AUTOFILLING DATA BASED ON ACCOUNT AUTHENTICATION USING A CONTACTLESS CARD

RELATED APPLICATIONS

[0001] This application claims priority to U.S. Patent Application Serial No. 17/530,837, entitled " AUTOFILLING DATA BASED ON ACCOUNT AUTHENTICATION USING A CONTACTLESS CARD " filed on November 19, 2021. The contents of the aforementioned application are incorporated herein by reference in their entirety.

BACKGROUND

[0002] Account identifiers for payment cards may include long numeric and/or character strings. As such, it may be difficult for a user to manually enter the account identifier correctly. Indeed, users often make mistakes and enter incorrect account numbers into payment interfaces on computing devices. Furthermore, processes have been developed that allow cameras or other malicious entities to capture and identify account identifiers entered in a device, thereby posing security risks.

BRIEF SUMMARY

[0003] In one aspect, a method, includes identifying, by a web browser executing on a processor of a device, a payment field in a web page, determining, by the web browser, that an application associated with an issuer of a contactless card is installed on the device, generating, by the web browser, a uniform resource identifier (URI) directed to the application, where the URI includes a merchant identifier parameter, a session identifier parameter, and an action identifier parameter, launching, by a mobile operating system (OS) executing on the processor, the application based on the URI, authenticating, by the application based on the action identifier parameter of the URI, an account associated with the contactless card based on a cryptogram generated by the contactless card, receiving, by the application based on the authentication, payment information associated with the contactless card, accessing, by a WebView component of the application, the web page, autofilling, by the WebView component of the application, the payment information into the payment field in the web page, and processing, by the WebView component of the application, a transaction based on the payment information autofilled into the payment field of the web page. Other embodiments are described and claimed.

BRIEF DESCRIPTION OF THE SEVERAL VIEWS OF THE DRAWINGS

[0004] To easily identify the discussion of any particular element or act, the most significant digit or digits in a reference number refer to the figure number in which that element is first introduced.

[0005] FIG. 1A illustrates an aspect of the subject matter in accordance with one embodiment.

[0006] FIG. 1B illustrates an aspect of the subject matter in accordance with one embodiment.

[0007] FIG. 1C illustrates an aspect of the subject matter in accordance with one embodiment.

[0008] FIG. 1D illustrates an aspect of the subject matter in accordance with one embodiment.

[0009] FIG. 1E illustrates an aspect of the subject matter in accordance with one embodiment.

[0010] FIG. 2A illustrates an aspect of the subject matter in accordance with one embodiment.

[0011] FIG. 2B illustrates an aspect of the subject matter in accordance with one embodiment.

[0012] FIG. 2C illustrates an aspect of the subject matter in accordance with one embodiment.

[0013] FIG. 2D illustrates an aspect of the subject matter in accordance with one embodiment.

[0014] FIG. 2E illustrates an aspect of the subject matter in accordance with one embodiment.

[0015] FIG. 3A illustrates an aspect of the subject matter in accordance with one embodiment.

[0016] FIG. 3B illustrates an aspect of the subject matter in accordance with one embodiment.

[0017] FIG. 3C illustrates an aspect of the subject matter in accordance with one embodiment.

[0018] FIG. 3D illustrates an aspect of the subject matter in accordance with one embodiment.

[0019] FIG. 4 illustrates a routine 400 in accordance with one embodiment.

[0020] FIG. 5 illustrates a routine 500 in accordance with one embodiment.

[0021] FIG. 6A illustrates a contactless card in accordance with one embodiment.

[0022] FIG. 6B illustrates a contactless card 104 in accordance with one embodiment.

[0023] FIG. 7 illustrates a data structure 700 in accordance with one embodiment.

[0024] FIG. 8 illustrates a computer architecture 800 in accordance with one embodiment.

DETAILED DESCRIPTION

[0025] Embodiments disclosed herein provide techniques to securely autofill data in a web browser using a contactless card. Generally, a web browser may load a web page that includes one or more payment form fields. The browser may detect the payment form fields and determine whether an application is installed on the device, where the application is associated with an issuer of the contactless card (e.g., an account management application provided by a financial institution associated with the contactless card). If the browser determines that the application is installed, the browser may generate a uniform resource identifier (URI) that is directed to the application. The browser may include, as parameters of the URI, a merchant identifier (ID) parameter, a session ID parameter, and an action ID parameter.

[0026] An operating system on the device may process the URI to launch the account application. The application may process the parameters of the URI and determine to output, based on the action identifier, an account authentication page of the application. The authentication page may include one or more functions to authenticate an account, such as via login/password, biometrics, or one-tap authentication based on a cryptogram generated by the contactless card based on tapping the card to the device.

[0027] In at least one embodiment, once the account is authenticated, the application may reload the web page in a WebView component of the application. By using the merchant ID and the session ID of the URI, the WebView component receives and restores the user's browsing session that was initiated in the web browser. The account application may receive (e.g., from a server and/or the contactless card) or otherwise store payment information associated with the contactless card (e.g., an account number, expiration date, and card verification value (CVV)). The payment information may be autofilled into the payment

form in the WebView component. Once autofilled, the purchase may be completed by submitting the form in the WebView component.

[0028] In at least one embodiment, once the account is authenticated, the application may initiate a local server on the device, where the local server is only accessible to applications executing on the same device as the local server. A connection between the local server and the web browser may be established. The web browser may receive the payment information from the local server, and autofill the payment information into the form fields in the web browser. Once autofilled, the purchase may be completed by submitting the form in the web browser.

[0029] Advantageously, embodiments disclosed herein provide secure autofilling of data in web browsers. By leveraging cryptograms generated by contactless cards, embodiments of the disclosure may securely verify the identity of the user with minimal risk of fraudulent activity. Furthermore, doing so ensures that autofill operations are only performed when the user has access to a contactless card that facilitates the cryptogram verification with the server. Furthermore, by leveraging the WebView component or the local server, certain restrictions imposed on the web browser may be avoided. For example, some operating systems and/or web browsers may not allow the web browser to directly communicate with the account application. Therefore, by using the WebView component and/or the local server, these restrictions may be overcome, allowing users to securely autofill payment information for a purchase. Furthermore, by providing the disclosed autofill functionality in web browsers, many different web sites can leverage the autofilling without requiring integration into every web site or application.

[0030] With general reference to notations and nomenclature used herein, one or more portions of the detailed description which follows may be presented in terms of program procedures executed on a computer or network of computers. These procedural descriptions and representations are used by those skilled in the art to most effectively convey the substances of their work to others skilled in the art. A procedure is here, and generally, conceived to be a self-consistent sequence of operations leading to a desired result. These operations are those requiring physical manipulations of physical quantities. Usually, though not necessarily, these quantities take the form of electrical, magnetic, or optical signals capable of being stored, transferred, combined, compared, and otherwise manipulated. It proves convenient at times, principally for reasons of common usage, to refer to these signals as bits, values, elements, symbols, characters, terms, numbers, or the like. It should be noted, however, that all of these and similar terms are to be associated

with the appropriate physical quantities and are merely convenient labels applied to those quantities.

[0031] Further, these manipulations are often referred to in terms, such as adding or comparing, which are commonly associated with mental operations performed by a human operator. However, no such capability of a human operator is necessary, or desirable in most cases, in any of the operations described herein that form part of one or more embodiments. Rather, these operations are machine operations. Useful machines for performing operations of various embodiments include digital computers as selectively activated or configured by a computer program stored within that is written in accordance with the teachings herein, and/or include apparatus specially constructed for the required purpose or a digital computer. Various embodiments also relate to apparatus or systems for performing these operations. These apparatuses may be specially constructed for the required purpose. The required structure for a variety of these machines will be apparent from the description given.

[0032] Reference is now made to the drawings, wherein like reference numerals are used to refer to like elements throughout. In the following description, for the purpose of explanation, numerous specific details are set forth in order to provide a thorough understanding thereof. It may be evident, however, that the novel embodiments can be practiced without these specific details. In other instances, well-known structures and devices are shown in block diagram form in order to facilitate a description thereof. The intention is to cover all modification, equivalents, and alternatives within the scope of the claims.

[0033] Figure 1A depicts an exemplary computing architecture 100, also referred to as a system, consistent with disclosed embodiments. Although the computing architecture 100 shown in Figures 1A-1E has a limited number of elements in a certain topology, it may be appreciated that the computing architecture 100 may include more or less elements in alternate topologies as desired for a given implementation.

[0034] The computing architecture 100 comprises one or more computing devices 102, one or more authentication servers 106, one or more contactless cards 104, and one or more merchant servers 108. The contactless card 104 is representative of any type of card, such as a credit card, debit card, ATM card, gift card, payment card, smart card, and the like. The contactless card 104 may comprise one or more communications interfaces 126, such as a radio frequency identification (RFID) chip, configured to communicate with a communications interface 126 (also referred to herein as a “card reader”, a “wireless card

reader", and/or a "wireless communications interface") of the computing devices 102 via NFC, the EMV standard, or other short-range protocols in wireless communication. Although NFC is used as an example communications protocol herein, the disclosure is equally applicable to other types of wireless communications, such as the EMV standard, Bluetooth, and/or Wi-Fi.

