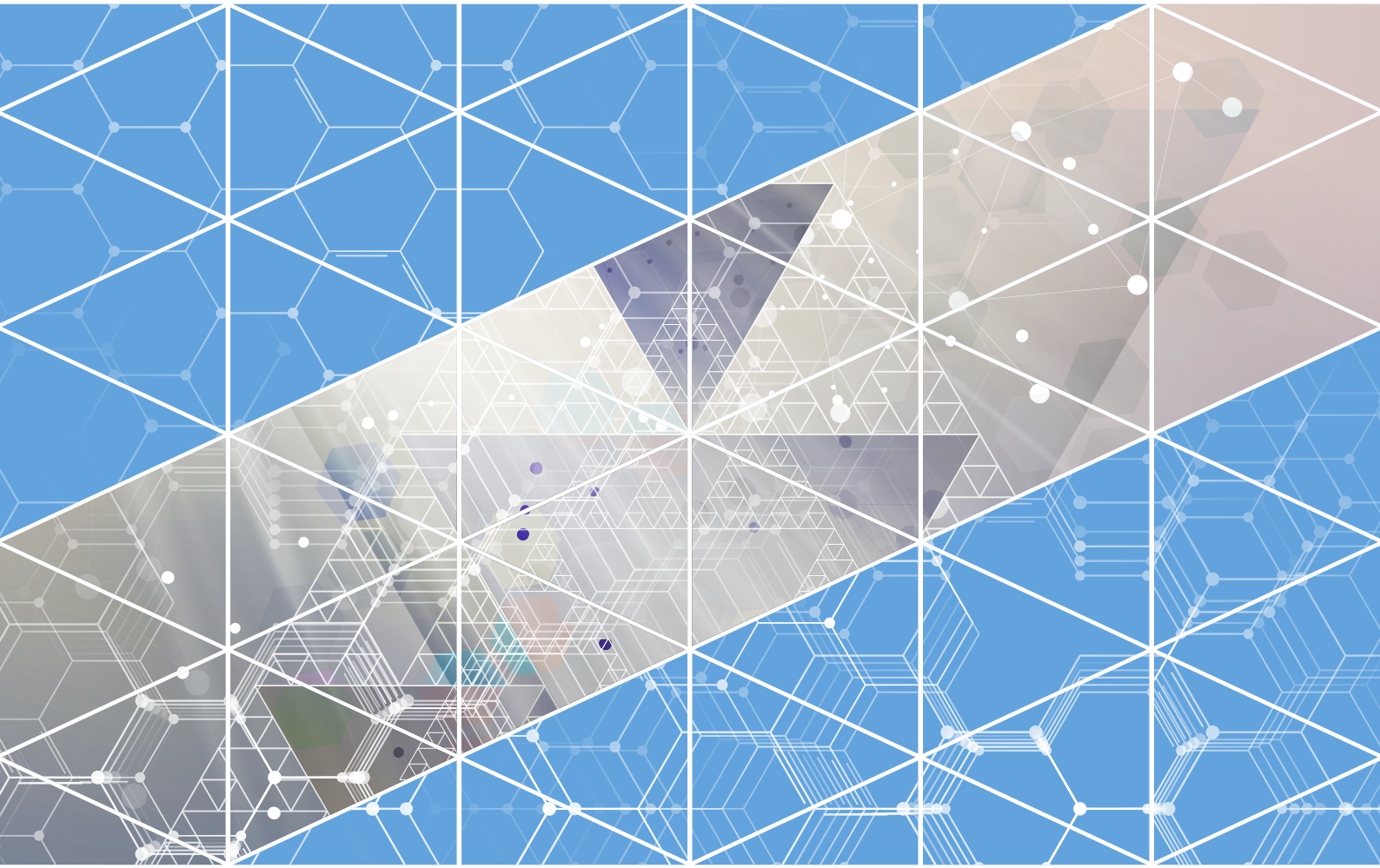
A SAFE Network Primer

An introductory guide to the world's first fully autonomous data network



February 2018

SAFE Network

The SAFE Network – a brief introduction

Technological progress is a perpetual process of automation and abstraction. Difficult and complex tasks are made simple by software and machines until they are completely taken for granted. Meanwhile technology moves on, taking down new barriers to progress as it goes.

One example of this process is cloud computing. The once specialized task of provisioning a server is now so simple that anyone can do it. The complexity has been abstracted away. Ultimately machines will become far better at provisioning other machines than people are, at which point there may be advantages in automating the task altogether. Indeed this is already starting to happen.

What if we were to take this process one step further and automate the entire network, making it autonomous, able to store, protect and deliver data without any human involvement at all? And what if all the data on the network were stored and transmitted under conditions of utmost security, with the most advanced encryption techniques available? And what if users of this platform could be as public or as private as they wanted to be?

Among the obvious benefits would be hugely enhanced confidence in the security and integrity of all data. With no third parties to introduce error and huge barriers to hackers wishing to steal, corrupt or compromise data, such a decentralized 'trustless' system would be invaluable in IoT applications such as driverless cars where any failure or cyber-kinetic attack would be life-threatening. And it would offer a much needed alternative to the current Internet, dominated as it is by a handful of enormously powerful corporations and used by governments to keep tabs on us all. This is a power dynamic that's ripe for change.

Above all, such a system would be simple, with the complexity of managing data abstracted away. This paradigm shift would be a huge shot in the arm for all sorts of industries. With developers no longer having to worry about low-level storage, routing and compute, whole new sets of applications could be built that are independent of the data they use. It would kickstart the nascent personal information economy in which individuals decide who can see what details about them, for what purpose, and for what possible financial recompense.

Such a system is the vision of MaidSafe, a Scottish software company working in the area of decentralized computer networking. The SAFE (Secure Access For Everyone) Network is an autonomous peer-to-peer network created by linking together users' computers and smartphones. It is designed to solve many of the current technical, managerial and societal problems with centralized networks: a lack of privacy and data security, censorship and the massive consolidation of control by a few powerful actors.

This guide outlines how the SAFE Network is constructed in order to achieve these aims. While it is somewhat technical in places, it's intended very much as a primer. By reading it, even those with very little technical knowledge should be able gain a good working understanding of the SAFE Network. Meanwhile for those requiring more depth there are plenty of pointers as to where they can find the relevant information.

Contents

1. A statement of intent

The Web of today is a far cry from the one created by Sir Tim Berners Lee and his colleagues less than three decades ago. Infinitely more powerful and pervasive than those scientists could ever have imagined, the Web has changed the world. Unfortunately it has also served to centralize power, wealth and control. Two corporations, Google and Facebook, now control 70 per cent of online advertising spending at the time of writing, probably more when you read this.

Elsewhere, giant retailers such as Amazon are using their scale, political clout and international footprint to destroy local competition creating a virtual monopoly in many areas, and these same companies are also the biggest providers of the cloud services that increasingly host all of our data. Everywhere you look it's the same story: centralization, consolidation, homogenization and monopolization.

> Centralizing data storage on corporate servers provides an irresistible honey pot for any would-be black hat looking to prove their chops

This centralized commercial infrastructure also provides the perfect ready-made framework for government surveillance, as Edward Snowden revealed so dramatically a few years ago. And as we become ever more dependent on it, this infrastructure becomes an increasingly juicy target for hostile state actors who not only use it to spread disinformation and discord, but can also take out critical infrastructure with well-targeted DDoS and malware attacks with plausible deniability and without having to put a single boot on foreign soil.

Of course, it's not just well-resourced, government-backed hackers we need to worry about. Centralizing data storage on corporate servers provides an irresistible honey pot for any would-be black hat looking to prove their chops. UK telecom giant TalkTalk was brought low by a couple of teenagers – literally from a back bedroom – and that's before we start talking about Yahoo, Equifax, Target, Tesco Bank and many more. Put simply: our data – meaning our online identities, the basis of how we operate in the modern world, our very 'digital souls' - is not safe on company servers.

