GORMOS: 分散型アプリケーションのための高パフォーマンスでスケーラブルなプロトコル設計

Loi Luu, Yaron Velner, Alex Xiong
{loiluu, yaron, alex}@kyber.network

翻訳: 篠原航, 加嵜長門
{shinohara-wataru, kasaki-nagato}@dmm.com

Working draft, last updated September 12, 2018

Abstract

我々は、高頻度なインタラクション、低レイテンシな決済、および優れた相互運用性を可能にする、高性能分散アプリケーションのための最初のプロトコル設計である GORMOSを紹介します。 GORMOSは Plasma [8] や sharding [11] や PeaceRelay [10] など、さまざまな技術の上に構築されています。プロトコルは理論的には、セキュリティと分散化を犠牲にすることなく、漸進的に最適化されたスケーリングを毎秒数百万回まで実行できます。このペーパーでは、分散型取引所の状況における GORMOSについて説明します。このアプローチは、ゲーム、支払い、ソーシャルネットワークなどのさまざまなアプリケーションに適用できます。

Table of Contents

1 導入

Ethereumは、2015年の導入以来、プロジェクトや企業がネットワーク上で分散アプリケーションを立ち上げることに大きな関心を集めています。その結果、Ethereumネットワークは、例えば、トークンの大量販売やCryptokittiesのようなゲームのために、過去にいくつかの混雑を抱えていました。そのような場合、ユーザーは取引を送信するためにはるかに高い手数料を支払うか、取引を確認するまで何時間も待たなければなりません。Bitcoinのトランザクションのピークは、1日あたり約425,000件[1]、Ethereumのトランザクション数は140万件[2]で、これは1億5000万人を超えるアクティブユーザーを抱えるTwitterや、ユーザー数14億人のFacebookに比べてはるかに少ない数です。さらに、Ethereumでの平均ブロック待ち時間が15秒になると、ほとんどのユーザーにとって使用可能なプラットフォームを構築する上でより多くの課題が生じます。スケーラビリティとレイテンシの両方の制約は、分散アプリケーションが主流になるのを困難にします。

分散型取引所を例に挙げてみましょう。既存の分散型取引所は、流動性が低く、ユーザーエクスペリエンスが低く、スケーラビリティが低い（インフラストラクチャのため）などの大きな問題があります。これらの課題に取り組むにはさまざまな提案がありますが、各ソリューションはセキュリティ、分散化、またはサポート機能のいずれかで大きな妥協を見せています。たとえば、現在のKyberNetworkの取引所では、完全にオンチェーンで実行することで最適なセキュリティを実現し、わずかなクリックでより良いユーザーエクスペリエンスを実現し、取引を完了することができますが、リミットオーダー、トレーディングレバレッジ、または高頻度取引はまだサポートできません。0x[3]やEtherDelta[4]のような他の取引所は、ハイブリッドモデル、すなわちオフチェーンの（集中化された）注文帳を持ち、チェーン上の決済を行うことによって、集中取引所での取引経験をシミュレートしようとします。ユーザーは本質的に注文書を保管して提供するサーバー/ウェブサイトに頼らざるをえないため、明確なセキュリティのトレードオフが存在します。いずれの解決策もEthereumブロックチェーン上で1つのオンチェーン取引を行うため、スケーラビリティの制約と待ち時間の長い決済が問題になります。執筆時点では、レイテンシは平均で約15秒であり、Ethereumブロックチェーンはトランザクションタイプに応じて毎秒約10-15回のトランザクションを処理できます。さらに、ブロックチェーンがCryptokittiesやいくつかの重要な公開トークンの販売などのイベントで混雑する際に、トランザクションのガス支払いにもっと高い取引手数料を支払わなければならないかもしれません。そのようなソリューションは、明らかに、Binance[5]やHuobi[6]などの集中化された取引所と競争することができません。現在のレポートでは[19]、Binanceはピーク時に毎秒40,000要求を処理すると主張し、すべての分散型取引所のパフォ

