ksch Whitepaper

Introduction

The emergence of Bitcoin makes it possible to go to a decentralized currency system. After several years of development, it has been found that the block chain technology of Bitcoin has great potential and it can be widely used in all walks of life. In order to make better use of the block chain technology, there appears a number of application platforms represented by Ethereum, which encapsulate the underlying protocol and build the infrastructure to provide the developer with a more friendly and more flexible interface, making the developer to focus on the business logic and improve the development efficiency greatly. The ksch system introdeced in this paper is also a development platform for decentralized application, the following part will elaborate the characteristics, principles and application scenarios of ksch system

Overview

Decentralized Application

Decentralized application has following features:

1. The application must be completely open-sourced and run autonomously, which means it should not be tampered by any centralized organization,

institution or individual. It can be revised to respond to the market requirments, but to the revision must be recognized by all users.

2. The application data should be stored in a distributed network that has to be secure, open, and redundant in order to avoid data manipulation and single node failure.

3. Application users need tokens to access, and application contributors can get tokens as rewards.

4. The application must apply a value-proved cryptographic algorithm to generate tokens.

The centrality application can invest the development of the application by authorizing stakeholders of the system, thus providing the potential for self-sufficiency. These kinds of application also have the advantages such as publicity, security, and no need to trust. Therefore, it is not strange that decentralized applications will have promoting prospectin following fields: payment, data store, cloud computing, e-business, and etc, and it is even possible that the value created by those applications will exceed the market value of giant international company such as VISA, Dropbox, or Amazon.

About Sidechain

Blockchain is a well-ordered collection of data blocks generated by using

cryptographic algorithm. Each block of data contains an amount of network transaction information which is used to validate the authenticity, as well as generate the next block. To a normal user it looks like a public ledger on which every single transaction will be recorded, and to anapplication developer it looks like a distributed database, which features are decentralized, open, self-autonomy, and tamper-resistant. The blockchain presents a very tight relationship with decentralized application which makes it very suitable for storage.

As a specific blockchain, the sidechain applies a “SPV (simplified payment verification) Pegged” technology to achieve the goal that the user’s asset can be transferred from one blockchain to another, which means users can use their current assets to utilize the new cryptocurrency system. There will be no need to worry about the disadvantage of Bitcoin that is hard to adapt the innovation and meet the new request. What need to be done is just create a sidechain and link it to the mainchain of Bitcoin. By inheriting and reusing the powerful blockchain of Bitcoin, other issues such as liquidity shortage and market fluctuation, often occurred when a new currency is introduced, can be averted. Furthermore, since the sidechain is a isolate individual system, any severe failure will only impact the sidechain itself instead of the whole blockchain, thus significantly reduce the risk and cost of innovations.

About ksch **Platform**

ksch is a decentralized application platform, providing a series of SDK and API to help developers creating decentralized applications based on JavaScript and sidechain technology. By providing a whole set of industry standard solutions including customized sidechain, smart contracts, application hosting, and etc., ksch intends to offer an easy-to-use, fully functional and plug & play ecosystem, by which developers can rapidly iterate their JavaScript applications, and easily deploy them into application store built in the system. These application can be smoothly downloaded and executed by every node and user located in everywhere of this ecosystem. The whole process will be secured by the ksch sidechain consensus network.

On the other hand ksch itself is also a fully opened, decentralized application, which has its own cryptocurrency known as XAS.XAS is capable of interfacing with sidechain or any decentralized application (DAPP) through “two-way peg” mechanism so that it can be the bridge or media for the asset transition among all DAPPs, and to achieve that purpose, the XAS will be sold to the investors before the deployment of the system. Once the ksch system is published, it will not under the control of the core development team, but the stakeholders of the system and the owners of XAS.

Who Should Use ksch Platform

Apart from some basic services, ksch platform also provides technology and toolkit support, whose main target audiences are as follow:

Developers

Developers can create and deploy DAPPs in ksch platform by following the criteria of application development and business activities. The business model of these DAPPs may be free or fixed price, or pay with value added service. It is all up to the developers themselves.

Enterprises

The tools provided by ksch platform can be used to create a complete blockchain easily, and more importantly, this blockchain can be pegged into main chain of ksch platform or even the blockchain of Bitcoin, thereby connected with well-developed cryptocurrency. It is a very attractive feature that is suitable for small and medium enterprises (SMEs) especially start-up companies.

