SQUARELINK

Simple and secure sign-in for apps and dapps

WHITE PAPER

October 2018

TABLE OF CONTENTS

ABSTRACT

The advent of blockchain technology has created platforms for individuals and organizations across every industry to engage in secure, transparent, efficient, and indisputable interactions. Since its popularization in 2009, we have seen incredible innovations in blockchain's base-layer protocols, applications, and developer ecosystem. However, in order for blockchain technology to achieve true mainstream adoption, we need to continue to find solutions for prominent pain points in the community. Most notably, we are still using the same key management concepts as 10 years ago, even as we recognize their limitations in terms of security or the user experience.

Squarelink has developed the first access tool for blockchain apps and services that enables users to securely recover lost private keys via our Private Key Recovery Technology.[1] With this new standard, we have created methods for bringing centralized user experiences and capabilities to the decentralized world without limiting the security advantages of decentralized applications.

1.0 INTRODUCTION

1.1 CURRENT SOLUTIONS

Interaction with blockchain technology requires the use of key-pairs consisting of a randomly-generated public key and private key. Whether a user intends to generate transactions, derive one's blockchain address, or encrypt information for secure storage, the user must keep their private key hidden from the outside world. Several key management methods have been popularized, including paper wallets, hardware wallets, deterministic wallets, browser tools like MetaMask, and custodial wallets like Coinbase.

Paper wallets are simply a key-pair printed and exclusively stored on a piece of paper. This method is highly secure but difficult and inconvenient to use. Hardware wallets are arguably more secure; these devices store a user's private key, hidden from the outside world – even the user. They internally generate transactions and execute other operations requiring the private key before releasing it to the user. MetaMask functions similarly, storing a user's private key in a browser extension. The extension automatically generates signed transactions without allowing third party websites to access the private key itself. Deterministic (HD) wallets provide an n-word mnemonic phrase as a means of regeneration (i.e. "dog cat mouse..."). Rather than a user-specified phrase, this is simply an alternative representation of a private key and must be handled accordingly. Custodial wallet services manage a user's private key on his or her behalf and provide access via centralized authentication methods. This provides a simple, familiar interface but creates a central point of failure in a decentralized ecosystem.

Apart from custodial wallets, the aforementioned methods require the owner of a key-pair to create secure backups. The owner cannot recover their key-pair once lost.

1 Patent Application #15/974,802: TECHNOLOGIES FOR PRIVATE KEY RECOVERY IN DISTRIBUTED LEDGER SYSTEMS.

1.2 PROBLEM

While key-pairs provide the cryptographic backbone for many of blockchain's intricacies, their inconvenience, and the risk associated with private key loss, have become a barrier to the widespread and unrestricted adoption of blockchain. First, users must reframe their understanding of digital account access from username-password combos and third-party authentication (Facebook/Google) to randomly generated key-pairs. Additionally, new users must also understand the gravity of losing their private key, which they cannot recover as they might expect in a centralized service.

Custodial wallets provide an effective soft-introduction for new blockchain users. Yet, their account centralization creates a glaring security risk. Between Mt. Gox, Coincheck, Bitfinex, and several other infamously hacked crypto exchanges, hackers have made away with nearly $1.5 billion in the past five years.

The current blockchain ecosystem has failed to comprehensively address these issues. Users need a solution that provides a secure interface that is easy to use and understand. They need a solution that provides intuitive, modern methods for recovering a blockchain account. Squarelink has developed the first access tool for blockchain apps and services that accomplishes all of this while preserving the decentralized ownership of each account in a blockchain.

2.0 THE SQUARELINK SOLUTION

2.1 OVERVIEW

Squarelink introduces a new key management tool to bridge the gap between usability and decentralized security. We've combined a traditional username/password interface with several novel encryption models to provide a simple, highly secure, and globally-accessible toolset for self-sovereign private key management.

