

SYSTEMS AND METHODS FOR LINKING DEVICES TO USER ACCOUNTS

CROSS-REFERENCE TO RELATED APPLICATION

[0001] The present application claims priority to U.S. Patent Application No. 14/286,248 filed May 23, 2014 entitled “SYSTEMS AND METHODS FOR LINKING DEVICES TO USER ACCOUNTS”, which is expressly incorporated by reference herein in its entirety.

FIELD

[0002] The present disclosure relates to magnetic stripe storage and transmission devices.

BACKGROUND

[0003] Transmission of magnetic stripe data has been done primarily by swiping a magnetic stripe card against a magnetic stripe reader (MSR) to enable payment, identification (ID), and access control functions. Mobile wallet applications on smartphones and tablets have had difficulty interacting with existing merchant point of sale (POS) devices or other devices with MSRs. Contactless reader enabled POS terminals (typically using, for example, an ISO 14443 standard) are not ubiquitous to accept contactless or near field communications (NFC) payments. It would be expensive and would take time to replace the millions of merchant POS devices or door locks that only accept magnetic stripe cards, just to interact with NFC phones or other transmission means like barcodes.

SUMMARY

[0004] The present disclosure relates to devices, systems, and methods including a magnetic stripe storage and transmission device (also referred to as a magnetic stripe transporter (MST)) for use in conjunction with a mobile wallet application to capture, store and transmit magnetic stripe card data to merchants' conventional point of sale (POS) terminals and other devices with magnetic stripe readers (MSRs) or checkout systems, in physical and virtual environments. The devices, systems, and methods provide secure binding, linking, or pairing of the MST to a user account. In some embodiments, this unique binding of the MST to a specific user account provides increased security.

[0004a] According to some embodiments of the present invention, there is provided a method of binding a device to a user account, comprising: sending a binding challenge to a magnetic stripe transporter (MST); receiving a response from the MST; sending a binding request to a server; receiving a binding token from the server in response to the binding request; and

sending the binding token to the MST for use in binding the MST to the user account.

[0004b] According to some embodiments of the present invention, there is provided a method of binding a device to a user account, comprising: receiving a binding challenge from a computing device; sending a response to the binding challenge to the computing device; receiving a binding token generated by a server from the computing device; verifying the binding token; and binding the device to the user account in response to the verification of the binding token.

[0004c] According to an aspect of the present invention, there is provided a method of binding a device to a user account, comprising: receiving a binding request including user input and information corresponding to a magnetic stripe transporter (MST); authenticating a user with the user account based on the user input; determining whether the information corresponding to the MST is valid and the MST is not bound to a second user account; and sending, to the MST, a binding token for use in binding the MST to the user account in response to the information corresponding to the MST being valid and the MST not being bound to the second user account.

BRIEF DESCRIPTION OF THE DRAWINGS

[0005] Embodiments of devices, systems, and methods are illustrated in the figures of the accompanying drawings which are meant to be exemplary and not limiting, in which like

association data of the MST 102 and user account 112, and one or more keys used by the MST 102 and/or the server 106. The MST 102 may be bound with the user account 112, as described in further detail below (it should be appreciated that the terms binding and pairing are used interchangeably herein).

[0014] As illustrated, the MST 102 may be a dongle that may be connected to and disconnected from the mobile communication device 104. The MST 102 may communicate with the mobile communication device 104 through an audio port and/or through other types of communication interfaces, for example including, but not limited to, a USB port, a 30pin or 9pin Apple interface, a Bluetooth interface, a near field communication (NFC), and other serial interfaces. While the MST 102 is illustrated as a dongle, the MST may be another type of peripheral device that communicates with the mobile communication device 104 through a contactless interface, such as Bluetooth or NFC.

[0015] In an aspect, a user may set-up the user account 112 on the server 106, for example, by downloading and/or installing a wallet application in the mobile communication device 104. The user may also set-up a user account 112 using a computer connected to the network 108 by accessing a user account web portal. To set up the user account 112, the user may specify a user name, password and a personal PIN. The password may be used to login to the wallet application on the mobile communication device 104. Once the user is logged in, the personal PIN may be used to enter a payment card section of the wallet application, authenticate with the MST 102, as well as to unlock the wallet application.

[0016] The user may optionally add the MST 102 to the user account 112 by specifying a globally unique identifier (GUID) of the MST 102 (also referred to herein as ID_{MST}). If the GUID specified by the user is valid and is “occupied,” meaning it is currently added under another user account, then it cannot be accepted under the user account 112. If the GUID is valid and not occupied, meaning it is not currently bound with a user account, the server 106 may generate a provisioning token or “binding token.” The provisioning token includes the personal PIN and is backed by the authority of the server. The provisioning token may then be securely injected to the MST 102 when the wallet application next communicates with the server 106. The personal PIN can be seen as a shared secret between the MST 102 and the user, allowing authentication to operate the MST 102 to be performed in the absence of server connectivity. The PIN (which only the user knows) is used to authenticate with the MST 102 to operate any card data stored on the MST 102. A copy of the PIN may also be stored on the server 106 and used as described below. Operation of the MST 102 using the PIN-based authentication can be done with or without the mobile communication device 104

being connected to the server 106 via the network 108. This allows the MST 102 to be operated to utilize the card data stored on the MST 102, even when no network connection exists.