For all its glories, the Internet is broken. It has strayed from its original vision. It is now a machine for censorship, propaganda, and central control that feeds on our personal data. It could and should have been much better than this.

MaidSafe's solution is to create a secure, autonomous, data-centric, peer-to-peer network as an alternative to the current server-centric model. Rather than residing on a central server or data centre, individual files are split up into pieces, encrypted and spread out over the network. Individuals retain full control of the files they create. It is resistant to DDoS, malware and hacking and it cannot be centrally co-opted and controlled by monopolistic corporate and government interests. As a platform, such a network could usher in whole new business models, just as the original Internet did at the turn of the century.

More and more people share this vision, including Sir Tim himself, and some of us are working hard to make it a reality.

Tell me more *(click on links)*

Autonomous Data Networks and Why The World Needs Them (MaidSafe blog)
The Power of the Crowd Series – Part One: The Problem (MaidSafe blog)
Tim Berners-Lee on the future of the web: 'The system is failing' (Guardian)
A decentralized web would give power back to the people online (Techcrunch)
The Abstracted Network for Enterprises and the Industrial Internet of Things (Meshdynamics)
Paradigm shifts for the decentralized Web (Ruben Verborgh blog)

2. A short history of decentralized networks

Decentralized or peer-to-peer (P2P) networks are not new. Since the release of Napster on June 1, 1999 they have taken the world by storm, particularly for file sharing. These networks allow users from all over the world to connect to each other and share data such as movies, books and music. In 2010 more than half of all Internet traffic was attributed to P2P.

But the use of these technologies is not limited to simple file sharing. Freenet was launched in March, 2000 allowing people to publish decentralized websites (Freesites). Freesites are not stored on central servers but instead are distributed across the machines of the encrypted network's users.

A little after that the BitTorrent protocol was created by Bram Cohen. BitTorrent was and still is particularly well suited to transferring large files in a P2P fashion, allowing simultaneous downloads from multiple peers.

The next notable development arrived after the 2008 financial crash which very nearly brought the global economy to its knees. In 2009 Satoshi Nakamoto released Bitcoin and gave the world a 'trustless' decentralized digital currency that is controlled by no bank, government or institution. The blockchain – the immutable ledger that records all Bitcoin transactions - was something very new, solving at a stroke the difficult and longstanding problem of creating a trustless source of the truth for transactions. The connections between Bitcoin nodes are not encrypted, but the ownership of 'addresses' in the network can be proven by the usage of private cryptographic keys in a Public Key Infrastructure (PKI). Hence the term cryptocurrency.

The SAFE Network is the next big step in the evolution of P2P networks, combining the vision of decentralized file sharing and web sites together with an internal cryptocurrency–Safecoin–and several additional innovations to enhance security, privacy, performance and stability. MaidSafe (a company based in Ayr, Scotland) has been researching and developing this project since 2006. The aim is to engineer and enable the following features:

- Fully encrypted data storage and file sharing with 100 per cent data integrity.
- The ability to log on to the network anonymously using a public key infrastructure.
- The ability to create censorship-resistant communication and publication services.
- Ensure all data is 100 per cent serverless, encrypted and decentralized.
- Ensure the network is autonomous and self-healing.
- Eliminate the need for centralized servers and trackers.
- Incorporate a fast and scalable cryptocurrency to allow anonymous exchange of value free from transaction fees.

These features combined give users the freedom to safely store data on the network. SAFE also allows them to share data with others securely and to publish websites using secure communication channels. New browsers and web apps communicating with the network via Application Programming Interfaces (APIs) enable improved private and highly secure forms of email, messaging, browsing; Domain Name System (DNS); chats; and all other technologies that are currently being used on the Internet.

Tell me more *(click on links)*

Napster (Wikipedia)
Freenet (Wikipedia)
BitTorrent (Wikipedia)
Bitcoin: A Peer-to-Peer Electronic Cash System (Whitepaper)

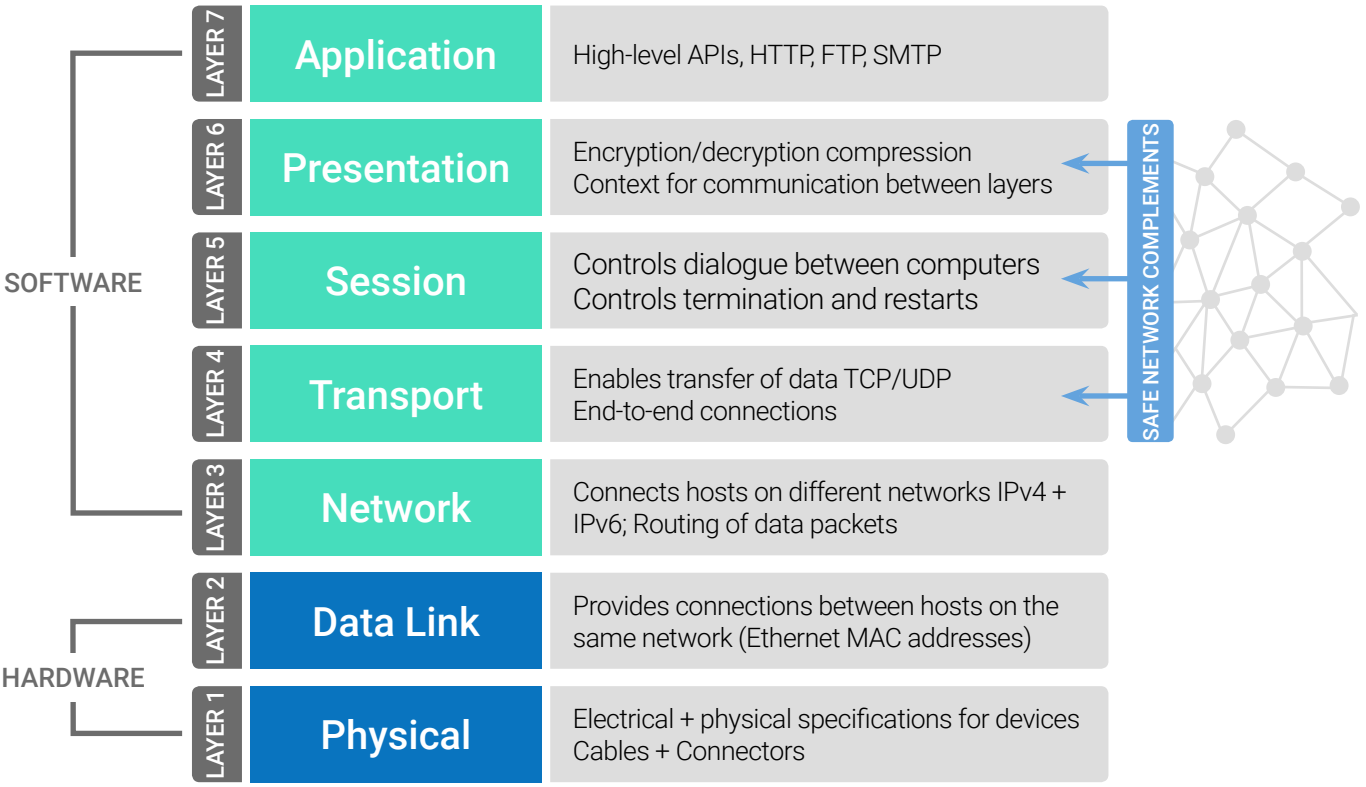
3. A fully autonomous data network

SAFE at its core is an 'autonomous data network'. This means it is capable of managing and optimizing workloads, routing, system redundancy, authentication, access control, and other networking and storage tasks without any human intervention. Unlike the current Internet, SAFE infrastructure is not defined by a set of federated servers, Virtual Machines (VMs), owned storage locations or identifiable nodes. Unlike blockchains it is designed to store and manage live data rather than pointers to data and to transact in real-time. And unlike BitTorrent, it does not rely on centralized components of the Web to locate and track files.