[0035] The computing device 102 is representative of any number and type of computing device, such as smartphones, tablet computers, wearable devices, laptops, portable gaming devices, virtualized computing systems, merchant terminals, point-of-sale systems, servers, desktop computers, and the like. A mobile device may be used as an example of the computing device 102, but should not be considered limiting of the disclosure. The authentication server 106 and merchant server 108 are representative of any type of computing device, such as a server, workstation, compute cluster, cloud computing platform, virtualized computing system, and the like. Although not depicted for the sake of clarity, the computing device 102, contactless card 104, authentication server 106, and merchant server 108 each include one or more processor circuits, e.g. to execute programs, code, and/or instructions.

[0036] As shown, a memory 110 of the contactless card 104 includes an applet 112, a counter 114, a master key 116, a diversified key 118, and a unique customer identifier (ID) 120. The applet 112 is executable code configured to perform the operations described herein. The counter 114, master key 116, diversified key 118, and customer ID 120 are used to provide security in the system 100 as described in greater detail below.

[0037] As shown, a memory 128 of the authentication server 106 includes an authentication application 130 and an account database 132. The account database 132 generally includes information related to an account holder (e.g., one or more users), one or more accounts of the account holder, and one or more contactless cards 104 of the account. For each contactless card associated with a financial institution associated with the authentication server 106, the authentication server 106 may store corresponding instances of the master key 116 and counter 114.

[0038] As shown, a memory 136 of the computing device 102 includes an instance of an operating system 138. Example operating systems include the Android® OS, iOS®, macOS®, Linux®, and Windows® operating systems. As shown, the operating system 138 includes an account application 140 and a web browser 142. The account application 140 allows users to perform various account-related operations, such as activating payment cards, viewing account balances, purchasing items, processing payments, and the like. In

some embodiments, a user may authenticate using authentication credentials to access certain features of the account application 140. For example, the authentication credentials may include a username (or login) and password, biometric credentials (e.g., fingerprints, Face ID, etc.), and the like. The web browser 142 is an application that allows the computing device 102 to access information via the network 148 (e.g., via the Internet). For example, using the web browser 142, the user may access one or more resources of the merchant server 108, such as the web page 146 stored in the memory 144 of the merchant server 108, which may be one of a plurality of web pages hosted by the merchant server 108 (or another hosting entity).

[0039] When accessing web pages or other content provided by merchant server 108, a user may select one or more products, services, or other items for purchase via the web browser 142. For example, the user may wish to purchase a basketball and a soccer ball, and may add these items to their shopping cart. To complete the purchase, the web page 146 may include a form with one or more payment fields. The payment fields may include fields for an account number, expiration date, CVV, customer name, and customer billing address. However, certain restrictions may prevent data from being autofilled into these payment fields. For example, the OS and/or web browser 142 may restrict the account application 140 from providing payment data to be autofilled into the form. Advantageously, however, embodiments disclosed herein provide solutions to autofill payment information into the form fields of the form.

[0040] To begin the autofill process, the web browser 142 may identify the one or more payment fields in the form of the web page 146, e.g., based on metadata of the form fields, receiving selection of one of the form fields, etc. The web browser 142 and/or the web page 146 may then determine whether the account application 140 is installed on the computing device 102. The web browser 142 and/or the web page 146 may use any feasible technique to determine whether the account application 140 is installed. For example, in iOS, the web browser 142 and/or the web page 146 may use the canOpenURL() method to determine whether a URI directed to the account application 140 may be opened. The method may generally return an indication of whether or not the URI can be opened. Doing so allows the web browser 142 and/or the web page 146 to determine that the account application 140 is installed on the computing device 102.

[0041] In some operating systems, such as the Android OS, the web browser 142 and/or the web page 146 may use a content provider service to determine whether the account application 140 is installed on the device 102. For example, the web browser 142 and/or the

web page 146 may provide a URI directed to the account application 140 to the content provider service, which may return an indication of whether or not the URI can be opened. Doing so allows the web browser 142 and/or the web page 146 to determine that the account application 140 is installed on the computing device 102.

[0042] In some embodiments, additional and/or alternate techniques may be used to determine whether the account application 140 is installed on the device 102 for any type of operating system. For example, some operating systems may require that native code in native applications call certain functions, e.g., the canOpenURL() method in iOS, or the content provider service in Android. In such embodiments, the web browser 142 and/or the web page 146 may execute code (e.g., JavaScript) to start a timer with a timeout threshold (100 milliseconds, 1 second, etc.) with a callback that will redirect to a web page that handles the case when the account application 140 is not installed (e.g., a page that generally indicates the account application 140 is not installed). Then web browser 142 and/or the web page 146 may then try to launch the account application 140. If the account application 140 is successfully launched before the timer elapses, the timer is cancelled, and autofill processing proceeds as discussed herein. If, however, the account application 140 is not launched before the timer elapses, the web browser 142 will be redirected to the timeout page, and the autofill process ends.

[0043] Once the web browser 142 and/or the web page 146 determines the account application 140 is installed, the web browser 142 and/or the web page 146 may generate a URI 152 directed to the account application 140. At least a portion of the URI 152 may be directed to the account application 140 based on the account application 140 being registered with the OS. Examples may include "example://auth" or "www.example.com/auth". Furthermore, the URI 152 may include one or more parameters. The parameters may include a merchant ID parameter, a session ID parameter, and an action ID parameter. The merchant ID parameter may be associated with a specific merchant, such as the merchant associated with the merchant server 108. Therefore, the account application 140 may uniquely identify each of a plurality of merchants using a respective merchant ID parameter of a plurality of merchant ID parameters. Doing so allows the account application 140 to identify addresses of the merchant server 108 associated with the merchant ID and/or identify addresses of any web pages 146 associated with the merchant ID. The session ID parameter may identify the browsing session in the web browser 142 vis a vis the merchant server 108. For example, the session ID parameter may be used to identify a shopping cart, items in the cart, pages previously visited, a current page displayed in the web browser 142

(e.g., the web page 146), and the like. The action ID may generally specify, to the account application 140, an action or operation to be performed. For example, in some embodiments, the action ID may instruct the account application 140 to open an authentication page. Therefore, the URI 152 may be a deep link to one or more pages of the account application 140. Examples of the URI 152 may include "example://auth?merchID=123&sessID=ABC&actID=456" or "www.example.com/?merchID=123&sessID=ABC&actID=456".

[0044] Once generated, the OS may process the URI 152, which causes the OS to open, access, or otherwise display the account application 140. Doing so further provides the URI 152 including the parameters to the account application 140. Based on the action ID parameter of the URI 152, the account application 140 may open an account authentication page to facilitate the autofill techniques described herein.

[0045] However, in some embodiments, the web browser 142 and/or the web page 146 generates and accesses the URI 152 without determining the account application 140 is installed. If the URI 152, when accessed, successfully launches the account application 140, and the autofill processing proceeds as described herein (with or without an explicit determination by the web browser 142 and/or the web page 146 that the account application 140 is installed). Otherwise, the account application 140 is not installed, and the web browser 142 is redirected to the timeout page indicating the account application 140 is not installed.

[0046] Figure 1B depicts an embodiment where the OS has accessed the URI 152 and the account application 140 has loaded the account authentication page. Generally, the account authentication page allows users to authenticate their account, e.g., via a login and password, biometrics, or a one-tap authentication based on a cryptogram 122 generated by the contactless card 104.

[0047] In the embodiment depicted in Figure 1B, the user may tap the contactless card 104 to the computing device 102 (or otherwise bring the contactless card 104 within communications range of the communications interface 126 of the device 102). The applet 112 of the contactless card 104 may then generate a cryptogram 122.

[0048] The cryptogram 122 may be based on the customer ID 120 of the contactless card 104. The cryptogram 122 may be generated based on any suitable cryptographic technique. In some embodiments, the applet 112 may include an unencrypted identifier (e.g., the customer ID 120, an identifier of the contactless card 104, and/or any other unique

identifier) as part of a data package including the cryptogram 122. In at least one embodiment, the data package is an NDEF file.

[0049] As stated, the computing architecture 100 is configured to implement key diversification to secure data, which may be referred to as a key diversification technique herein. Generally, the authentication server 106 (or another computing device) and the contactless card 104 may be provisioned with the same master key 116 (also referred to as a master symmetric key). More specifically, each contactless card 104 is programmed with a distinct master key 116 that has a corresponding pair in the authentication server 106. For example, when a contactless card 104 is manufactured, a unique master key 116 may be programmed into the memory 110 of the contactless card 104. Similarly, the unique master key 116 may be stored in a record of a customer associated with the contactless card 104 in the account database 132 of the authentication server 106 (and/or stored in a different secure location, such as the hardware security module (HSM) 134). The master key 116 may be kept secret from all parties other than the contactless card 104 and authentication server 106, thereby enhancing security of the system 100. In some embodiments, the applet 112 of the contactless card 104 may encrypt and/or decrypt data (e.g., the customer ID 120) using the master key 116 and the data as input a cryptographic algorithm. For example, encrypting the customer ID 120 with the master key 116 may result in the cryptogram 122. Similarly, the authentication server 106 may encrypt and/or decrypt data associated with the contactless card 104 using the corresponding master key 116.