[0017] Each MST 102 may be initially open to be bound with a user account 112. Once the MST 102 is bound, the MST 102 may be locked and have to be unlocked to change modes and parameters on the MST 102. The MST 102 can store cardholder data by either an initial load at manufacturing, loading via a wireless communication network after setting up the user account 112, and/or by the consumer loading his/her own card(s) data directly into the MST 102 using the mobile wallet application. In general, the user is a person that has set up a user account, for example, on the server 106 via a cloud computing infrastructure (such as via the network 108), and has initialized the wallet application on his/her mobile communication device 104.

[0018] A method 200 of initializing and binding the MST 102 to a user account 112 according to an illustrative embodiment is described with reference to FIG. 2. An MST is initialized for the first time to a user account by plugging in or connecting the MST to the mobile communication device, illustrated as block 202. Upon connecting the MST to the mobile communication device, the wallet application recognizes or determines the status of the MST as bound and unbound, illustrated as block 204.

[0019] When the MST dongle has already been bound to another user account, the wallet application will recognize the MST as bound to another user account, illustrated as block 206, and generate an authentication error, illustrated as block 208.

[0020] When the MST has been bound to the appropriate user account, the wallet application recognizes the MST as bound, illustrated as block 210. The MST and the user account may then perform a handshake, illustrated as 212, and send and receive commands, illustrated as block 214.

[0021] When the MST has not been bound and there is no user account bound to the MST, upon connecting the MST to the mobile communication device, for example, a smartphone with the wallet application thereon, the wallet application recognizes the MST as unbound, illustrated as block 216. The wallet application may then face a determination as to whether the MST should be bound to the user account, illustrated as block 218. If the appropriate user account user desires to bind the MST, a binding process begins and the MST is bound to the user account, illustrated as block 220. Upon binding the MST to the user account, the MST and the user account may then perform a handshake, illustrated as 212, and send and receive commands, illustrated as block 214.

[0022] Once the MST has been bound with the user account, the user can use the wallet application to load his/her cards by swiping the cards on a built in magnetic stripe reader (MSR) of the MST or a separate MSR that may be connected to the MST or the mobile communication device. The card data may be digitized and encrypted, and stored into the memory means or secure element of the MST for later use.

[0023] A method 300 of paring the MST 102 to the user account 112 according to an illustrative embodiment is described with reference to FIG. 3. As illustrated, upon connecting the MST 102 to the mobile communication device 104 operating the wallet application, the wallet application sends a binding challenge or query to the MST 102, illustrated as 302. The MST 102 responds to the binding challenge / query by sending a response, illustrated as 304, to the wallet application on the mobile communication device 104. The wallet application on the mobile communication device 104 then sends a binding request, illustrated as 306, to the server 106. The server 106 may authenticate the MST 102 and the request. The server 106 may then send a binding token, illustrated as 308, to the wallet application on the mobile communication device 104 to bind the MST to the user account 112. The wallet application on the mobile communication device 104 forwards the binding token to the MST 102, illustrated as 310.

[0024] In an embodiment, the MST 102 contains an ID_{MST} (such as 16-byte non-predictable ID) and a key K_{MST} (such as a 16-byte key) stored in memory. In this embodiment, the server 106 is capable of generating K_{MST} given the ID_{MST} . The K_{MST} is then a shared secret between the server 106 and the MST 102. Each MST may have a different K_{MST} and ID_{MST} for security purposes.

[0025] The MST 102 and server 106 communicate indirectly via the wallet application on the mobile communication device 104. The communications between the server 106 and the mobile communication device 104 may be secured using SSL3/TLS. Communications between the MST 102 and the mobile communication device 104 may be encrypted using a session key $K_{session}$ derived from the personal PIN and session random nonce.

[0026] In this embodiment, the mobile communication device 104 sends the binding challenge (302), including an indication to initiate binding (for example a random number or other type of initiation indication). The response to the binding challenge / query (304) sent from the MST 102 to the mobile communication device 104 includes the ID_{MST} and a random number R_{MST} (also referred to as a nonce) generated by the MST. The binding request (306), with input from the user includes the user's username, password, and the ID_{MST} and R_{MST} generated by the MST. The server 106 authenticates the user with the user account 112 using

the username and password. The server 106 then checks to see if the received ID_{MST} is valid and that the MST 102 is currently not bound to any other user account. The server 106 computes K_{MST} using the ID_{MST} , and sends back a binding token (308) signed using K_{MST} . The binding token may include R_{MST} , a server generated time-stamp R_s , the PIN, and may also include some auxiliary information, such as a verification component that will have to be transported along with the signature in order for it to be verifiable by the MST 102. The wallet application on the mobile communication device 104 forwards this binding token to the MST 102 (310). The MST verifies the binding token and matches R_{MST} . If everything looks fine, the MST installs the PIN. At this moment, the MST is said to be bound or bound to the user account 112, and the user can operate the MST using the personal PIN.

[0027] In this embodiment, the handshake (illustrated as block 212 in FIG. 2) may be performed by the wallet application first sending an Exchange Nonce (EN) command to the MST 102 along with a random challenge R_W^1 generated by the wallet application. The MST 102, upon receiving the message, generates and returns a random nonce R_{MST}^1 . The MST 102 also echoes EN, by sending EN back to the wallet application. At this stage, both the wallet application and the MST 102 know the other's fresh nonce. During subsequent communications, the sender always acknowledges the receiver's nonce as part of the message payload. The purpose of the handshake for exchanging nonce can be seen as an effort to defray any replay attacks. The counter-party's nonce is in service until another handshake is performed, for example, until the wallet application sends the next EN message. There may also be a life-span associated with each handshake, and the MST 102 and/or the wallet application may request a new handshake if a previous handshake has expired.