> There is absolutely no human intervention in the SAFE Network apart from running a piece of self-configuring, self-healing software

The ramifications of a fully autonomous network that takes care of all aspects of securing, managing and transmitting data are far reaching. There are no people setting prices, no altering configurations to make things work, no tweaking data on disk, no gaming the rules that govern the nodes–absolutely no human intervention apart from users running a piece of self-configuring, self-healing software, together with other users doing the same thing. The network decides prices, rewards, how to protect data, communications and calculations. We can all go and do something more valuable with our time!

The SAFE Network replaces three of the seven networking layers in the OSI model for improved security, privacy and autonomy.



Two sides of the network – Clients and Vaults

The SAFE Network can be seen as a secure, encrypted layer that sits on top of the current Internet allowing for autonomous data storage and networking by replacing three of the seven OSI networking layers. It has two basic components: Vaults and Clients.

KEEP IT SIMPLE!

The SAFE Network is comprised of its users' machines, linked together by software. There are two main kinds of node: Clients and Vaults. Client software allows users to access the network, while Vaults provide the storage, routing and data security functionality and a way of earning the Safecoin currency. A single computer can be both types of node at the same time.

The core of the SAFE Network is formed of nodes called Vaults. Vaults are programs that people (Farmers) run on their computers, connecting the devices to each other using existing protocols like TCP, UDP and µTP. Vaults allow users to let the network store data on their devices, potentially earning the Safecoin currency as they do so (see Chapter 7). Vaults also manage the data and route it around the network, pass messages and ensure events occurring on the network are valid.

WHAT DOES THAT MEAN?

Vault – the nodes that make up the SAFE Network are called Vaults. They consist of software that runs on users' devices and communicates with other Vaults. Vaults also store data in the form of encrypted chunks, for which they can earn Safecoin by Farming when that data is requested.

Client – a program that allows users to connect to the network and make use of its services. At the present time this is incorporated into the Peruse browser.

Peruse – a browser for surfing the SAFE web.

Farming – the act of earning Safecoin by providing storage capacity, computation and bandwidth connections that make up the network

Safecoin – the currency of the SAFE Network, earned by storing data, spent by uploading data, and otherwise exchangeable amongst network users.

The Vault

The Vault software is a small executable file that connects the user's machine to the SAFE Network. It manages the storage of data chunks on the user's computer and in that way it provides storage capacity to the network. It also routes and caches encrypted data chunks over the network making use of fully encrypted connections to other Vaults.

By routing and storing data, Vaults form the core of the SAFE Network. Vaults are clustered logically into small groups, each of which is responsible for looking after the data stored within a certain range of addresses, called a Section (see Chapter 4).

The forming, merging and splitting of these groups of nodes happens in a fully autonomous way. The same goes for the routing of chunks across the network. No central servers or agents (like BitTorrent trackers) are needed to form this network. The Vaults follow a set of rules to create and maintain the network, requiring no central authority to oversee the proceedings. Vaults have 'personas' to help them manage the various tasks (see Chapter 8).

Running a Vault in the SAFE Network is called Farming because users look after the data until it is consumed at which point they may earn payment for their efforts.

> Running a Vault is called Farming because users look after the data until it is needed at which point they may earn payment for their efforts

The Client

Just as you don't need to run a web server to use the Web, so you don't have to run a Vault to access the SAFE Network. Ordinary users interact with the network via the Client. This is a piece of software, at present bundled with the Peruse browser, that allows secure connection to the Vaults that constitute the SAFE Network while hiding the IP address of the user from the network itself. Any user (even if they're not logged in) is able to request (GET) data from the network for free. For example they can browse a safe:// site or download a publically posted song or movie, and it will cost them nothing. It's only when the user wants to store (PUT) data onto the network that an account with a small amount of Safecoin is needed.

Tell me more *(click on links)*

On the Verge of Autonomous Networking (Enterprise Networking Planet)
Autonomous Network (MaidSafe whitepaper)
An Implementation of a SAFE Network Vault (Github)
OSI model of network layers (Wikipedia)
SAFE Client libraries (Github)
Peruse browser (Github)

4. The architecture of the SAFE Network

It is a fundamental goal of MaidSafe that users should be able to browse, create and share files just as easily as they do with the current Internet but with greatly enhanced privacy, security and control over their data. The SAFE Network is being designed with ease of use in mind, and for developers it's just a short leap from what they are familiar with in terms of using APIs. However, those wishing to perform more complex systems-level tasks will need to go deeper into its architecture. There are obviously some big differences between traditional client-server and decentralized systems. This chapter provides a brief introduction to the topic.

Sections

The first step to understanding the architecture of the SAFE Network is to take a brief look at distributed hash tables (DHTs).

KEEP IT SIMPLE!

To find data on a distributed network you need a map. A distributed hash table (DHT) fulfils this function by providing a lookup service for locating data. Data is stored according to a unique property – its hash. This hash is the same as its network address. The SAFE Network is divided into Sections with a small group of nodes (Vaults) responsible for managing the data in a single Section. The more nodes there are on the network, the more Sections there will be, and the more resistant it becomes to failure or attack.

Petar Maymounkov and David Mazières released the Kademlia distributed hash table in 2002. The idea is that nodes in a network overlay the basic network layer with a different node identification system. So a node (in this case a Vault) could have an IP address of 96.251.182.97 while it uses 17846cb8a4b53c9e44c616d2415a15984283eee975a1dac8f488dd91d0aed1cd as a unique 256-bit address in XOR space.

Bitwise exclusive OR (XOR) has the feature that each address has a unique distance to any other address in the entire address range. XOR distance bears no relation to physical distance. Indeed two pieces of data on the network may be very close XOR-wise but be sitting on machines located on opposite sides of the world.

MaidSafe came up with an enhancement to Kademlia by splitting the whole 256-bit address range into so-called 'Disjoint Sections', or Sections for short. A 256-bit address space has 2^{256} -1 possible addresses which is an extremely large number to manage, but it can be split up into smaller Sections based on address, with each Section being managed by a small group of Vaults.

WHAT DOES THAT MEAN?

Distributed hash table – a map of where data is stored on a distributed network.

XOR networking – a way of randomizing the physical location of data on a distributed network and ensuring each location is unique.

Section – a subset of all the addresses on the network. Data stored in each Section is looked after by a dedicated group of nodes.

group_size – a parameter stipulating the minimum number of nodes that can look after a Section.

Bootstrapping – starting up the SAFE Network by connecting together a minimum number of nodes. Bootstrapping is also used to describe a new node (Vault) joining the network.

Bootstrap servers – a small number of servers operated by MaidSafe to which new Vaults connect (via a proxy) when they join the network. Their IP addresses are hard coded into the Vault software.

Hash function – a function used to map data of arbitrary size to data of fixed size (e.g. a 256-bit string of characters) called a hash. Any change to the original data will result in a completely different hash. SAFE uses the SHA-3 hash function.

Datachain – a record of Section membership and network events that's held by each Vault in that section.

