Présentation Technique
Vega

vega.xyz

Juillet 30, 2019

1 Introduction

Vega est un protocole technique qui autorise les réseaux décentralisés publiques ou privés à faciliter l'automatisation totale du commerce et l'exécution des produits financiers. Les réseaux sont sécurisés avec une couche de consensus tolérante aux fautes byzantines et implémentent de la vente à découvert pseudonyme utilisant un programme original de commerce base sur l'incitation de liquidité pour résoudre le problème d'attirer et allouer des ressources importantes du marché dans un système décentralisé.

Les choix technologiques et le modèle de Vega sont poussé par des critères qui s'alignent avec notre approche d'ingénierie et la vision générale de Vega.

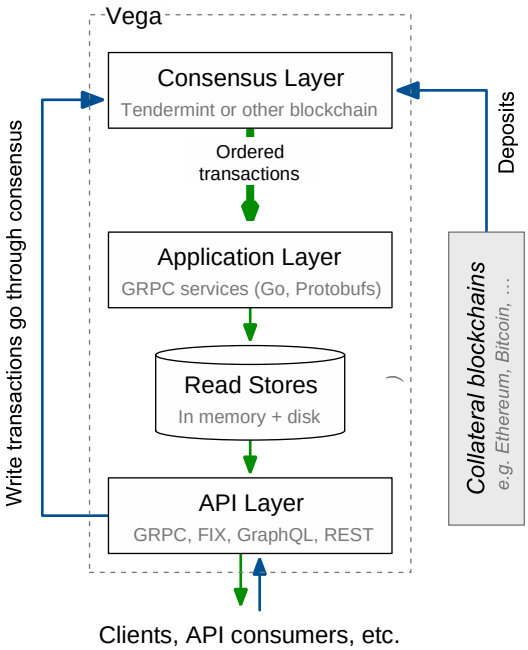
- **Sécurité et exactitude:** Vega doit être conçue avec sécurité et testabilité en tête.
- **Performance blockchain:** Vega sera et restera à l'avant-garde de la performance des blockchains publiques en terme de latence et de débit.
- **Performance d'application:** la couche application doit performer à Egalite avec les systèmes professionnels de commerce ; inclus sont ceux qui ne sont pas basé sur la blockchain.
- **Flexibilité:** Vega ne peut être liée à aucune blockchain spécifique ou quelconque cryptomonnaies pour les opérations ou le commerce et les règlements.
- **Expérience développeur:** Construire au-dessus de Vega doit être facile pour tous les types de développeurs et les cas d'utilisation.

Ce court papier couvre quelques un des aspects plus techniques des réseaux utilisant le protocole Vega, et de la référence d'implémentation développée par l'équipe Vega. Pour une description du protocole lui-même, consultez le white Paper du protocole Vega[1].

2 Architecture

On utilise *Command Query Responsibility Segregation*[2] (CQRS) et un design modulaire pour imposer une séparation stricte entre le *consensus* (blockchain), *l'application*, et les niveaux d'*API* dans l'implémentation de référence. Les transactions sont des messages **Protocol Buffers** passé de niveau consensus au niveau application, dans un ordre qui est garanti par l'algorithme de consensus pour afin d'être le même pour tous les nœuds[3].

Les nœuds Vega lisent(read) d'autres blockchains qui sont utilisées pour garantie, et publient(post) des transactions sue le réseau Vega quand ils reconnaissent un *deposit(dépôt)* ou un *retrait(withrawal)* sur cette blockchain. Vega, par conséquent , supporte des garanties d'une multitude de blockchains, et des règlements cross-chain.



[1] https://vega.xyz/papers/vega-protocol-whitepaper.pdf

[2] https://martinfowler.com/bliki/CQRS.html [3] Vega par conséquent requiert une blockchain a finalité immédiate.

Niveau blockchain

Vega opère son propre réseau blockchain *proof-of-stake* pour la performance, la scalabilité, et la flexibilité. On utilise actuellement **Tendermint**[2] comme consensus, qui fournit un temps de bloc d'une seconde et peut traiter de 1000 jusqu'à 4000 transactions pas secondes(tps). Les transactions subissent une validation initiale avant d'être acceptée, et sont traitées par l'application Vega quand chaque bloc est finalisé.

