



Dash Evolution

DAPI: Decentralized Application Programming Interface

Overview

Rev 1

Evan Duffield - evan@dash.org

NOTE: All Dash Evolution documentation, designs and source are in the research & development phase and subject to change. To access all materials, follow the development, or contribute, visit the [Dash Evolution Github \(https://github.com/evan82/dash/\)](https://github.com/evan82/dash/)



Contents

- [1 Introduction](#)
- [2 Separation of Network Infrastructure and End-user Network](#)
- [3 DAPI Architecture](#)
 - [3.1 First Tier - Bitcoin Core](#)
 - [P2P Network Topology](#)
 - [3.2 Second Tier - Masternode Network](#)
 - [P2P Network Topology](#)
 - [Masternode Quorums](#)
 - [Masternode Quorum Selection](#)
 - [Wallet Integration](#)
 - [3.3 Third Tier - Light Devices](#)
- [4 DAPI Diagrams](#)
 - [4.1 DAPI Components](#)
 - [4.2 DAPI Communication](#)
- [5 Security](#)
- [6 Scalability](#)
- [7 Quorum Group Design](#)
- [8 Stable Network Structure](#)
- [9 Access Design](#)

1 Introduction

Using Satoshi-style decentralized networks, users have limited options to consider when using a network. Such options include running a full node, which can be cost prohibitive, difficult to setup/maintain, or light clients that historically have been serviced through centralized SPV installations. When scaling a decentralized network, administrators of the network desire to keep as much decentralization and scalability as possible.

We propose a network topology that acts with the efficiency and speed of centralized services, but is in fact entirely decentralized and controlled by the network's proof-of-work. No one can control the topology of the network at any given time, therefore we can perform secure tasks such as reading/writing user data and serving data to users.



2 Separation of Network Infrastructure and End-user Network

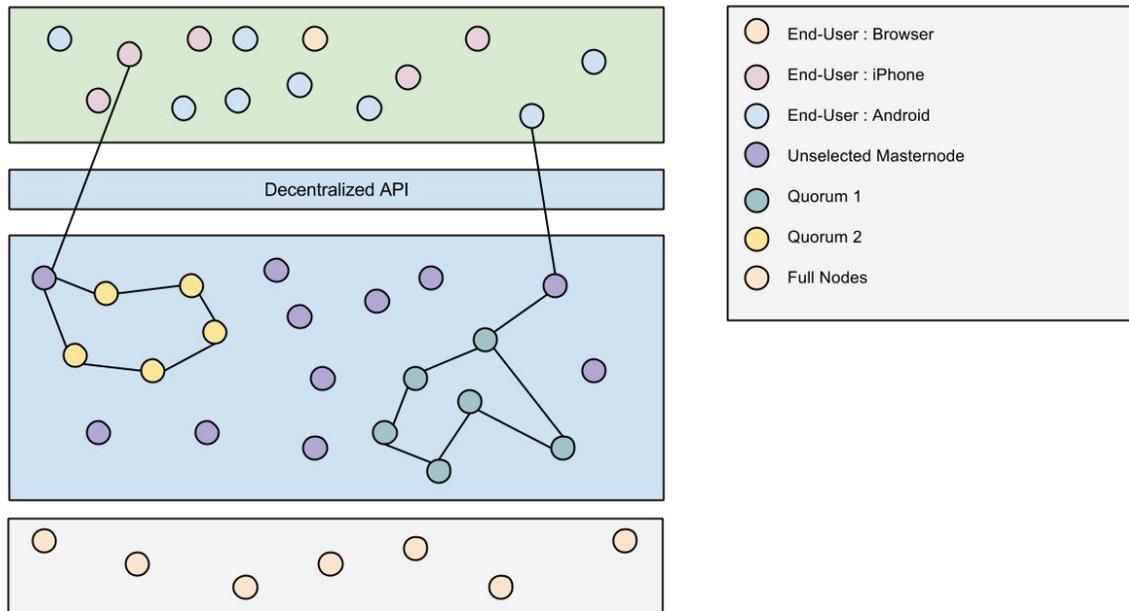
The highest form of security one can have is by running a full-node, which fully validates everything that happens on the network. You can then “trust” what that daemon will tell you and utilize network services through it.