[0050] In some embodiments, the master keys 116 of the contactless card 104 and authentication server 106 may be used in conjunction with the counters 114 to enhance security using key diversification. The counters 114 comprise values that are synchronized between the contactless card 104 and authentication server 106. For example, the counters 114 may comprise a number that changes each time data is exchanged between the contactless card 104 and the authentication server 106 (and/or the contactless card 104 and the computing device 102). Generally, the applet 112 may provide the master key 116, unique customer ID 120, and a diversification factor as input to a cryptographic algorithm, thereby producing a diversified key 118. In some embodiments, the diversification factor is the counter 114. The diversified key 118 may then be used to encrypt some data, such as the diversification factor (e.g., the counter 114) or other sensitive data. The applet 112 and the authentication server 106 may be configured to encrypt the same type of data to facilitate the decryption and/or verification processing of the cryptogram 122.

[0051] More generally, when preparing to send data (e.g., to the authentication server 106 and/or the computing device 102), the applet 112 of the contactless card 104 may increment the counter 114. The applet 112 of the contactless card 104 may then provide the master keys 116, customer ID 120, and counter 114 as input to a cryptographic algorithm, which produces a diversified key 118 as output. The cryptographic algorithm may include encryption algorithms, hash-based message authentication code (HMAC) algorithms, cipher-based message authentication code (CMAC) algorithms, and the like. Non-limiting examples of the cryptographic algorithm may include a symmetric encryption algorithm such as 3DES or AES107; a symmetric HMAC algorithm, such as HMAC-SHA-256; and a symmetric CMAC algorithm such as AES-CMAC. Examples of key diversification techniques are described in greater detail in United States Patent Application 16/205,119, filed November 29, 2018. The aforementioned patent application is incorporated by reference herein in its entirety.

[0052] The applet 112 may then encrypt some data (e.g., the unique customer ID 120, the counter 114, a command, and/or any other data) using the diversified key 118 and the data as input to the cryptographic algorithm. For example, encrypting the unique customer ID 120 the diversified key 118 may result in an encrypted unique customer ID 120(e.g., a cryptogram 122).

[0053] In some embodiments, two diversified keys 118 may be generated, e.g., based on one or more portions of the input to the cryptographic function. In some embodiments, the two diversified keys 118 are generated based on two distinct master keys 116, the unique customer ID 120, and the counter 114. In such embodiments, a message authentication code (MAC) is generated using one of the diversified keys 118, and the MAC may be encrypted using the other one of the diversified keys 118. The MAC may be generated based on any suitable data input to a MAC algorithm, such as sensitive data, the unique customer ID 120, the counter 114, etc. More generally, the applet 112 and the authentication server 106 may be configured to generate the MAC based on the same data. In some embodiments, the cryptogram 122 is included in a data package such as an NDEF file. The account application 140 may then read the data package including cryptogram 122 via the communications interface 126 of the computing device 102.

[0054] Figure 1C depicts an embodiment where the account application 140 transmits the data package including the cryptogram 122 to the authentication server 106. The authentication server 106 may provide the cryptogram 122 to the authentication application 130 and/or the HSM 134 for verification based at least in part on the instance of the master

key 116 stored by the authentication server 106. In some embodiments, the authentication application 130 and/or the HSM 134 may identify the master key 116 and counter 114 using the unencrypted customer ID 120 provided to the server 106 with the cryptogram 122. In some examples, the authentication application 130 may provide the master key 116, unique customer ID 120, and counter 114 as input to the cryptographic algorithm, which produces one or more diversified keys 118 as output. The resulting diversified keys 118 may correspond to the diversified keys 118 of the contactless card 104, which may be used to decrypt the cryptogram 122 and/or verify the MAC once decrypted. For example, the authentication server 106 may generate a MAC based on the same data as the applet 112, e.g., the sensitive data, the unique customer ID 120, and/or the counter 116. If the MAC generated by the authentication server 106 matches the decrypted MAC in the cryptogram 122, the authentication server 106 may verify or otherwise authenticate the cryptogram 122.

[0055] Regardless of the verification technique used, the authentication application 130 and/or the HSM 134 may successfully decrypt the cryptogram 122 and verify the MAC, thereby verifying or authenticating the cryptogram 122.

[0056] If the decryption is successful, the authentication application 130 may transmit a decryption result 150 to the account application 140. In some embodiments, the decryption result 150 may include data to be autofilled into the payment forms to pay for a purchase with the merchant server 108. However, if the authentication application 130 is unable to decrypt the cryptogram 122 to yield the expected result (e.g., the customer ID 120 of the account associated with the contactless card 104), the authentication application 130 does not validate the cryptogram 122. In such an example, the authentication application 130 determines to terminate the autofill process. The authentication application 130 may transmit an indication of the failed decryption to the computing device 102.

[0057] Figure 1D depicts an embodiment where the authentication application 130 transmits a decryption result 150 to the account application 140. The decryption result 150 generally reflects whether or not the cryptogram 122 was decrypted. In the example depicted in Figure 1D, the decryption result 150 may indicate the authentication server 106 decrypted the cryptogram 122. Doing so may allow the account application 140 to determine that the cryptogram 122 was successfully decrypted prior to continuing the autofill process, thereby improving security.

[0058] As shown, the authentication server 106 may further transmit autofill data 124 to the account application 140. Although depicted as being transmitted with the decryption result 150, the autofill data 124 may be transmitted separate from the decryption result 150.

Furthermore, in some embodiments, the autofill data 124 may be received from the server 106 responsive to another tap of the contactless card 104 to the computing device 102, which causes another cryptogram to be generated, which is verified by the authentication server 106. In other embodiments, the autofill data 124 is received directly from the contactless card 104 (e.g., via a direct read via the communications interface 126). In other embodiments, the account application 140 retrieves the autofill data 124 in a local database stored on the computing device 102. The autofill data 124 may generally include an account number of the contactless card 104, an expiration date of the contactless card 104, a CVV of the contactless card 104, a customer name associated with the contactless card 104, and customer billing address associated with the contactless card 104. In some embodiments, the account number may be a one-time use virtual account number associated with the contactless card 104.

[0059] In some embodiments, the authentication server 106 may generate a payment token as the autofill data 124. In such embodiments, the authentication server 106 may provide the payment token to the merchant server 108 and/or the computing device 102.

[0060] As shown, responsive to receiving the decryption result 150, the account application 140 may launch a WebView component 154. The WebView component 154 is generally configured to access and display web content, such as the web page 146, but lacks some features of the web browser 142. Stated differently, the WebView component 154 is an in-app web browser of the account application 140 that is distinct from the web browser 142. Generally, the account application 140 may launch the WebView component 154 within the account application 140 to load web-based content within the account application 140.

[0061] The account application 140 may cause the WebView component 154 to receive and restore the user's browsing session from the web browser 142 with the merchant server 108. For example, the account application 140 may use the merchant ID and session ID of the URI 152 to generate a URL (and/or URI) directed to the web page 146. The URL directed to the web page 146 may include the parameters to allow the web page 146 to be loaded in the WebView component 154 while maintaining the browsing session from the web browser 142. For example, the merchant ID may be associated with a base URL (e.g., www.example.com) and the session ID may be included as a parameter of the base URL to create a URL directed to the web page 146 and/or the merchant server 108. Examples may include “www.example.com/page.html?sessID=ABC.” Once the URL generated by the account application 140 is accessed by the WebView component 154, the WebView

component 154 may load the web page 146 to replicate the user's browsing session from the web browser 142. For example, the web page 146 including a payment form may be rendered in the WebView component 154 allowing the user to purchase one or more items the user previously added to their shopping cart in the web browser 142. Continuing with the above example, the WebView component 154 may load the web page 146 which reflects the user's shopping cart, which includes a basketball and a soccer ball.

[0062] Advantageously, the WebView component 154 may autofill the autofill data 124 into the one or more payment fields of the web page 146. In some embodiments, the WebView component 154 uses an autofill service provided by the account application 140. In some embodiments, the WebView component 154 uses an autofill service provided by the operating system 138. Regardless of the autofill technique used, the user may submit the form including the autofilled data to complete the purchase.

[0063] In embodiments where the authentication server 106 generates a payment token, the web page 146 loaded by the WebView component 154 may include the payment token. The payment token may be provided by the merchant server 108 to the WebView component 154, or by the account application 140 to the WebView component 154. The payment token may be used to process the purchase.

[0064] Figure 1E depicts an embodiment where the WebView component 154 generates a transaction package 156 to process a payment using the autofill data 124 filled into the form fields of the web page 146. Generally, the transaction package 156 may be transmitted according to the hypertext transfer protocol (HTTP). Once received, the merchant server 108 may process payment for the transaction using the autofill data 124. The merchant server 108 may then create a transaction record 160 for the transaction in a transaction database 158. As stated, however, in some embodiments, the payment token is used instead of the autofill data 124 to pay for the transaction.

