Litepaper

Abstract

BlendSafe is an omnichain wallet developed on the Internet Computer platform, with a primary focus on ensuring programmable transaction management across diverse blockchain ecosystems.

Each BlendSafe has access to a signing and address generation mechanism unique per wallet. The message signing is cryptographically distributed in a network that allows for a decentralized, programmable signing flow from within smart contracts.

Building on top of a robust multi-signature scheme BlendSafe is a versatile and extendable infrastructure to orchestrate and manage transactions across different chains, including non-smart-contract platforms such as Bitcoin.

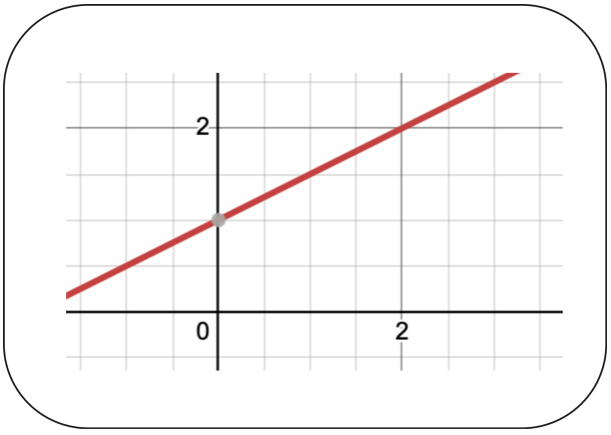
BlendSafe has developed a full-featured PoC that is ready to demo and is currently building an MVP with multichain yield farming as an initial use case.

A no-scary-math-symbols primer to BlendSafe cryptography

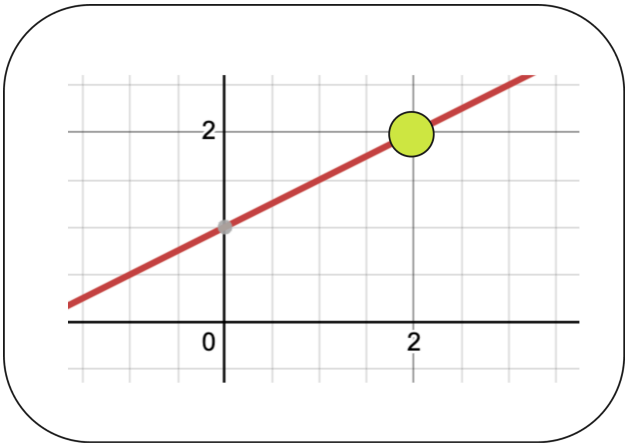
BlendSafe stands on the shoulders of an important cryptographic concept, known as *ECDSA Threshold Signing*, that allows us to store a private key in a decentralized way.

A lot of the defining features of BlendSafe such as cross chain treasury and multichain signing as well as use cases of our infrastructure e.g. real Bitcoin AMMs and the likes are based on this technology. Here's how it works.

Imagine a line like this:

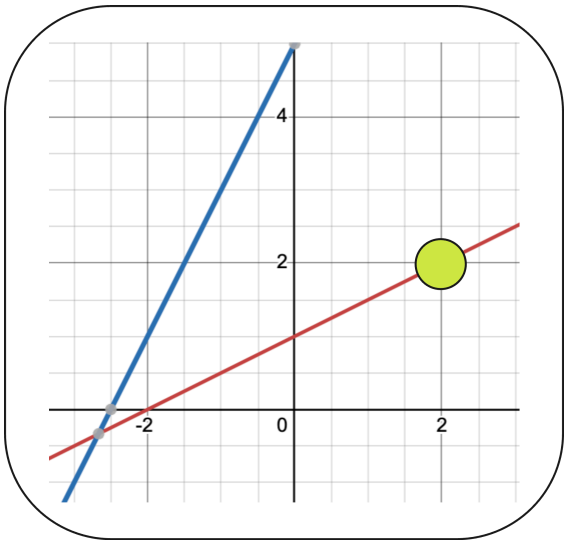


As you might remember from highschool, this line has an infinite number of points. Let's focus on exactly one point for now, the point at *2;2*:

