthn authentication ceremony, an authenticator receives an "authenticatorGetAssertion" request to provide cryptographic proof of user authentication. PROX_MSFT_002849, 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). PROX_MSFT_002849, Client to Authenticator Protocol (CTAP) (fidoalliance.org), § 6.2 (Defining the input parameters of the authenticatorGetAssertion as including a required relying party identifier.). The authenticatorGetAssertion is called in response to get request issued by the relying party attempting authentication. PROX_MSFT_003098, Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 5.1.4 ("WebAuthn Relying Parties call navigator.credentials.get({publicKey:..., ...}) to discover and use an existing public key credential, with the user's consent… If the user picks a credential source, the user agent then uses § 6.3.3 The authenticatorGetAssertion Operation to sign a Relying Party-provided challenge and other collected data into an assertion, which is used as a credential."). When get() is called by the relying party, "The RP ID defaults to being the caller's origin's effective domain unless the caller has explicitly set options.rpId when calling get()."PROX_MSFT_003098, Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 5.1.4. The RP ID, therefore, is provided by the relying party attempting to authenticate the external authenticator. As an authenticator will only return credentials corresponding to the RP ID access key provided by the external relying party, the authenticator has 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. When a user wishes to use a Microsoft Authenticator passkey stored on their Android or iOS device to access a resource, the user is presented with either a QR code. *See* PROX_MSFT_003575, Sign in with passkeys in Authenticator for Android and iOS devices (preview) - Microsoft Entra ID | Microsoft Learn ("A QR code should appear on screen. Now, on your iOS device, open the camera app and scan the QR code.") and ("A QR code should appear on screen. Now, on your Android device, open the system camera app and scan the QR code. Alternatively, you can also use the camera in Authenticator. Navigate to the passkey account tile and tap on it. Under **Passkey details (preview)**, you'll see a button in the bottom-right corner to scan the QR code."). The user scans a QR code displayed on the screen of a device of the device on which they want to sign in to Microsoft Entra ID.

		Proximity between the devices is then verified using Bluetooth, and the Bluetooth connection is also utilized to establish a secure connection between the devices via the internet. Utilizing the internet connection, the Microsoft Authenticator app on the Android or iOS device sends one-time passkey signature, including a device-bound credential identifier, to the device on which the user is attempting to sign in to Microsoft Entra ID Establishing the connection over the internet would require utilizing either the device's Wi-Fi or cellular capabilities – both of which are wireless protocols. Accordingly, tablets and devices implementing the Android and iOS and running the Microsoft Authenticator app 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, Android iOS devices running the Microsoft Authenticator app include an integrated PDK. *See also:* the hybrid device including … the integrated RDC Android and iOS devices running the Microsoft Authenticator app must include an RDC to communicate passkey signatures over an encrypted connection, through the internet, via Wi-Fi, and/or cellular protocols.

	receiving a first signal at the integrated RDC via the first wireless link from the external PDK;	During the authentication process discussed *supra*, the Microsoft Authenticator App receives an "authenticatorGetAssertion" request to provide cryptographic proof of user authentication. PROX_MSFT_002849, 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."). Browsers, such as Google Chrome or Microsoft Edge, and/or the OS forward the authenticatorGetAssertion to the t Microsoft Authenticator app running on an iOS or Android device. As such, the devices will be connected to forward to the Microsoft Authenticator app the authenticatorGetAssertion request received by the device on which the user is attempting to sign into Microsoft Entra ID. The CTAP specification details a first wireless signal using a hybrid BLE protocol to perform a handshake via a secure tunnel server to establish a secure connection and to prove proximity. https://fidoalliance.org/specs/fido-v2.2-rd-20230321/fido-client-to-authenticator-protocol-v2.2-rd-20230321.html#ble; https://learn.microsoft.com/en-us/windows/security/identity-protection/passkeys/?tabs=windows%2Cintune ("For passkey cross-device authentication scenarios, both the Windows device and the mobile device must have Bluetooth enabled and connected to the Internet."); MSFT_PROXE000198405 at 198504, 198511-198523. "The client platform speaks first to prove