[0065] Figure 2A depicts an example schematic 200 for using a local server to autofill data into the web browser 142, according to various embodiments. Although the computing architecture 200 shown in Figures 2A-2E has a limited number of elements in a certain topology, it may be appreciated that the computing architecture 200 may include more or less elements in alternate topologies as desired for a given implementation.

[0066] As shown in Figure 2A, the user may use the web browser 142 to select one or more items for purchase via the merchant server 108. For example, the user may wish to purchase an orange and an apple, and have these items in their shopping cart. To complete the purchase, the web page 146 may include a form with one or more payment fields. The

payment fields may include fields for an account number, expiration date, CVV, customer name, and customer billing address. As stated, however, certain restrictions may prevent data from being autofilled into these payment fields. For example, the OS and/or web browser 142 may restrict the account application 140 from providing payment data to be autofilled into the form. Advantageously, however, the account application 140 may use a local server to autofill payment information into the form fields of the form of the web page 146.

[0067] To begin the autofill process, the web browser 142 may identify the one or more payment fields in the form of the web page 146. The web browser 142 and/or the web page 146 may then determine whether the account application 140 is installed on the computing device 102. The web browser 142 and/or the web page 146 may use any feasible technique to determine whether the account application 140 is installed. For example, in some operating systems such as iOS, the web browser 142 and/or the web page 146 may use the canOpenURL() method to determine whether a URI directed to the account application 140 may be opened. The method may generally return an indication of whether or not the URI can be opened. Doing so allows the web browser 142 and/or the web page 146 to determine that the account application 140 is installed on the computing device 102. Furthermore

[0068] In some operating systems, such as the Android OS, the web browser 142 and/or the web page 146 may use the content provider service to determine whether the account application 140 is installed on the device. For example, the web browser 142 and/or the web page 146 may provide a URI directed to the account application 140 to the content provider service, which may return an indication of whether or not the URI can be opened. Doing so allows the web browser 142 and/or the web page 146 to determine that the account application 140 is installed on the computing device 102.

[0069] Once the web browser 142 and/or the web page 146 determines the account application 140 is installed, the web browser 142 and/or the web page 146 may generate a URI 202 directed to the account application 140. The URI 202 may be generated based on the same techniques to generate to the URI 152. At least a portion of the URI 202 may be directed to the account application 140 based on the account application 140 being registered with the OS. Furthermore, the URI 202 may include one or more parameters. The parameters may include a merchant ID parameter, a session ID parameter, and an action ID parameter. As stated, the merchant ID parameter may be associated with a specific merchant, such as the merchant associated with the merchant server 108. Therefore, as stated, each of a plurality of merchants may be uniquely identified by a respective merchant

ID parameter. Doing so allows the account application 140 to identify the merchant server 108 associated with the merchant ID and/or identify any web pages 146 associated with the merchant ID. The session ID parameter may identify the browsing session in the web browser 142 vis a vis the merchant server 108. For example, the session ID parameter may be used to identify a shopping cart, items in the shopping cart, pages previously visited, a current page displayed in the web browser 142 (e.g., the web page 146), and the like. The action ID may generally specify, to the account application 140, an action or operation to be performed. For example, in some embodiments, the action ID may instruct the account application 140 to open an authentication page. Therefore, the URI 202 may be a deep link to one or more pages of the account application 140.

[0070] Once generated, the OS may process the URI 202, which causes the OS to open, access, or otherwise display the account application 140. Doing so further provides the URI 202 including the parameters to the account application 140. Based on the action ID parameter of the URI 202, the account application 140 may open an account authentication page to facilitate the autofill techniques described herein.

[0071] As stated, in some embodiments, the web browser 142 and/or the web page 146 may use the timer and attempt to launch the URI 202 without predetermining that the account application 140 is installed. In such embodiments, if the account application 140 is successfully launched based on the URI 202, the autofill processing proceeds as described herein. If, however, the time expires without the account application 140 launching on the device, the autofill processing ends.

[0072] Figure 2B depicts an embodiment where the OS has accessed the URI 202 and the account application 140 has loaded the account authentication page. Generally, the account authentication page allows users to authenticate their account, e.g., via a login and password, biometrics, or a one-tap authentication based on a cryptogram 122 generated by the contactless card 104.

[0073] In the embodiment depicted in Figure 2B, the user may tap the contactless card 104 to the computing device 102 (or otherwise bring the contactless card 104 within communications range of the communications interface 126 of the device 102). The applet 112 of the contactless card 104 may then generate another cryptogram 122 as described above. The account application 140 may then read the another cryptogram 122 via the communications interface 126.

[0074] Figure 2C depicts an embodiment where the account application 140 transmits the data package including the another cryptogram 122 to the authentication server 106. The

authentication server 106 may verify or otherwise authenticate the cryptogram 122 as described above.

[0075] Based on the successful decryption, the authentication application 130 may transmit another decryption result to the account application 140. As stated, the decryption result may include data to be autofilled into the payment forms to pay for a purchase with the merchant server 108.

[0076] Figure 2D depicts an embodiment where the authentication application 130 transmits a decryption result 204 to the account application 140. In the example depicted in Figure 2D, 1the decryption result 204 may indicate the authentication server 106 decrypted the another cryptogram 122 of Figures 2B-2C. Doing so may allow the account application 140 to determine that the another cryptogram 122 was successfully decrypted prior to continuing the autofill process, thereby improving security.

[0077] As shown, the authentication server 106 may further transmit the autofill data 124 to the account application 140. Although depicted as being transmitted with the decryption result 204, the autofill data 124 may be transmitted separate from the decryption result 204. Furthermore, in some embodiments, the autofill data 124 may be transmitted responsive to another tap of the contactless card 104 to the computing device 102, which causes another cryptogram to be generated and verified by the authentication server 106. In other embodiments, the autofill data 124 is received directly from the contactless card 104. In other embodiments, the account application 140 retrieves the autofill data 124 in a local database stored on the computing device 102. The autofill data 124 may generally include an account number of the contactless card 104, an expiration date of the contactless card 104, a CVV of the contactless card 104, a customer name associated with the contactless card 104, and customer billing address associated with the contactless card 104. In some embodiments, the account number may be a one-time use virtual account number associated with the contactless card 104.

[0078] As shown, responsive to receiving the decryption result 204, the account application 140 may launch a local server 210. The local server 210 may be any type of server, such as a TCP/IP server, HTTP server, Hypertext Transfer Protocol Secure (HTTPS) server, a streaming server, and the like. However, only local applications (e.g., applications executing on the computing device 102) may access the local server 210. The OS may restrict attempts to access the local server 210 from external sources (e.g., via the network 148). The account application 140 may initiate the local server 210 on a specific port number. The account application 140 may select the port according to any feasible selection

scheme, such as randomly generating port numbers, using a predetermined port number, and the like. The use of a local server to exchange data is described in greater detail in United States Patent Application 16/876,473, filed May 18, 2020, and Application 16/876,549, filed May 18, 2020. The aforementioned patent applications are incorporated by reference herein in their entirety.

[0079] The account application 140 may cause the web browser 142 establish a connection with the local server 210. In some embodiments, the account application 140 may generate a URI directed to the web browser 142, and include the relevant parameters needed to establish the connection with the local server 210 (e.g., IP address, username, password, port, etc.). The OS may access the URI, which launches the web browser 142 and provides the parameters to the web browser 142. The web browser 142, which still maintains the browsing session with the merchant server 108, may then connect to the local server 210, and receive the autofill data 124 from the local server 210.

[0080] Advantageously, once received from the local server 210, the web browser 142 may autofill the autofill data 124 into the one or more payment fields of the web page 146, e.g., to pay for the orange and apple. In some embodiments, the web browser 142 uses JavaScript® to autofill the autofill data 124 to the form fields, e.g., an autofill service provided by the web browser 142. In some embodiments, the web browser 142 uses an autofill service provided by the operating system 138 to autofill the autofill data 124 to the form fields. Regardless of the autofill service used, the user may submit the form including the autofilled data to complete the purchase via the web browser 142

[0081] In some embodiments, the authentication server 106 may generate a payment token as the autofill data 124. In such embodiments, the authentication server 106 may provide the payment token to the merchant server 108 and/or the computing device 102. In such embodiments, the account application 140 may receive the payment token from the authentication server 106 or the merchant server 108. The payment token may be provided by the account application 140 to the local server 210, which provides the token to the web browser 142. The payment token may be used to process the purchase via the web browser 142.

[0082] In some embodiments, the account application 140 may provide the autofill data 124 to the content provider service of the operating system 138 rather than initiating the local server 210. In such embodiments, the web browser 142 and/or the web page 146 may receive the autofill data 124 from the content provider service. The web browser 142 may

then autofill the autofill data 124 into the form fields using the autofill service of the web browser 142 and/or the autofill service of the operating system 138.

[0083] Figure 2E depicts an embodiment where the web browser 142 generates a transaction package 206 to process a payment for the apple and the orange using the autofill data 124 filled into the form fields of the web page 146. Generally, the transaction package 206 may be transmitted according to the hypertext transfer protocol (HTTP). Once received, the merchant server 108 may process payment for the transaction using the autofill data 124. The merchant server 108 may then create a transaction record 208 for the transaction in a transaction database 158. As stated, however, in some embodiments, the payment token is used instead of the autofill data 124 to pay for the transaction.