SMEs can utilize blockchain technology to provide the information and data that were enclosed inside companies or Internet in the past, and to connect with the system data owned by authority so that they canenhance the business transparency and improve the companies' image. Consequently more trustiness can be brought to the investors and financial institutions to make it easy to get the funds or contracts.

It is inevitable for SMEs to public their business information. SMEs found it is difficult to prevent social community to know their secrets. Therefore we

can assume that blockchain will play a vital role in SMEs everyday business activities.

Normal users

Normal users can download, install and use numbers of decentralized applications through ksch built-in app store, which is similar with application store in current smart phone. ksch system supports different kinds of decentralized application on which Normal users can make profit by contributing the program or content. It is possible for developers and users to create a vivid ecosystem together.

Design Philosophy

Turing-Completeness Scripting vs Sidechain

The scripting engine is a highlight in design of Bitcoin on which not only can transfer the currency but can realize the smart contracts such as multi-signature, escrow and arbitration, and gambling. Bitcoin's scripting system, however, applies a concise design setting a series of limitations due to the consideration of security and difficulty of implementation, for example, it cannot support loop, or must strict the length of its scripts, or only supports limited numbers of standard transactions.

The most attractive characteristic of Ethereum is that it significantly

expands the scale of functionality of this scripting engine. In detail, Ethereum adds several instructions such as reading blockchain, charging, jumping, etc., as well as releases the limitation of the invoke depth of stacks and functions, and length of the script. Ethereum claims that its scripting engine is Turing-completeness by which developers can carry out almost any computation that can be represented by mathematic model.

Although with Ethereum, expanding script became a prevail approach of decentralized application platform, it still has one disadvantage that cannot be ignored: the application itself and the data generated by it have to be stored in the same blockchain, which makes the blockchain grows extremely rapidly. Ethereum tried to slow this trend down by optimizing or compressing the program code and data, but still cannot solve the problem completely. Moreover, since those applications based on scripting have to share the same ledger, some parameters, i.e., created-time of blockchain, cannot be changed, which prevents developers from customizing the application.

Sidechain mechanism, in contrast, deals with scalability in another way. Every sidechain is running in a different distributed network node, having separate users, investors and development team. This partition-like solution has resolved the blockchain expansion problem, and has made every application has its own ledger, on which the consensus mechanism, block parameter, and transaction mode can be customized. For this reason, we believe sidechain

mechanism is a lower-cost, more flexible and easier to use solution than a Turing-completeness transaction scripting.

Account vs UTXO

Within Bitcoin and derived systems, there is no so-called “account” to save the user’s balance. The balance has to be dealt with by the transition of the whole transaction system’s status. Let’s introduce a new term, UTXO, standing for unspent transaction outputs. Every UTXO has its own nominal price and its owner and each transaction has multiple inputs and outputs. Each input includes a reference of current UTXO and a cryptographic signature generated by private key related to owner’s address. If a user own this private key, he/she can consume the value of this UTXO, in other words, the balance this user has is the sum of the currency value of all the UTXO related to the user’s private keys. The main advantage of UTXO is high level privacy, which means the user can generate a new address for each of his/her transaction so that he/she cannot be traced. It is a great thing for a currency involved in transaction, but to variety of DAPPs, it is not sure. Compare to UTXO, account has following benefits:

1. Space efficiency. For instance, it will take a user 300 bytes space when he/she has 5 UTXOs: (20+32+8) x 5 = 300 bytes (20 bytes for address, 32 bytes for transaction No., and 8 bytes for transaction amount). Meanwhile if using account, the storage space will be limited to 30 bytes: 20+8+2=30 bytes (in this case, 20 bytes for address, 8 bytes for balance, and 2 bytes for random number).

2. Easy to monitor. Due to the account, the cryptocurrencies are easy to recognized, since all we have to know is the account that the currency is coming from.

3. Simple, easy to code and understand.

4. Referring in a fixed time. A light client can access all the data in a user's account in a fixed time, while with UTXO system, the access time will change once there is a transaction.

ksch platform is not a pure currency system because it has to contain various applications. In general, account mechanism is a better option under this circumstance.