		possession of the BLE advert. The authenticator thus needs only to receive the client platform's handshake message and send a reply in order to complete the handshake. The pre-exchanged public key was passed to the authenticator in the QR code, and the pre-shared symmetric key is derived from the QR secret and decrypted BLE advert. (The full BLE advert is included in the PSK derivation to ensure that any future additions to the advert format are automatically authenticated.) . . . This transport requires a proof of proximity to help prevent attacks, thus notification of the connection attempt comes in the form of a BLE advertisement . . . The connection nonce is the value that demonstrates possession of the BLE advert, and thus proximity to the authenticator. The authenticator needs two values to start communicating on the tunnel: the link ID so that it knows which client platform is contacting it (and thus which keys to use), and a nonce from the client platform."
	generating an enablement signal 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 an external RDC;	generating an enablement signal enabling one or more of an application, a function and a service on … a device associated with an external RDC During a WebAuthn authentication ceremony, an external authenticator (Android or iOS device running the Microsoft Authenticator app) receives an "authenticatorGetAssertion" request to provide cryptographic proof of user authentication. PROX_MSFT_002849, 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."). When successfully invoked, the authenticatorGetAssertion causes the authenticator to return a response including "the credential identifier whose private key was used to generate the assertion." PROX_MSFT_002849, Client to Authenticator Protocol (CTAP) (fidoalliance.org), §6.2.2. As noted above, a credential can only be accessed when an authenticator is applied by the appropriate relying party identifier, such that the relying party identifier is an access

		key. When generating a response, therefore, the device running the Microsoft Authenticator app only retrieve credentials corresponding to the RP ID provided to it by the external FIDO server. PROX_MSFT_002849, 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."). Authentication is a service provided by a FIDO server operated by Microsoft, and the credential ID is necessary for Microsoft's FIDO server to perform the authentication function. Upon receiving the response (i.e., enablement signal), the WebAuthn/FIDO server 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." PROX_MSFT_003098, Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 13.1. As the proper credential ID is needed for Microsoft's FIDO server to authenticate a user, and the credential ID is included within a response to a get request having the appropriate relying party ID received from Microsoft's FIDO server, the response to the authenticatorGetAssertion request generated by the device running the Microsoft Authenticator is an enablement signal enabling authentication by Microsoft's FIDO server. Microsoft's FIDO server is associated with either the OS and/or an instance of the Edge or Chrome browser running on the Windows PC linked to the device running the Microsoft Authenticator app. PROX_MSFT_002819, All about FIDO2, CTAP2, and WebAuthn - Microsoft Community Hub ("CTAP2 and WebAuthn define an abstraction layer that creates an ecosystem for strongly authenticated credentials. Any interoperable client (such as a native app or browser) running on a given 'client device' can use a standardized method to interact with any interoperable authenticator – which could mean a platform authenticator that is built into the client device or a roaming authenticator that is connected to the client device through USB, BLE, or NFC."). The device running the Microsoft Authenticator app, accordingly, generates an enablement signal enabling one or more of an application, a function, and a service on a device associated with an external RDC.

	sending the enablement signal to one or more of the hybrid device and the device associated with an external RDC.	The responsibility for sending responses received from the Microsoft Authenticator app falls upon the WebAuthn Client. PROX_MSFT_003098, Web Authentication: An API for accessing Public Key Credentials - Level 2 (w3.org), § 4 ("A WebAuthn Client is an intermediary entity typically implemented in the user agent (in whole, or in part). Conceptually, it underlies the Web Authentication API and embodies the implementation of the [[Create]](origin, options, sameOriginWithAncestors) and [[DiscoverFromExternalSource]](origin, options, sameOriginWithAncestors) internal methods. It is responsible for both marshalling the inputs for the underlying authenticator operations, and for returning the results of the latter operations to the Web Authentication API's callers."). Microsoft identifies its Edge Browser as the WebAuthn client. PROX_MSFT_002819, All about FIDO2, CTAP2 and WebAuthn – Microsoft Community Hub ("Microsoft Edge plays the part of a WebAuthn Client. Edge can handle the UI for the above listed WebAuthn/CTAP2 features, and also supports the AppID extension. Edge is capable of interacting with both CTAP1 and CTAP2 authenticators, which means that it can facilitate the creation and use of both U2F and FIDO2 credentials…"). As such, Microsoft's Edge Browser will forward the enablement signal received from the Microsoft Authenticator app to Microsoft's FIDO server. Other web browsers, like Chrome, also act as a WebAuthn client and are compatible with Microsoft's UPA (e.g., by forwarding the enablement signal received from a roaming authenticator to Microsoft's FIDO server. See e.g. PROX_MSFT_003062, Passkey support on Android and Chrome | Authentication | Google Developers ("Chrome on all desktop platforms supports using passkeys from mobile devices."); and PROX_MSFT_003062, Passkey support on Android and Chrome | Authentication | Google Developers (Providing a table indicating "Can sign in with phone" is supported on Chrome running on the macOS, iOS, and Windows.).