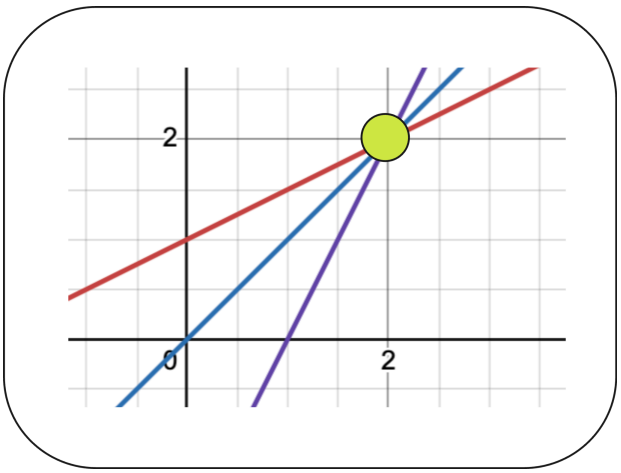


Let's assume that you don't know the underlying line - what information can you gain from this point about the line? It turns out: Not that much.

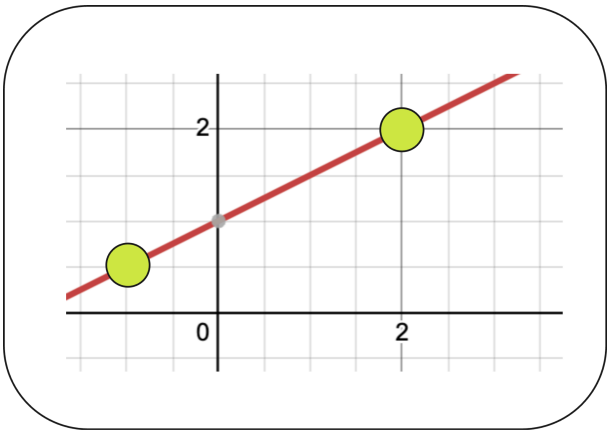
Sure, you can rule out an infinite number of lines that do not have the point *2;2*, so e.g. this blue line can't be the line we are looking for:



However, there are still an infinite number of lines possible that do share this point, here are a few:



It actually turns out that you need to have exactly two points to uniquely identify a line, but any two points on the line will do:



A line is what is known in the business as a polynomial of degree one. The degree is determined by the highest power of the variable x.

A polynomial is something in the form of $ax^3 + bx^2 + cx + d$, where the degree is not limited at all. So the number of coefficients (a, b, c and d in this example) can be quite large.