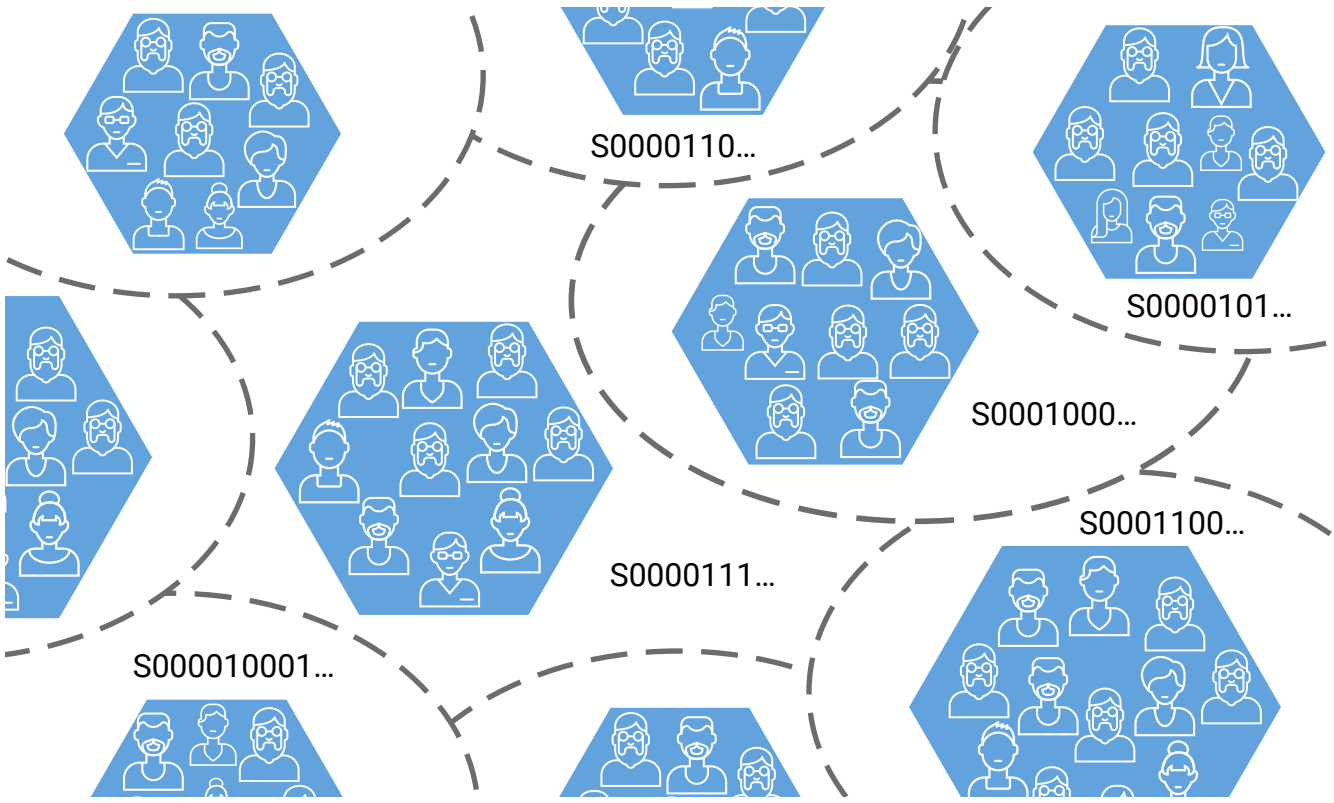
On joining or rejoining the SAFE Network a Vault cannot simply pick its own XOR address. Instead it has to wait for a Section to pick it up and make it part of the network. When this occurs, the Vault receives the Section's Routing Table and learns the address range of the data that it is responsible for.

Data stored on the SAFE Network is first broken into chunks, hashed and then encrypted. Those chunks are run through a hashing algorithm to create a unique 256-bit code or hash for each chunk. Only chunks that are exactly identical will have the same hash value. This hash serves as the XOR address on the network where that chunk will be stored, which in turn determines the group of Vaults that will manage it.

So chunks with hashes that lie within a certain address range (say 000010... to 000011...) will be secured and stored by the group of Vaults that manages that particular Section. The membership of this group will change over time as nodes disconnect from and reconnect to the network. When the network is bootstrapped, a minimum of eight Vaults are responsible for the whole 256-bit address range (2^{256} -1 addresses). As more Vaults join, the group will split in two. And if these new groups attract new Vaults, too, this process repeats itself. So with 1,000 groups the address space is split into 1,000 Sections. With 100,000 groups it is split into 100,000 Sections, and so on.

> The hash of a chunk of data serves as the XOR address on the network where it will be stored, which in turn determines the group of Vaults that will manage it

Each Section (range of XOR addresses) is managed by a group of Vaults. The most trusted Vaults in a group are called Elders. Elders have voting rights and communicate with Elders in neighbouring Sections.



If a group of Vaults managing a particular Section grows significantly larger than average (the mode average is about 12 Vaults), in general it will split into two smaller groups with each managing a smaller Section (range of addresses). Usually this will happen before group membership exceeds 22 Vaults. Likewise, if, as a result of Vaults leaving during churn events the number of nodes drops below a certain level specified by a parameter called `group_size`, then it will be triggered to seek a merger with a sibling group (one managing a nearby Section). At present, `group_size` is 8.

Consensus and quorum

The group of Vaults managing a Section will always try to reach consensus among themselves on any state and action. They also 'group sign' messages that travel over the wider network so other Vaults in other groups can cryptographically verify each message and action. These group signatures are stored in 'Datachains' which are secured and held by all Vaults in the group. All events such as groups forming, splitting and merging are recorded and stored that way.

KEEP IT SIMPLE!

In order for something to happen on the network, for example the storing of a data chunk at a network address, the group of Vaults responsible for that address must decide that the action is legal and valid. If the required quorum size (for example, 5 Vaults out of 8) is achieved the action will go ahead. A record of all such events is kept in a container called a Datachain. Any new Vault joining the group can read the Datachain. Groups also share some information about events and current state with other groups that are close to them.

Because of this all group events are cryptographically verifiable by any Vault that joins the group. These events are also stored by other close groups (as measured in XOR distance) in the network. So a group of nodes managing a Section S(0000110) knows about all the nodes in the group managing S(0000111) as well. It also has access to the part of the Datachain containing all the information on this group. Because they hold a record of the state of the Sections, Datachains allow the network to rebuild itself in the event of a major failure.

The closer a Vault is to a certain address space on the SAFE Network the more information it has about data stored at that address. And logically the further away a Vault is, the less information it has. Except for when the network is very small (fewer than 22 Vaults), no Vault in the SAFE Network has a complete overview of the network. The bigger the network becomes, the more secure it will get because an individual Vault will have influence over a diminishing range of addresses.

An action or event happening in a Section is only valid once a quorum of the group has approved it. A change of state needs to be signed by a certain proportion of that group known as the quorum size. Vaults follow certain rules that stipulate the quorum size. In a group of eight Vaults, five are needed to approve an action (such as a request by a Client to store a chunk of data); in this case the quorum size is 62.5 per cent.

Notice that there are no external trackers or managers involved in any of these group decisions. A group of Vaults (no matter how small or big) operates in a fully autonomous way within the network.