La séparation entre la blockchain et l'application signifie que Vega est *blockchain independent*(indépendante de la blockchain) , parce que le niveau application peut traiter des transactions valides et ordonnées de n'importe quelle source. Ceci autorise Vega de migrer a un nouveau protocole de consensus si une meilleure technologie devient disponible. L'indépendance blockchain signifie en outre que le protocole Vega et l'implémentation fondamentale peut facilement être réutilisée dans d'autre environnement basés sur serveurs qui sont décentralises, distribue ou même traditionnel pour satisfaire une plus large gamme de cas d'utilisations.

Niveau application

L'application (alias *trading core*) traite les transactions entrantes du niveau consensus —sous la forme de messages protocol buffers — séquentiellement et de façon déterministe. Cela garantit que tous les nœuds arrivent at exactement au même état. Ce protocole en entier a besoin actuellement de juste 11 types de transactions :

governance(gouvernance): proposition d'ouvrir un marché, proposition de fermer un marché, proposition de mettre à jour un paramètre, voter sur une proposition.

trading(commerce): soumettre une instruction (order), modifier une instruction, annuler instruction, notifier un observable (oracle data).

collateral(garantie): notification de dépôt (sur la chaine de garantie ou collateral chain), demande de retrait, validation de retrait.

Le trading core dans notre implémentation est écrit entièrement en **Go**, choisit comme c'est un langage mature qui s'est avéré idéale pour écrire des applications serveur sûre, maintenable, et haute performance. L'application Vega est divisée dans des composantes fonctionnelles, qui sont décrites plus loin dans ce document.

Read stores

Les *read stores*, implémentées aussi en Go, ingèrent et interprètent un courant d'évènements ou stream dans le trading core et conservent les données résultantes en mémoire et des structures de données sauvegardées sur disque désignées pour servir les requêtes API ou API queries. Les évènements incluent les changements d'etats pour les commandes, exécutions d'échanges, chiffres de prix et risques, règlements de cashflow, dépôts et retraits, et les actions de gouvernance.

Niveau API

Les clients se connectent a de divers API, qui effectuent des *queries* ou requêtes sur les read stores et publient des *commands* sur le niveau du consensus. Les APIs sont désignés pour fournir une excellent expérience de développement pour de différent types de systèmes clients.

Le GRPC et FIX APIs assurent un trading haute performance et une intégration de système de données ; tandis que les API REST et GraphQL, qui incluent un support pour diffuser les données du marché, sont conçus pour une construction rapide et facile d'applications front-end haute performance et le scripting.

3 Composantes trading core

Le trading core de Vega est une application modulaire avec une séparation fonctionnelle entre les composantes qui permet une configuration maximale, incluant un usage sélectif d'un sous-ensemble de composantes dans des déploiements permissif qui ne requiert pas la fonctionnalité totale du protocole.

- **Matching engine:** un carnet de commandes à cours limité qui fonctionne soit en trading continu ou en mode auction(enchères) dans les marché ouvert (open markets). Il soutiendra également *request for quote*(RFQ) et *matched trades* pour trading sur les marchés *over the* counter(OTC).
- **Risk engine:** évalue le *risk model* ou modèle de risque pour chaque marché afin de calculer les exigences de marge des position ouverte nette de chaque participant. Le risk engine assure ensuite qu'une marge suffisante est allouée à chaque position nette avec des requêtes d'allocations pour le gestionnaire de garanties ou collateral manager, et sinon, initie des échanges *closeout* ou de liquidation.
- **Collateral engine:** maintient la solde pour chaque crypto-atout ou crypto-asset déposé par chaque participant selon le traitement des notifications des dépôts par les blockchains de garanties et les instructions de règlements du settlement engine. Il traite aussi les allocations de garanties aux marchés de marge.
- **Settlement engine:** génère des instructions de règlement pour l'engin de garanti ou collateral engine quand les marchés arrives à maturité, des produits créent des intérims de cashflow, et chaque fois une position est fermée entièrement ou partiellement. Emploie aussi *l'algorithme de résolution de position* si i l y a un déficit au règlement.