Relational Database vs Non-relational Database

At present most blockchain system choose to use lightweight non-relational databases, such as Berkeley DB, LevelDB, and so on. Usually these kinds of database support some simple data structures, like B-tree, hashtable, queue,etc. and they are not able to manipulate data through SQL. Generally non-relational database is good enough at the application of cryptocurrency system, but speaking of application platform, it is not the case, especially when we talk about finance, bank, e-commerce and so on. Within these fields, the relational database is wildly used because of its following advantages:

5. Transaction processing

6. Very low cost of data updating

7. Capability of complex query (using join, for example)

We decide to adopt SQLite, a high-performance lightweight embedded rational database. SQLite can support up to 2T data volume, and the data file can be shared smoothly among different byte order machines. The most attractive feature is SQL, which is very suitable for DAPP developers.

System Highlights

Ease of Use

Development Language

Developers can use JavaScript and numerous npm libraries to create their application. Unlike C++ and stack script for Bitcoin and new Solidity for Ethereum, JavaScript is more popular, has wilder audience, and is easier to master. Furthermore, by introducing relational database, ksch platform make decentralized application development more like coding traditional web application, hence it become the easiest way among all the current development approaches.

Toolkits

ksch system provides a command-line tool to rapidly create a sidechain on which any kind of applications can be created and deployed. Also the system provides a series of API, involving consensus mechanism, strong random number, database, and cryptography, to help users creating complex smart contract applications.

Deployment

To deploy the applications, all the developers need to do is committing their DAPPs onto Github, and registering in web wallet or light wallet. After that DAPPs will be presented in App store to download.

Flexibility

Developers can freely customize all parameters in their sidechain, such as generating speed of block, transaction type, and transaction fee, etc., even can carry out a new consensus mechanism, for example, developers can use the POS or POW to replace the default DPOS

Security

One well-known highlight of ksch system is a consensus algorithm based on improved DPOS, in detail, the system embedded a high-performance Byzantine fault tolerance algorithm upon original DPOS hence significantly reduce the possibility of network forks. As long as there is no more than 1/3

nodes which are attempting to attack together, the system will not fork, so that the risk of double payment will be avoided. Secondly, ksch system closely examined every detail with security consideration, such as passphrase hint based on BIP39 algorithm, second secret, and multi-signature, etc.

Technique Specification

Consensus Mechanism

The consensus mechanism used by ksch system is based on DPOS, a mechanism of delegates vote, but applied an optimized variant algorithm of PBFT, which can make loyal nodes reach an agreement, i.e., not fork, in O(n^2) message complexity and O(1) time complexity when t < n/3. Here “t” refers to the number of Byzantine node (on which any situation can happen, such as network delay, machine shutdown, or malicious attacks) and “n” refers to the number of all nodes.

Delegates vote

The delegates vote mechanism of ksch is similar with the one of DPOS, which is composed of 101 delegate nodes. A delegate is a trustful account voted by community, and 101 delegates who have the most votes are allowed to running the nodes which maintain the network and generate the blocks. Those accounts who cannot be the top 101 are called “candidates”, who can be delegate once they get enough votes to become the top 101.

Every ksch user has the right to vote up to 101 delegates, and the weights of vote is determined by the amount of XAS owned by this user.

Each voting round will generate 101 blocks, and the ranking change of each voting and delegates will be shown in the next round. The new block will be generated in 10 seconds after the previous one, and then be broadcasted into network and added in the blockchain. Once new block is added, the confirmation count of all transactions before will be increased 1 time. After gaining 6 confirmations, a transaction can be treated as “safe”. If the amount of transaction is small, less confirmation count is allowed. In other words, larger confirmation count is necessary when it is a large transaction to improve the security.

If some failures occurred in a few delegates, such as under attacks or shutdown, the block will miss, and this issue will be recorded. This problem has an impact on the online rate of this node hence the voting of community. Therefore the election of delegate should be treated solemnly, for example, a delegate needs to have experience on running a website, ability to maintain the nodes stable, so that the security and stability of the whole system will be strengthened.

Byzantine Fault Tolerance

The major difference between ksch system and DPOS is reflected in the

second part of the algorithm.

What the DPOS applied is, first, randomly sorting the delegate list of current round (to make sure the delegate order of each round will be different and the delegate of next round cannot be predicted), then let delegates create block in turn through a round-robin way. The main issue of this algorithm is that, if a delegate attempt to attack, he may broadcast many different blocks, which may contain double payment trades, to “fork” the whole network. It is no doubt that if only one delegate betrays, the fork will be solved by syncing with the next longest chain. However, it will take a much longer time to eliminate betrayed nodes as their amount is increasing. A few betrayed nodes are enough to have a serious impact on system security, which means even the transaction gain 6 confirmations, it still may not be safe.