[0028] After the handshake is complete, both the wallet application and the MST 102 are ready to send and receive commands (illustrated as block 214 in FIG. 2). The authentication is performed on a per message basis; that is, the sender must demonstrate its knowledge of the shared secret, in this case, this is the personal PIN that the user specified during set-up of the user account. A command CMD may be sent from the wallet application to the MST 102 by sending the CMD and R_{MST}^1 signed using the PIN or a derivation of the PIN. Similarly, subsequent messages in the other direction (the MST 102 to the wallet application) are sent by sending the CMD and R_W^1 signed using the PIN. The use of the combination of the PIN and nonce ensure proper authentication and defense against replay for both parties. However, the protocol may still be susceptible to replay attack within a same handshake session. Therefore, a counter may be included in the CMD within a session. The sender may then increment the counter every time a new CMD is sent, and the receiver may check the counter

to verify whether the counter is monotonically increasing.

[0029] In another embodiment, the server 106 stores a public-private key bind (K_S and K_S^{-1}). The server 106 may generate, for example, a self-signed certificate ($Cert_S$), a root certificate, intermediate certificate, signing certificate, etc. This certificate is used to verify certificate chain locally at the wallet application and the MST 102. The wallet application associated with the user account 112 also has a public-private key bind (K_W and K_W^{-1}). The private key is stored in a password-protected keystore or a keychain. The public key, user account ID and optionally some auxiliary information (used for verification purposes) are sent to the server 106 for certification. The wallet application securely possesses its identity certificate $Cert_W$, which is signed by server 106, and the wallet application securely possesses $Cert_S$ in a trusted store. The MST 102 also has a public-private key bind (K_{MST} and K_{MST}^{-1}) generated at manufacturing. The MST 102 possesses its identity certificate $Cert_{MST}$ and $Cert_S$, both assigned and installed at manufacturing.

[0030] Using the certificates and keys described above, the wallet application on the mobile communication device 104 can obtain the binding status of the MST 102 without the need for a network connection. To perform this function, the wallet application detects that the MST 102 is connected to the mobile communication device 104. The wallet application generates a random challenge R_W (such as, a time-stamp and a random number) and sends it to the MST 102. In response, the MST 102 generates a random challenge R_{MST} . The combination of R_W and R_{MST} represent a mutually-verifiable fresh nonce, and the MST 102 signs it with K_{MST}^{-1} . The MST 102 sends R_{MST} , the signature and its identity certificate $Cert_{MST}$ to the wallet application.

[0031] The wallet application knows R_W and its freshness and can thus verify the signature as newly computed by the MST 102, thereby ruling out replay attack. Moreover, the wallet application can authenticate the MST 102 from the signature. If everything verifies, the wallet application generates a session key $K_{session}$ and a random sequence number Seq_W and then signs $[R_W, R_{MST}, K_{session}, \text{and } Seq_W]$. The resulting signature and the wallet application's identity certificate $Cert_W$ is sent to the MST 102. The MST is then able to authenticate the wallet application. The secrecy of the session key is guarded by the encryption using the MST's public key. At this stage, the MST 102 checks its internal state and answers if it is ready to perform new binding or it is currently bound with a user account.

[0032] In this embodiment, a method 400 for performing a new binding is described with reference to FIG. 4. The wallet application sends the session key $K_{session}$ to the MST 102 (402). The MST 102 sends a binding ready signal (modeled as a constant PR) as well as a

challenge $[R_W, R_{MST}]$ and acknowledgement of the receipt of the session key $K_{session}$ to the wallet application (404). From this moment, the wallet application and the MST 102 use the session key $K_{session}$. The wallet application decrypts the response from the MST 102 (404) using $K_{session}$ and obtains the MST state constant PR (406). The wallet application first informs the server 106 that a binding is to be performed (408); this intention is encoded with a constant pairing signing request (PSR), as well as the certificates of the MST 102 and the wallet application ($Cert_W, Cert_{MST}$). Upon receiving PSR, the server 106 generates a fresh challenge R_S (including a server time stamp, ID_W and ID_{MST}) (410) and sends the challenge to the wallet application (412).

[0033] The wallet application does not need to authenticate R_S since the transmission is over an SSL/TLS (RFC6101/RFC2246) session. The wallet application passes along R_S , together with R_{MST} , and the PSR message to the MST 102 signed by $K_{session}$ (414). The MST 102 decrypts the message using $K_{session}$ (416). The MST 102 can therefore assert that the message is from the wallet application and verify its freshness from R_{MST} . The MST 102 also verifies that the IDs inside R_S are itself and the wallet application (418). The MST 102 returns its signature of the request and therefore expresses its willingness to perform binding with the user account (420). The MST 102 returns R_S and PSR signed by K_{MST}^{-1} , all signed by $K_{session}$ to the wallet application. The wallet application verifies that the message is signed by the MST 102 with $Cert_{MST}$ (422). The wallet application also signs the same inner content, resulting in R_S and PSR signed by K_W^{-1} and thereby expresses its willingness to perform binding (424). Finally, the wallet application returns both signatures (R_S and PSR signed by K_{MST}^{-1} , and R_S and PSR signed by K_W^{-1}) to the server 106 over the secure channel (426).

