Prakriti
An Advanced Blockchain Built with TRUST:
Teamwork, Reliability, Unbreakable, Sustainable and Trust.

Abstract: The Prakriti platform is a highly advanced blockchain which is designed for flexibility, scalability and ease of usage for users. It will be a trusted platform for collaborations among the DApp developers and providers as it gives them full control over managing their own applications and business profitability. The platform provides accounts, authentication, securities, fraud detection, monitoring, databases, asynchronous communication, and managed APIs. Various advanced features have been incorporated to allow for maximum income-generation for our users.

1. **Background**

We are in a world where new cutting-edge technologies are constantly being released and adopted. One of these new technologies was the blockchain with the introduction of Bitcoin in 2009. Since then the market cap of bitcoin has increased to an incredible $73B USD as of October 2017. This is quite a feat since majority mainstream technology companies haven’t adopted this technology yet.

However, with the rise of startup companies like Ethereum and Ripple, Blockchain has increased in popularity and is becoming more common. The key feature of the blockchain is that it is a ledger where every transaction is recorded. Transactions can never be executed without approval from the network and therefore it is much safer than other forms of network whereby the lack of any oversight can lead to fraud and so forth.

Moreover, as blockchain is a decentralized system with end-to-end encryption for all data; users can be rest assured that their data is always safe and only accessible by them.

2. **Problem**:

Blockchain is an excellent technology for decentralized, secured data network however not many users have the opportunity to use such technology. Most blockchains are difficult to use or navigate and set up barriers to entry for developers as they are required to purchase some form of crypto currency before they can start developing and testing their applications on the blockchain. Other current challenges are, not limited to:

i. Complicated to use and navigate
ii. Expensive for app developers to start with.
iii. Scalability – transactions per second are not up to par

3. **Introduction:**

Unified Capital Group (UCG) is excited to bring you the future of blockchain – Prakriti, an advanced blockchain build with TRUST – Teamwork, Reliability, Unbreakable, Sustainable and Trust.

Prakriti is a platform on which new DApp creators can bootstrap and onboard their apps on blockchain. The main intent of this platform is to scale well and help the DApp creators with creating, managing and operating their applications. This is to basically ensure that the DApp user will only have to think about writing codes and launching their DApps without worrying about the infrastructure, such as storage, processing etc.

For example, company X wants to build a messaging app P and they believe that their application will eventually scale up to have a cryptocurrency platform and hence want to have that as part of their products. They do not have to build up everything from scratch. X can now make use of Prakriti, through which they get the infrastructure to build, test and deploy their apps running on blockchain.

4. **What does Prakriti offer?**

Prakriti allows a developer/company to onboard into its platform. They are provided with an onboarding UI and after the UI form is filled and the agreements are accepted, Prakriti provides them an infrastructure with the Prakriti stack, which could be used for building a transaction feature or transaction based app.

In short, the developer is entitled to get an environment to

1. Build and test his app – through docker images
2. An environment to deploy, use block-chain based feature or app. This app can either be a mobile or system app.

The value add for the user lies in the details of what is being offered by the platform.

When the onboarding form is filled, the DApp users will be offered different value adding software that they may need to use as part of their stack. Based on the nature of their application, the DApp users may choose the services they deem fit for them.