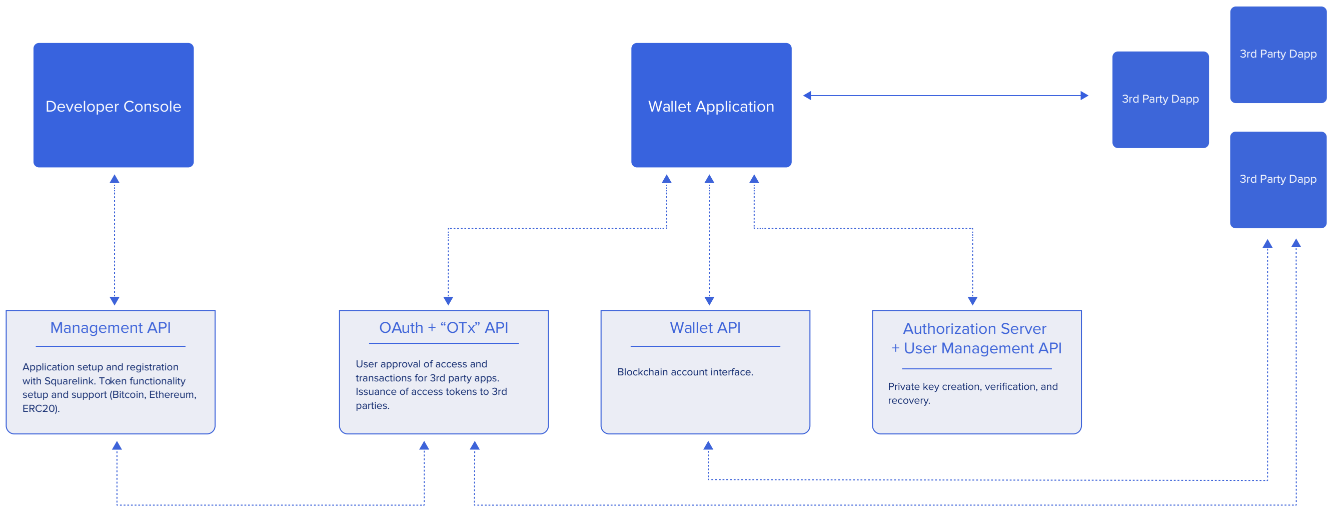
With Squarelink, a user owns a 512-bit Master Key that cannot be accessed by Squarelink nor any other parties. This Master Key is derived from the user's email address, user-specified password, and randomly generated 256-bit cryptographic salt[2] that resides securely on Squarelink servers. Squarelink allows users to protect, access, and use any of their private keys via their Master Key. Whether signing blockchain transactions, encrypting data for secure storage, or accessing self-governed account data, we ensure that keys are never accessible to applications, websites, or third parties — not even to Squarelink.

However, we couldn't boast the usability that centralized services offer without account recovery capabilities. We have therefore moved beyond n-word mnemonic phrase recovery to create self-sovereign versions of traditional centralized recovery methods. We place zero burden on the user to keep track of recovery information and simultaneously ensure that the only person who can recover their Master Key is the user themself. Squarelink also provides integration tools for blockchain applications to easily interact with their users via Squarelink. Through this service, applications can request transactions or account information, and users may approve or sign using their Squarelink app — similar to the way a hardware wallet works but as simple as OAuth or Facebook Connect.

2 A Cryptographic Salt is a random value used to increase entropy (randomness) in a key derivation or hash.

2.2 THE SQUARELINK TOOL

With our solution, users interact with blockchain apps and services via the Squarelink application. Depending on the user's desired functionality, Squarelink offers several interfaces of the application.


Developer Console
Wallet Application
3rd Party Dapp
3rd Party Dapp
3rd Party Dapp
Management API
Application setup and registration with Squarelink. Token functionality setup and support (Bitcoin, Ethereum, ERC20).
OAuth + "OTx" API
User approval of access and transactions for 3rd party apps. Issuance of access tokens to 3rd parties.
Wallet API
Blockchain account interface.
Authorization Server + User Management API
Private key creation, verification, and recovery.