[4] https://tendermint.com/

- **Governance engine:** gère la création et la fermeture de marchés sur le réseau, et la modification de paramètres. Des actions sont prises en réponse à un vote qui se produits suite a l'acceptance d'une proposition de transaction par un ou plusieurs participants.

4 Performance

Blockchain

- L ;utilisation de proof-of-stake permet d'importantes améliorations(par plusieurs ordre de grandeur) de performance sue d'existantes chains proof-of-work.
- Chiffres de performance Tendermint:

  Temps de block(block time): 1 seconde

  Latence: 0.5–1.5 seconde

  Débit: 1000–4000 tps[3]

- La séparation architecturale du niveau blockchain du niveau applications permet à de nouvelles améliorations dans le future si des solutions plus performantes deviennent disponibles.

Application

- Le matching engine de Vega à été teste avec des carnets de commandes contenant des millions de commandes et à traité avec consistance les instructions de soumissions, modifications et d'effacements en 5–15μs sur un ordinateur portable standard.
- Les modèles de risque sont exécuté se le « métal nu » et peuvent par conséquent prendre des capacités totales du matériel sous-jacent(hardware), y compris l'accélération GPU et la parallélisation.

Niveau API

- L'API GRPC utilise des messages Protocol Buffers binaires, et les données sont récupères par des sauvegardes en mémoire optimises pour des requêtes haute performance.

5 Echelonnement ou Scaling

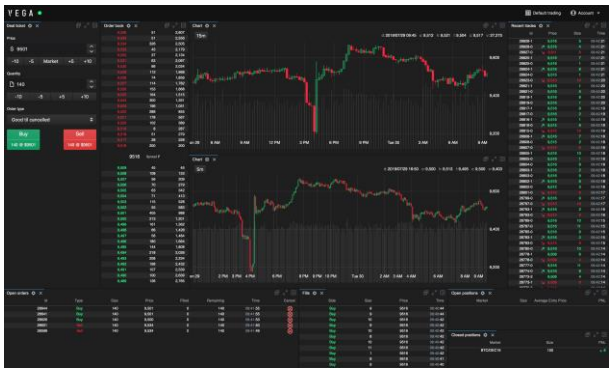
Vega a été désigné des le début pour échelonner au-delà de la phase d'adoption précoce, jusqu'au niveau requis par des usages réels et sérieux, dans le cadre de l'infrastructure du monde financier.

- Le protocole permet au réseau d'être « éclaté » ou *sharded* par *risk universe*[4], ce qui signifie effectivement que chaque marché peut avoir son propre réseau. Ça signifie que Vega possède essentiellement une évolutivité horizontale illimitée et peut par conséquent approvisionner n'importe quel nombre d'instruments et de marchés.
- Chaque blockchain, et par conséquent chaque marché, est limitée en débit de transaction par la technologie blockchain utilisée aussi bien que le réseau physique et l'infrastructure computationnelle en place. Cette limite augmentera avec le temps avec les mises à niveau matérielles et logicielles, qui peuvent comprendre migrer a un diffèrent protocole de consensus et d'implémentation. Pour les marchés qui sont à la limite, l'agrégation de transaction permettra la participation de continuer à augmenter, au prix de latence pour quelques transactions.
- Les modèles de risque peuvent être démarrés en mode asynchrone, avec des résultats passant a travers le consensus, permettant des modèles de risque de type Monte Carlo plus lent mais plus complexe, aussi bien que des calculassions de formes fermées.

6 Interface utilisateur (UI) du trading de référence

Tandis qu'on excepte plusieurs utilisateurs de se connecter au divers API directement, Vega fournira une référence d'implémentation complétement fonctionnelle de l'interface ou UI du trading pour Vega.

La capture d'écran ci-dessous est d'un prototype préliminaire de cette application, qui va être développée en une application de trading d'un degré et d'une qualité professionnelle avec une variété d'outils de trading et de gestions des risques. Le front-end trading sera également extensible avec des extensions de troisième groupe (third party).