5. **Onboard**

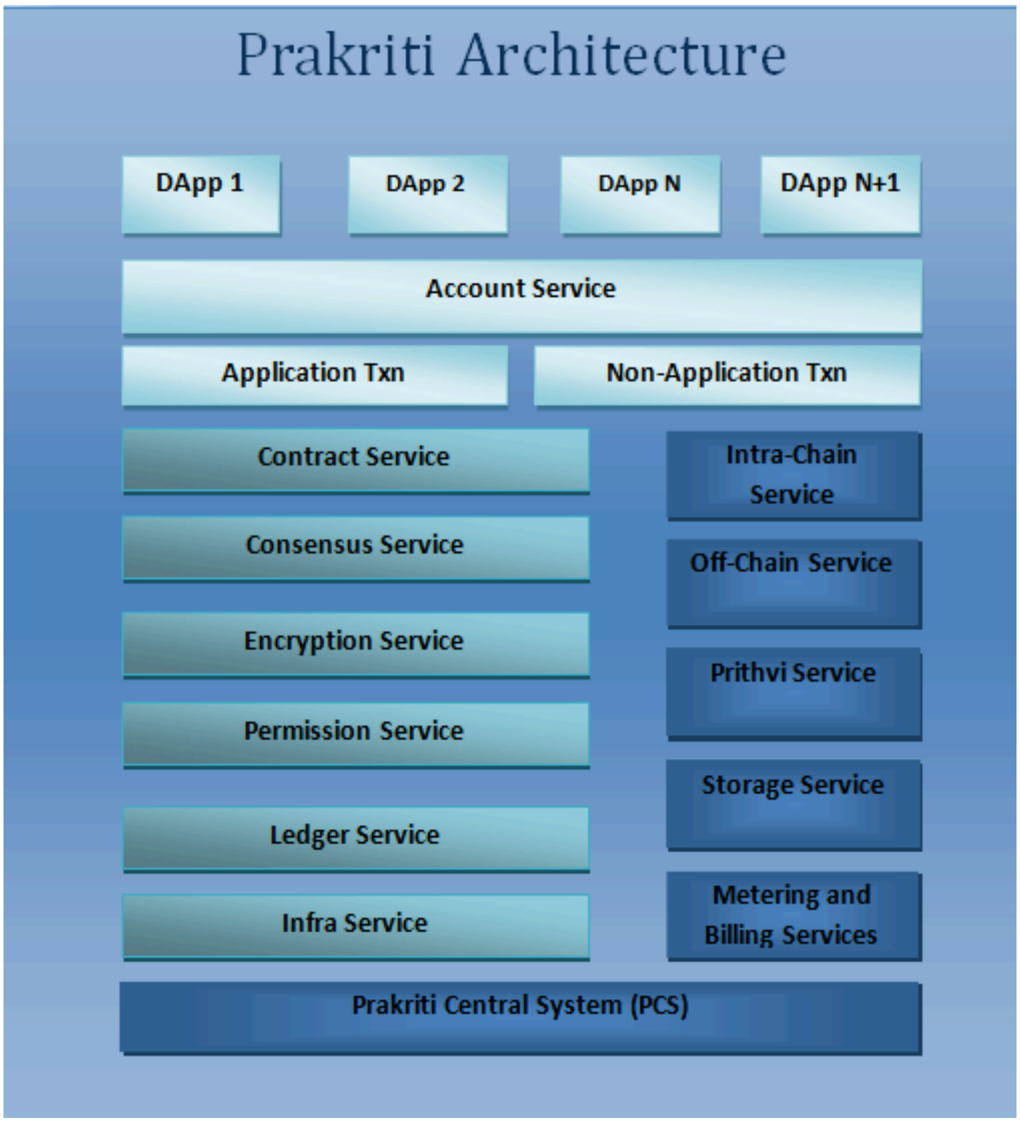
Once the developer has completed the form and onboarded into Prakriti with the app, the name and type of currency, the user can fill out two options. The user can either choose to have the environment 'Elastic' or 'On Request'. The architecture of the solution would be provided in the later part of the document.

a. **Elastic Environment**
When the transaction rate for the customer X increases and possibly starts hampering the performance, the system would be smart enough to increase or decrease the nodes at the right places.

b. **On Request Environment**
This option basically keeps the environment static and doesn't expand or contract the environment based on the traffic stats.

Users will also be given an option to move between option (a) and option (b) based on their requirements.

6. Architecture of Prakriti



Services in Prakriti Stack

7. Account Service

This service would have the following functionalities:

i. User Creation
ii. User Modification
iii. User Deletion
iv. User Authentication

The moment a DApp developer onboards to Prakriti, the developer will be provided a separate master key. The developer can use this master key to create a private and public key for the users who create a login. The users may login using their phone numbers or emails. The service would send an activation message through one of these mediums and upon confirmation, their login gets enabled.

Along with this, we have also introduced dual-factor authentication as an option. This will let user's login based on two authentications for extra security. It will be the DApp's choice to opt for it or not.

When the users are created, they may assume 2 roles. They may either be participants or/and vendors. A vendor is a role which has the ability to provide the hardware utilities to the ecosystem (DApp). For example, the user may provide unused HDD space, a whole storage space or a whole system etc.