[0034] The server 106 verifies with $Cert_W$ and $Cert_{MST}$ and recognizes the freshness of this signing request from R_S (428). The server 106 then performs a signing over R_S and effectively approves binding of the wallet application and the MST 102 to the user account 112, with a time-stamp signified from within R_S (R_S signed by K_S^{-1}) (430). The server 106 then sends this provisioning packet to the wallet application (432). The server 106 also saves the cryptogram ($\{R_S, PSR, Cert_W, Cert_{MST}, \{R_S, PSR\}K_{MST}^{-1}, \{R_S, PSR\}K_W^{-1}\}$) as evidence of issuing the provisioning packet or token (308).

[0035] The wallet application extracts the content using the root certificate $Cert_S$ and verifies that ID_W and ID_{MST} are correct (434). If all are correct, the wallet application forwards the provisioning packet or token ($\{\{R_S\}K_S^{-1}\}K_{session}$) to the MST 102 (436). The wallet application saves the cryptograms ($\{\{R_S\}K_S^{-1}, Cert_{MST}\}$) as a record for the binding. The MST 102 verifies the binding and extracts the content with root certificate $Cert_S$ (438). The

MST 102 then verifies that ID_{MST} is itself and ID_W is the correct user account associated with the wallet application. At this stage, it promotes its internal state to “bound” (440). The MST 102 also saves the cryptograms ($\{\{R_S\}K_S^{-1}, Cert_W\}$) for later handshakes with the same user account.

[0036] In this embodiment, the handshake (illustrated as block 212 in FIG. 2) may be performed as described below. The MST 102 is currently bound with a user account. The MST 102 compares the $Cert_W$ it received (after authenticating the wallet account and/or the wallet application) and a wallet account ID from its stored provisioning packet $\{R_S\}K_S^{-1}$ that it received as described above. If the two match, the MST 102 sends a handshake complete signal (modeled as a constant “HC”), a randomly generated sequence number Seq_{MST} as well as the challenge $\{R_W, R_{MST}\}$ and its acknowledgement of receipt of session key $K_{session}$. From this moment, the wallet application and the MST 102 switch to using session key $K_{session}$.

[0037] The wallet application reads the cipher text, decrypts and sees R_W so it understands the freshness of the message. The wallet application also sees HC, so it knows that the MST 102 has accepted the handshake. Finally, the wallet application compares the identity ID_{MST} with the one from R_S described above, if the two match, the wallet application promotes its internal state to handshake complete.

[0038] After the handshake is complete, both the wallet application and the MST 102 are ready to send and receive commands (illustrated as block 212 in FIG. 2). The combination of the session key (described above) and randomly generated sequence numbers from both parties are used to ensure proper security during this operation. Before sending/receiving any commands, both the wallet application and the MST 102 possess its own and know the other's sequence number. Seq^i denotes the sequence number of a principal (in this case either the wallet application or the MST 102) prior to its $(i + 1)^{th}$ message transmission as a sender. Thus initially $Seq^0_W = Seq_W$ and $Seq^0_{MST} = Seq_{MST}$. Where Seq_W and Seq_{MST} are sequence numbers as described above. Additionally, a deterministic function f that is known to both the wallet application and the MST 102 is defined, for either the wallet application or the MST 102 and the i^{th} sequence number Seq^i : $f(Seq^i) = Seq^{i+1}$.

[0039] The message protocol and enforcement constraints are as follows: suppose at some stage principal X (i.e. the MST or the wallet application) has sent i number of commands to principal Y (i.e., the other of the MST or the wallet application) and the principal Y has sent j number of commands to principal X, and suppose that X is now sending the $(i + 1)^{th}$ command to the Y. The format of the message may be : $X \rightarrow Y: \{Seq^j_Y, Seq^{i+1}_X,$

$CMD\}_{K_{session}}$, assuming without loss of generality that X received Seq_Y^i . Where, CMD is the specific command that X is sending to Y. Y decrypts the message using session key $K_{session}$. Y compares Seq_Y^i with its currently stored sequence number, and takes the latest sequence number of X that Y received (which is Seq_X^i) and verifies that with Seq_X^{i+1} . The combination of the session key (described above) and the sequence numbers from both parties are used to ensure proper security during this operation.

[0040] Once the MST 102 is bound with the user account 112, the MST 102 can be used to interact with a merchant point of sale (POS) by transmitting magnetic stripe data from a magnetic field transmitter to a magnetic stripe reader (MSR) of the merchant POS. As illustrated in FIG. 5, the MST 102 includes a microprocessor 502, a light-emitting diode (LED) indicator 504, a power source 506, optionally a magnetic stripe reader (MSR) 508, a memory storage component or secure element 510, an input/output interface 512 (for example, a 3.5mm or other standard audio port, a USB port/jack interface or other communication interface, including but not limited to a 30pin or 9pin Apple interface, a Bluetooth interface, and other serial interfaces), and a magnetic field transmitter 514 which includes a driver and an inductor for transmitting magnetic pulses to be received by any POS device with a MSR, such as the POS 516.