To deal with this problem, we introduced an algorithm known as PBFT (Practical Byzantine Fault Tolerance). Same as DPOS, PBFT also chooses delegates in the way of round-robin. The different part is there will not be a block generated after delegates are voted. In fact, there will be a piece of propose issued to determine the hash code of next block. Only when more than 2/3 of all nodes agree with this propose, next block will be generated. The hash of next block must be the same as that agreed with all delegates of current round. Essentially, the introduction of PBFT solved the issue of abusing the delegates’ right, and improved the control capability of ledgers.

Sidechain and DAPP

ksch system provides a command tool by which developers can easily create a basic sidechain system. Developers are also able to fully customize their sidechain system, including its own database, consensus mechanism, trading mode and account architecture. Sidechain system can be hosted in the clusters of nodes of individual delegates, so that a partitioned mechanism can be naturally generated hence avoid the rapid expansion of the main blockchain.

Every DAPP is related to a sidechain. A sidechain's core code is developed by nodejs, but UI can be programmed by any of front-end techniques such as QT, HTML, or JavaScript. The communication between front-end and back-end is usually implemented under JSON RPC protocol. The developer or owner of a DAPP can trace the usage of their application. Compare to cryptocurrency that is based on community consensus, DAPPs are more like a private company. All the transactions within a DAPP are handled by main node, which is run by DAPP's owner. The owner of DAPP must have an ksch account, like a multi-signature account, whose major objective is to create a consensus of a new block and to sign it in DAPP's main node, just like multi-signature wallet. Once a new DAPP is created, and signed in main node, the SHA256 hash of this block need to be calculated, and submitted to the main blockchain of ksch system as a DAPP block by the owner of it. When ksch 's main blockchain received information of a trade including DAPP hash, the delegate will compare it with the previous one

and then store it into ksch blockchain. Then lately when main node is in sync with network, the user will validate all DAPP blocks through ksch blockchain, which means it is impossible to remove the previous DAPP block from main blockchain. Developers can replace ksch blockchain with Bitcoin blockchain to use the same functionality since the usage of ksch API will be the same. The security of DAPP can also be guaranteed by Bitcoin blockchain. Developers can take advantage of both XAS and BTC as the currency of their DAPP. In the case of money in-and-out when using a DAPP, the XAS or BTC will be delivered to a DAPP address, and appropriate amount of money will be shown in that DAPP's account, ready to use. The deposit mechanisms of both BTC and ksch are the same, which is to deliver to a particular DAPP address, and then the appropriate amount will be shown in the account. In every DAPP the account is created by the developer, so all deposited BTC or ksch coin will be saved in related address. Considering the security, only those DAPP accounts owned by trustable signers with multi-signature are recommended. A withdrawal from DAPP is main node's responsibility. When someone requests a withdrawal of money, DAPP main nodes will acquire the request and move the currency from DAPP address to ksch blockchain or Bitcoin blockchain. Developers are allowed to issue a token in their own DAPP, and to use it as the currency of this DAPP. These kinds of token can be circulated in DAPP like XAS or BTC, but they are not allowed to transfer from one DAPP to another directly but through ksch main blockchain.

Sandbox and VM

Sandbox is a execution environment that restricts program's activities under the security policy. In early time it was used to test suspicious software, for example hackers usually use sandbox technique to run some virus or unsafe program. The implementation of a classic sandbox is usually to intercept the system calls, to monitor program's activities, then control the usage of computer's resource such as disk I/O according to user's policy.

ksch system implemented sandbox mechanism by using VM module of nodejs. The VM module is an encapsulation of JavaScript V8 engine, which is used to run the pure JavaScript code. But it cannot use system APIs like file system or network communication. It also is lack of "require" function therefore the third-party libraries cannot be imported smoothly and even worse, the modular development cannot be conducted. This requires DAPP developers to use a "browserify" technique to bundle all the third-party libraries they used to a JavaScript file, so that ksch system can load it and run. Some necessary system level API will be provided to sidechain through IPC (Inter-Process Communication), which takes into account both security and functional completeness.