This has a few drawbacks. It’s expensive to run a full-node and will continue to get more and more expensive due to space and bandwidth requirements. As the network grows, so do the requirements for running a full-node.

Another option is Simplified Payment Verification (SPV), which validates from a specific point in the blockchain. With SPV, users connect to a centralized server, which gives them the current longest chain to validate transactions against. SPV security is very good, but the design is centralized and requires trust.

We work around this issue by introducing the first ever decentralized API (DAPI). By utilizing an open network design and Masternode quorums, we can give users a completely secure experience without any of the complexity or cost associated with existing solutions. When accessing the network, you will use seven random Masternodes that will do the work you request in parallel. If the results match, all seven will sign and return you the result. Your client will know the public keys of each of these Masternodes and which Masternodes were elected to do the work, so verification is simple and effective.

3-Layer Network Design



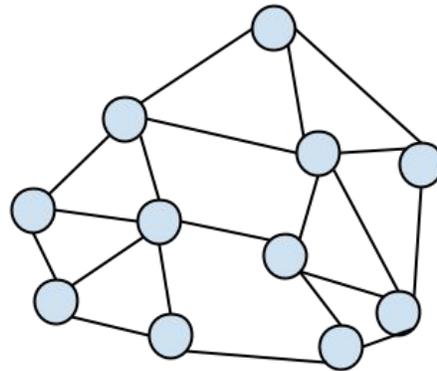
By splitting out different sections of the network, we have separated the main network from our end-user network. They will communicate through a static API that changes very little over time. We can then leverage this to continue to develop new systems and strategies without worrying about reverse compatibility of the network itself.

We can update the core network as often as needed without disturbing our end-user ecosystem. This will allow us to provide a high level of service, with higher security and lower cost to our users. End-user APIs will use a permanent versioning system for reverse compatibility, example `/dapi/v1/commands`, then after a new update we will introduce `/dapi/v2/commands`. This allows us to keep reverse compatibility with the end-users forever and allow developers more time when updating to a more recent API.

3 DAPI Architecture

3.1 First Tier - Bitcoin Core

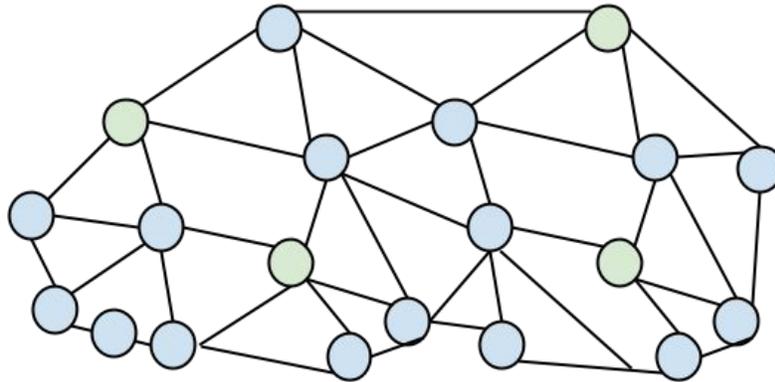
P2P Network Topology



All technology we build exists on top of a simple p2p decentralized network implementation. This framework allows us to build our decentralized service on top of a decentralized currency. Bitcoin provides all of the complex groundwork for making a cryptographic currency tick, like managing the proof-of-work, approving cryptographic signatures, running complex transaction scripts, approving blocks, and so on. Used by itself, Dash is inherently complex. But by adding layers on top of this framework, we can reduce the currency's complexity so that it's easy to understand and use while still being robust enough to support the an enormous user-base.