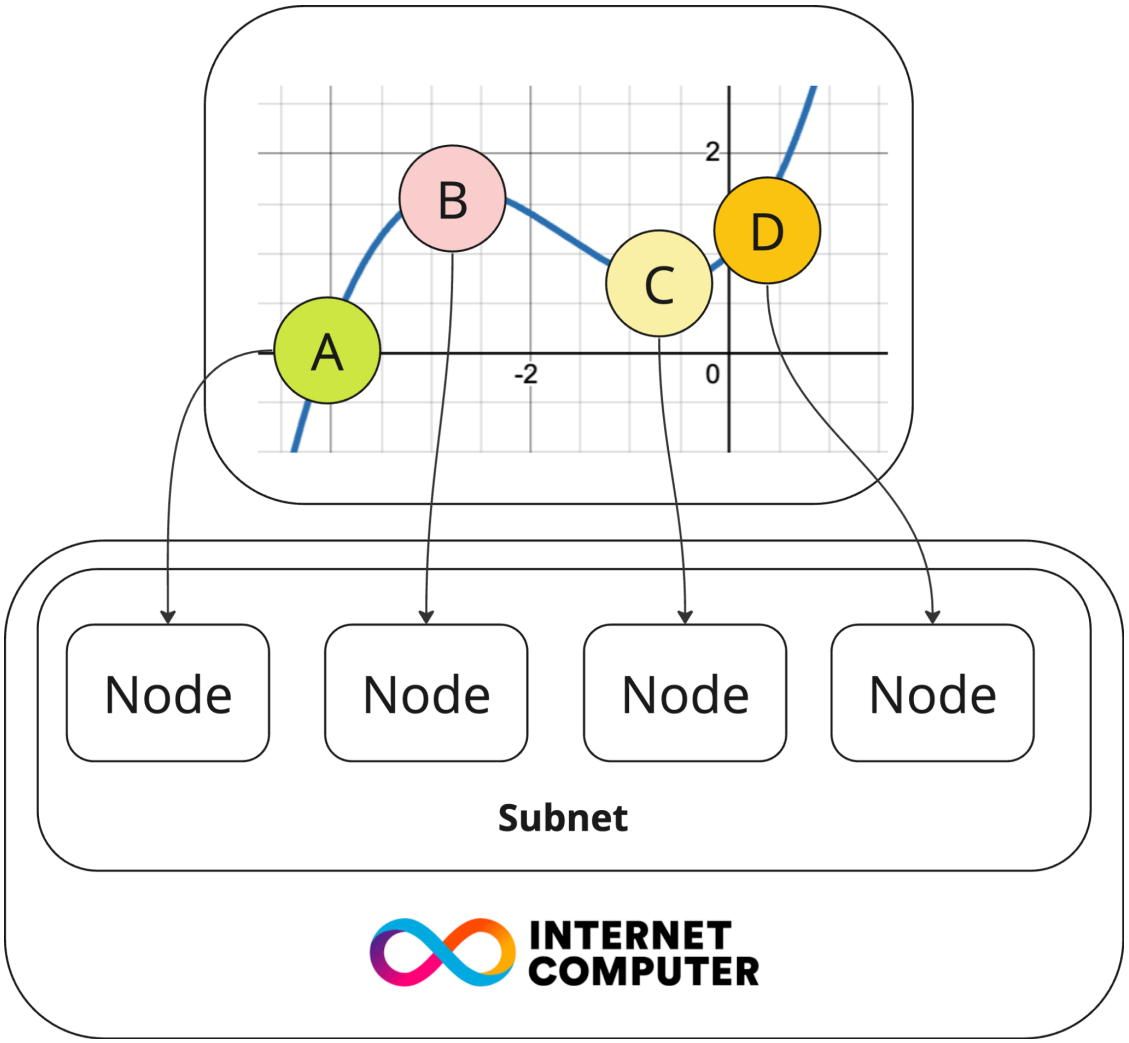
The fundamental theorem of algebra states, in loose terms, that the number of roots is equal to the degree of the polynomial. A few pages down the math book follows polynomial interpolation - the mathematical concept related to this intuitive idea that we always need one more point than the degree of the polynomial to uniquely identify the underlying curve.

Following from this: We can split the uniqueness of a polynomial in a set of points. Each point contributes a piece of information in order to construct the polynomial back from the raw points.

What we want to design is a polynomial with enough coefficients so we can encode a private key within these coefficients! A private key is just a number, usually a big one and we can split this number up into smaller numbers and make those the coefficients of our polynomial!

There are a few more details in the actual implementation and we've simplified a bit for clarification (e.g. we didn't discuss elliptic curves and finite fields that have some neat cryptographic properties), but we're still reasonably close to how it works under the hood.

BlendSafe utilizes the subnet architecture of the Internet Computer, that distributes the number of points required cryptographically secure among the nodes of a subnet.