A novice blockchain user might use Squarelink via our out-of-the-box web portal. Users can sign transactions or offer their account details to any blockchain application offering Squarelink integration. A "Log in with Squarelink" button/link will redirect the user to Squarelink's application where they can sign into their account (creating or regenerating their Master Key) and generate the requested information. Squarelink will then redirect back to the site and return the information without ever exposing keys to the application.

For users looking to experience more functionality or further enhanced security, we alternatively offer browser extension and mobile application versions of the Squarelink tool. Any Squarelink-enabled client applications can receive the same information via these varied Squarelink applications. For example, a user with the Squarelink Chrome extension can approve an application's request through the Chrome extension itself without the user ever leaving the site. Users with either the iOS or Android versions can perform this same task via their mobile device. Furthermore, all client applications also come with traditional blockchain wallet capabilities and can generate/broadcast custom transactions outside the context of a Squarelink-integrated application similar to any popular wallet.

With these client-versions of Squarelink, a user has the option to generate their Master Key using an additional salt. This new 256-bit "Client Key" is only ever stored on the user's device, thus requiring the device itself to generate a Master Key. In addition to enhanced security, these applications provide tools such as blockchain explorers, interfaces for viewing encrypted account-related data, and key-import tools for adding other blockchain accounts like Bitcoin. However, users without a client version of the Squarelink application can still manage these account features via a Squarelink web portal.

Users further increase their account security by adding 2-Factor Authentication[3] to their salt. Squarelink provides a variety of 2FA methods for the user to choose from, including verification of their identity via SMS, email, or challenge questions to retrieve their salt from Squarelink servers and generate their Master Key.

3 2-Factor Authentication (2FA) – an additional verification of a user's identity via email, SMS, OTP, etc.

Again, our users have full-ownership of their Master Key. As such, we do not lock users into our service. We offer key-export tools for users to delete their account and/or manage their keys by other means.

2.3 ACCOUNT RECOVERY

Our proprietary, non-custodial private key recovery technology is the cornerstone of Squarelink's solution, allowing the Squarelink application to effectively bridge the gap between ease-of-use and decentralized applications. If a user forgets their password for a centralized application, they can simply reset it by verifying their identity via email, SMS, or other means. Squarelink has modified these common recovery methods, increasing security while still allowing users to recover and reset their Master Key using a familiar process in the event they forget the password used to construct it.

During the Squarelink setup process, users activate one of several Master Key recovery methods. Currently, Squarelink offers account recovery via any combination of email with PGP encryption, challenge questions, and/or Universal Second Factor (U2F)[4]. To accomplish this, we encrypt a "Recovery Seed" – a value that comprises at least a part of the user's Master Key – with a "Recovery Key" that only the user can access or reproduce. This ensures that Squarelink or other third parties cannot access the Recovery Seed. Squarelink then uses the user's password hash[5] as the Recovery Seed. The Squarelink tool makes novel use of S/MIME[6], PGP[7], challenge question answers, and/or U2F keys to encrypt the Recovery Seed as described later in detail. The encrypted seed is then protected on Squarelink servers until a user needs to recover their account.

Account recovery requires a user to reproduce their Recovery Key via their previously determined recovery methodology. Completed correctly, this recovery process decrypts the Recovery Seed, allowing the user to regenerate their original Master Key. With email, SMS, or U2F, a message containing the Recovery Seed is encrypted with the public key of their respective account or device. The plaintext (unencrypted) message never touches Squarelink servers, so no person can access it without the corresponding private key of the key-pair. Similarly, a user must reproduce the same answers to any challenge questions they provided during setup to decrypt their Recovery Seed. Users can then specify a new password to generate a new Master Key. Finally, Squarelink will take care of re-encrypting any keys or data encrypted by the original Master Key and transfer ownership to this new Master Key.

2.4. INTEGRATING BLOCKCHAIN APPLICATIONS

Squarelink technology doesn't benefit blockchain users exclusively. Private key management creates challenges to blockchain application developers as well. Currently, developers may choose to generate private keys through a client-side application and instruct their users to keep track of them. While simple, this approach places the burden on the user and, in many instances, discourages the user from using the application. Alternatively, a developer may choose to develop a custodial wallet solution. This requires incredible vigilance and diverts many invaluable resources toward building massive security infrastructures. No matter the complexity, custodial wallets create a central point of failure, requiring developers to continually improve and monitor their solutions. In both of these scenarios, users are forced to make a trade-off between either security or accessibility, and developers are ultimately inconvenienced.