3.2 Second Tier - Masternode Network

P2P Network Topology

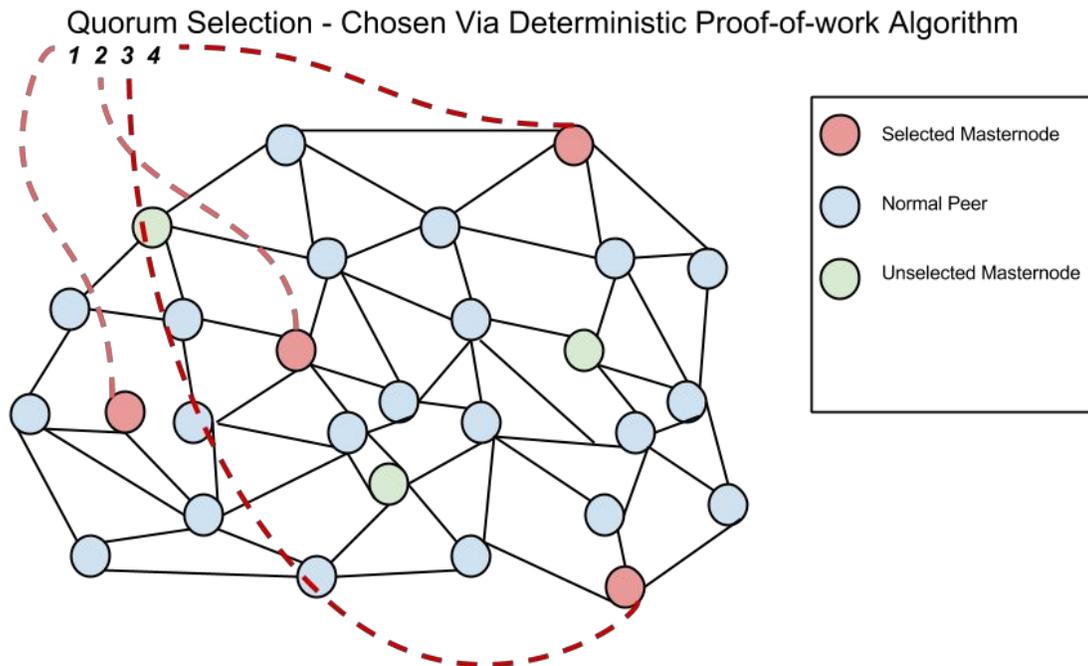


The second tier exists as a protocol extension on top of the original Bitcoin framework. This allows us to support services that add value beyond the decentralized ledger, such as instant transaction confirmation, improved privacy, decentralized financing of the project, economic expansion, decentralized polling for direction of the currency, and a voting mechanic for getting a final answer from the network for decisions that need to be made.

Masternode Quorums

In the Second Tier, Masternodes provide the actual services of the network by responding to what are called Masternode quorums. A Masternode quorum is a secure subset of Masternodes making a decision as an independent oracle on the network. Since we can't trust one individual Masternode, we use randomly selected subsets of the whole Masternode network and have them do identical work, then report the results. If all results match, the work was successful and information from that process can be trusted. By using this strategy throughout our network we achieve a service as easy to use as a centralized service, but also more decentralized than the Bitcoin network.

Masternode Quorum Selection



In addition, Masternodes allow direct connections to their peers to perform services that aren't propagated over the network. On the current network this includes a separate communication protocol capable of sending private and instant transactions. After T3 is implemented, this will be a simple API that T3-Light-Clients can access for quick questions about the network. These direct messages are resolved via quorum, then the answer plus quorum signatures are sent back to the device.

Wallet Integration

The Second Tier (T2), is a private network that only the Masternodes can access. It's run on local hardware and has guaranteed high speed API access and return time for actions. It's made to support any number of wallets and clients, so users can switch between different wallet products without losing any data.