8. Transaction Types

a. Application Transaction:

An application transaction is a type of transaction which may be a monetary or a non-monetary transaction such as book keeping of messages. This transaction would be a one of the core aspects of the DApp which would require the book keeping from the Prakriti's end. A typical transaction would involve two parties at a time.

For example:

If Person A wants to transfer 100 Units to Person B, the typical transaction proof or transaction receipt would look something like this

```
{
  txnId : "Unique_Number"
  txnType : "Application Transaction"
  from : "abcde-12345-fghij"
  to : "efghi-23456-uvwxy"
  txnTime : "12-12-2017:00:00:00UTC"
  units : "100"
  txnDetails : "appId:txnType:txnSubType:Functionality"
}
```

b. Non-Application Transaction:

A non-application transaction would generally be a non-monetary based transaction. It may be an instance like adding a validating node to the node service, or adding extra storage to storage service, removing the storage or node from the pool, etc.

```
{
  txnId : "Unique_Number"
  txnType : "Non-App Transaction"
  from : "abcde-12345-fghij"
  to : "efghi-23456-uvwxy"
  txnTime : "12-12-2017:00:00:00UTC"
  units : "100"
  txnDetails : "appId:nonApp:nodeTxn:AddNode"
}
```

c. Secured Transactions:

A transaction can be secured through our own defined way of transaction. Recently, there has been a frequent increase in cryptocurrency heists, which happen because, the senders do not know if they are sending to the right receivers or not. However, there are certain scenarios, where both the senders as well as the receivers maintain absolute transparency towards transactions. Thus, such users will be benefitted with Prakriti's secured transactions feature.

Assuming that User A looks for a crowd-funding through one of the money transferring DApps and assuming that user A expects 1000 tokens for funding.

Current Standard Transaction:

i. User A publishes his/her wallet address in the website
ii. Other users transfer tokens to User A's wallet

If User A's website was hacked and if it is replaced with a hacker's wallet address, the funders are easily tricked into sending their money to wrong addresses. This is how most crypto currency heists have been conducted so far.

In Prakriti Secured Transaction method:

i. User identifies or registers to the system with KYC
ii. Through this, the user gets a unique id to the system and that will be his or her unique KYC_Id that will help to initiate secured transactions
iii. User A can initiate a transaction request to the crowd funders. This is with the assumption that, there is a list of accounts which are white listed. Or User A must have a basic understanding that he is expecting funds from a list of public wallets.
iv. An authorization request goes to the funders, with option to authorize. This will have the information on who is requesting the transfer.
v. The funder can ignore, cancel or authorize the transaction.
vi. Once Authorize is clicked, the user can input the number of tokens he or she wants to transfer and then transact. This will ensure that neither the funder, nor the receiver loses the money.

9. Contract Service

Contract can be defined as a set of one or more transactions. These transactions can be either monetary or non-monetary based.

The contract service can be an event driven program, where the next event in contract gets executed based on the outcome of the previous event. For example, a customer can define a SMART contract with an event driven program such as the one below. Where a customer will pay a vendor X units of the currency per storage GB that is being used and assuming 90% of the time his storage unit was up every month.

Alternatively, a user as a vendor role, may own a validation node from the pool of nodes that will be used for consensus of a transaction, especially for PoW. In these cases, based on the uptime of the node and based on the amount of validation/verification made, the vendor can be paid in the local currency of the DApp.

A contract may have specific validators or witnesses being added and every transaction event will be trusted and approved by the witnesses, prior to adding it into the ledger.

Pseudocode

```
Until(contractExpires){
Var lastPayMentTime = billingInfo.getLastPaymentTime()
  If lastPayMentTime >= 30 days

 if vendor[0].getStorageUpTime() >= ((0.9) * (30))
     vendor[0].payUnits(getStorageUsedInGB() * 5)
  billingInfo.setLastPayMentTime(currentTime)
     }
```

These contracts are generally time sensitive and event based.

10. Consensus Service

A consensus could be arrived at using traditional block chain methods such as PoW or PoS. Prakriti also introduces one more way to validate a transaction. It is called Witness of System (WoS). This is one of the unique features that will be provided by our consensus services.