Transaction

ksch system has built a abstract transaction layer, on which almost all the functionalities of system core are established, such as transferring, voting, application store, deposit, and withdraw. The sidechain itself can also implement

its own type of transaction. The main difference between each transaction is their types and assets. The data structure of a basic transaction is as follow, and the extension of transaction will be saved in different asset table according to its type.

```
Transaction {
required VARCHAR(20)      id;
required VARCHAR(20)      blockId;
required TINYINT          type;
required INT              timstamp;
required VARCHAR(21)senderId;
optional VARCHAR(21)recpientId;
required BIGINT           amount;
required BIGINT           fee;
required BINARY(64)       signature;
optional BINARY(64)       signSignature;
optional TEXT             signatures;
required BINARY(32)       senderPublicKey;
}
```

Let us take a vote transaction as an example: the vote entity is connected to a basic transaction through transaction Id like follows:

```
Asset_Votes {
required VARCHAR(20)      transactionId;
optional TEXT             votes;
}
```

Account System

Every account in ksch system consists of one master secret, a pair of public/private key and an address. Users can also set their own second secret. Note that the difference with Bitcoin here is that each account is related to only one address, instead of multiple addresses and private keys related to one wallet in Bitcoin.

Master secret is a mnemonics used to produce real wallet in line with BIP39 standard. The mnemonics is much easier to remember for people than other binary or hexadecimal characteristics. The way in which a master secret is generated is to convert entropy on a multiple of 32bit into several words, particular in ksch system, the length of entropy is 128bit and it will be converted into 12 words. The master secret, as a top-level password, is maintained by users instead of published in public. Once users lose their

passphrase they would lose the ownership of their account. A master secret is generally like follows:

barely decline dust stamp protect color certain cup arena busy latin shell

A pair of keys includes public key and private key, which uses SHA256 hash of master secret as seed and is generated through ed25519 Edwards-curve Digital Signature Algorithm (EdDSA). It looks like this:

```
Public Key:
9989388b220a13465e49f52df5ba28ba08eb1e7a973320347f9687a107dc2f9
a

Private Key:
91e891f653e3ed0232d8c7de2e72b625d50d48593fc0fb570c0db25c5e4456
9a9989388b220a13465e49f52df5ba28ba08eb1e7a973320347f9687a107dc2f9a
```

The account address is a big number generated by reversely converting the first 8 bits of SHA256 hash of public key, shown as follows:

```
5034187504202890358
```

Client

ksch platform provides three types of clients:

The full version client is a perfect solution for those super users, delegates and developers, running on Windows, Mac OSX and Linux, but only Linux system can run delegate nodes on which blocks forging is enabled.

The light wallet client allows users to connect network through full version wallet or API, but it is necessary to open all the rights of this API by the owner of this full version wallet. A full version wallet can download the whole blockchain from other full version wallets through peer-to-peer network.

An ordinary user usually use light wallet to manage his/her ksch account. The light wallet can run on Windows and Mac OS platform without installation. It utilizes the system built-in browser. It needs to be noted that the light wallet cannot run as a network node since it would not download the blockchain data but only links with other nodes through HTTP protocol, which have several advantages: first, no blockchain data downloaded would keep the volume small; second, no private key broadcasted to network make it possible to make all types of trade since all the data will be signed locally. If you want to run a delegate node, you can use light wallet to register a delegate account, but cannot run delegate node through it to produce block. In order to run it, downloading and running a full version wallet in Linux would be necessary. The DAPP users can take advantage of light wallet to manage installed DAPP. The APIs of both DAPP and node are available for developer to invoke, so that it can make them rapidly and easily creating JavaScript DAPP through Node.js.

The mobile version client allows users to manipulate their own ksch accounts on their mobile devices, both iOS and Android platform. They can download the client from AppStore or Google Play. The backend of the client is based on our solution applied in our desktop version, but mobile client applies responsive UI instead to fit the mobile device screen and changes some interface mode according to the device characteristics. This mobile application brings mobile user-friendly interface, like Bitcoin or some major bank application, and of course, the mobile client can also support all the DAPP you like running inside of it.

Performance

All the information of a transaction usually occupies about 100 bytes. Let us calculate how much network bandwidth will it be demanded when system achieve 10,000 TPS. Since the interval of block production is 10 seconds, each block needs to carry 100,000 transactions, which make the volume become 10 MB. This 10 MB data needs to be broadcasted to whole network, so even under the best situation, 10 nodes will be achieved in the first step, and then 100 nodes in the next step, and each step is supposed to complete in 5 seconds, which makes the necessary bandwidth be 10MB x 10 /5 = 20 MB. Considering the bandwidth loss and unexpected circumstance, we think that bandwidth needs to be at least 40 MB to satisfy the 10,000 TPS throughput. Although this volume of bandwidth is not easy to gain, it can be sure that it will bring much more benefit

to delegates than the cost. In Nov 11, 2014, the peak throughput of Alipay payment system reached 85900 per second. There is still possibility for optimization of trade throughput in ksch system, which is our major investment in the future.

