Pistol – decentralized data access

Pistol is a decentralized data store where every peer has the ability to store and deliver data or share resources with other peers in a secure trust-less manner. This allows the user to own and control their own data instead of some centralized service provider.

You will run applications that use the shared resources. You are your own facebook, twitter, YouTube...

Pistol also keeps track of who has requested data and provides updates when the data changes.

In order to incentivize use, there are several revenue streams for not just the creators, but also for the users.

* Fees for transferring data
* Fees for access to data and services
* Income from the sale of products and services over the network
* Fees to setup network nodes
* Staking in network nodes

Nodes on the network are able to set their own fee structure. User peers on the network would discover paths on the network based on latency and cost and can set parameters to balance the two.

The network would have its own native network token which can be exchanged for other crypto tokens.

Tokens are introduced to the network through a proof-of-participation mechanism. *** Finish here ***

Why create something new, why not use Blockchain?

Blockchain was invented to solve two problems. The problem of trust-less transactions and the double spend. Obviously, it handles both these problems brilliantly. However, blockchain was not created to store data. While blockchains can be used to store data, it adds a lot of overhead and bloat that is unnecessary and leads to problems.

Blockchain nodes must store a record of every transaction. Unless they have every transaction, there is no way to calculate the Merkel proofs needed to verify transactions. So, if Alice stores a value, then updates that value, then updates again, all three transactions, plus all of the overhead will be stored permanently on every node. Instead, pistol stores just the updated value.

Blockchains would delay transactions for storing data due to the time to reach consensus. If I want to store a piece of data, it would have to go through consensus first before it could be stored and read by others. While it is true that some blockchains use a fast consensus mechanism, they still can not scale to millions of transactions per second.

What about the blockchain storage solutions?

Blockchain storage solutions work not by storing the file on the blockchain, but by storing the file someplace else and storing a hash proof for that file on the blockchain. The blockchain becomes a place to store receipts for storing the actual file.

This works fine for file storage where transactions are low, but fails as above when you have giga-bytes to tera-bytes of data transactions per second with updates.

How does Pistol store data?

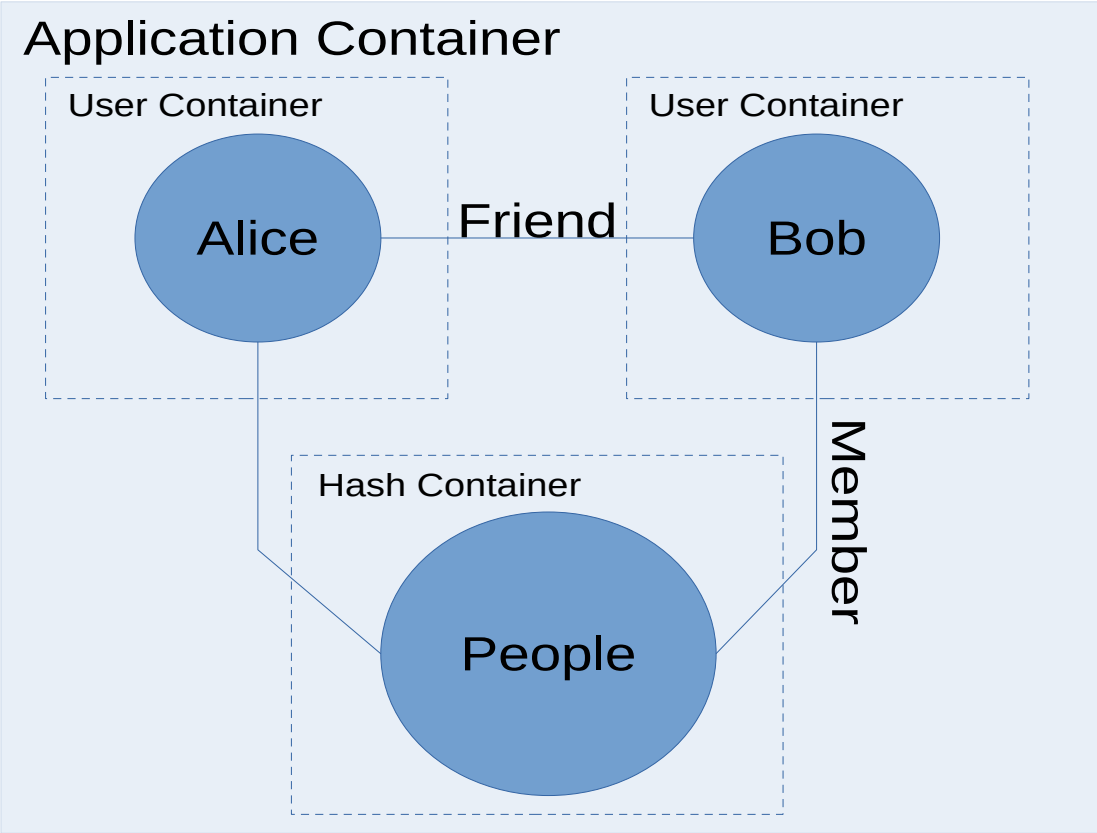
Pistol is a graph data structure. Applications will define sub-graphs within the graph using "containers". A container defines the rules for how to secure, update and read data in the sub-graph governed by that container.