[0084] Figure 3A is a schematic 300 depicting an example embodiment of autofilling data into a web browser using a contactless card. As shown, Figure 3A includes a mobile computing device 102 executing a web browser 142. The web browser 142 may display a web page, such as the web page 146. For example, the web page 146 may be a web page 146that allows a user to place an order and provide payment information for the order. As shown, the web page 146 includes a payment form having fields 301-305, where field 301 is a name field, field 302 is an account number field, field 303 is an expiration date field, field 304 is a CVV field, and field 305 is an address field.

[0085] The web page 146 further includes a selectable element 308 that allows the user to initiate the autofill process to autofill payment information into the form fields 301-305. Although discussed with reference to the WebView example, Figs. 3A-3D are equally applicable to the local server example. Embodiments are not limited in this context.

[0086] Figure 3B is a schematic 310 illustrating an embodiment where the user has selected the element 308 to initiate the autofill process. Generally, doing so may cause the web browser 142 to generate a URI directed to the account application 140, which causes the computing device 102 to display the account application 140. Based on the parameters of the URI, the account application 140 outputs an authentication page that requests the user tap their contactless card 104 to the computing device 102 as indicated by the notification 306. Doing so causes the contactless card 104 to generate a cryptogram that is verified by the authentication server 106. As shown, the authentication server 106 verifies the cryptogram in FIG. 3B. Doing so may cause the authentication server 106 to transmit payment information (e.g., account number, expiration date, CVV) to the account application 140. The authentication server 106 may further provide the account holder's name and address if the account application 140 does not have these values stored locally.

[0087] Figure 3C is a schematic 320 depicting an embodiment where the account application 140 receives an indication from the authentication server 106 specifying the server verified the cryptogram generated by the contactless card 104. As stated, the authentication server 106 may include the payment information to autofill in the form fields 301-306. In response, the account application 140 launches the WebView component 154. The WebView component 154 uses a URL generated by the account application 140 to replicate the browsing session of FIG. 3A in the WebView component 154, e.g., based on the merchant ID and session ID of the URI generated in FIG. 3A.

[0088] As shown, the WebView component 154 may autofill the payment information and any personally identifiable information into the form fields. More specifically, the WebView component 154 may autofill the user's name to the name field 301, the account number to the account number field 302, the expiration date to the expiration date field 303, the CVV to the CVV field 304, and the address to the address field 305. The user may then complete the purchase using the button 311, which causes the merchant server 108 to process the payment using the autofilled data.

[0089] Figure 3D is a schematic 330 illustrating a confirmation page in the WebView component 154. The confirmation page generally reflects that the purchase was completed using the data autofilled into the form fields 301-305. The embodiments are not limited in this context.

[0090] Operations for the disclosed embodiments may be further described with reference to the following figures. Some of the figures may include a logic flow. Although such figures presented herein may include a particular logic flow, it can be appreciated that the logic flow merely provides an example of how the general functionality as described herein can be implemented. Further, a given logic flow does not necessarily have to be executed in the order presented unless otherwise indicated. Moreover, not all acts illustrated in a logic flow may be required in some embodiments. In addition, the given logic flow may be implemented by a hardware element, a software element executed by a processor, or any combination thereof. The embodiments are not limited in this context.

[0091] Figure 4 illustrates an embodiment of a logic flow 400. The logic flow 400 may be representative of some or all of the operations executed by one or more embodiments described herein. For example, the logic flow 400 may include some or all of the operations to autofill data using the WebView component 154. Embodiments are not limited in this context.

[0092] In block 402, routine 400 identifies, by a web browser 142 executing on a processor of a computing device 102, a payment field in a web page 146. The payment field may be one of multiple payment fields of a form. In block 404, routine 400 determines, by the web browser 142, that an account application 140 associated with an issuer of a contactless card 104 is installed on the computing device 102. The web browser 142 may use one or more functions provided by the operating system 138 to determine that the account application 140 is installed. In block 406, routine 400 generates, by the web browser 142, a uniform resource identifier (URI) directed to the account application 140, wherein the URI comprises a merchant identifier parameter, a session identifier parameter, and an action identifier parameter.

[0093] In block 408, routine 400 launches, by an OS executing on the processor, the account application 140 based on the URI. In block 410, routine 400 authenticates, by the account application 140 based on the action identifier parameter of the URI, an account associated with the contactless card based on a cryptogram generated by the contactless card 104. In block 412, routine 400 receives, by the account application 140 based on the authentication, payment information associated with the contactless card 104. The payment information may include an account number, expiration date, and a CVV of the contactless card 104. The payment information may further include an account holder name and/or address. In block 414, routine 400 accesses, by a WebView component 154 of the account application 140, the web page 146. The merchant server 108 may then transmit all data associated with the user's browsing session and the web page 146 to the WebView component 154. Doing so replicates or otherwise restores the user's browsing session from the web browser 142 in the WebView component 154. In block 416, the WebView component 154 autofills the payment information into the payment field in the web page 146. If multiple fields are present, the WebView component 154 may autofill the information into the appropriate fields. In block 418, routine 400 processes, by the WebView component 154, a transaction based on the payment information autofilled into the payment field of the web page.

[0094] Figure 5 illustrates an embodiment of a logic flow 500. The logic flow 500 may be representative of some or all of the operations executed by one or more embodiments described herein. For example, the logic flow 500 may include some or all of the operations to autofill data using the local server 210. Embodiments are not limited in this context.

[0095] In block 502, routine 500 identifies, by a web browser 142 executing on a processor of a device, a payment field in a web page 146. The payment field may be one of multiple

payment fields of a form. In block 504, routine 500 determines, by the web browser 142, that an account application 140 associated with an issuer of a contactless card 104 is installed on the device. The web browser 142 may use one or more functions provided by the operating system 138 to determine that the account application 140 is installed. In block 506, routine 500 generates, by the web browser 142, a uniform resource identifier (URI) directed to the account application 140, wherein the URI comprises a merchant identifier parameter, a session identifier parameter, and an action identifier parameter.

[0096] In block 508, routine 500 launches, by a mobile operating system (OS) executing on the processor, the account application 140 based on the URI. In block 510, routine 500 authenticates, by the account application 140 based on the action identifier parameter of the URI, an account associated with the contactless card 104 based on a cryptogram generated by the contactless card. In block 512, routine 500 receives, by the account application 140 based on the authentication, payment information associated with the contactless card 104. In block 514, routine 500 initiates, by the account application 140, a local server 210 accessible only to applications executing on the computing device 102.

[0097] In block 516, routine 500 establishes a connection between the local server 210 and the web browser 142. In block 518, routine 500 receives, by the web browser 142, the payment information from the local server 210. In block 520, routine 500 terminates, by the web browser 142, the connection with the local server by issuing a termination command to the local server 210. In block 522, routine 500 autofills, by the web browser 142, the payment information into the payment field. If multiple fields are present, the web browser 142 may autofill the information into the appropriate fields. In block 524, routine 500 processes, by the web browser 142, a transaction based on the payment information autofilled into the payment field of the web page 146.

[0098] FIG. 6A is a schematic 600 illustrating an example configuration of a contactless card 104, which may include a payment card, such as a credit card, debit card, or gift card, issued by a service provider as displayed as service provider indicia 602 on the front or back of the contactless card 104. In some examples, the contactless card 104 is not related to a payment card, and may include, without limitation, an identification card. In some examples, the transaction card may include a dual interface contactless payment card, a rewards card, and so forth. The contactless card 104 may include a substrate 604, which may include a single layer or one or more laminated layers composed of plastics, metals, and other materials. Exemplary substrate materials include polyvinyl chloride, polyvinyl chloride acetate, acrylonitrile butadiene styrene, polycarbonate, polyesters, anodized

titanium, palladium, gold, carbon, paper, and biodegradable materials. In some examples, the contactless card 104 may have physical characteristics compliant with the ID-1 format of the ISO/IEC 7816 standard, and the transaction card may otherwise be compliant with the ISO/IEC 14443 standard. However, it is understood that the contactless card 104 according to the present disclosure may have different characteristics, and the present disclosure does not require a transaction card to be implemented in a payment card.

[0099] The contactless card 104 may also include identification information 606 displayed on the front and/or back of the card, and a contact pad 608. The contact pad 608 may include one or more pads and be configured to establish contact with another client device, such as an ATM, a user device, smartphone, laptop, desktop, or tablet computer via transaction cards. The contact pad may be designed in accordance with one or more standards, such as ISO/IEC 7816 standard, and enable communication in accordance with the EMV protocol. The contactless card 104 may also include processing circuitry, antenna and other components as will be further discussed in FIG. 6B. These components may be located behind the contact pad 608 or elsewhere on the substrate 604, e.g. within a different layer of the substrate 604, and may electrically and physically coupled with the contact pad 608. The contactless card 104 may also include a magnetic strip or tape, which may be located on the back of the card (not shown in FIG. 6A). The contactless card 104 may also include a Near-Field Communication (NFC) device coupled with an antenna capable of communicating via the NFC protocol. Embodiments are not limited in this manner.