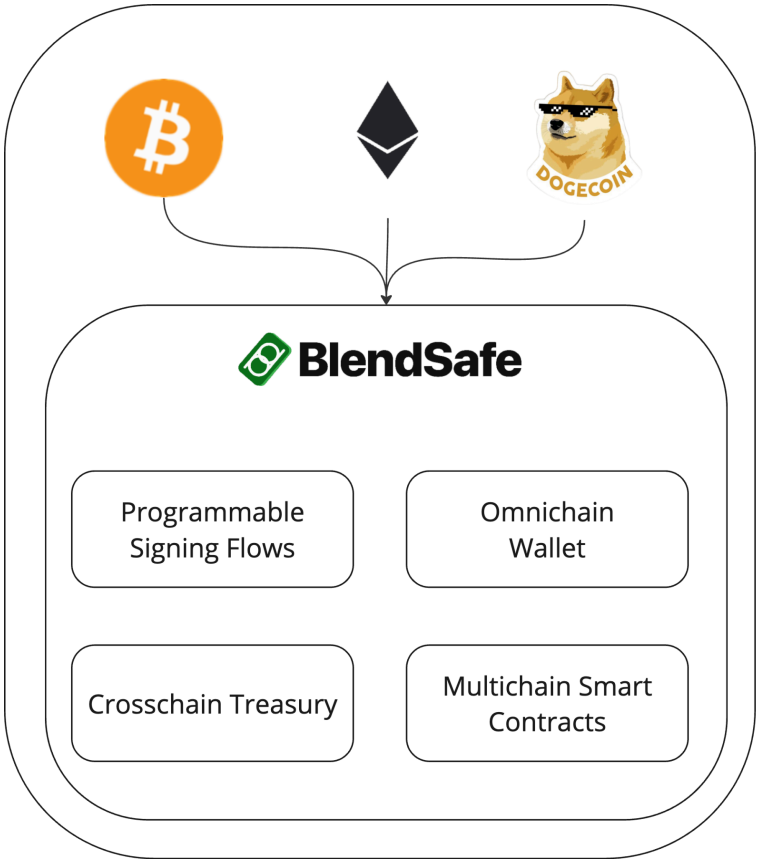
Remember that the coefficients of the underlying polynomial are determining the private key that we want to share across a network.

Now the nodes need to jointly come together and contribute their point to the signing process. No node on its own has the knowledge to do so.

There are various ceremonies such as partial signing or multi-part computations that allow the final signature being created without all parts involved being revealed to a single party. The number of coefficients and therefore required nodes and various other properties - e.g. redundancy in order to allow for thresholds - can be tuned to form a robust and secure mechanism.

Messages are signed by the network with its own private key that is distributed among nodes whilst nobody actually knows the underlying private key in its entirety.

BlendSafe packages this functionality into a protocol and infrastructure that can sign arbitrary messages and orchestrate them across different blockchains. Each BlendSafe has its unique and protected set of public / private keypairs that can be used to derive addresses of any blockchain.