Tell me more *(click on links)*

A Survey of DHT Security Techniques (Globule)
Kademlia: A Peer-to-peer information system based on the XOR Metric (Whitepaper)
Distributed Hash Table (MaidSafe Whitepaper)
DHT-based NAT Traversal (MaidSafe Whitepaper)
Disjoint-set data structure (Wikipedia)
Disjoint Sections (MaidSafe RFC)
Routing - specialised storage DHT (MaidSafe repository)
SAFE Network Architecture (Ian Coleman)
Datachains: What? Why? How? (David Irvine, Metaquestions blog)

5. Node Age and Proof of Resource

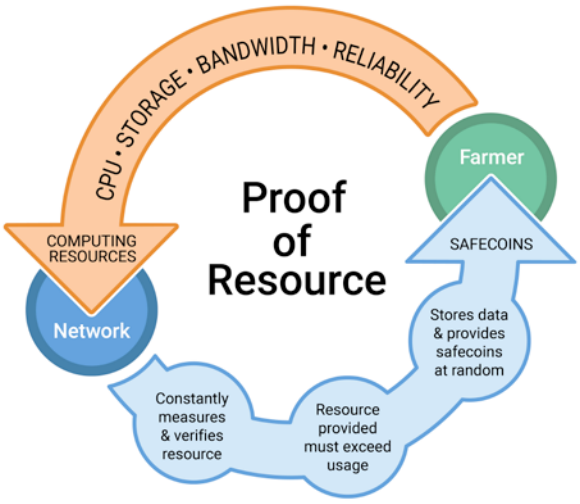
For a Vault to join the network it requires the IP addresses of other Vaults to connect to. For this purpose MaidSafe provides a small number of bootstrap servers. These are internet-connected machines running the Vault software. The public keys of the bootstrap servers are hard-coded into the Vault's binary, meaning that all communication is encrypted from the beginning.

KEEP IT SIMPLE!

Just as children are not allowed to vote in elections, so a node (Vault) may not vote on network events–such as a new member joining or the storage of a data chunk or Section splits and mergers–until it has proven itself to be reliable (Proof of Resource). A node gains trust by moving from Section to Section and acting reliably, each time increasing its Node Age by 1. Once it is among the oldest in its group (in terms of Node Age), it may be given voting rights. Forcing nodes to prove themselves in this way is an important security measure.

A Vault connects to other Vaults over the so-called 'Crust' layer, sending out a message that it wants to join the network. A group of Vaults that has an open spot may reach consensus to allow the new Vault to join the group. In this case the Vaults will send out a Proof of Resource request in which the new Vault has to prove that it can provide a certain amount of bandwidth and CPU capacity.

If the new Vault successfully completes the Proof of Resource test the group will assign it an address on the network within their Section and it will become



WHAT DOES THAT MEAN?

Crust – short for Connected Rust, Crust is a software library, written in the Rust programming language, for establishing and maintaining reliable peer-to-peer network connections across a wide variety of network conditions and protocols.

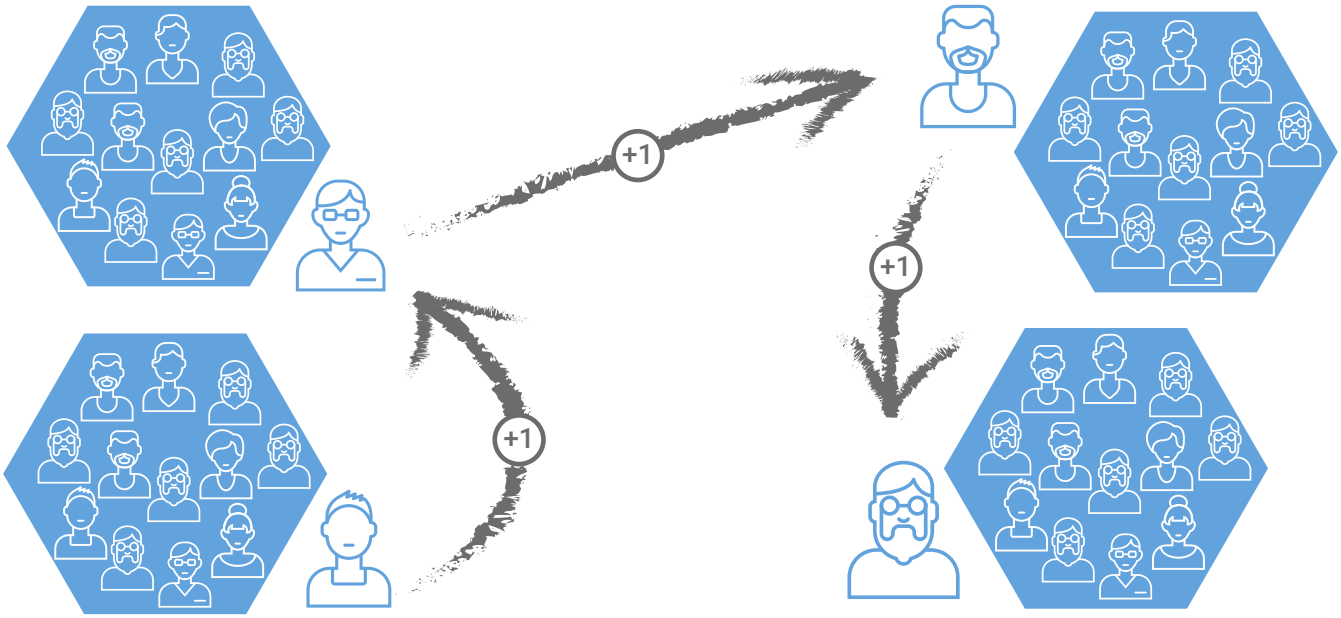
Proof of Resource – 1. a test whether a Vault that wants to join the network has sufficient bandwidth and CPU power. If it fails the test it will not be allowed to join. 2. Random checks are occasionally made by managing nodes to ensure that the Vault is indeed maintaining chunks it is supposed to be storing. If it fails the challenge (by not providing the demanded proof) its Node Age is diminished.

Node Age – a measure of the trustworthiness of a node (Vault). After the initial connection, a Vault gets moved at random from Section to Section and builds its reputation or Node Age. Once its Node Age reaches a certain value it can be an active participant in group decisions.

Churn – the action of Vaults leaving a group or new Vaults joining, requiring a reorganization of relationships and responsibilities. Churn means that groups are not static for long.

Elder – a node with voting rights in its section. Elders are simply those nodes with the greatest Node Age in a Section.

A Vault's journey to maturity. A Vault must prove itself in a number of sections before it has a chance to be granted voting rights as a Section Elder.



a member with a low 'Node Age', meaning it doesn't yet have voting rights and is liable to be moved off to another Section at any moment (churn).

When a node leaves the group, it is sent at random to another Section in the network, i.e. it is given a new random XOR address. If the group managing that new Section reaches full consensus on this action the Vault will join it and its Node Age will increase by 1.

> Node ageing and churn make targeting a particular Section on the SAFE Network by an attacker close to impossible

Node Age is an exponential function. Newer Vaults are more likely to be moved to a new Section (and thus increment their Node Age) than older more trusted ones. Vaults must prove their worth (CPU, bandwidth, reliability) to the others in order to earn trust. If that trust is lost, it must be re-earned. Only once a Vault reaches a certain Node Age can it vote on network events. Because new Vaults must prove their worth in various Sections before they can vote, targeting a particular Section on the SAFE Network by an attacker is close to impossible. Node Ageing and churn are thus vital security features.

Only the most trusted Vaults in a Section, those Vaults with the greatest Node Age, have voting rights. These Vaults are called Elders. Other Vaults in a Section simply receive notification of their decisions.

Tell me more *(click on links)*

SAFE Network Explained
Introduction & Technical Overview of SAFE Consensus
Datachains - Deeper Dive
Crust- Reliable p2p network connections in Rust with NAT traversal
Understanding Churn in Peer-to-Peer Networks (Sigcomm, Research paper)

6. Everything's encrypted

All data on the SAFE Network is protected by several layers of encryption. Even public data (like a blog safe://website/blog) is protected by encryption, but in this case the keys to remove the encryption are shared with visitors to make the data available to them.

KEEP IT SIMPLE!

All data on the SAFE Network is encrypted or kept in encrypted containers. To store a file on the network it is first broken into chunks, hashed and then encrypted. These chunks are themselves encrypted using the hash of another chunk from the same file. This is self-encryption – a method patented by MaidSafe.