[0100] As illustrated in FIG. 6B, the contact pad 608 of contactless card 104 may include processing circuitry 610 for storing, processing, and communicating information, including a processor 612, a memory 110, and one or more communications interface 126. It is understood that the processing circuitry 610 may contain additional components, including processors, memories, error and parity/CRC checkers, data encoders, anticollision algorithms, controllers, command decoders, security primitives and tamper proofing hardware, as necessary to perform the functions described herein.

[0101] The memory 110 may be a read-only memory, write-once read-multiple memory or read/write memory, e.g., RAM, ROM, and EEPROM, and the contactless card 104 may include one or more of these memories. A read-only memory may be factory programmable as read-only or one-time programmable. One-time programmability provides the opportunity to write once then read many times. A write once/read-multiple memory may be programmed at a point in time after the memory chip has left the factory. Once the memory is programmed, it may not be rewritten, but it may be read many times. A read/write

memory may be programmed and re-programed many times after leaving the factory. A read/write memory may also be read many times after leaving the factory. In some instances, the memory 110 may be encrypted memory utilizing an encryption algorithm executed by the processor 612 to encrypted data.

[0102] The memory 110 may be configured to store one or more applet 112, one or more counters 114, a customer ID 120, one or more master keys 116, and one or more diversified keys 118. The one or more applet 112 may comprise one or more software applications configured to execute on one or more contactless cards 104, such as a Java® Card applet. However, it is understood that applet 112 are not limited to Java Card applets, and instead may be any software application operable on contactless cards or other devices having limited memory. The one or more counter 114 may comprise a numeric counter sufficient to store an integer. The customer ID 120 may comprise a unique alphanumeric identifier assigned to a user of the contactless card 104, and the identifier may distinguish the user of the contactless card 104 from other users of other contactless cards 104. In some examples, the customer ID 120 may identify both a customer and an account assigned to that customer and may further identify the contactless card 104 associated with the customer's account.

[0103] The processor 612 and memory elements of the foregoing exemplary embodiments are described with reference to the contact pad 608, but the present disclosure is not limited thereto. It is understood that these elements may be implemented outside of the contact pad 608 or entirely separate from it, or as further elements in addition to processor 612 and memory 110 elements located within the contact pad 608.

[0104] In some examples, the contactless card 104 may comprise one or more antenna(s) 614. The one or more antenna(s) 614 may be placed within the contactless card 104 and around the processing circuitry 610 of the contact pad 608. For example, the one or more antenna(s) 614 may be integral with the processing circuitry 610 and the one or more antenna(s) 614 may be used with an external booster coil. As another example, the one or more antenna(s) 614 may be external to the contact pad 608 and the processing circuitry 610.

[0105] In an embodiment, the coil of contactless card 104 may act as the secondary of an air core transformer. The terminal may communicate with the contactless card 104 by cutting power or amplitude modulation. The contactless card 104 may infer the data transmitted from the terminal using the gaps in the power connection of the contactless card 104, which may be functionally maintained through one or more capacitors. The contactless card 104 may communicate back by switching a load on the coil of the contactless card 104

or load modulation. Load modulation may be detected in the terminal's coil through interference. More generally, using the antenna(s) 614, processor 612, and/or the memory 110, the contactless card 104 provides a communications interface to communicate via NFC, Bluetooth, and/or Wi-Fi communications.

[0106] As explained above, contactless card 104 may be built on a software platform operable on smart cards or other devices having limited memory, such as JavaCard, and one or more or more applications or applets may be securely executed. Applet 112 may be added to contactless cards to provide a one-time password (OTP) for multifactor authentication (MFA) in various mobile application-based use cases. Applet 112 may be configured to respond to one or more requests, such as near field data exchange requests, from a reader, such as a mobile NFC reader (e.g., of a mobile computing device 102 or point-of-sale terminal), and produce an NDEF message that comprises a cryptographically secure OTP encoded as an NDEF text tag. The NDEF message may include the cryptogram 122, and any other data.

[0107] One example of an NDEF OTP is an NDEF short-record layout (SR=1). In such an example, one or more applet 112 may be configured to encode the OTP as an NDEF type 4 well known type text tag. In some examples, NDEF messages may comprise one or more records. The applet 112 may be configured to add one or more static tag records in addition to the OTP record.

[0108] In some examples, the one or more applet 112 may be configured to emulate an RFID tag. The RFID tag may include one or more polymorphic tags. In some examples, each time the tag is read, different cryptographic data is presented that may indicate the authenticity of the contactless card. Based on the one or more applet 112, an NFC read of the tag may be processed, the data may be transmitted to a server, such as a server of a banking system, and the data may be validated at the server.

[0109] In some examples, the contactless card 104 and server may include certain data such that the card may be properly identified. The contactless card 104 may include one or more unique identifiers (not pictured). Each time a read operation takes place, the counter 114 may be configured to increment. In some examples, each time data from the contactless card 104 is read (e.g., by a mobile device), the counter 114 is transmitted to the server for validation and determines whether the counter 114 are equal (as part of the validation) to a counter of the server.

[0110] The one or more counter 114 may be configured to prevent a replay attack. For example, if a cryptogram has been obtained and replayed, that cryptogram is immediately

rejected if the counter 114 has been read or used or otherwise passed over. If the counter 114 has not been used, it may be replayed. In some examples, the counter that is incremented on the contactless card 104 is different from the counter that is incremented for transactions. The contactless card 104 is unable to determine the application transaction counter 114 since there is no communication between applets 112 on the contactless card 104. In some examples, the contactless card 104 may comprise a first applet 440-1, which may be a transaction applet, and a second applet 440-2. Each applet 440-1 and 440-2 may comprise a respective counter 114.

[0111] In some examples, the counter 114 may get out of sync. In some examples, to account for accidental reads that initiate transactions, such as reading at an angle, the counter 114 may increment but the application does not process the counter 114. In some examples, when the mobile device 10 is woken up, NFC may be enabled and the computing device 102 may be configured to read available tags, but no action is taken responsive to the reads.

[0112] To keep the counter 114 in sync, an application, such as a background application, may be executed that would be configured to detect when the computing device 102 wakes up and synchronize with the server of a banking system indicating that a read that occurred due to detection to then move the counter 114 forward. In other examples, Hashed One Time Password may be utilized such that a window of mis-synchronization may be accepted. For example, if within a threshold of 10, the counter 114 may be configured to move forward. But if within a different threshold number, for example within 10 or 1000, a request for performing re-synchronization may be processed which requests via one or more applications that the user tap, gesture, or otherwise indicate one or more times via the user's device. If the counter 114 increases in the appropriate sequence, then it possible to know that the user has done so.

[0113] The key diversification technique described herein with reference to the counter 114, master key, and diversified key, is one example of encryption and/or decryption a key diversification technique. This example key diversification technique should not be considered limiting of the disclosure, as the disclosure is equally applicable to other types of key diversification techniques.

[0114] During the creation process of the contactless card 104, two cryptographic keys may be assigned uniquely per card. The cryptographic keys may comprise symmetric keys which may be used in both encryption and decryption of data. Triple DES (3DES) algorithm may be used by EMV and it is implemented by hardware in the contactless card 104. By

using the key diversification process, one or more keys may be derived from a master key based upon uniquely identifiable information for each entity that requires a key.

[0115] In some examples, to overcome deficiencies of 3DES algorithms, which may be susceptible to vulnerabilities, a session key may be derived (such as a unique key per session) but rather than using the master key, the unique card-derived keys and the counter may be used as diversification data. For example, each time the contactless card 104 is used in operation, a different key may be used for creating the message authentication code (MAC) and for performing the encryption. This results in a triple layer of cryptography. The session keys may be generated by the one or more applets and derived by using the application transaction counter with one or more algorithms (as defined in EMV 4.3 Book 2 A1.3.1 Common Session Key Derivation).

[0116] Further, the increment for each card may be unique, and assigned either by personalization, or algorithmically assigned by some identifying information. For example, odd numbered cards may increment by 2 and even numbered cards may increment by 5. In some examples, the increment may also vary in sequential reads, such that one card may increment in sequence by 1, 3, 5, 2, 2, ... repeating. The specific sequence or algorithmic sequence may be defined at personalization time, or from one or more processes derived from unique identifiers. This can make it harder for a replay attacker to generalize from a small number of card instances.

[0117] The authentication message may be delivered as the content of a text NDEF record in hexadecimal ASCII format. In another example, the NDEF record may be encoded in hexadecimal format.

[0118] FIG. 7 illustrates an NDEF short-record layout (SR=1) data structure 700 according to an example embodiment. One or more applets may be configured to encode the OTP as an NDEF type 4 well known type text tag. In some examples, NDEF messages may comprise one or more records. The applets may be configured to add one or more static tag records in addition to the OTP record. Exemplary tags include, without limitation, Tag type: well known type, text, encoding English (en); Applet ID: D2760000850101; Capabilities: read-only access; Encoding: the authentication message may be encoded as ASCII hex; type-length-value (TLV) data may be provided as a personalization parameter that may be used to generate the NDEF message. In an embodiment, the authentication template may comprise the first record, with a well-known index for providing the actual dynamic authentication data. The data structure 700 may include the cryptogram 122, and any other data provided by the applet 112.