La référence UI est une *dApp* ou decentralized application- de l'anglais, application décentralisée, composée de JavaScript + HTM, construite avec **TypeScript**, **React**, et **GraphQL** (Apollo) et peut être démarrée d'un serveur hotté (hosted server) ou par la machine locale de l'utilisateur directement. Comme le reste de Vega, elle est complètement

[5] Performance de production estimée base sur des tests préliminaires .

[6] Un marché ou ensemble de marches en relation qui autorisent des prises parfaites, par exemple, plusieurs maturités de futurs contrats, ou options avec des strikes diffèrent sous-jacents.

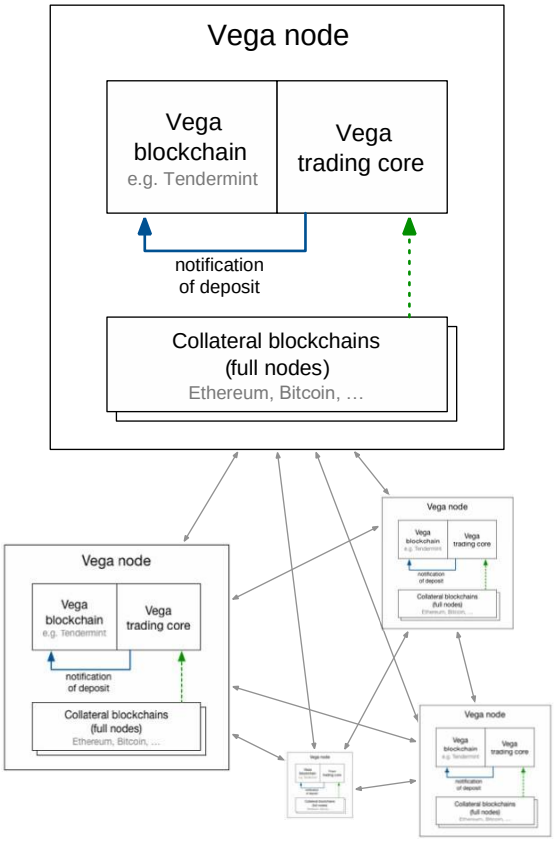
décentralisée, exigeant seulement une connexion à un nœud Vega complet ou passif pour démarrer.

7 Garantie chaine multiple ou multi-chain collateral

Vega est conçu pour le trading et les règlements haute performance, mais ne « host » pas les capitaux en échange. Au lieu, Vega est conçu pour marcher avec un large éventail de digital coins, tokens et autre atouts stockés sur les blockchains existantes.

Cependant, bien qu'elle gère les soldes de n'importe qu'elle capitaux déposés avec le réseau, la blockchain Vega elle-même ne tient pas les crypto-atouts. Dans ce sens, Vega est une solution « second niveau » (*second layer*) ou *side-chain*, fournissant fonctionnalité en plus de celles fournies par d'autres blockchains comme Bitcoin et Ethereum, qui tiennent ;est atouts sous-jacents étant échangés.

Pour achever cela, chaque opérateur de *nœud complet* Vega devra également gérer des nœuds complets sur chaque blockchain collatérale supportée.



Par exemple, un *smart contract* sur la blockchain Ethereum est utilisé pour tenir des fonds déposés à Vega et règlementer les retraits. Paiements dans le smart contract sont reconnus et prêt à l'emploi sur Vega quand ils ont été observés par deux tiers des nœuds. Pour retirer des fonds, une requête se fait via une transaction Vega. Si elle est valide, les nœuds Vega signent une transaction de retrait a signature multiple ou *multi-sig* qui peut être publiée sur la blockchain Ethereum une fois qu'elle reçoit assez de signatures pour achever la transaction ; de nouveau, deux tiers des nœuds.

Une approche similaire peut être prise pour chaque blockchain collatérale de garantie, mais en pratique nous allons, a un moment donné, chercher des solutions multi-chain/cross-chain pour supporter des collatéraux de chaînes sources plus larges avec une seule intégration.