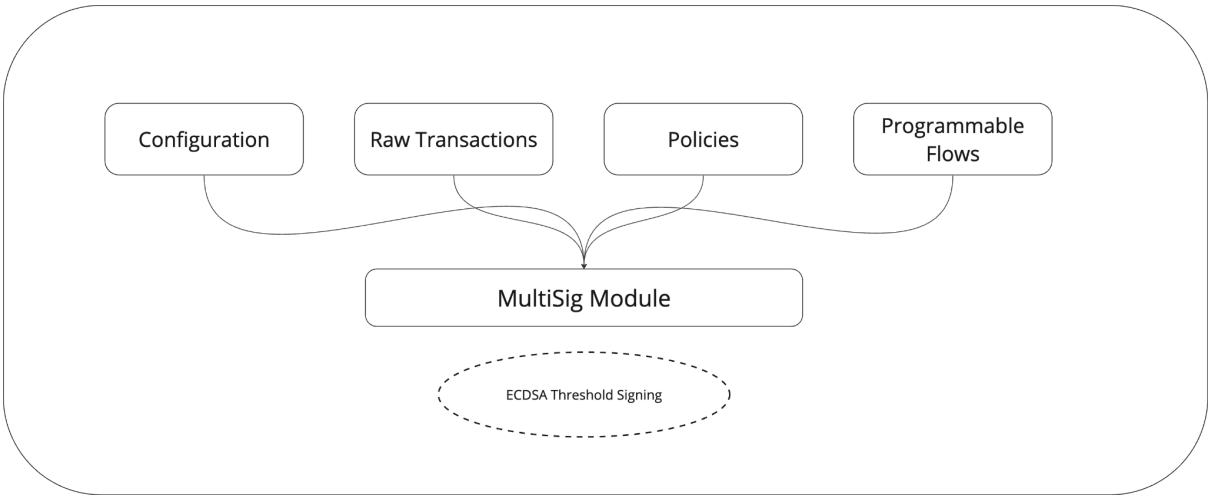


BlendSafe is a programmable omniwallet for the multichain world and an infrastructure hub to build trustless, decentralized and programmable transaction flows agnostic of the underlying blockchain technology.

Architectonic Overview

At the heart of BlendSafe lies a multisignature contract alongside an abstraction for programmable flows.

It allows for multiple inputs into the signature engine and a fine-grained control of what a BlendSafe will ultimately sign or reject:



Configuration manages the multi-signature transaction rules, such as the required number of signatures and authorizing signatory power. This module requires multisignature capabilities but does not utilize crosschain signing, as it configures the state of the blend safe itself. Note that a multisignature contract with just one signer is a special case that would be equal to a single-user wallet.

Raw Transactions are the transaction payload of a blockchain, e.g. a bitcoin or ethereum transaction. They can be sent to a BlendSafe alongside and be authorized by the multisignature committee and subsequently be broadcasted to the target chain. The BlendSafe maintains its own address and becomes the proxy holder of any assets sent to this address.

Policies can be put into place for transactions that can bypass the MultiSig Module. For this to happen, the transaction must adhere to a format that is greenlighted by the policy and the policy itself has to be authorized by the MultiSig committee. An example use case would be a hedge fund that uses BlendSafe and wants to preauthorize a daily spending limit for its trading desk and a less-strict process (e.g. manager only must sign off higher limits) for bigger transactions to streamline strategies agnostic of the chain where vaults are deployed.

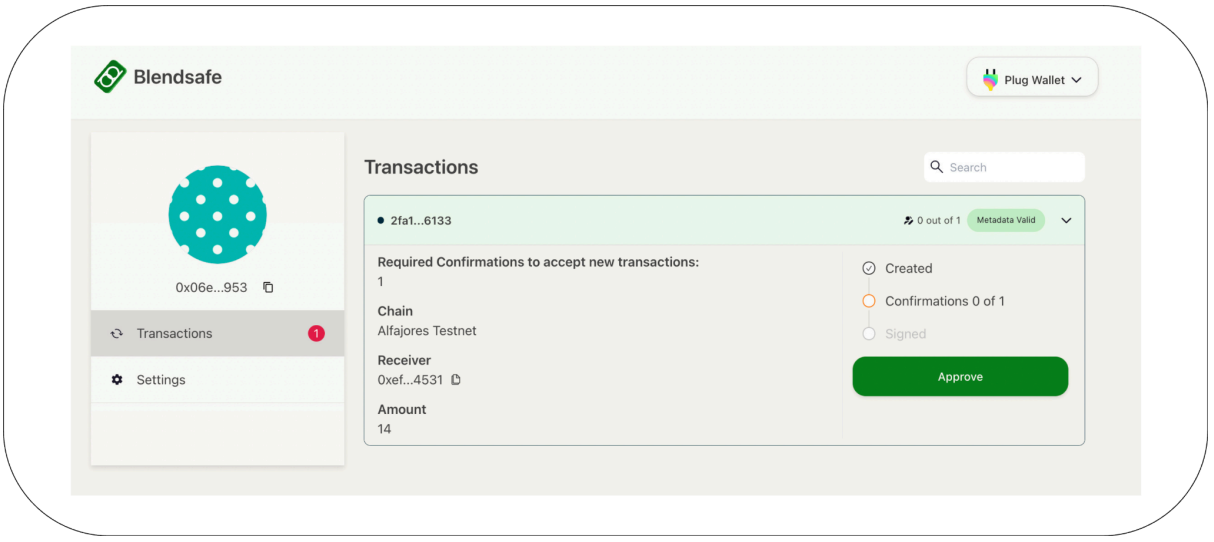
Programmable Flows are crosschain flows that require different transactions on different chains that have to be executed in order. An example would be to bridge eth from arbitrum to