This makes the system infinitely flexible since new container types can be defined by an application at any time. Containers have two sections, the ACL (Access Control List) and a data section. Below is a list of some of the "types" of containers. In reality the difference will just be entries in the ACL for that container.

The different default container types are:

User Container	A user is the only one who can read or write to this type of container. Different users will have their own user containers. A user can have multiple user containers.
Application Container	This is a container owned by an application owner which they can control.
Hash Container	This is a write only container
Public Container	This is a container where anyone can read and write. (probably rarely used)`
Group Container	This is a container that will have rules for a group of users

Since Pistol is a graph structure, links between containers are necessary to write any useable application. Here is an example:



Links are two way. So, Alice and Bob are members of People and People contains Alice and Bob. Users can also be separate from all applications as well as links between nodes within application containers.

How does Pistol handle a large volume of data?

Pistol is a unique data model. The larger it grows, the better it scales. Since peers store data, the data storage capacity is the capacity of every peer attached to the network. If more storage is needed, there will be financial incentives to add peers to the network, the higher the demand, the higher the possibilities to earn.

How will Pistol handle network traffic?

Pistol uses a custom network routing protocol. Each node on the network will advertise its resources which will be added to a radix based routing table. Radix makes lookups O(k) where k equals the longest path in the graph.

Routing is modelled on CIDR routing in IP based networks. If every router needed to know every resource, they would have very large routing trees. Pistol routers in the network will have rules to summarize routes that are remote to them. Routing information will get more specific closer to where that resource exists.

What about security?

Users can store their data using several different security models.

Who can access?			Method			
Owner	Group	World	Encryption	Signature	Group Cert	Hash
R/W			Yes	Yes		
R/W	R		Yes	Yes	Yes (RO)	
R/W	R/W		Yes	Yes	Yes (RW)	
R/W		R		Yes		
R/W		R/W				
R	R	R				Yes
R	R				Yes (RO)	Yes
R			Yes			Yes

How do you have a public user repository?

Users can have their own sub-graphs of nodes using a User Container. Only that user can update a value within their own area of the graph through the use of a cryptographic signature accompanying updates which is verified with the public key of the user.

Logins are done using a combination of cryptographic verification and encryption.

Nodes in the network can be of one of three categories and may be combined:

* Dedicated storage node
* Routing node

* Caching node
* Service node
* Computational node
* Client application

Dedicated Storage Nodes

A dedicated storage node (DSN) will store data for participants in the network including individual and institutional data to increase reliability and allow higher bandwidth and throughput rates. Data could even be moved around based on demand.

Routing Nodes

A routing node (RN) will route traffic around the network using various algorithms to ensure reliability, timeliness and elimination of duplicate traffic where possible. Free-riders will be eliminated through the use of a Merkel Proof to ensure that routing nodes do not claim data they did not actually get from another node.

Caching Storage Node

A caching node (CSN) is similar to a DSN except it will only store data for a short period of time where it is used most. CSNs can move data access to the boundaries of the network to ensure faster delivery of frequently accessed data.