[0041] Microprocessor 502 handles security and communications with the mobile communication device 104. The microprocessor 502 can also transmit and receive encrypted card data to and from the secure element 510. The magnetic field transmitter 514 transmits magnetic stripe data of a cardholder to the POS device 516 by transmitting magnetic impulses to the MSR of the POS device 516. The MST 102 may also be used for reading other magnetic stripe cards by using the optional MSR 508. The MSR 508 may be used for loading payment card data onto the secure element 510 and for capturing card track data.

[0042] The mobile communication device 102 includes the wallet application, and may also include a display with key pad or touchpad display and a central processing unit (CPU). The wallet application initializes and unlocks the MST 102, interacts with the MST 102 and accepts card payment data from the MST 102.

[0043] The card data may be encrypted, and the encrypted data may be transmitted to the mobile communication device 104. The wallet application may transmit the data to the server. The data may be decrypted at the server and the primary account number (PAN) data, card number, expiration and name of the cardholder is stripped from the track data. The wallet application or the server may also make a determination as to whether the magnetic card is a payment card or a non-payment card. If the magnetic card is a non-payment card

the MST 102 can automatically store the track data in the memory for non-payment transmission. If the magnetic card is a payment card, for example, having a specific format recognizable to the system, the card may be detected as a payment card and the system determines if the name on the payment card matches the name of the user account. If the name does not match, an error message may arise. If the name on the payment card matches the name of the user account, the system may determine if the PAN number matches an existing card already stored on the server, to either create a new account or leave the existing one. If a new card is created, the system may store the track data in a payment section of MST's secure memory encrypted.

[0044] The MST 102 has the ability to load any type of magnetic stripe card into the memory means, not just payment cards. Non-payment cards may be stored separately with less security for convenience. For example, some non-payment applications may include cards to open doors, loyalty cards, etc. The loading of payment data vs. non-payment data may be separated into two separate fields or storage areas. In an example, payment cards may not be loaded into non-payment storage. For example, payment data may have a specific format that can be detected and may not be allowed to be loaded into the non-payment storage area. The payment cards may also require authentication with the application before being transmitted. On the other hand, default non-payment data may be transmitted without authentication.

[0045] The devices, systems, and methods disclosed herein provide for the magnetic card track data to be captured and stored in the MST's secure memory means directly by the user without modification, and to be used later with a POS or other MSR device. The unique binding of a MST to a specific user account such that the MST can be only used with that account for track data storage and transmission use provides better security.

[0046] The MST is capable of connecting to mobile communication devices via different interfaces beyond audio jack and USB connections. The devices, systems, and methods allow for the loading of encrypted magnetic stripe track data into the memory means of the MST that can later be decrypted and transmitted to the POS, or can be transmitted encrypted to the mobile communication device and then routed to the payment server for decryption and processing for loading a user account on the server or processing a POS transaction. The devices, systems, and methods provide for the ability to use the stored track data or swiped track data for virtual checkout environments for a more secure and lower cost transaction for merchants. The devices, systems, and methods provide for the remote loading and transmission of track data from a card issuer to the wallet server provider, to the wallet application on the mobile communication device, and to the SE or memory means of the

MST for later use. The devices, systems, and methods also provide for the ability to load loyalty account information along with the payment card data into one or more discretionary fields of the track data to be read by the issuer during or after a transaction, which can lead to offers and loyalty programs combined with a payment transaction.

[0047] The mobile communication device may be a laptop computer, a cellular phone, a personal digital assistant (PDA), a tablet computer, and other mobile devices of the type. Communications between components and/or devices in the systems and methods disclosed herein may be unidirectional or bidirectional electronic communication through a wired or wireless configuration or network. For example, one component or device may be wired or networked wirelessly directly or indirectly, through a third party intermediary, over the Internet, or otherwise with another component or device to enable communication between the components or devices. Examples of wireless communications include, but are not limited to, radio frequency (RF), infrared, Bluetooth, wireless local area network (WLAN) (such as WiFi), or wireless network radio, such as a radio capable of communication with a wireless communication network such as a Long Term Evolution (LTE) network, WiMAX network, 3G network, 4G network, and other communication networks of the type.

[0048] While “binding” is discussed herein effectively as a pairing of device to account, those skilled in the art should appreciate that in addition to one-to-one binding, one-to-many binding may be effected according to the disclosure. That is one specific user device/MST may be bound to one or more specific, owned accounts, or one account may be bound to one or more specific, owned devices.”

[0049] Although the devices, systems, and methods have been described and illustrated in connection with certain embodiments, many variations and modifications will be evident to those skilled in the art and may be made without departing from the spirit and scope of the disclosure. The discourse is thus not to be limited to the precise details of methodology or construction set forth above as such variations and modification are intended to be included within the scope of the disclosure.

CLAIMS

What is claimed is:

1. A method of binding a device to a user account, comprising:
receiving a binding request including user input and information corresponding to a magnetic stripe transporter (MST);
authenticating a user with the user account based on the user input;
determining whether the information corresponding to the MST is valid and the MST is not bound to a second user account; and
sending, to the MST, a binding token for use in binding the MST to the user account in response to the information corresponding to the MST being valid and the MST not being bound to the second user account.
2. The method of claim 1, wherein the receiving the binding request includes receiving a username and a password, and an identification and a random number generated by the MST.
3. The method of claim 2, wherein the authenticating the user with the user account includes authenticating the user with the user account using the username and the password.
4. The method of claim 2, wherein the determining whether the information corresponding to the MST is valid includes determining whether the identification generated by the MST is valid.
5. The method of claim 4, further comprising computing a key corresponding to the MST using the identification generated by the MST.
6. The method of claim 5, wherein the binding token includes the random number generated by the MST, a server generated time-stamp, and a personal identification number signed using the key.
7. The method of claim 1, wherein the binding request is received from a computing device.