At the network level, the SAFE Network uses the TCP and µTP protocols and all the data moved by these protocols is encrypted from the first bit of data transmitted to the network. The first connection to the SAFE Network that a Vault or Client makes is to a MaidSafe bootstrap server, one of a number of servers run by MaidSafe to allow new machines to join. The public keys for these bootstrap servers are hard-coded into the Client software, so the communications between the network and the user are always encrypted, never in plain text.

Clients and Vaults in the network get a list of addresses and public keys of other users to connect to. These connections are also encrypted from the first bit. That way the lowest connection level (Crust) forces all communications to be encrypted.

Proxy Node for Clients

To retain anonymity, the identity of a Client connecting to the network must be obfuscated from the nodes (Vaults) that comprise it. For this reason connections between Clients and Vaults in the SAFE Network always occur via a Proxy Node. The Proxy Node knows the Client's IP address and will allow it to connect to a group of Vaults on proving its bandwidth. The Vaults cannot see the Client's IP address but they know its public key and XOR address. Of course, the connections between the Client and the group(s) are fully encrypted.

The Proxy Node is able to provide a service to the user, connecting him or her to the network without having any knowledge of activity thereafter. The group of Vaults to which the user is connected might know a little about what the user is doing on the network but they can only identify the user by their XOR address and not their IP. In this way, complete anonymity is assured.

Self-encryption of data

When a Client uploads a piece of data to the network (for example an mp4 video) it is first broken into pieces with a maximum size of 1 Mb and those pieces, or chunks, are then 'self-encrypted', a process patented by MaidSafe by which each chunk is encrypted using the hash of another chunk from the same file. These encrypted chunks are then sent out into the network to be stored with the hash of an encrypted chunk equaling its network address. A number of copies of each chunk are stored, most likely on machines distributed around the world. If one copy is lost another is generated immediately ensuring a high level of redundancy.

> The SAFE Network provides a platform for applications that is both highly secure and anonymous by design

The Client retains the keys to decrypt the data locally. That way no keys or passwords need leave a person's computer or mobile phone. Meanwhile the chunks are stored on the SAFE Network in a fully encrypted way. Users can choose to share these files with others by sharing their keys with them. They can also choose to make the files fully public, in which case the keys required to decrypt the files are made publicly available, as with the example of a blog.

Multilayered encryption

As shown above, the SAFE Network uses several layers of encryption to protect a user's anonymity and privacy. Several extra layers are active when people use direct messaging or create a public profile. The network is meant to be as 'zero knowledge' as possible even to the extent that Farmers cannot possibly figure out what chunks from which file they are storing - even if it's their own.
By utilizing multiple levels of encryption as well as obfuscating the identity of its users after the first hop, the SAFE Network provides a platform for applications that is both highly secure and anonymous by design.

Tell me more *(click on links)*

Connecting to the Sodium crypto library
Crypto 101
Self-encryption - data that encrypts itself, with itself (MaidSafe)
Self-encrypting Data (MaidSafe Whitepaper)
UDP Hole Punching (MaidSafe RFC)

---

7. Farming for Safecoin

The main incentive for Vault operators to join the network and cooperate toward the goal of secure data storage is the ability to earn Safecoin. The idea behind Safecoin is similar to that of Bitcoin: to ensure that cooperative participation is a more rational course of action than uncooperative participation. Safecoin can be spent on the network or exchanged for other currencies. The amount of data that a user can store on the SAFE Network depends on the Safecoin balance of the user's account.

KEEP IT SIMPLE!

Farming is the process by which users who lend out spare storage can earn Safecoin, the currency of the SAFE Network. Safecoin can be spent on the network, for example by uploading files.

When a user of the network requests some data, for example by browsing a website, a number of things happen. First the client software makes a request for the required data chunks. This message (a GET request) is then propagated across the network, and when the chunk is found there is a competition between the Vaults in that Section to deliver it to the network where it will be routed back to the requester. The first Vault to deliver will have a chance of being rewarded with one Safecoin. This process is described as a Farming Attempt.

WHAT DOES THAT MEAN?

Farming – a Vault delivers data chunks that it is storing to the network and earns Safecoin in return.

Farming Attempt – by delivering data chunks when requested, a Vault occasionally gains the opportunity to make a Farming Attempt. This consists of sending a validated request to a random Safecoin address. If an owned Safecoin already exists at that address the Attempt fails. If there is no Safecoin there, one is created and awarded to the requesting Vault, i.e., a successful Farming Attempt has been made.

farming_rate – a variable used to attract or discourage farming in order to maintain a certain level of free space (about 30 per cent of the total capacity).

MaidSafeCoin – a cryptocurrency token that will be exchangeable for Safecoin once the network is live.

A successful Farming Attempt will be rewarded by payment in Safecoin. The probability of gaining a Farming Attempt is dependent on the `farming_rate`, which is a variable related to the quantity of available storage resources in the network.

The network will always try to maintain free space of at least 30 per cent of the total capacity to cover a disconnection or outage in certain parts of the network. So when the free space drops below 30 per cent of the total capacity the `farming_rate` will go up and more Farming Attempts will be allowed, and more Safecoins will be awarded. This way Farmers make more money providing chunks to the network. It works the other way around as well. When there are too many Farmers providing storage space the `farming_rate` will go down. This happens automatically, and the effect is to create more incentive for farmers to provide storage when the overall spare capacity is low, and less incentive when the amount of free space is high.

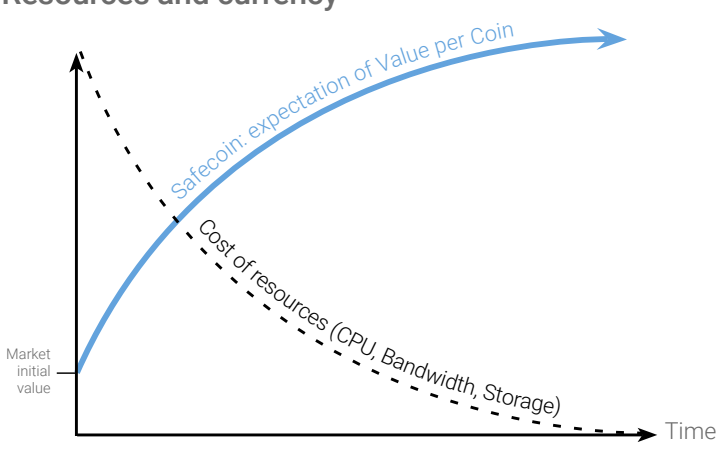
> There's more incentive for Farmers to provide storage when the overall spare capacity is low, and less incentive when the amount of free space is high

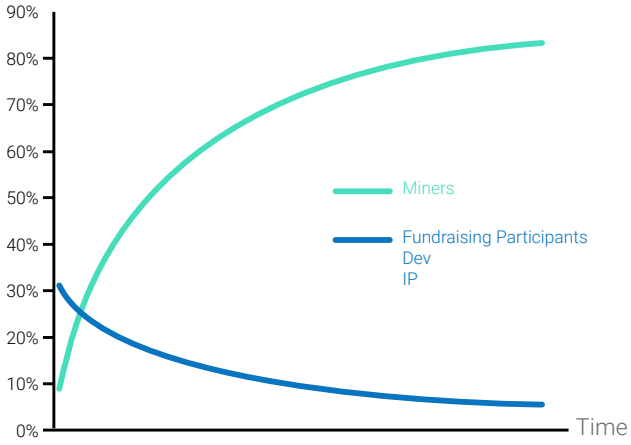
The network will balance itself in a way that's completely independent of the price of Safecoin. By adjusting the `farming_rate` according to the amount of free available space on the network users storing data are charged at the optimum rate. While the storing rate is high (available free space is lower) users are discouraged from storing thus helping to free up more space. This dynamic pricing should translate into very competitive prices for data storage.