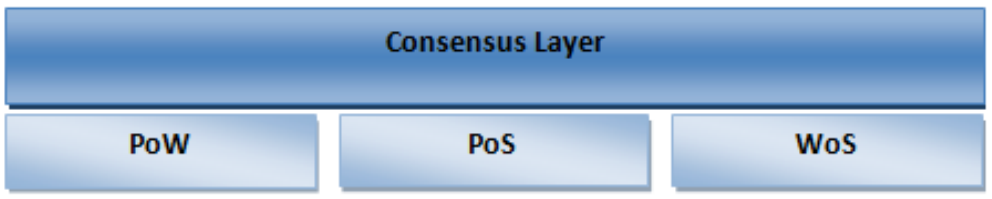
Though PoW is considered as a traditional consensus algorithm to use, it has two innate disadvantages

a. Energy consumption is high
b. High latency towards approval

Whereas, PoS believes in the richest account holder in the system to have a transaction trusted – with the belief that the largest stakeholders would be less greedy, which may not be the best factor to trust in all the events.

In Prakriti, we believe that a user who is part of the system is not an imposter or hacker, as we make sufficient security checks at the right places. Hence any transactions would be approved by Prakriti directly and this ensures that there is no double spending problem. This basically ensures that the transaction made updates the ledger directly. This is what we call as WoS, where the system witnesses and approves the transaction enabling security and efficiency.

Every time a transaction goes smoothly between two parties and when the users have good feedback, are trusted and dedicated to the system and other users they get Karmic points which the system keeps track of. These points may be useful when the DApp wants to build some privileged service for the users who are reliable, trustworthy and dedicated to the system.



One use case of where these Karma points could be used is – when there is a contract, such as buying an asset or property, where we need to ensure that we need to have the contract with users with good karma points and to be added as witnesses for approval. Ideally, our WoS would ensure that the transactions are approved by the system. However, if the user would want to introduce witnesses to validate the transaction and if they require an external trust – they can do that. This would add the witnesses into the deal and their approval would be added to the transaction info updated in the ledger. The system would still be responsible for updating the ledger. The only difference is it has the witness information.

The intention of this design is to ensure that the user must have absolute power over the system and he should be able to make choices for himself, rather than system making it for him.

11. Encryption Service

The transaction approved from consensus layer can be encrypted using a user defined encryption service. If needed, this encryption service would require a separate KMS (Key Management System) hosted in it and it would use separate algorithms for encryption. The encryption layer would also choose to have a pluggable quantum-proof or non-quantum-proof encryption services.

These encryptions would happen based on the encryption policy created by the DApp. For example, a monetary transaction and account information would require strong encryption; whereas, a network transaction info would be slightly lesser in criticality. So, a DApp administrator can define these policies in the Encryption Services, so that the right services are chosen during the right transactions and then it enters into the permission services.

The DApp may also choose to store any info without encryption and this layer can be bypassed if needed.

12. Permission and Ledger Services

The transaction is considered completed/approved when the following actions happen:

a. System approves the transaction and makes an entry to the ledger.
b. If the user prefers explicit witnesses, the system would write the witness info as part of the transaction into the ledger.

Ledger update is completely taken care of by the WoS process or threads, which is the only Witness users who have permission to write from the ledger. All the other users will have read permissions based on their accounts.

The ledger service will be used and will be updated by the system threads. They will have separate private and public keys and separate authentications to the ledger service. This is to ensure that a user on their own, with their own credentials and access, cannot influence the ledger services.

13. Infra Service

Infra management service is the lowermost and the most important service of Prakriti. When a DApp wants to get onboard, the DApp can provide a basic estimate on what traffic they expect with their App. However, it would be more of an educational guess and not a perfect number. With the DApp admins providing a traffic estimate, the infra management layer would basically check if the Admins have opted for either:

a. Elastic Environment Service, or
b. On Request Environment Service

Based on the customer requirement, the infra detection will provide an estimated cost of running an infra structure monthly, the minimum and maximum amount may have to be paid by the developer based on the usage. The layer will also provide information in detail on how the stack will be deployed and with what type of instances.

Once the customer/DApp developer is comfortable with the proposal, he or she may go ahead with the deployment and the infra layer would deploy an infrastructure, specific for the customer. The only layer which will know some information on Infra Services is Prakriti Central System or PCS. This will be explained in detailed as and when we move closer to it.