8. The method of claim 7, wherein the binding token is sent to the computing device.

9. The method of claim 7, wherein the binding token is sent to the MST by the computing device.

10. The method of claim 9, wherein the binding token is validated by the MST for authenticity.

FIG. 1

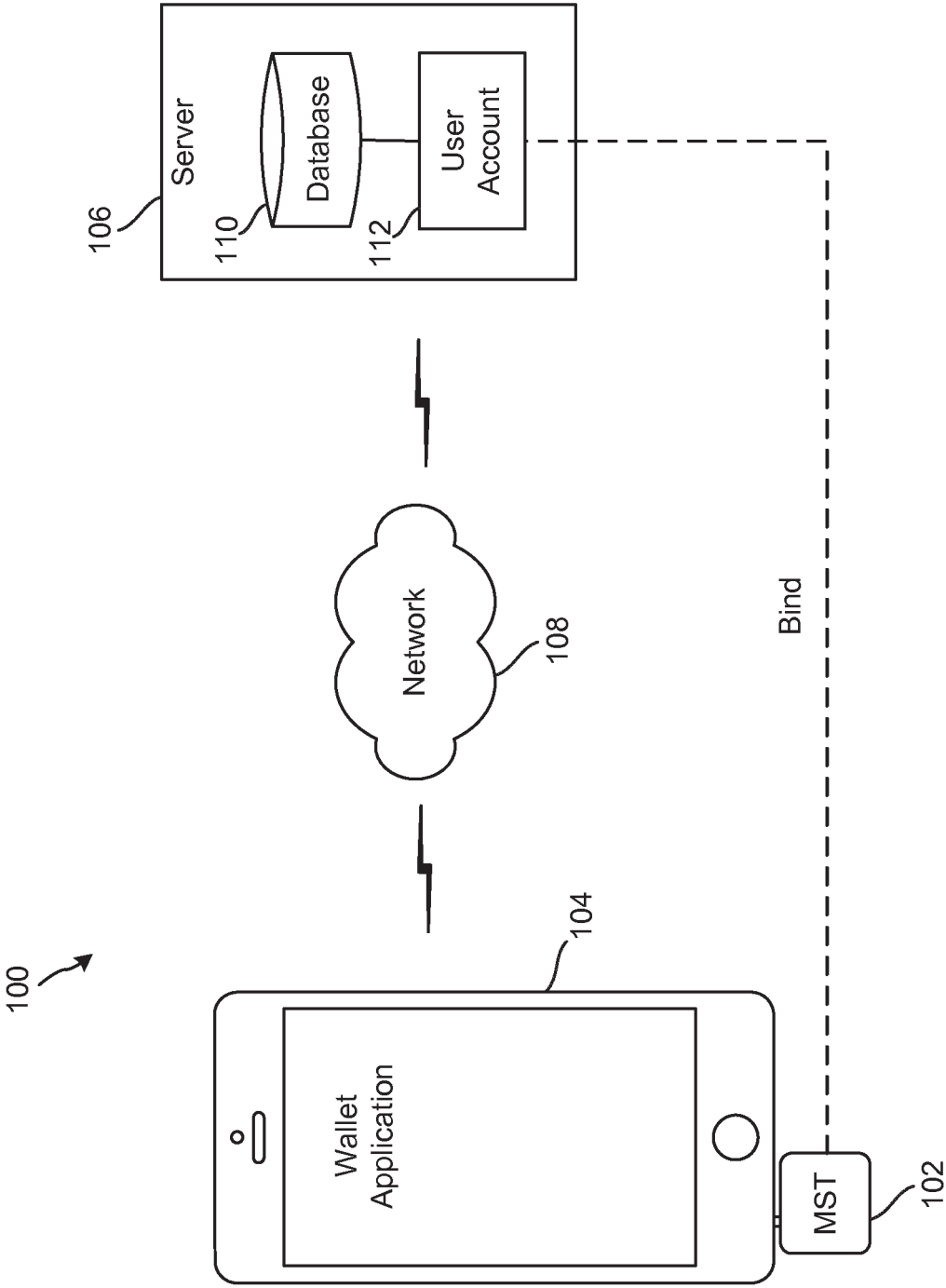