[1] https://blockchain.info/charts/n-transactions?timespan=all
[2] https://etherscan.io/chart/tx
[3] https://0xproject.com/
[4] https://etherdelta.com/
[5] https://www.binance.com/
[6] https://www.huobi.pro/

ーマンスを大きく引き離しました。重要な設計上の変更がない場合、分散された変更はほんの一部でしか使えず、プロのトレーダーは言うまでもなく主流のユーザーを引き付けることはできません。

最近、Plasmaは、別のブロックチェーンにトランザクションをプッシュし、ユーザーを重要なブロックチェインバリデータから保護するためにクリプトエコノミクスを使用することで、有望なスケーラビリティで救助に到達しました [8]。しかし、既存のプラズマ設計は高レベルであり、分散型取引所には最適化されていません。 Plasmaのスケーリングは、Plasmaバリデーターの物理的容量によっても制限されます。したがって、ユーザーエクスペリエンスとセキュリティーの保証を損なうことなく、最適なスケーリングを提供しません [7]。Binanceの規模を実現するとなると、各プラズマノードは、テラバイトのデータとネットワーク帯域幅を格納しなければならない場合があります。それにかかわらず、プラズマに関しては相互運用性に関して議論されていない。

スケーラビリティは、分散取引所だけでなく、ソーシャルネットワーク、ゲームなどの他の多くのアプリケーションにとっても明らかに問題です。例えば、Etheremon [8]とPeepeth [9]は現在、より良いユーザビリティを提供するために複数のアクティビティを束ねてチェーンにコミットするというトレードオフを採用しています。まだコミットされていないアクティビティはプロジェクトサーバ内にローカルに保存されているため、システムのセキュリティが大幅に低下します。

本稿では、低レイテンシの安全でスケーラブルなブロックチェーンのための新しいプロトコル設計であるGormosを紹介します。上位レベルでは、GormosはPlasmaとShardingの間のスウィートスポットを見つけ出します。これにより、i) ブロック時間がはるかに短くても安全にサイドチェーン上でトランザクションを実行できるようにし、 ii）ネットワーク内のノードまたはバリデータから多くのリソースを必要とすることなく、線形スケーリングを可能にします。 Gormosは一般的な設計ではなく、異なるコンポーネントを持ち、互いに相互作用することはめったにない種類のアプリケーションに対してのみ有効です。例えば、分散型取引所の場合、各取引ペアを取引所の1つの構成要素と見なすことができ、異なる取引ペアの間の取引は、通常の取引ペアでの取引よりもはるかに少ないとみなすことができます。したがって、分散化された取引に対してシャーディングを効率的に適用して、ワークロードを異なるシャードに分散し、より高いスケーラビリティを達成することができます。さらにGormosは、その設計における相互運用性、すなわちBitcoin、Litecoin、ETC、または異なるブロックチェーンからの暗号通貨をPlasmaチェーンに移すことも考慮に入れています。同様に、GormosはEtheremonやPeepethのようなアプリケーションにも適用でき、ユー

[7] To be precise, the Plasma whitepaper mentions the concept of Plasma child chain. Technically it works and offers infinite scalability, but with the cost of user experience since they have to move their assets to many layers. Not to mention that the Plasma child chain is another layer of tradeoff for security guarantee.

[8] https://etheremon.com

[9] https://peepeth.com

ザーエクスペリエンスを犠牲にすることなく、より良いセキュリティ保証を提供します。

本稿では理解を容易にするために、以降の節では分散型取引所の文脈でGormosについて説明し、 4.7項にて他のアプリケーションに適用する方法について述べます。

1.1 望まれる性質

最先端のさまざまな分散取引所の事例や、KyberNetworkのリリース後数ヶ月にわたってユーザーから得られたフィードバックから、高性能な分散型取引所（DEX）の望ましい特性を次のようにまとめます。