Wallets will be exceptionally easy to make by using the first ever decentralized API. Information coming from the API can be trusted due to the Masternodes quorums, so a full node installation or SPV service are not needed. We call this API security model SSRI (Simple Secure RPC Interface). Wallets for the Dash Network will then be very light and efficient, while remaining highly secure and easy to use.



3.3 Third Tier - Light Devices

The third tier is how regular users access and use the Dash Network trustlessly. These devices will connect to the Masternode network by utilizing the decentralized API, capable of carrying out all basic operations such as syncing the user's public and private profile information, transaction history, friends list, and any other information a user stores on the network.

Light devices can be anything from a phone, to a small app on iOS, a package on Linux, or Javascript in an HTML web browser. The application that runs on these devices will have a very small footprint and be lightning fast to use. As time goes on and the network grows to millions of users, this application will grow more secure and remain as fast or even get faster as we optimize common requests.

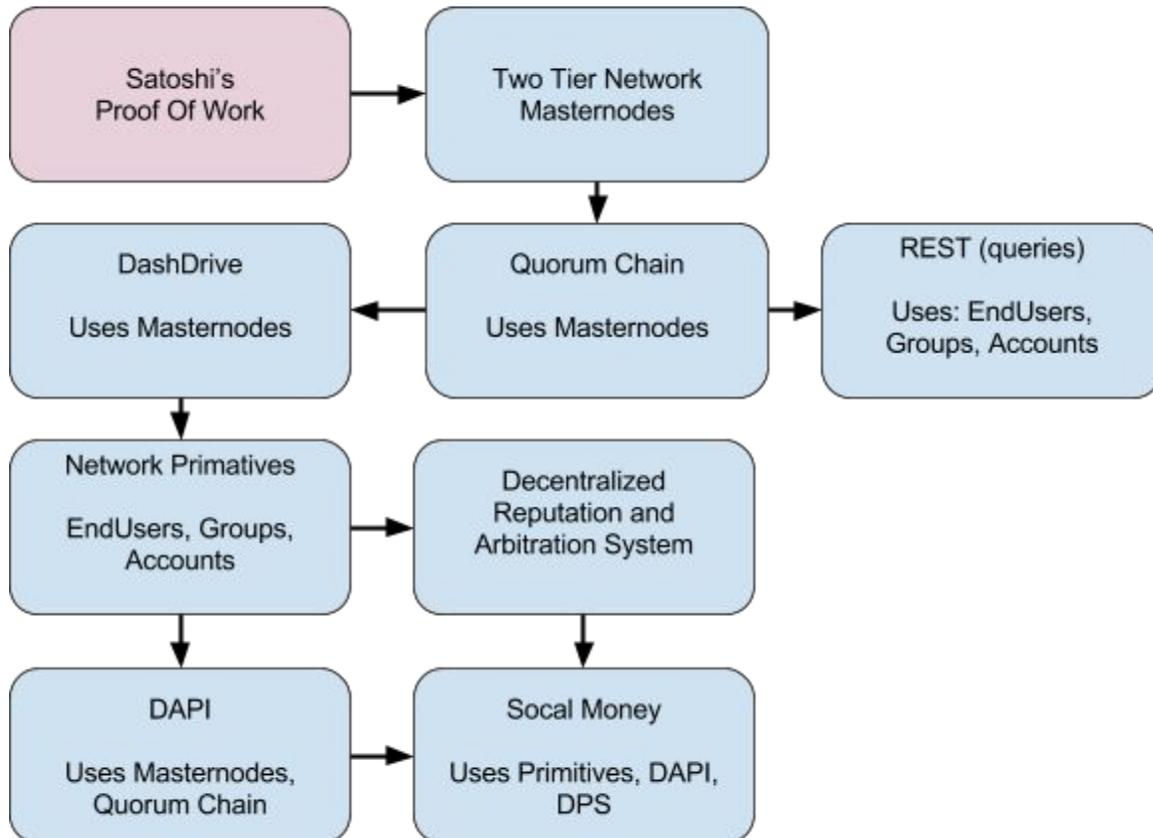
The third tier network will have a fee structure based on the amount of processing time used, in order to deter spam attacks. These fees will always be low even when the network is transacting millions/billions of transactions a day. Even micropayments will be allowed on the T3 network. Users will be allowed a certain amount of free usage on the network per-day, per-month, and per-year. As users utilize the network's services, they will sign messages with their private key, proving who they are, which will in turn update their profile with the amount of processing time they have used. After using up their allocated amount of free processing, users can purchase more for a small fee. Merchants will often have to do this due to the amount of payments they process.