FIG. 2

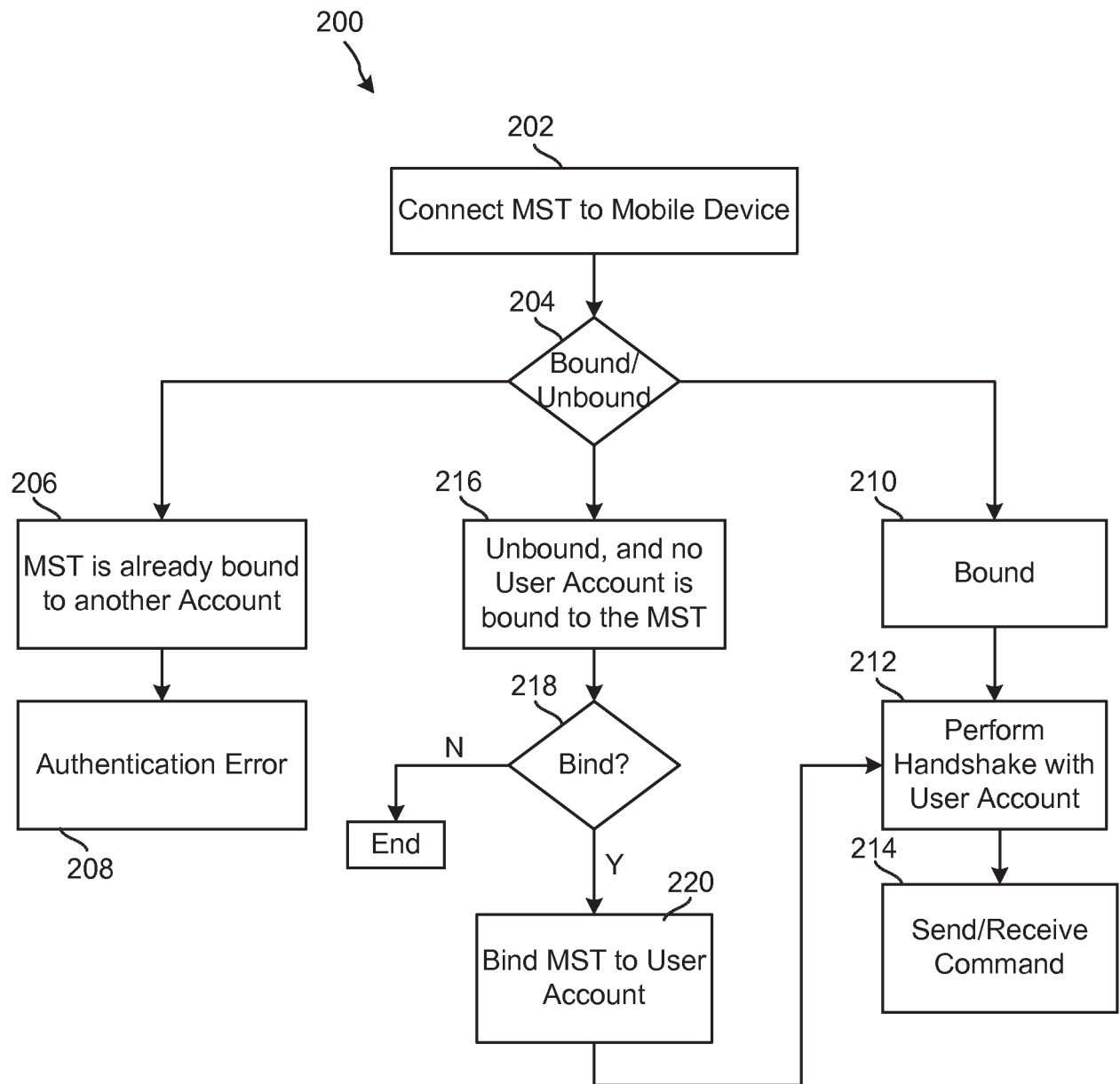


FIG. 3

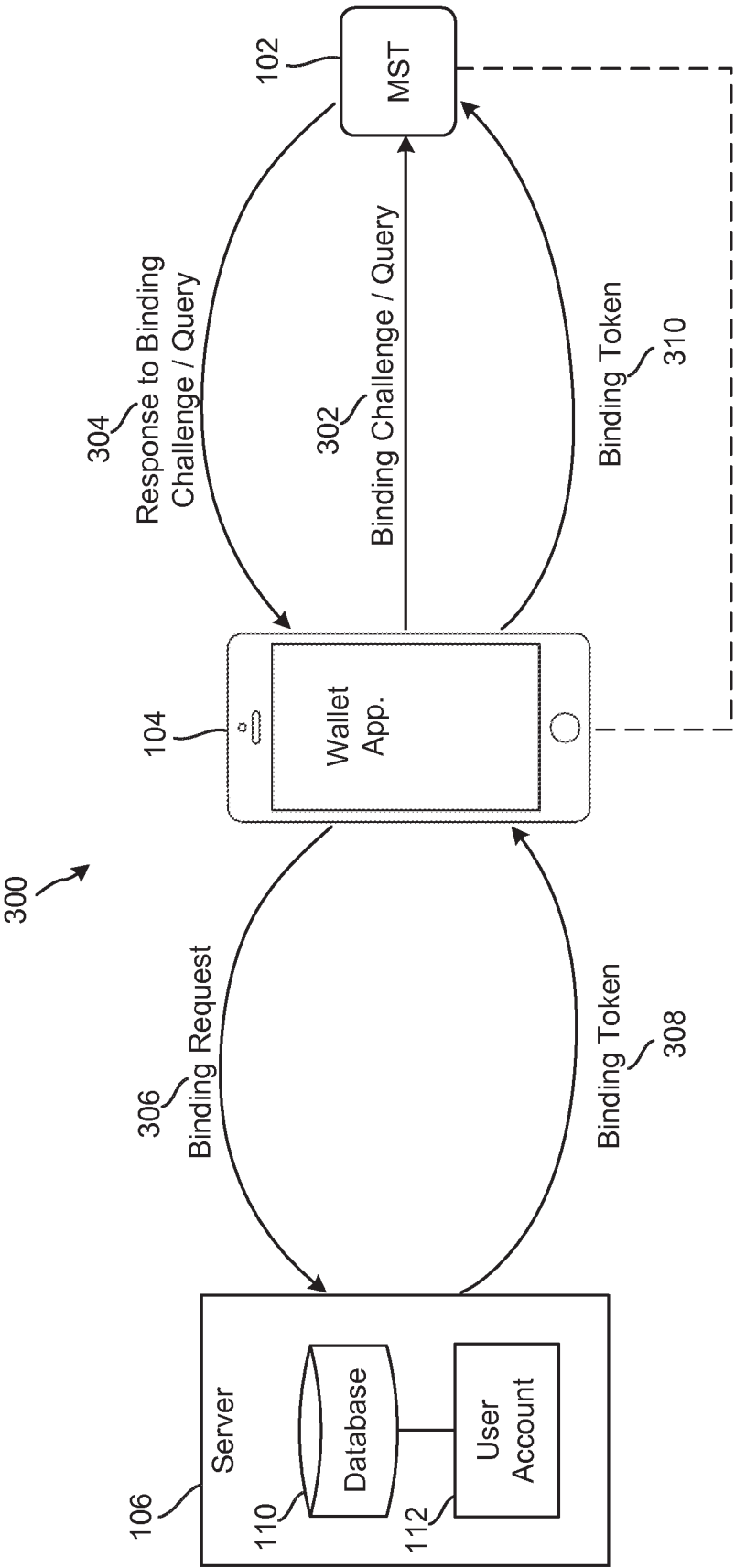


FIG. 4

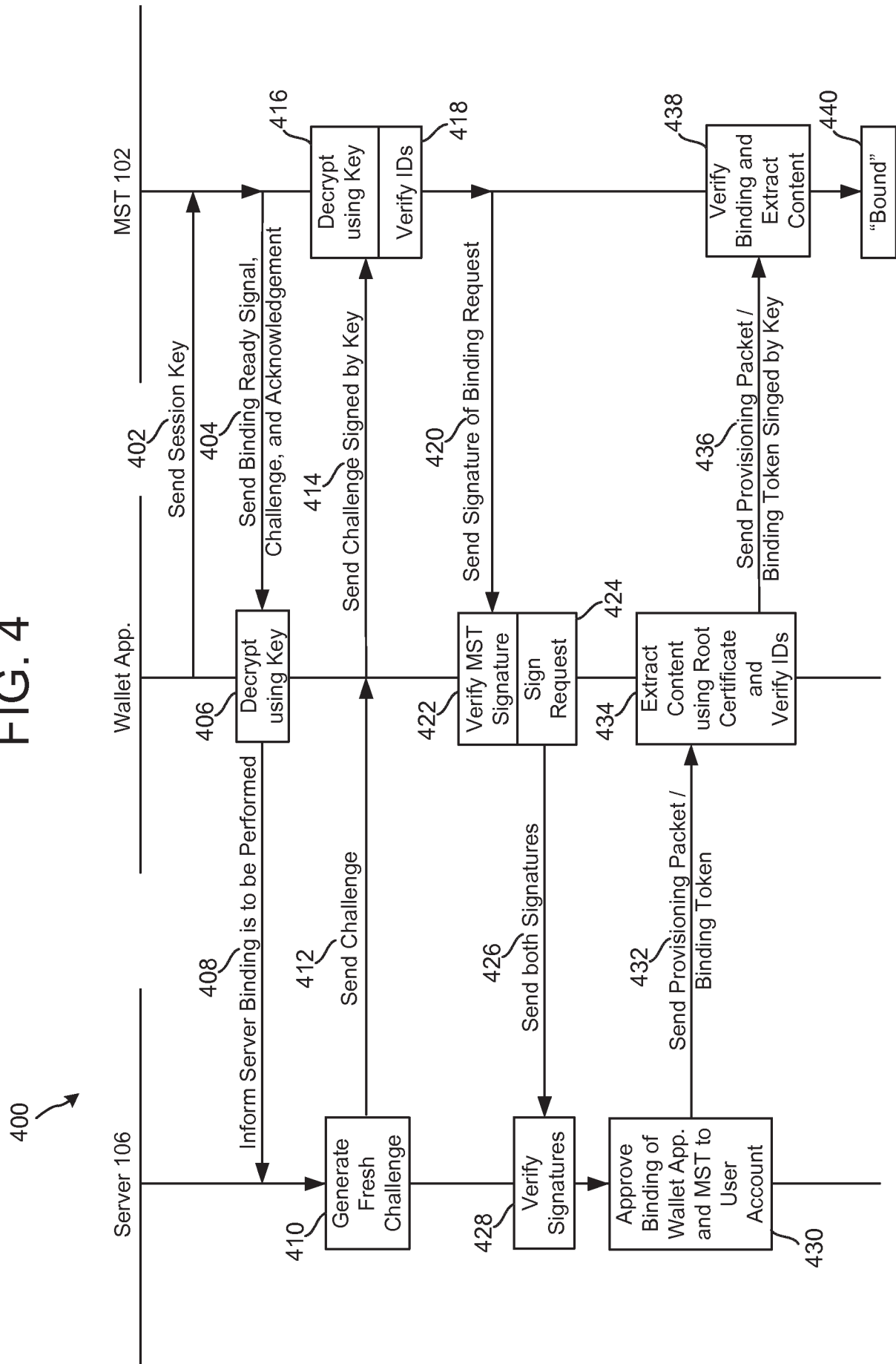


FIG. 5