Infra Service functions are, not limited to:

a. Predicting an infrastructure that is optimal for a customer and estimating billing charges.
b. Deploying an environment for the customer with all the mandatory services.
c. During onboarding, the customer would also be able to suggest some of the value-added services, which could be installed as part of the deployment.
d. If the customer opted for an elastic deployment, the Monitoring and Billing Service will keep monitoring the traffic and it will add/delete/manage nodes based on the traffic that is currently hitting Prakriti.
e. The system will also provide specific monitoring services along with monitoring technical and business metrics that are keys for the blockchain. The customers will have business metrics from the platform, such as, not limited to:
    i. Total Revenue of DApp / Day
    ii. Total Active Users
    iii. Total Profit / Total Loss
    iv. Total Operation Cost of the DApp in the blockchain environment.
    v. Total Transactions Completed
    vi. Max Peak Transaction
    vii. Mean transactions / min
    viii. Per transaction cost to the DApp
    ix. Revenue per transaction for the DApp

The bottom-line is the Infra management service which can create, manage and delete environments to help the DApp running their App on top of Prakriti.

14. Off-Chain Service

The Infra management service would also communicate very closely with other managed APIs – such as Off-chain services, Prithvi service, Storage services, etc. Some of the Off-chain services that could be provided are something like an interface to 3rd party services or DB services or to any external data sources.

For example – if there is a smart contract defined between two parties A & B and if checking the credit score of party A is mandatory as part of the contract, then a DApp developer can include a service added in off-chain service, which would make a call to get the credit score and make decisions.

15. Prithvi Service

The Prithvi service is an optional service that a customer may use. Say for example – if DApp users run an App where Data security is one of the key focuses or goals, we provide an environment which will allow the DApp users to assume a role known as vendor. These users can provide a part of their mobile/gadget storage for Prithvi, which in turn will be used for decentralized storage. The advantage of such an approach is

a. Data Security – A hack/break into a system will not let a person lose all data
b. Users can play much superior role than just being users and can contribute to the system and get benefits based on the contribution made.

Prithvi, in short will make a user a vendor, who can provide storage as well as processing abilities in the future, in return for tokens or fiat currency. We use IPFS (Inter Planetary File System), which internally uses bit torrent/P2P protocol for file storage and distribution. With users lending some % of their computing processing abilities, we try promoting multiple edges for processing on needful scenarios, which provides a win-win situation to the users and vendors. However, as IPFS and CPU sharing has not started or matured inside a mobile environment, we are starting this with computers and laptops first and then going on to the mobile environment in the future.

More Info on Prithvi and its APIs will be published separately.

16. Storage Service

The difference between Storage Service and Prithvi Service is – Storage service provides APIs to centralized/distributed storage environment – such as Amazon S3 – where as Prithvi APIs help towards storing in a decentralized environment. DApp users may make a choice of storing what at which environment based on their needs and requirements. Moreover, Prakriti will have its own cloud storage keeping security and performance in mind – to which a user can provide their HDD and get benefits in monetary and karmic point form.

The storage service would also be used to store/archive the transaction information (previous blocks) and use it when needed.

17. Monitoring Service

This service would be run on top of all the services and it will basically measure the performance and resource consumption stats. We will be using the cloud/infra monitoring APIs and write a service which will measure a DApp environment's performance and consumption. This monitoring service has an option to trigger an emergency support from the Prakriti team – if they feel that the Prakriti is not behaving the way it should normally do.

That way, we ensure that we support our customers on a need and emergency basis.
Some of the metrics that Prakriti monitored are:

a. Any spike in CPU usage beyond certain levels
b. If lot of 5XX error comes up in the load balancer of Prakriti
c. If the key business metrics aren't moving anywhere for a defined time, say 15 mins. For example – if number of transactions are 0 for the past 30 mins.

This monitoring will render an alert which would be handled by the respond team.

Furthermore, this monitoring system will send a message regarding the traffic in DApp to the Prakriti Central System (PCS). PCS will observe the pattern based on these messages and decide whether to add/delete or modify the nodes or take any corrective actions to rebalance the system and make it optimal. More information about PCS is explained as we go towards the end.