4 Universal Second Factor (U2F) – a device used to generate cryptographic signatures without releasing its private key.
5 Hash – A function where its output can be replicated by the same input, but never reversed to obtain the original input.
6 S/MIME – an email protocol supporting encrypted mail via digital certificates (public/private-key pairs).
7 PGP – an SMS protocol supporting encrypted messages via digital certificates (public/private-key pairs).

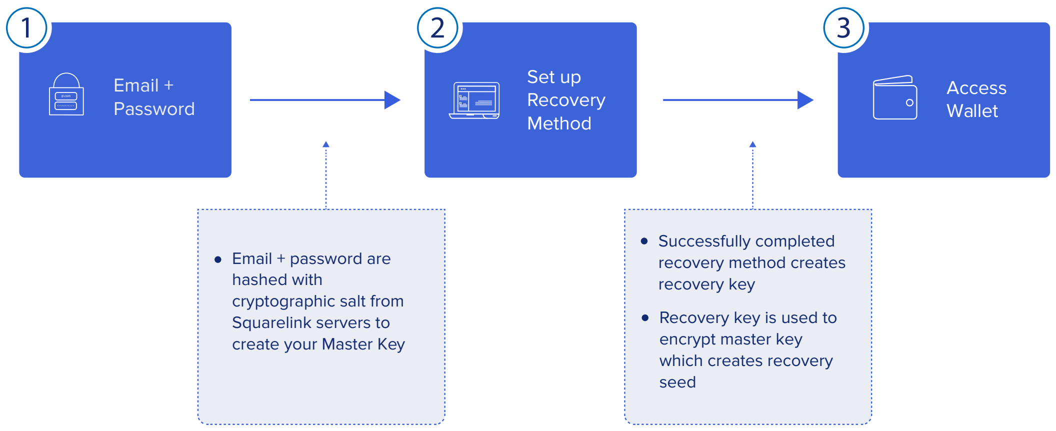
Squarelink seeks to address these issues. Blockchain developers may offer Squarelink integration in addition to, or in replacement of, their custom key-management solutions. For non-custodial applications, Squarelink vastly improves customer acquisition and retention with its simple, intuitive interface. Squarelink allows custodial applications to offload the stress and burden of managing private keys and additionally remove any central point of failure – no hack can feasibly compromise all Squarelink users.

3.0 TECHNOLOGY & SECURITY

This section provides a comprehensive technical overview of ideas previously presented and is further detailed in United States Patent Application no. 62625745

3.1 ACCOUNT CREATION

When a user first creates their Squarelink account, they enter a valid email address and a strong password. Squarelink may require email validation via traditional validation methods. To generate the user's Master Key, their password is first hashed to a 256-bit output length using PBKDF2[8] with *n* iterations where *n* is a random number between 20000 and 200000. These iterations add processing time that is unnoticeable to the user but costly for hackers performing brute force attacks. A 256-bit, high-entropy, cryptographic salt is then generated on the client-side of the application. Together, the email, password hash, and salt are appended to one another and hashed again using PBKDF2 with *n* iterations, resulting in the user's 512-bit Master Key. To make use of asymmetric encryption[9] later, we use Elliptic Curve Cryptography (ECC)[10] to derive a public key from the user's Master Key. Along with the salt and *n*, this public key is then sent to Squarelink servers and stored relationally with the user's email. Users are then prompted to setup a recovery method where their password hash (Recovery Seed) is encrypted with their Recovery Key, a value derived from information only the user owns and stored on Squarelink servers as described later.


1
Email + Password
2
Set up Recovery Method
3
Access Wallet
• Email + password are hashed with cryptographic salt from Squarelink servers to create your Master Key
• Successfully completed recovery method creates recovery key
• Recovery key is used to encrypt master key which creates recovery seed

8 PBKDF2 – A popular and effective key derivation function or hashing algorithm.
9 Asymmetric Encryption - this allows for encryption via one key (public) and decryption via another (private) or vice versa.
10 Elliptic Curve Cryptography – A widely-used and open-sourced protocol for asymmetric encryption.