[0119] FIG. 8 illustrates an embodiment of an exemplary computer architecture 800 suitable for implementing various embodiments as previously described. In one embodiment, the computer architecture 800 may include or be implemented as part of computing architecture 100.

[0120] As used in this application, the terms "system" and "component" are intended to refer to a computer-related entity, either hardware, a combination of hardware and software, software, or software in execution, examples of which are provided by the exemplary computing computer architecture 800. For example, a component can be, but is not limited to being, a process running on a processor, a processor, a hard disk drive, multiple storage drives (of optical and/or magnetic storage medium), an object, an executable, a thread of execution, a program, and/or a computer. By way of illustration, both an application running on a server and the server can be a component. One or more components can reside within a process and/or thread of execution, and a component can be localized on one computer and/or distributed between two or more computers. Further, components may be communicatively coupled to each other by various types of communications media to coordinate operations. The coordination may involve the uni-directional or bi-directional exchange of information. For instance, the components may communicate information in the form of signals communicated over the communications media. The information can be implemented as signals allocated to various signal lines. In such allocations, each message is a signal. Further embodiments, however, may alternatively employ data messages. Such data messages may be sent across various connections. Exemplary connections include parallel interfaces, serial interfaces, and bus interfaces.

[0121] The computer architecture 800 includes various common computing elements, such as one or more processors, multi-core processors, co-processors, memory units, chipsets, controllers, peripherals, interfaces, oscillators, timing devices, video cards, audio cards, multimedia input/output (I/O) components, power supplies, and so forth. The embodiments, however, are not limited to implementation by the computing architecture 800.

[0122] As shown in FIG. 8, the computer architecture 800 includes a computer 812 comprising a processor 802, a system memory 804 and a system bus 806. The processor 802 can be any of various commercially available processors. The computer 812 may be representative of the computing device 102 and/or the authentication server 106.

[0123] The system bus 806 provides an interface for system components including, but not limited to, the system memory 804 to the processor 802. The system bus 806 can be any of several types of bus structure that may further interconnect to a memory bus (with or

without a memory controller), a peripheral bus, and a local bus using any of a variety of commercially available bus architectures. Interface adapters may connect to the system bus 806 via slot architecture. Example slot architectures may include without limitation Accelerated Graphics Port (AGP), Card Bus, (Extended) Industry Standard Architecture ((E)ISA), Micro Channel Architecture (MCA), NuBus, Peripheral Component Interconnect (Extended) (PCI(X)), PCI Express, Personal Computer Memory Card International Association (PCMCIA), and the like.

[0124] The computer architecture 800 may include or implement various articles of manufacture. An article of manufacture may include a computer-readable storage medium to store logic. Examples of a computer-readable storage medium may include any tangible media capable of storing electronic data, including volatile memory or non-volatile memory, removable or non-removable memory, erasable or non-erasable memory, writeable or re-writeable memory, and so forth. Examples of logic may include executable computer program instructions implemented using any suitable type of code, such as source code, compiled code, interpreted code, executable code, static code, dynamic code, object-oriented code, visual code, and the like. Embodiments may also be at least partly implemented as instructions contained in or on a non-transitory computer-readable medium, which may be read and executed by one or more processors to enable performance of the operations described herein.

[0125] The system memory 804 may include various types of computer-readable storage media in the form of one or more higher speed memory units, such as read-only memory (ROM), random-access memory (RAM), dynamic RAM (DRAM), Double-Data-Rate DRAM (DDRAM), synchronous DRAM (SDRAM), static RAM (SRAM), programmable ROM (PROM), erasable programmable ROM (EPROM), electrically erasable programmable ROM (EEPROM), flash memory, polymer memory such as ferroelectric polymer memory, ovonic memory, phase change or ferroelectric memory, silicon-oxide-nitride-oxide-silicon (SONOS) memory, magnetic or optical cards, an array of devices such as Redundant Array of Independent Disks (RAID) drives, solid state memory devices (e.g., USB memory, solid state drives (SSD) and any other type of storage media suitable for storing information. In the illustrated embodiment shown in FIG. 8, the system memory 804 can include non-volatile 808 and/or volatile 810. A basic input/output system (BIOS) can be stored in the non-volatile 808.

[0126] The computer 812 may include various types of computer-readable storage media in the form of one or more lower speed memory units, including an internal (or external)

hard disk drive 814, a magnetic disk drive 816 to read from or write to a removable magnetic disk 818, and an optical disk drive 820 to read from or write to a removable optical disk 822 (e.g., a CD-ROM or DVD). The hard disk drive 814, magnetic disk drive 816 and optical disk drive 820 can be connected to system bus 806 the by an HDD interface 824, and FDD interface 826 and an optical disk drive interface 828, respectively. The HDD interface 824 for external drive implementations can include at least one or both of Universal Serial Bus (USB) and IEEE 1394 interface technologies.

[0127] The drives and associated computer-readable media provide volatile and/or nonvolatile storage of data, data structures, computer-executable instructions, and so forth. For example, a number of program modules can be stored in the drives and non-volatile 808, and volatile 810, including an operating system 830, one or more applications 832, other program modules 834, and program data 836. In one embodiment, the one or more applications 832, other program modules 834, and program data 836 can include, for example, the various applications and/or components of the system 100.

[0128] A user can enter commands and information into the computer 812 through one or more wire/wireless input devices, for example, a keyboard 838 and a pointing device, such as a mouse 840. Other input devices may include microphones, infra-red (IR) remote controls, radio-frequency (RF) remote controls, game pads, stylus pens, card readers, dongles, fingerprint readers, gloves, graphics tablets, joysticks, keyboards, retina readers, touch screens (e.g., capacitive, resistive, etc.), trackballs, track pads, sensors, styluses, and the like. These and other input devices are often connected to the processor 802 through an input device interface 842 that is coupled to the system bus 806 but can be connected by other interfaces such as a parallel port, IEEE 1394 serial port, a game port, a USB port, an IR interface, and so forth.

[0129] A monitor 844 or other type of display device is also connected to the system bus 806 via an interface, such as a video adapter 846. The monitor 844 may be internal or external to the computer 812. In addition to the monitor 844, a computer typically includes other peripheral output devices, such as speakers, printers, and so forth.

[0130] The computer 812 may operate in a networked environment using logical connections via wire and/or wireless communications to one or more remote computers, such as a remote computer(s) 848. The remote computer(s) 848 can be a workstation, a server computer, a router, a personal computer, portable computer, microprocessor-based entertainment appliance, a peer device or other common network node, and typically includes many or all the elements described relative to the computer 812, although, for

purposes of brevity, only a memory and/or storage device 850 is illustrated. The logical connections depicted include wire/wireless connectivity to a local area network 852 and/or larger networks, for example, a wide area network 854. Such LAN and WAN networking environments are commonplace in offices and companies, and facilitate enterprise-wide computer networks, such as intranets, all of which may connect to a global communications network, for example, the Internet.

[0131] When used in a local area network 852 networking environment, the computer 812 is connected to the local area network 852 through a wire and/or wireless communication network interface or network adapter 856. The network adapter 856 can facilitate wire and/or wireless communications to the local area network 852, which may also include a wireless access point disposed thereon for communicating with the wireless functionality of the network adapter 856.

[0132] When used in a wide area network 854 networking environment, the computer 812 can include a modem 858, or is connected to a communications server on the wide area network 854 or has other means for establishing communications over the wide area network 854, such as by way of the Internet. The modem 858, which can be internal or external and a wire and/or wireless device, connects to the system bus 806 via the input device interface 842. In a networked environment, program modules depicted relative to the computer 812, or portions thereof, can be stored in the remote memory and/or storage device 850. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers can be used.

[0133] The computer 812 is operable to communicate with wire and wireless devices or entities using the IEEE 802 family of standards, such as wireless devices operatively disposed in wireless communication (e.g., IEEE 802.11 over-the-air modulation techniques). This includes at least Wi-Fi (or Wireless Fidelity), WiMax, and Bluetooth™ wireless technologies, among others. Thus, the communication can be a predefined structure as with a conventional network or simply an ad hoc communication between at least two devices. Wi-Fi networks use radio technologies called IEEE 802.11 (a, b, g, n, ac, ax, etc.) to provide secure, reliable, fast wireless connectivity. A Wi-Fi network can be used to connect computers to each other, to the Internet, and to wire networks (which use IEEE 802.3-related media and functions).

[0134] The various elements of the devices as previously described with reference to FIGS. 1-7 may include various hardware elements, software elements, or a combination of both. Examples of hardware elements may include devices, logic devices, components,

processors, microprocessors, circuits, processors, circuit elements (e.g., transistors, resistors, capacitors, inductors, and so forth), integrated circuits, application specific integrated circuits (ASIC), programmable logic devices (PLD), digital signal processors (DSP), field programmable gate array (FPGA), memory units, logic gates, registers, semiconductor device, chips, microchips, chip sets, and so forth. Examples of software elements may include software components, programs, applications, computer programs, application programs, system programs, software development programs, machine programs, operating system software, middleware, firmware, software modules, routines, subroutines, functions, methods, procedures, software interfaces, application program interfaces (API), instruction sets, computing code, computer code, code segments, computer code segments, words, values, symbols, or any combination thereof. However, determining whether an embodiment is implemented using hardware elements and/or software elements may vary in accordance with any number of factors, such as desired computational rate, power levels, heat tolerances, processing cycle budget, input data rates, output data rates, memory resources, data bus speeds and other design or performance constraints, as desired for a given implementation.