optimism to invest it into a yield opportunity or taking an arbitrage opportunity over there. This sequence of transactions can be preauthorized within the programmable flow and be executed asynchronously. Once preauthorized, the broadcasting and synchronization of the actual transactions becomes a permissionless agent that are performed by bots and programmable agents within the BlendSafe infrastructure.

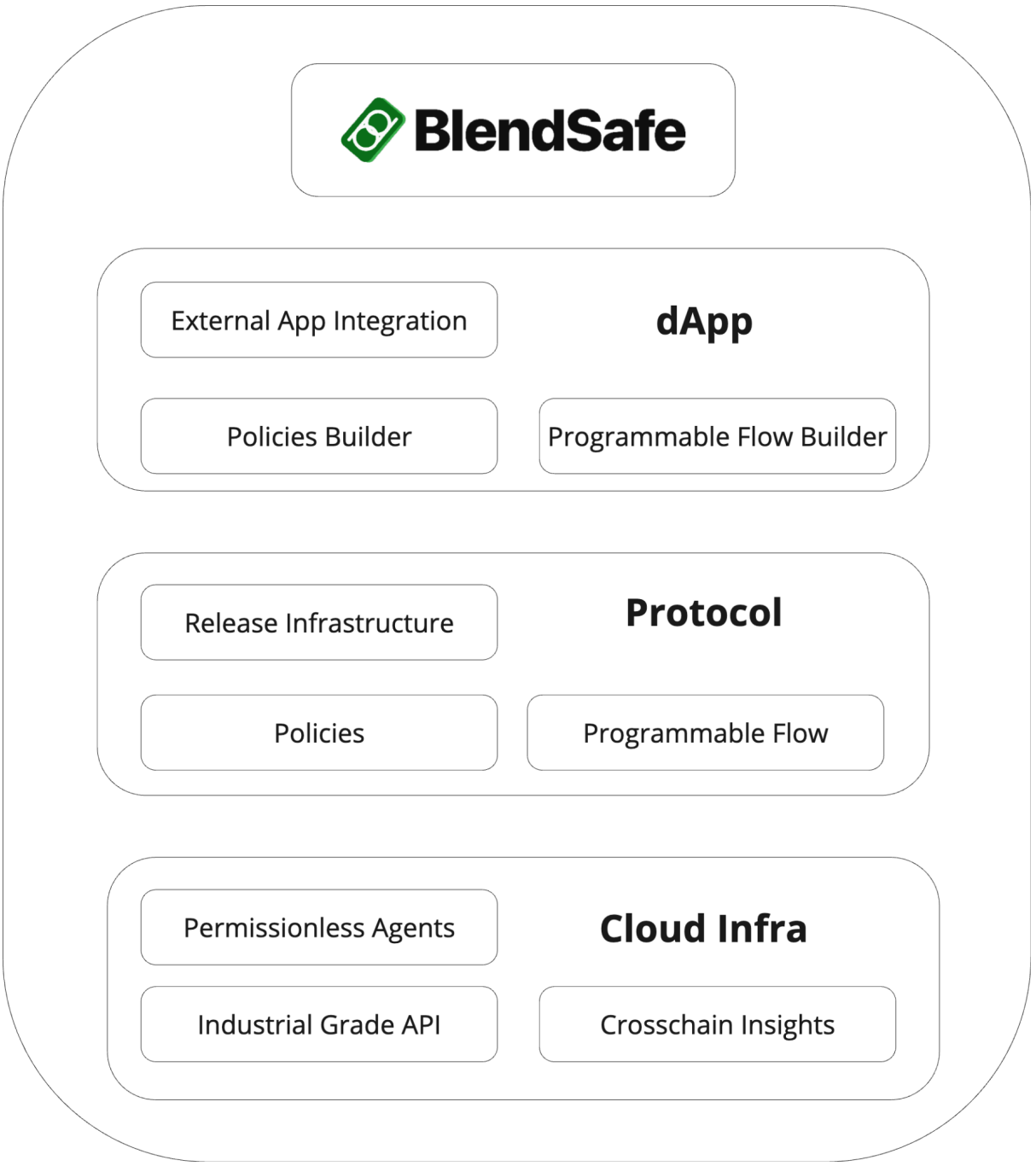
Roadmap

Ultimately all of the aforementioned features ultimately rely on a multisignature scheme that is able to sign arbitrary messages in a fully decentralized manner.