All transactions happening on T3 will be broadcast and stored immediately. By doing so the blockchain will always be up-to-date and accurate. Please see the DashDrive paper for more information.

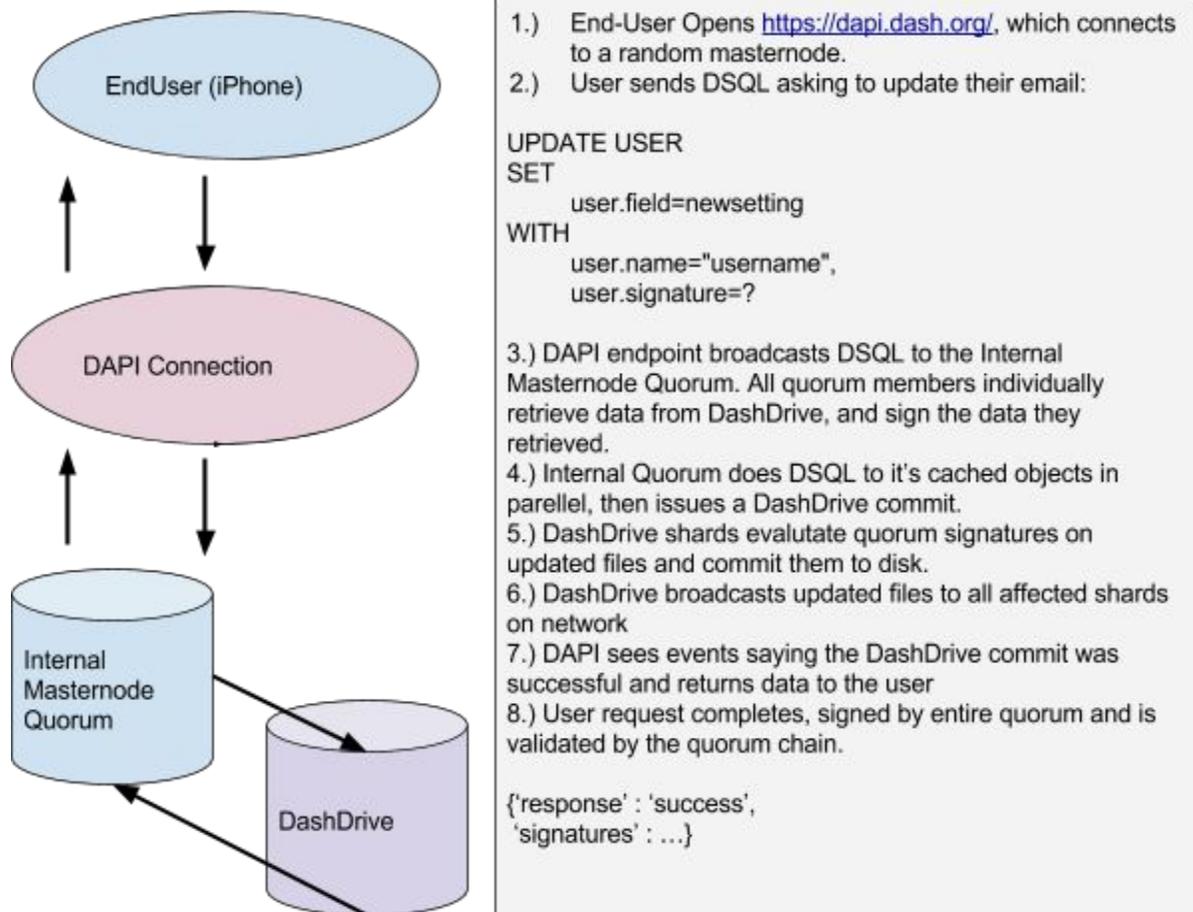


4 DAPI Diagrams

4.1 DAPI Components



4.2 DAPI Communication



5 Security

Network-designed topology is calculated ahead of time using only deterministic algorithms. This provides a map of which second tier nodes are connected to which other second tier nodes. By using such a structure, we can protect the nodes from attack. For example, each node will have a list of all second tier nodes they should be connected to at a given point, which will increase attack resilience, because during an attack, the nodes can simply disable all non-second tier connections, reducing the connections to those who are identified and collateralized.

By using predefined topology, we create small groups of servers known as sub-quorums, which are groups Masternodes to do work for users simultaneously. Each sub-quorum consists of 7



random masternodes and 1024 shard connections. This allows the sub-quorums 1-hop access to all data stored on the network.

When handling user requests, a given user will connect directly to a random masternode, which then relays the request to all sub-quorum members and any required shards. User signatures are then checked by the shards, to determine access and the DashDrive objects are cached in the sub-quorum awaiting enough data to fully process the command. After permissions have been validated and objects have been cached, the command is executed simultaneously as a group action. Each member of the quorum will perform any required updates to cached objects, then sign the resulting hash of the object. Users can validate the data was not tampered with by looking at the seven signatures given back to them.

Requests are made over HTTP via a REST interface on the endpoint Masternode the user is connecting to. Users will have access to a After being processed by a web service running on the entire second tier, the endpoint server will relay a REST command locally to the already connected sub-quorum this endpoint is connected to.