Scenarios

Token System

The “hello world” program generated by ksch toolkit would be a basic token system.

Developers may not need to write even a line of code but adjust some initial parameters in genesis.json, to publish a token system. Like tokens based on Ethereum, the token in ksch system can represent gold, stock, mortgage, or any other type of asset. These tokens can be traded with XAS transferred in sidechain in a decentralized way, and with any other currencies in a centralized exchange.

Mediatory Contract

Let us assume that a buyer wants to trade with a person he/she does not familiar with. Generally if the trade is well-doing, both would not hope the interference from a third-party. But if there is any issue occurred in the process, like the buyer does not satisfy with the items, they would like a mediator to help

solving the problem. The mediator will ask both sides to present some evidences and then make a decision, for example, ask seller making a refund. This business process can be described as follows:

1. Both the buyer and seller of the transaction choose a mediator together.

2. The buyer creates a 2-3 multi-signature account by using a public key shared with all the three parties. Then the buyer transfers the money into this account, then signs and publishes a transaction in which the buyer's account and seller's one are treated as promotor and recipient respectively. At this moment this transaction would not be confirmed immediately until two of these three people have signed.

3. The seller deliver the item to the buyer.

4. If the buyer received the item and satisfy with it, he/she will sign the transaction with his/her private key. After that when the seller sign the transaction again, it will be completed successfully.

5. If the buyer does not satisfy with the item, he/she can can appeal to the mediator and show evidences about the item with faults. Then the seller also can show evidences against buyer's one. Finally the mediator would make an agreement with one of the two sides of trade and sign this trade together to complete it.

Decentralized Exchange

There are two types of decentralization according to legal currency support. If legal tender is not supported, the full decentralization can be realized; otherwise only half decentralization would be supported. In this case, legal tender can circulate through gateway but transaction information would be made public.

Full decentralized exchange can still be divided into two types: one is peer-to-peer trade through “atomic cross chain exchange API”. The other one is pending order, which require seller transfer an amount of asset to ksch sidechain from other blockchain. This transferring should be achieved by the SPV proof of frozen assets of main chain. Beside, with the support of relational database, it is easy to create a well performance matching engine by using union-table query and index.

Existence Proof

Existence proof is used to register copyright, patent and so on, which is based on the idea that to prove an existed particular file by saving hash code of this file into ksch sidechain. In addition we can add metadata such as timestamps, owners’ digital signatures to prove when they own these files. This kind of information cannot be forged or tempered, also will not expose the privacy data and ready to verify at any time without athird-party authority.

IoT (Internet of Things)

There are numerous of online devices existed in IoT, which make it difficult to manage all the devices and identification of every nodes by a centralized institution. ksch sidechain is an excellent solution for it. Firstly, it can deal with trustiness among different nodes. It means that all devices connected to each other to be a distributed network, and trade between devices can be verified by consensus algorithm, and can be traced, audited, and analyzed. Secondly, different types of devices can be connected to different sidechains, and this natural partition mechanism that we mentioned above can prevent main ledger from explosive increment. Let us imagine that a vending machine in a sidechain based IoT can not only monitor and report its own stock, but also analyze historic trading data to select suppliers through bidding, and then complete payment automatically.

Conclusion

ksch system is a decentralized application platform, which is designed to lower the threshold for developers, such as using JavaScript as develop language, supporting relational database to save transaction data, and making DAPP development be similar with traditional Web application. It is sure that these characteristics are very attractive to developers and SMEs. The ecosystem of the whole platform cannot be improved until developers make a huge progress on productivity. Also, ksch platform is designed to be open for various fields, not

limited to some particular parts such as finance, file storage, or copyright proof. It provides underlying and abstract API which can be combined freely to create different types of applications. In consensus mechanism, ksch inherits and enhances DPOS algorithm, by which the possibility of forks and risk of double payments would be significantly reduced. Furthermore, ksch sidechain mode not only can mitigate the pressure of blockchain expansion, but also make DAPP more flexible and personalized. ksch system, as a proactive, low-cost one-stop application solution, will surely be the next generation incubator of decentralized applications.