For users that desire even more security, an additional 256-bit Client Key may be randomly generated on any client application version of Squarelink. This Client Key is hashed along with the email, password, and first salt to generate the Master Key. In browser extensions, this key is stored via the browser's local storage. For mobile versions, we make use of the Android KeyStore and the iOS Keychain – both of which can be further protected by fingerprint authentication. We additionally provide tools to securely add the Client Key to other devices. This key never touches Squarelink servers, adding an additional layer of security. Users may upgrade to this feature after initial setup such that they generate a new Master Key and re-encrypt any account information with this new key. However, should the user lose their device, a user will be unable to recover their account.

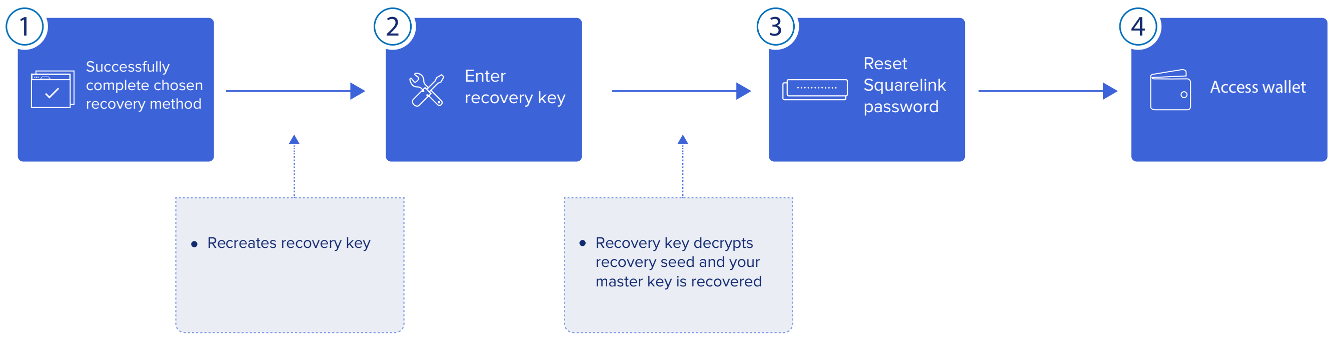
3.2 ACCOUNT ACCESS

Account access can be customized depending on the security desired by the user. Initially, a user is presented with a traditional email and password interface. The email entered is used to lookup the user's salt and the number of hash iterations n on Squarelink servers and return them back to the application via REST API. Users have the option to add a Second Factor (2FA) method to this process to further protect access to this salt. Upon receipt of the salt, the user's Master Key can be regenerated following the same steps as above.

At this point, any encrypted keys or data owned by the Master Key are secured on Squarelink servers. Although this information is encrypted, we still secure it until we can ensure the user has reproduced their original Master Key. A 128-bit value is randomly generated on our servers, encrypted via ECC with the public key corresponding to the user's Master Key, and sent along with the salt. The client-side application will attempt to decrypt it with the generated Master Key and pass it back to Squarelink's API. If the client application was successful, Squarelink then returns any encrypted data, which can then be decrypted on the client-side via the Master Key.

3.3 ACCOUNT RECOVERY

Should the user forget the password used to construct their Master Key, they can recover the information necessary to reconstruct their Master Key and reset their account. As mentioned earlier, the user's password hash is used as this Recovery Seed. Through various methods listed below, the user may decrypt this Recovery Seed, at which point their original Master Key is recovered, and they're able to set a new user-specified password. By using a password hash rather than the password itself, Squarelink ensures that the password can never be accessed in plaintext during recovery.