This flow has been fully implemented with a robust implementation on the Internet Computer alongside with an end-to-end frontend to orchestrate the flow. Functionally this is equal to a decentralized omnichain multisignature smart contract, it has features for transaction validation, signing flows, configuration and omnichain address generation.



In order to transform this into an infrastructure the team aims to implement a set of features into the protocol that forms our Version 1.



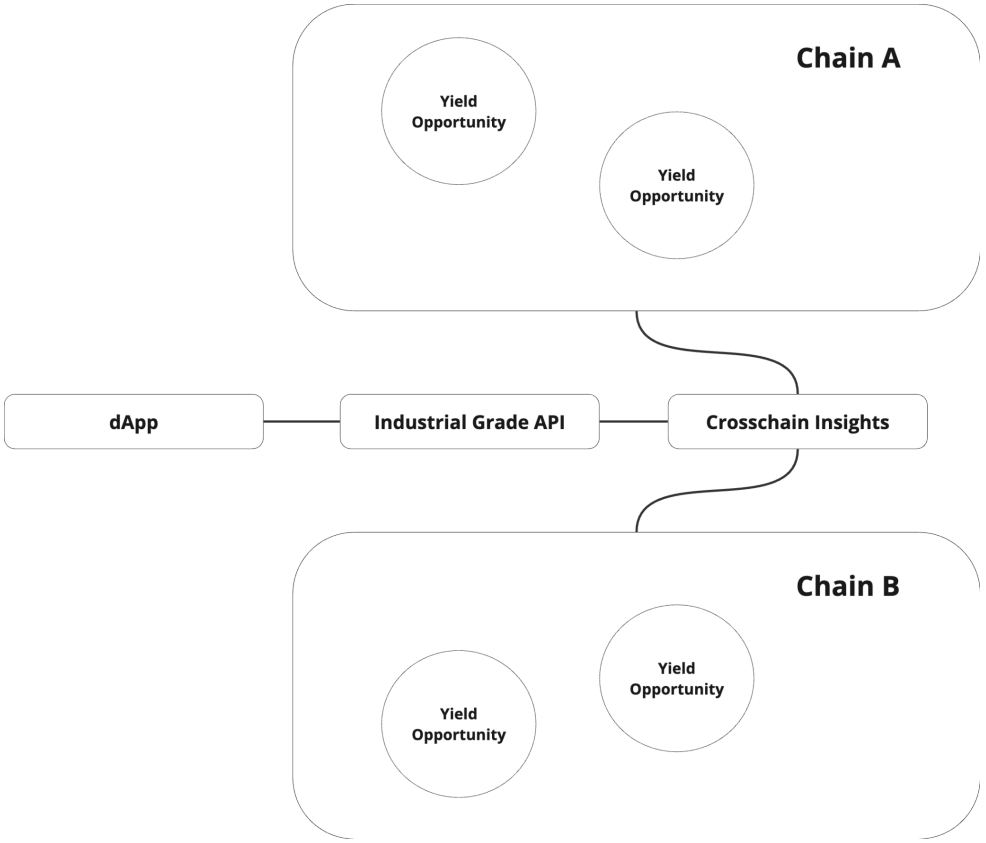
Those are components in the dApp and in the protocol for composable policies and flows. This is the biggest chunk of work for both protocol architecture and UI/UX integration.