- **スケーラビリティ**. DEXは、DEXのバリデーター（またはノード）からの重要なハードウェアおよびネットワーク帯域幅を必要とせずに、毎日何百万人ものユーザーが取引できるようにする必要があります。さらに、いくつかの一般的な取引ペアがDEXのすべての能力を占める可能性があり、その結果、他のペアのユーザーの取引経験に影響を及ぼす可能性があります。スケーラブルなDEXは、このような問題を防止し、ユーザーがすべてのユーザーのペアを円滑に交換できるようにする必要があります。
- **低レイテインシ**. DEXは待ち時間の短い取引、すなわち注文が提出されてから確認されるまでの時間をミリ秒単位ではないとしても数秒にする必要があります。私たちは、私たちのデザインで2秒の確認とインスタント決済を達成することを目指しています。
- **セキュリティと非中央集権性**. これは、DEXの基本的な特性であり、人々が中央集権的取引所よりもDEXを好む理由です。非中央集権は、DEXにおいて特に重要です。なぜなら、より良い透明性と検閲に抵抗する取引の需要が大きいからである。最近、中央集権的取引所がどのように透明であるかについての疑問が投げかけられています [10]。
- **相互運用性**. DEXは、Bitcoin、Ethereum、Litecoin、Ethereum Classic、EOSなどを含む、これらに限定されない異なる暗号通貨を取引できるようにすべきです。これは、他のプロパティを妥協することなく行う必要があります。

これらは、中央集権的取引所と競争できるようにするためにDEXが持たなければならない主な望ましい特性です。高い流動性、優れたユーザーエクスペリエンスなどを含む展開と採用に関しても重要な非プロトコルの理想的な特性があります。しかし、これらの特性は、我々が構築している高性能な分散型交換を構築する主な問題とは正反対です。

2 関連プロジェクト

なぜ**Plasma**チェーンのツリー構造を用いないのでしょうか？慎重な読者は、GormosとPlasmaチェーンのツリー構造との違い、すなわち既存のPlasmaチェ

[10] https://medium.com/@sylvainartplayribes/chasing-fake-volume-a-crypto-plague-ea1a3c1e0b5e,
https://cointelegraph.com/news/okex-resolves-futures-price-slip-impact-as-trader-threatens-suicide

ーンの上に別の子Plasmaチェーンを有することとの違いが何であるか疑問に思うかもしれません。それは、PlasmaがLayer-2のレイヤー2であるから、すなわち、ルートチェーンの上に別個の層を形成するからです。一般に、レイヤー2のソリューションにはある種のセキュリティのトレードオフがあります。たとえば、Plasmaユーザーは、何か悪いことが起こっていないかどうかを確認し、手遅れになる前にexitをする必要があります。 Plasmaチェーンの上にレイヤー2のソリューションを増やすことで、よりセキュリティのトレードオフが導入され、ユーザーのフォローが難しくなります。また、開発者は、プラットフォームを構築するか、プラットフォームのセキュリティ保証について理由を判断することも難しくなります。一方、シャーディングは、セキュリティの妥協をせずにコンセンサス層でスケーラビリティを提供します。シャーディングの主なトレードオフは、クロスシャードトランザクションによるユーザビリティです。しかし、GORMOSでは、ネットワークを異なる取引ペアごとに異なるシャードに分割し、クロスシャード取引の量を最小限に抑える明確なロジックがあります。

Zilliqaのような既存のシャーディングチェーンはどうでしょうか？ Zilliqaがシャーディングを使用してスケーラブルなブロックチェーンを構築しているため、Zilliqaの上にプラットフォームを構築しない理由が不思議に思うかもしれません。主な違いは、Zilliqaはステートシャーディングをサポートしておらず、分散型取引のプラットフォームやその他のアプリケーションには最適化されていません。何十億ものトランザクションを処理する際のストレージと状態の更新の要件は、ストレージ、帯域幅、および計算リソースの追加のコストをZilliqaで必要とします。一方、GORMOSは、状態処理、取引活動、および保管を明確に分離しています。また、Plasmaを活用して、シャード内のバリデータセットの数を減らし、トレーディングプラットフォームのレイテンシを短くします。