1
Successfully complete chosen recovery method
2
Enter recovery key
3
Reset Squarelink password
4
Access wallet
• Recreates recovery key
• Recovery key decrypts recovery seed and your master key is recovered

3.3.1. CHALLENGE QUESTION RECOVERY

The first method we introduce leverages challenge questions to facilitate account recovery. In this approach, a user selects three challenge questions to answer out of a Squarelink-vetted set of questions. Questions are chosen to minimize risk of phishing, answer discovery, brute-force reproduction, or guessing of answers. For instance, it might be trivial for an attacker to discover what Elementary School a person attended, so we avoid these questions. Their answers, along with their server-side salt, are hashed using PBKDF2 with n iterations resulting in their Recovery Key. Their password hash is then encrypted using AES and sent, along with the chosen questions and the public key derived from the Recovery Key, to Squarelink servers for secure storage. The answers themselves are never stored to ensure that neither Squarelink nor other parties can reproduce the Recovery Key.

To recover plaintext password hash, a user may simply enter their email to retrieve their salt and challenge questions. Once the user has entered their answers, Squarelink can then regenerate the Recovery Key. Although the password hash is encrypted, Squarelink further secures access by verifying that the user has generated the correct Recovery Key using the method described in Section 3.2 (using the public key derived from the Recovery Key). At this point, the encrypted password hash is returned to the user and can be decrypted by the Recovery Key. The user then follows the account reset process described earlier. For additional security, a Second Factor (2FA) may be required before the challenge questions and salt are provided to the user.

3.3.2. EMAIL/SMS RECOVERY

This recovery method makes novel use of digital certificates in email and SMS via S/MIME7 and PGP8 respectively. Through Squarelink, a user can provide the public key of the digital certificate for their email or SMS applications. Squarelink then drafts a message containing instructions for the user on how to reset their account along with their password hash in plaintext. This message is encrypted with the public key of their digital certificate and sent to Squarelink's servers for secure storage. The plaintext message never touches Squarelink servers.

To recover an account, a user must simply request that this encrypted message be sent to their chosen account. With the private key of the corresponding digital certificate, the user can view the message and follow instructions to enter their password hash into an input field using the Squarelink application. With their email address and original salt, their Master Key can be recovered and their account reset as described earlier.

3.3.3. U2F RECOVERY

Similar to the email/SMS method, we make use of public key cryptography via the public key of a U2F key owned by the user. The U2F key, when connected to the user's device, can be registered with Squarelink. The Squarelink application will encrypt the user's password hash with the public key of this U2F key and store it securely on Squarelink's servers.

A user may recover their account by simply connecting their U2F key to their device. The encrypted password hash is given as a challenge to the U2F key which can then decrypt and return the plaintext password hash as part of a U2F signature to Squarelink. Again, the Master Key may be recovered and the user's account reset.

11 Advanced Encryption Standard (AES) – a widely accepted standard for symmetric encryption.

3.3.4. DAISY-CHAINING

For increased security, any of these methods can be used in combination with one another via daisy-chaining. If two methods or more are desired, the encrypted output of one method would be used as the Recovery Seed in the next method. For instance, if a user desires to use both email and challenge questions, the user may first answer their challenge questions, encrypt their password hash with the resulting Recovery Key, and insert the result into a message to be encrypted by the digital certificate of the user's email account. For recovery, the respective recovery processes would be executed in reverse order of encryption.

3.4. ADDING KEYS

Squarelink becomes more and more useful as users add additional keys and/or data protected under the Master Key. Users may manually add previously-owned existing keys or have new keys generated for new blockchain applications. In either event, the key is simply encrypted with the user's master key using AES encryption and sent to Squarelink's servers for storage. Additional account data is handled similarly.

3.5. DATA TRANSPORT & STORAGE

We've engineered the Squarelink solution to withstand even the worst cyber-attacks. Contrary to custodial wallets and centralized services, in the unlikely event that an attacker breaches Squarelink servers, we still leave no central point of failure and no way to easily access any given account. Still, the security, availability, and permanence of account data stored on Squarelink servers is critical. We have created our data transport and storage practices with this in mind.