Service Node

A service node (SN) can connect the Pistol network to various services. The service nodes will be addressable at a location just like data (ex. ‘app.some.service.data’) and return data to the application just like a regular DSN, however data will be retrieved from some outside service.

Computational Nodes

A computational node (CN) is similar to a SN, however, its main task is to provide a computational service rather than a network service. Examples of a CN could be scientific calculations or AI.

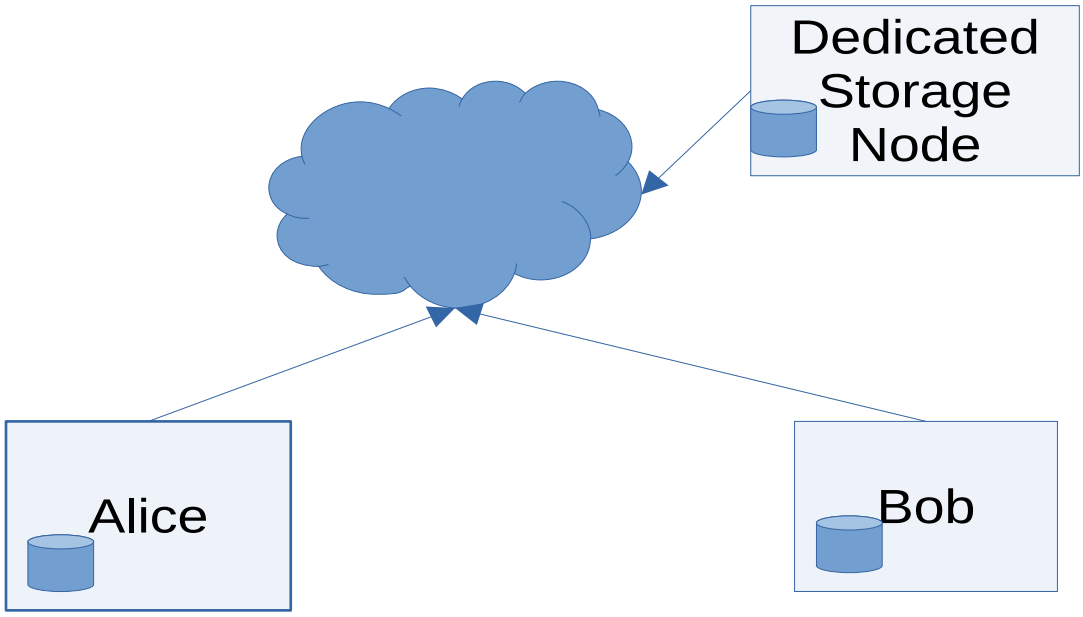
How does Pistol handle concurrent updates?

Concurrent updates are handled through a use of CRDT (Conflict-free replicated datatype). Each update is accompanied by a state clock. Since the rules on all nodes are the same, they will all resolve to the same value, even in the case of a conflict.

If a bad actor tries to game the system by passing in a state that is outside of the acceptable range, that update will be held until such a time as it can be processed. This eliminates the incentive to lie since, unlike a time based update system, setting your update to a future time does not mean that you always win.

Peers are nodes

Unlike blockchain based solutions and other peer-to-peer data stores, applications are also peers in the network that can provide data. For example, if Alice creates a post, it will be written to Alices local DB. It can then be shared with a dedicated storage node if Alice and the node agree. When Bob wants that piece of data, he can retrieve it directly from Alice or a dedicated storage node depending on which is online and closer at the time.



ERC-20 Tokenomics

Pistol development will be funded by the sale of a ERC-20 token. The following table breaks down the initial allocations for the token.

50%	Protocol emissions
25%	For the development team
20%	For investors
5%	For users of the platform

Tokens will be distributed in accordance to the following vesting schedule.

** NEED SCHEDULE HERE **

Network Token

The pistol network itself will also have a token. Tokens will be used to control spam and prevent sybil attacks. Tokens will be introduced through a proof-of-participation model. Good actors on the network will be rewarded with tokens that they can use on the network. Bad behaviour on the network will result in slashing of a nodes tokens.

Initially this will be achieved through the use of a DAO model of trusted nodes (nodes that have shown to be trustworthy and have staked ERC-20 tokens. The DAO nodes will also be rewarded for their participation in governance of the network.