Atomic Cross-Chain Swap (ACCS). [18]は、最も初期に考案されたシンプルなクロスチェーン取引の設計の1つで、以前にコミットされたハッシュの元データを明らかにすることによって払い戻しが可能な、nLockTimeによるコントラクトも用いる手法です（正式な記述はTesseract [1]のセクション2にあります）。ACCSの主な問題は3つあります。第1に、タイムロックの導入によってリアルタイムの交換が難しくなってしまったことによる高いレイテンシです。これは基本的に*fair exchange problem* [14]の結果です。第2に、複数のタイムロックを並行運用するマルチパーティACCSの文脈では、複雑さのためスケーラビリティが劣ります。第3に、注文の発見の遅延と流動性の課題です。それと比較して、GORMOSは他の暗号化された通貨をトークン化し、相互運用性に対する複雑なアプローチを大幅に削減します。 GORMOSは、それ自体でクロスチェーン転送で長い待ち時間を解決するわけではありません。しかし、いったんペッグされたトークンは、他のトークンと同様に簡単に取引され、GORMOSで確定したものと見なされます。

Payment ChannelやState Channel. [12, 15] は、Blockchainからロック状態の預金を介してトランザクションをオフロードし、最終的には更新された状態でチェーンを解決する仕組みです。特に、最近公表された*generalized state channel*は、ブロックチェーンにブロードキャストすることなく、新しい機能をインストールするための反証的にインスタンス化された契約を利用し

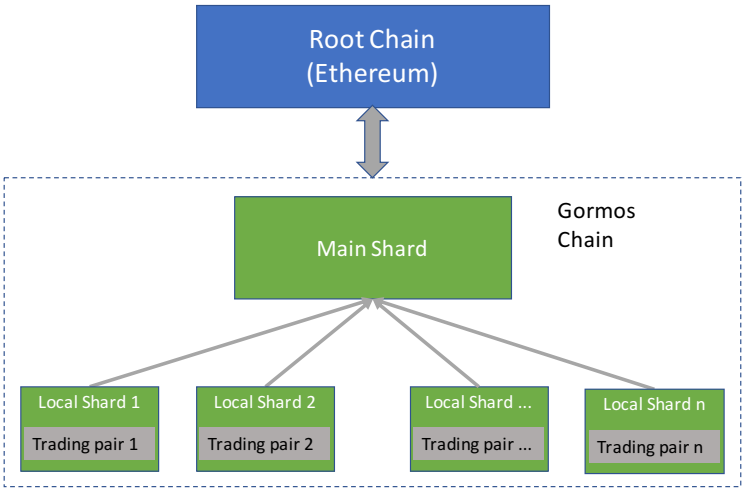

Fig. 1: GORMOS System Architecture

ます [5, 6]。ブロックチェーンのスループットを効率的に調整する一方で、DEXのようなアプリケーションでは、州内の参加者をチャンネルに誘導するだけで、グローバルオーダーブックを構築することはほとんど不可能です。

3 デザイン

私たちの設計のほとんどは、Plasma [8]の既存の作業とシャーディング [11]の以前の作業、ブロックチェーンの相互運用性 [10]を活用しています。私たちの目標は、前述のすべての望ましい特性を、セキュリティと分散化を最小限に抑えて達成することです。

3.1 Design Overview

GORMOSの高レベルのデザインはFig.1に示されています。 GORMOSは、Plasmaとシャーディングの両方の素晴らしい技術を採用しています。 Shardingアーキテクチャが組み込まれたこのPlasmaベースのアプローチでは、既存の取引プラットフォームにスケーリングの負荷を加えずに[11]、無限の取引ペアをリストすることができます。

私たちは新しいチェーンのバリデーターの中でビザンチン合意を採用して、低レイテンシとより迅速なファイナリティを達成しています。さらに重要なことに、このPlasmaチェーンの上にシャーディングアーキテクチャを設計して、無限のスケーリングを可能にします。私たちの調査では、異なる取引ペアのアクティビティを明確に分離し、異なるシャードにローカライズできます。例えば、ETH / KNCとBTC / OMGとの間の取引は分離することがで

[11] More accurately, asymptotically optimal (i.e. constant factor cost) in main shard with increasing token pairs/shard, linear scaling within shard with increasing computations and communication costs incurred by more users.