Everyone with a suitable device and sufficient bandwidth is welcome to farm and thereby contribute to the network - including data centre owners. To minimize the risk of centralization, however, the use of huge farming rigs will be economically disadvantageous compared with running multiple smaller nodes.

While it cannot be guaranteed, the indications are that bandwidth and storage capacity will continue to increase rapidly for the next decade or two. This will mean data never having to be deleted, with all information stored indefinitely for the foreseeable future.

Resources and currency





Safecoin

Unlike Bitcoins which are numbers on a blockchain, Safecoins are actual files called MutableData (MD) entities (see Chapter 9). When a Safecoin changes hands the transaction is recorded as an entry in the MD. The full history of the coin is not recorded, unlike with blockchain-based currencies, just the previous and current owner, making Safecoin more like cash. The supply of Safecoin will be limited to 4.3 billion coins, and each Safecoin will have its own unique digital identity. Safecoins will be recycled when users exchange them for network services, ensuring there is always a supply for other users to earn.

Note: At the time of writing the SAFE Network is still pre-release (Alpha) and Safecoin is not yet implemented. However, a proxy token called MaidSafeCoin (MAID) can be purchased on cryptocurrency exchanges. When the network goes live MAID will be exchanged for Safecoin on a 1:1 basis.

Tell me more *(click on links)*

Farming (SAFE Network Wiki)
Safecoin Implementation (MaidSafe RFC)
Farm Attempt (MaidSafe RFC)
Introduction to MaidSafe: what it is, how it works, and how it compares to Bitcoin (Blanshey)

8. Vault personas

The Vaults that form the SAFE Network have different functions to fulfil. They route and store chunks of data. They cryptographically check messages and split into new groups or merge back if their group is becoming too small. They also take on more defined roles which are called personas.

The Client Manager

The Client Manager persona holds an account for each Client which is close to it in the network address space – i.e. in its Section. Each Client account is managed by around eight Vaults.

A Client account contains details of the number of chunks of data that have been stored on the network by that Client and how many new chunks can still be uploaded. If a Client account indicates that no more chunks can be put to the network (because of insufficient Safecoin), the Client Managers for that Client disallow any further PUT requests, responding with a LowBalance error.

Clients can retrieve their account balances by sending a specific request to their Client Managers (a GetAccountInfo request).

A SAFE Network Client account is completely different from a Gmail or Facebook account as it is not linked to any identity. The Client Managers know the account balance of a Client but to them it's just an address in the network. They don't know the IP address of the Client nor do they have any knowledge of a username, public identity or anything else that could link the Client to an individual.

Data Manager

The Data Manager persona manages a chunk store where data chunks are held and is responsible for those chunks in its Section. Not every Data Manager in a given Section will necessarily hold that chunk, but each will be aware of which peers do hold it.

Tell me more *(click on links)*

Data Manager - consensus without a blockchain (MaidSafe blog)
Introduction & Technical Overview of SAFE Consensus (MaidSafe blog)
Safe_vault (Github repository)

9. Data types

The SAFE Network provides two data types for storing and retrieving data: MutableData (MD) and ImmutableData. As the names suggests, MutableData can be changed whereas ImmutableData cannot.

MutableData
A MutableData structure is composed of entries. An entry is a key-value pair (e.g. a MD with key 1: value `bananas`, key 2: value `apples` has two entries). Entries can be inserted, updated or removed. A MutableData entity can hold up to 1,000 entries and contain a maximum of 1 Mb of data. MutableData can be used in different ways: Public (e.g. websites), Private (private files) and Shared (private messaging groups) depending on whether or how it has been encrypted.

ImmutableData
An ImmutableData structure can only store a single value. The ImmutableData's network address is derived from the hash of its file content. This means the file cannot be edited in any way after it has been uploaded – any change would alter the hash. A single ImmutableData entity is also limited to 1 Mb, but by using a data map to keep track of their locations files larger than 1 Mb can be split into chunks with those chunks stored as separate ImmutableData entities.

> Data deduplication is a unique feature of the SAFE Network which is achieved by the process of self-encryption

Data deduplication is a unique feature of the SAFE Network which is achieved by the process of self-encryption (see Chapter 6). Two identical chunks will have the same hash value, and therefore only one need be stored on the network. Binary data and other types of files can be good candidates for storing in the network as ImmutableData. ImmutableData types are cached by the clients and fetching the same can be quicker.

SAFE also has a feature called Opportunistic Caching in which more copies of popular data are created closer to where it is being requested, so popular websites and other data feeds will actually speed up as they get more visitors, rather than slowing down as they do on today's Web.

Data is saved using a combination of MutableData and ImmutableData to create an emulated file system on top of the network called NFS (Network File Storage). NFS saves a file's content as ImmutableData. It then creates an entry in a MutableData entity, with the file name as the entry's key and the ImmutableData's address as the entry's value. The file can be updated by uploading a new ImmutableData file and then updating the file's address in the MutableData structure to point to the new file.

Websites on the SAFE Network can be identified using URLs thus safe://service_name.public_id (e.g. safe://mysite.alice). Working in a similar way as the familiar Internet Domain Name System (DNS), these human-readable addresses are translated into network addresses on SAFE using the Decentralized Naming System - also abbreviated DNS.

On the SAFE Network the DNS takes a hash of the Public_ID, so 'alice' in our example becomes a string of 256 characters. The browser takes this hash and uses it as the address to find the corresponding MutableData entity from which it obtains the address of the site on the NFS that matches the Service Name.

The browser finds the Services container for the public ID 'alice', and it retrieves the entry whose key matches the service name 'mysite'. The value stored in this entry contains a reference to the NFS container (a MutableData entity with type tag set to 15002) where the browser can retrieve the names of the files that make up the website along with the addresses of the ImmutableData entities where these files' content is stored. With this information it can start retrieving the website's files and rendering them on the UI.

WHAT DOES THAT MEAN?

Network File Storage (NFS) – an API that allows a Client to access a collection of files stored over the SAFE Network.

Decentralized Naming System (DNS) – analogous to the Domain Name System on the Internet (also DNS), this is a system that translates a human-readable web address into a network XOR address.

Public ID – chosen name for an account (e.g. alice or maidsafe). An account may register any number of Public IDs so long as they have not been registered previously.

Service name – the name of a service such as a website, email or chat app. So if Alice's website was called mysite the URL would be mysite.alice.

Data map – a record retained within a User Account which contains all the information necessary to decrypt a user's file stored on the SAFE Network.

Type tag – allows apps to identify the type of data: 15001 Public ID; 15002 Service Name; 15003 Email ID; 15004 Email Archive.

Opportunistic caching - automatic creation of more copies of popular data close to where it is being requested, so popular websites and other data feeds will actually speed up as they get more visitors, rather than slow down as they do on today's web.