All sub-quorum members will then process the REST command and retrieve any needed files from DashDrive. Required files are temporarily cached in RAM, then the REST command is executed locally when all required files are available (users, groups, accounts, etc). Commands can have a result, which is then relayed using a signed message based on the Masternode collateral transaction keys.

Users then validate the results by checking the signatures against the resulting data. Each Masternode signed their result, so all results must match in order for the command to have succeeded on the network.

6 Scalability

Because of the quorum system of command execution, we can assume much higher network efficiency. Quorums are made up of seven individual nodes on the network and the current Masternode network is made up of 3300 Masternodes, giving us the ability to execute 471 commands at once on the network. We will provide an event-based implementation of command processing, which will allow servers to process multiple commands at once. It will be common for events to be caching for 100-200ms, which allows hundreds of events to be processed at a time.

Each node will have many connections open to the shards on the network, this allows for 1-hop communication from DashDrive to the individual quorum groups. We will use deterministic connection information between all nodes on the network to provide a secure, well designed network topology that no one controls.



7 Quorum Group Design

By relaying information to a Masternode's entire sub-quorum, all required shards will be queried for the required cached files.

Name	Connected To Shards
Quorum Member 1	1-146
Quorum Member 2	147-292
Quorum Member 3	293-438
Quorum Member 4	439-584
Quorum Member 5	585-730
Quorum Member 6	731-876
Quorum Member 7	877-1024

8 Stable Network Structure

When using the network, endusers connect to permanent API versions on the network, using a common URL pattern, <http://tier2.dash.org/api/v1/command?parameters>

9 Access Design

To make use of current generation internet technology, we allow usage of the network via HTTP using a REST API. By allowing users to download an SDK of all DAPI functionality, we allow websites to do on-website purchases. The transaction can be performed entirely on the Dash network, from start to finish by reaching out to a random Masternode, executing the command on a local quorum, then retrieving data or updating data on DashDrive.