We as well want to change the release infrastructure in order for communities, organizations and individuals to customize the infrastructure with extensions and third party tooling for their needs. Lastly the protocol and dApp is accompanied by a set of cloud infrastructure such as data for transaction states, treasury insights and so on. Permissionless agents are bots that perform pre-authorized actions by protocols once some specific preconditions are met (such as an asset bridge has been finalized).

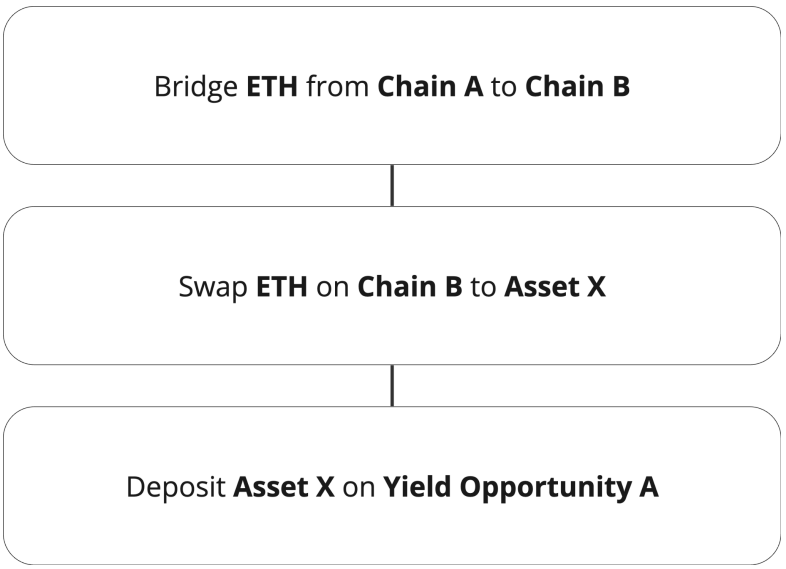
This setup defines our MVP that we are eager to launch in a few months. Whilst the protocol is already useful as-is, this will mark the moment where real omnichain operations become seamless and users can interact with dApps with the underlying chains being abstracted away: Cross Chain Yield Aggregation.

Sample Use Case: Cross Chain Yield Aggregation

Yield Opportunities occur across different chains and typically users learn about them on social networks or aggregation sides. Whilst all apps building on BlendSafe are free to use their own data sources, we maintain our own infra to provide our dApps and services with the data required to render their services.



Once a yield opportunity is required typically a series of steps is required on behalf of the user in order to take advantage of said opportunity. These series of steps are programmable flows within BlendSafe:



This series of synchronous events can be preauthorized in one go for a given timeline. The user can ignite this flow at the beginning and subsequently requires a set of actions that have to be taken at a later stage.

This is where **permissionless agents** come into place: Little bots that observe the state of our BlendSafes and the connected chains and can trigger the execution of transactions after a specific precondition is met - e.g. a specific amount of eth has arrived on Chain B and can now be swapped to Asset X. Since the user has already pre authorized the programmable flow this interaction does not require permissions nor funds - the BlendSafe is covering the transaction fees.

The process can be fine tuned with **policies** where e.g. third party yield hunters get limited privileges on a BlendSafe in order to take Yield Opportunities.