[0135] One or more aspects of at least one embodiment may be implemented by representative instructions stored on a machine-readable medium which represents various logic within the processor, which when read by a machine causes the machine to fabricate logic to perform the techniques described herein. Such representations, known as “IP cores” may be stored on a tangible, machine readable medium and supplied to various customers or manufacturing facilities to load into the fabrication machines that make the logic or processor. Some embodiments may be implemented, for example, using a machine-readable medium or article which may store an instruction or a set of instructions that, if executed by a machine, may cause the machine to perform a method and/or operations in accordance with the embodiments. Such a machine may include, for example, any suitable processing platform, computing platform, computing device, processing device, computing system, processing system, computer, processor, or the like, and may be implemented using any suitable combination of hardware and/or software. The machine-readable medium or article may include, for example, any suitable type of memory unit, memory device, memory article, memory medium, storage device, storage article, storage medium and/or storage unit, for example, memory, removable or non-removable media, erasable or non-erasable media, writeable or re-writeable media, digital or analog media, hard disk, floppy disk, Compact Disk Read Only Memory (CD-ROM), Compact Disk Recordable (CD-R),

Compact Disk Rewriteable (CD-RW), optical disk, magnetic media, magneto-optical media, removable memory cards or disks, various types of Digital Versatile Disk (DVD), a tape, a cassette, or the like. The instructions may include any suitable type of code, such as source code, compiled code, interpreted code, executable code, static code, dynamic code, encrypted code, and the like, implemented using any suitable high-level, low-level, object-oriented, visual, compiled and/or interpreted programming language.

[0136] The foregoing description of example embodiments has been presented for the purposes of illustration and description. It is not intended to be exhaustive or to limit the present disclosure to the precise forms disclosed. Many modifications and variations are possible in light of this disclosure. It is intended that the scope of the present disclosure be limited not by this detailed description, but rather by the claims appended hereto. Future filed applications claiming priority to this application may claim the disclosed subject matter in a different manner, and may generally include any set of one or more limitations as variously disclosed or otherwise demonstrated herein.

CLAIMS

What is claimed is:

1. A computer-implemented method, comprising:

identifying, by a web browser executing on a processor of a device, a payment field in a web page;

determining, by the web browser, that an application associated with an issuer of a contactless card is installed on the device;

generating, by the web browser, a uniform resource identifier (URI) directed to the application, wherein the URI comprises a merchant identifier parameter, a session identifier parameter, and an action identifier parameter;

launching, by a mobile operating system (OS) executing on the processor, the application based on the URI;

authenticating, by the application based on the action identifier parameter of the URI, an account associated with the contactless card based on a cryptogram generated by the contactless card;

receiving, by the application based on the authentication, payment information associated with the contactless card;

accessing, by a WebView component of the application, the web page;

autofilling, by the WebView component of the application, the payment information into the payment field in the web page; and

processing, by the WebView component of the application, a transaction based on the payment information autofilled into the payment field of the web page.

2. The method of claim 1, further comprising:

generating, by the application, a URL of the web page based on the merchant identifier and the session identifier, wherein the merchant identifier is associated with a merchant providing the web page, wherein the session identifier is associated with a browsing session in the web browser; and

accessing, by the WebView component of the application, the web page based on the generated URL of the web page.

3. The method of claim 1 or 2, wherein the payment information comprises an account number received from an authentication server.

4. The method of claim 1 or 2, wherein the payment information comprises a payment token received from a web server hosting the web page.

5. The method of claim 1 or 2, wherein the web browser determines the application is installed on the device based on a function provided by the mobile operating system.

6. The method of claim 1 or 2, wherein the URI is a deep link to the application, wherein the application loads an authentication page of the application based on the action identifier.

7. The method of claim 1 or 2, wherein the authenticating comprises:

transmitting, by the application, the cryptogram to an authentication server; and

receiving, by the application from the authentication server, an indication specifying the authentication server decrypted the cryptogram.

8. The method of claim 1 or 2, further comprising prior to determining that the application is installed on the device:

receiving, by the web browser, input selecting the payment field of the web page;

displaying an indication specifying to process the transaction using the application; and

receiving input specifying to process the transaction using the application.

9. A non-transitory computer-readable storage medium, the computer-readable storage medium including instructions that when executed by a processor of a device, cause the processor to:

identify, by a web browser, a payment field in a web page;

determine, by the web browser, that an application associated with an issuer of a contactless card is installed on the device;

generate, by the web browser, a uniform resource identifier (URI) directed to the application, wherein the URI comprises a merchant identifier parameter, a session identifier parameter, and an action identifier parameter;

launch, by a mobile operating system (OS), the application based on the URI;

authenticate, by the application based on the action identifier parameter of the URI, an account associated with the contactless card based on a cryptogram generated by the contactless card;

receive, by the application based on the authentication, payment information associated with the contactless card;

access, by a WebView component of the application, the web page;

autofill, by the WebView component of the application, the payment information into the payment field in the web page; and

process, by the WebView component of the application, a transaction based on the payment information autofilled into the payment field of the web page.

10. The computer-readable storage medium of claim 9, wherein the instructions further cause the processor to to:

generate, by the application, a URL of the web page based on the merchant identifier and the session identifier, wherein the merchant identifier is associated with a merchant providing the web page, wherein the session identifier is associated with a browsing session in the web browser; and

access, by the WebView component of the application, the web page based on the generated URL of the web page.

11. The computer-readable storage medium of claim 9 or 10, wherein the payment information comprises an account number received from an authentication server.

12. The computer-readable storage medium of claim 9 or 10, wherein the payment information comprises a payment token received from a web server host the web page.

13. The computer-readable storage medium of claim 9 or 10, wherein the web browser determines the application is installed on the device based on a function provided by the mobile operating system.

14. The computer-readable storage medium of claim 9 or 10, wherein the URI is a deep link to the application, wherein the application loads an authentication page of the application based on the action identifier.

15. The computer-readable storage medium of claim 9 or 10, wherein the authenticating comprises instructions that when executed cause the processor to:

transmit, by the application, the cryptogram to an authentication server; and

receive, by the application from the authentication server, an indication specifying the authentication server decrypted the cryptogram.

16. The computer-readable storage medium of claim 9 or 10, wherein the instructions further cause the processor to, prior to determining that the application is installed on the device:

receive, by the web browser, input selecting the payment field of the web page;

display an indication specifying to process the transaction using the application; and

receive input specifying to process the transaction using the application.

17. A computing apparatus comprising:

a processor; and

a memory storing instructions that, when executed by the processor, cause the processor to:

identify, by a web browser, a payment field in a web page;

determine, by the web browser, that an application associated with an issuer of a contactless card is installed on the apparatus;

generate, by the web browser, a uniform resource identifier (URI) directed to the application, wherein the URI comprises a merchant identifier parameter, a session identifier parameter, and an action identifier parameter;

launch, by a mobile operating system (OS) executing on the processor, the application based on the URI;

authenticate, by the application based on the action identifier parameter of the URI, an account associated with the contactless card based on a cryptogram generated by the contactless card;

receive, by the application based on the authentication, payment information associated with the contactless card;

access, by a WebView component of the application, the web page;

autofill, by the WebView component of the application, the payment information into the payment field in the web page; and

process, by the WebView component of the application, a transaction based on the payment information autofilled into the payment field of the web page.

18. The computing apparatus of claim 17, wherein the instructions further cause the processor to:

generate, by the application, a URL of the web page based on the merchant identifier and the session identifier, wherein the merchant identifier is associated with a merchant providing the web page, wherein the session identifier is associated with a browsing session in the web browser; and

access, by the WebView component of the application, the web page based on the generated URL of the web page.

19. The computing apparatus of claim 17 or 18, wherein the payment information comprises an account number received from an authentication server.

20. The computing apparatus of claim 17 or 18, wherein the payment information comprises a payment token received from a web server host the web page.

21. The computing apparatus of claim 17 or 18, wherein the web browser determines the application is installed based on a function provided by the mobile operating system.

22. The computing apparatus of claim 17 or 18, wherein the URI is a deep link to the application, wherein the application loads an authentication page of the application based on the action identifier.

23. The computing apparatus of claim 17 or 18, wherein the authenticating comprises instructions that when executed by the processor cause the processor to:

transmit, by the application, the cryptogram to an authentication server; and

receive, by the application from the authentication server, an indication specifying the authentication server decrypted the cryptogram.

24. The computing apparatus of claim 17 or 18, wherein the instructions further cause the processor to, prior to determining that the application is installed:

receive, by the web browser, input selecting the payment field of the web page;

display an indication specifying to process the transaction using the application; and

receive input specifying to process the transaction using the application.