We ensure no third party can sniff data packets in transit by employing use of end-to-end encryption via HTTPS/TLS for all connections to our REST API. Additionally, data access is layered to ensure that a user only has the information needed at each step in the Master Key generation/recovery process to move further. For instance, a user cannot access their encrypted account data until they prove they have generated the correct Master Key.

The permanence of Squarelink account data is imperative as users rely on it to generate their keys. To ensure no data can be deleted, accidentally or otherwise, and still remain highly secure, we have chosen to build our data infrastructure on top of Hyperledger Fabric – an open source blockchain-for-enterprise solution developed and maintained by the Linux Foundation. Hyperledger Fabric enables highly-permissioned, private blockchain for storing encrypted account data. Squarelink has full control of the peers on the network and, with its distributed nature, we ensure no loss of data and zero down-time.

3.6. INTEGRATION TOOLS

For blockchain developers looking to quickly bootstrap their key management systems, strengthen the security of their custodial solution, or give access to users desiring to log in with Squarelink, we provide a suite of integration tools. Squarelink offers several client-side integration tools for web applications, Android apps, and iOS apps via JavaScript, Java, and Swift SDKs respectively. These client SDKs allow users to generate their Master Key via Squarelink and additionally allow integrated applications to request transactions or data.

12 Transport Layer Security (TLS) – This protocol allows for end-to-end encryption between server and client.

In order to fetch transactions, data, or user account information (once authorized by the user), we provide a public REST API with plans to build wrappers for many common languages and frameworks including Node.js, Python, Go, Java, and Swift. This adds a second layer of protection between users and applications and, with the use of application client-secret keys, allows Squarelink to monitor active users and application-specific consumption of our API.

Some developers may want more complete control over their user's private keys and/or branding. To accommodate this, we offer a custom SDK. Developers can build their own login forms (or use Squarelink-provided forms) and integrate our key-management technology directly into their site. In this instance, the application is responsible for its own security, as well as ensuring a user's keys never touch their servers. To make sure a client application's security doesn't compromise the user's Master Key, we completely separate their Squarelink account and create a new single-application account for each of these custom applications.

4.0 ROADMAP

PROOF OF CONCEPT APPLICATION USING SQUARELINK

Speedlink (https://speedlink.io) facilitates "one-click" registration for conferences and events. Users simply fill in their personal details and payment information for one event and reuse for other events in one click. All user data is encrypted by the user's Master Key which users generate via the web-version of Squarelink.

PRIVATE BETA FOR BLOCKCHAIN APPLICATION DEVELOPERS

We will soon offer restricted access to our developer portal and client SDKs for blockchain application integration. Supported languages include JavaScript, Java, and Swift. We will additionally provide platform-agnostic API documentation.

SQUARELINK FOR CHROME

We plan to develop the first client version of Squarelink on Chrome. Our Chrome Extension will be open-sourced for public validation.

OPEN SOURCE CLIENT SDKS AND SQUARELINK API WRAPPERS

Following a successful beta, we plan to open source our client SDKs for blockchain developers in addition to API wrappers for Node.js, Python, and Go.

SQUARELINK FOR ANDROID AND IOS

We plan to complete the foundation for the Squarelink ecosystem with open-sourced Android and iOS versions of Squarelink.

SQUARELINK FOR THE ENTERPRISE

Finally, we plan to enter the enterprise market, scaling our technology to offer businesses the security and innovation of blockchain without any extra burden to the "help desk" and no loss of user accessibility.

5.0 DISCUSSION

Data breaches, hacks, censorship, net neutrality, and numerous other concerns about the future of the Internet have dominated the news in recent years. It is no secret that our world is shifting toward a decentralized future, and we need to prepare for it. Several problems stand in the way of blockchain's mass adoption, but its steep learning curve and usability are among the largest. Custodial wallets provide a solution to this issue but effectively eliminate all advantages of decentralization.

Squarelink introduces a new standard for key management at the intersection of usability and decentralized security. Our breakthrough in private key recovery gives peace of mind to any person or company interacting with blockchain and makes wider adoption of blockchain technologies possible.

Squarelink is growing quickly, so if you have interest in joining the team or our private beta, please reach out to us at contact@squarelink.com