Tell me more *(click on links)*

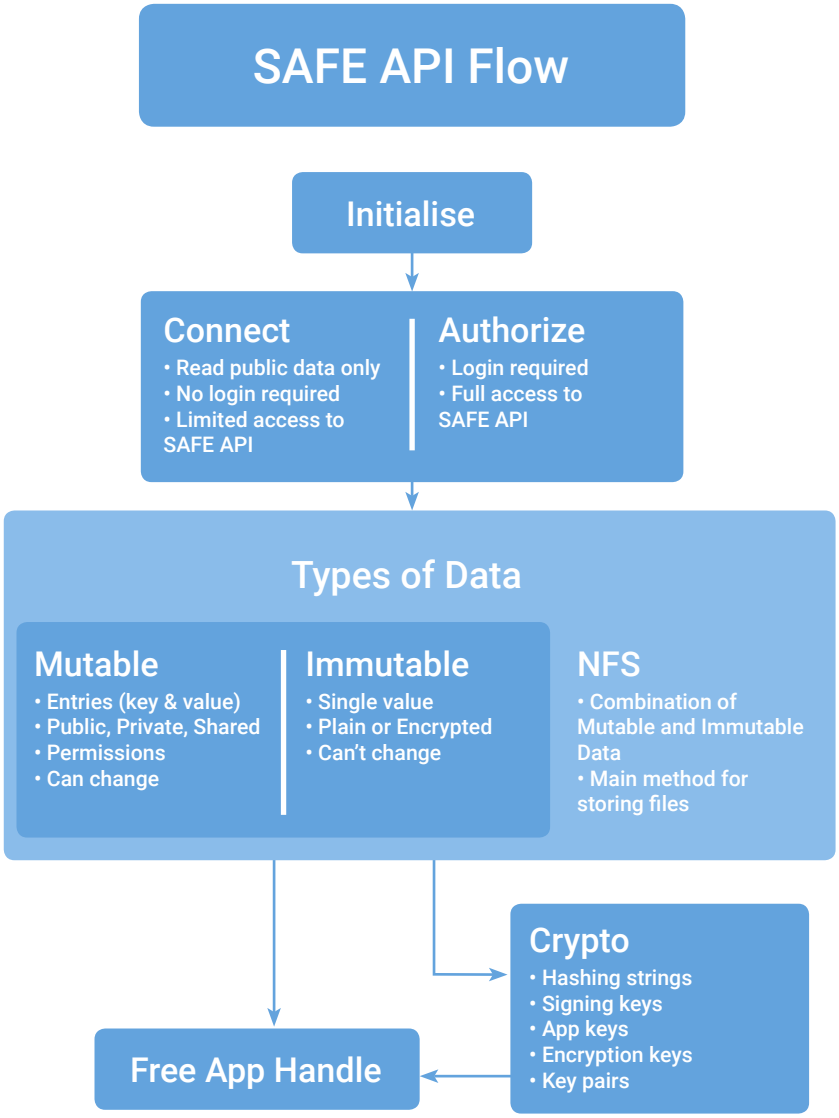
MutableData (MaidSafe RFC)
Distributed network with opportunistic data caching (MaidSafe)
Maidsafe NFS API documentation

10. The SAFE API

The SAFE API is used by developers to interact directly with the SAFE Network. It is available to use in JavaScript, Node.js, Java and C#.

Apps connecting to the SAFE Network are granted different levels of access to data through the API depending on whether they are authorized or not. Apps that are not authorized can only access public data, such as websites. Authorized Apps can access the full range of network functionality.

The SAFE Network has default containers in which certain types of files are stored. For example, `_documents` is used to store document-related data; `_downloads` is the container for downloaded content; `_music` is the place to store music files, and so on. Two special cases are `_public` - to store unencrypted data (the container is encrypted even if its contents are not), and `_publicNames` - to store Public IDs which can be looked up for public information.



Credit: Joseph Meagher

Application development for the SAFE Network is no different from standard practice. There are `safe_app` libraries based on the platform the application is being built on. As mentioned, Node.js, Javascript and C# are best supported at present. Web applications can be built using the DOM API of the Peruse browser.

Authorization

Applications need to be authorized before they can access data on the network.

In a similar way to the familiar OAuth process, the application sends a request using the library for authorization. When the authorization is approved by the user, the application receives a token which is used to connect to the SAFE Network. Authorization is achieved via an application call to the Authenticator, which is currently bundled with the SAFE Network browser Peruse.

Authorization is fine grained. An application can create its own container and request access to default containers or other applications' containers through the authorization request. READ, WRITE, UPDATE, DELETE, MANAGE permissions can be requested for every container.

The API features many methods for allowing apps to interact with and deploy MutableData and ImmutableData types (see Chapter 9) and to write and retrieve data from the network.

CipherOpt and Crypto APIs

The `safe_app` library also provides crypto API functions. The safeCipherOpt API provides functions to create different encryption options to be applied while storing the data in the network. Sign key handling APIs were a work in progress at the time of writing.

There are three types of CipherOpts:

- Plain - Data will not be encrypted.
- Symmetric - Data is encrypted with a symmetric key.
- Asymmetric - Data is encrypted using a key pair.

The safeCrypto API functions provide handy cryptographic functions, including hashing and generating key pairs.

DOM API

A web application can communicate with the SAFE Network and Authenticator by interacting directly with the SAFE Browser's DOM API, i.e. window.safe* functions.

This API is very similar to the Node.js API, the main difference being that the web application receives handles for each of the objects that are instantiated when interacting with the API, e.g. SAFEApp and MutableData instances. The web app is required to release the handles provided by calling a specific 'free' function on each of the tokens received.

Tell me more *(click on links)*

SAFE Network DOM API
SAFE Network Node.js API
safeCrypto API
safeCipherOpt API
New Auth Flow (MaidSafe RFC)
Async safe_core (MaidSafe RFC)

11. The promise of the SAFE Network

The SAFE Network is still in development. While many features and functionalities have already proven themselves under test conditions others, including Datachains and Safecoin, are still to come. As with any cutting-edge experimental technology, the proof of the pudding will be in the eating.

But let's assume for a moment that the network is successful and is widely adopted for various use cases including Internet browsing, IoT connectivity, data security, personal information management, medical records and more. What would that world look like?

First, most cyber attack strategies deployed today would be dead in the water. DDoS would not work as the network would simply route around the affected nodes. Viruses and malware would be extremely limited in their depth of penetration. Ransomware would not raise a single dollar. Cyber-kinetic attacks aimed at disabling national infrastructure or taking control of a driverless car would be extremely hard to pull off. Medical records and other personal data would be ours and ours alone, to share as we see fit.

> There would be a rebalancing of power from the data haves to the data have-nots

For users there can be a single sign-on to multiple services. XOR networking with opportunistic caching promises faster speeds, data storage should be extremely cheap and the network will offer high levels of availability. For developers having a single storage architecture to address has the potential to simplify the systems programmer's job. And data deduplication would allow for both simplicity and resource savings.

The Internet giants of today would no longer be able to harvest our data without our say so, nor could government spooks eavesdrop over their shoulders. There would be a rebalancing of power from the data haves to the data have-nots. Censorship would be impossible and data could be not be erased.

Because the cost of entry will be low and access unrestricted, the ongoing net neutrality debate will end, and neutrality will have won the day. People in places with poor or restricted access to information will have those blockers lifted. Some may even make a decent living earning Safecoin.

New business models based on consent would spring up in a world where data storage and networking and eventually compute is a commodity, and the world of information will be a much more level playing field.

Isn't this all a bit idealistic? Well yes of course, but that's the nature of visions. Not everything will work as planned and we need to be hard headed about that. Techno-utopianism is a dangerous thing. There are both predictable and unpredictable consequences of taking on the status quo, and indeed of deploying any new technology, and not all of them will be positive.

Nevertheless, given where we are today, and where we are headed with the IoT, something like the SAFE Network is most definitely needed to redress the power imbalance and to secure the data-driven future.

About MaidSafe

MaidSafe is a company founded by David Irvine in 2006, which has a mission to provide security and privacy for everyone by building a better internet platform. This new platform is the SAFE Network, which is the world's first autonomous and decentralised data network. The network is made up of the unused hard drive space, processing power and bandwidth of its users. The SAFE Network will include storage, peer-to-peer communications, transactions, internet functionality and a wide variety of apps to name a few of its features.

Visit: **www.maidsafe.net**

This paper was written and produced by members of the SAFE Network Forum independently of MaidSafe.

Join the debate at **safenetforum.org**