18. Metering and Billing Services

Pay-As-You-Use is the crux of our metering service. The customers pay based on how much they have used the services. Furthermore, this metering and billing policies will change based on the nature of the App the customer is onboard.

For instance – if the customer deploys a DApp, then the billing policy will have the following actions:

1. The DApp developer would be charged to cover the infrastructure cost
2. The metering and billing service will analyze what is the allowed number of requests that a DApp can make per day and how much would he would be charged after crossing the daily limit.

19. Intra-Chain Service

This service would ensure that, if needed, it allows DApp developers to communicate or form partnerships with one another within the Prakriti system.

For example, DApp-A and B are side chains on Prakriti. If there is a business need where these Apps need to partner, Prakriti will allow them to unite with less hassle. In this case, the DApp1, owning sidechain-1, can share access with DApp2 owning sidechain-2. DApp2 can use the access credentials shared by DApp1 and use the system. There will be a defined restriction created by DApp2 here, based on their business agreement.

20. Prakriti Central System (PCS)

PCS is the central layer which will monitor and govern the happenings inside Prakriti. It also works very closely with Infra Management Layer. PCS keeps track of:

a. Application transactions
b. Non-application transactions

These transactions are monitored on a per customer basis and will be used for further analytics and data processing to improvise customer experience and usability.

Moreover, when an environment needs to be elastic, the infra management APIs are effectively used by PCS. For example, if DApp1 uses 5 nodes and their traffic starts surging, then the monitoring system would send an alert to PCS, specifying that a traffic surge is detected in DApp1's environment.

Checking the traffic surge pattern, PCS will decide how much more nodes are to be brought up to normalize the traffic. Similarly, when the traffic is less, PCS will cut down unnecessary nodes from DApp's ecosystem to save cost. The point here is – the monitoring system would report to PCS once in a minute regarding traffic updates and PCS will make decisions based on it.

21. Ease of Development

We will also release Prakriti solution as a docker image to benefit the DApp developers by creating a docker environment for development. The reason for this approach is to eradicate an environmental cost for test and dev phase. We will have a docker script bundled with an image and config file. This config file would have environment details, such as how many containers would be launched for what services. After a user configures this file, he can run the script which will bring up the whole stack of Prakriti locally or remotely in any host.

The users can start developing their DApp and have them pointing against this Docker environment, until they are production ready. This will help them to

a. Develop the App
b. Test – unit and integration test

22. Securities of Prakriti

Prakriti is designed to be a reliable, trusted, and highly secured blockchain. Below are current securities measurements, not limited to:

i. Users would have a highly secure private and public key, which would be needed to enter the system and make a transaction.
ii. The public and private keys of users would be protected inside KMS through symmetric keys. The very first effort that we would do is to encrypt these keys with quantum proof encryption as and when needed, to avoid easy breakage of keys.
iii. Frequent rotation of keys.
iv. An optional encryption layer – which a DApp owner needs to choose. This encryption layer would encrypt the messages that will be floating inside the system and store them in the blocks in the ledger in an encrypted manner.
v. Users can initiate transactions. However, it will be the system in the end which will update the ledger and not the users. The system will have its own private and public key generated frequently and will be rotated on a frequent basis. Using this authentication – the system would update the ledgers.
vi. Building a fraud detection and prevention pipeline.
vii. Build a user anomaly detection pipeline to ensure that, a fraud doesn't hog the system with DoS attacks. This could be found using anomaly detection algorithms, which can find the outliers from genuine transactions and take corrective measures. For example – if a rogue user had decided to hog the system – he somehow entered the system with two accounts and starts sending messages between two of his accounts infinitely. This anomaly detection should be able to find out that this is a fake conversation and should be able to block these accounts – until the user verifies them much more clearly.
viii. Tracking abnormal transactions and taking corrective measures. For example, if a user who is staying in US suddenly transacts from South Sudan, then there is a chance that he has been hacked. So, such transactions would be tracked and a corrective response would be given.
ix. During the onboarding of a DApp, the developer would be given a choice to install IDS and IPS in the gateway of APIs. This will be absolutely the DApp writer's choice to use it or not and he is responsible for the choice.
x. Applying preventive solutions for anti DDoS attacks.