き、ETH / KNC間で取引するユーザは、BTC / OMGを取引するユーザに影響を与えません。この設計では、既存のリスティングペアのパフォーマンスと取引のユーザー体験に影響を与えることなく、プラットフォームがより多くのトークンを扱う場合に、より多くのシャードを追加することができます。このような設計は、多数の取引ペアに拡張することができ、毎秒数百万回の取引をサポートしますが、バリデーターのリソースの必要量は最小限に抑えられます。 GORMOSのバリデーターは、状態全体を保管する必要もなく、プラットフォーム内で起こっているすべての取引を処理する必要もありません。代わりに、彼らは自分のシャードの状態を保存し、シャードに記載されている対応する取引ペアのすべての取引を処理すればよいだけです。より多くのトレーディングペアを追加しても、既存のシャードのバリデータには影響しません。

GORMOS設計の理論的根拠. GORMOSは、シャーディングとPlasmaの両方のアプローチを組み合わせ、お互いの限界を補い合うという利点を利用しています。具体的には、GORMOSはシャーディングの待ち時間を短縮するためにPlasmaを活用し、シャーディングを採用してPlasmaをさらにスケールアップします。 GORMOSは両者の統合を、一般性（もしくは普遍的な適用性）のトレードオフにより実現します。つまり、GORMOSは応用分野に特化したものであり、特定の種類の分散型アプリケーションでのみ機能し、サブコンポーネントが明確に分離されています。

ここまで説明したコアな調査結果を統合して、GORMOSのヒューリスティックな理論的根拠を確立することができました。シャーディング、PlasmaおよびGORMOSのより詳細な比較を表1に示します。

	Scalability	**Latency**	**Applicability**
Plasma	100x	Low	Generic
Sharding	100x	High[12]	Somewhat Generic
GORMOS	**1000x- 10000x**	**Low**	**Application Specific**

Table 1: Plasma vs. Sharding vs. GORMOS

3.2 Plasma: 取引活動を安全に新しいサイドチェーンに移す

上位レベルでは、Plasmaは悪意のあるバリデータがユーザーを不正行為するのを防ぐために、クリプトエコノミクス的なインセンティブを持つサイドチェーンを持つことでセキュリティパフォーマンスのトレードオフを提供します。 Plasmaチェーンのユーザは、ルートチェーン（すなわち、Ethereumチェーン）からPlasmaチェーンに資産を移動することができ、ルートチェーン内でトランザクションを行うことなく、このPlasmaチェーン内で安全に取引し、動作させることができます。何かが起こった場合、たとえばPlasmaチェーン内のバリデーターは、ユーザーに適切なバランスを与えたり、信用したりしないため、ルートチェーンに ”fraud proof” を提出して、資産を取り戻すことができます。この設計のおかげで、より多くのトランザクションが ”ルートチ

[12] for standalone blockchain with sharding

ェーン外”で処理されますが、ユーザーはまだ良好なセキュリティ保証を得ることができます。

Plasmaの1つの利点は、セキュリティ保証に影響を与えずにバリデータセットを小さくすることができることです。つまり、Plasmaチェーンは1つのバリデータで動作することさえできます。したがって、Plasmaチェーンのパフォーマンスは、集中化されたサーバーと同じくらい良いものになります。しかし、より良い耐障害性と耐検閲性を提供するために分散化の必要性があるため、バリデータセットにはより多くのバリデータが必要です。このトレードオフと高速決済を実現するため、私たちは約20のバリデーターセットを使用することにします。第4章でバリデーターセットの選択について詳しく説明します。

3.3 最適な水平スケーラビリティを提供するシャーディング

前述のように、私たちのシャーディングソリューションの目的は、異なる取引先を別々のシャードに分けることです。シャードごとに数組のトークンペアがあります。シャーディングにおける高レベルのアイデアは、バリデーターを異なるサブセット（またはシャード）に配布すること、バリデーターの各サブセットを維持すること、またはPlasmaチェーン内のアクティビティーの別個の部分を担当することです。以前の研究では、いくつかの一般的なシャーディングプロトコル [9, 11, 17]が提案されていましたが、分散取引所に最適な具体的な設計は提案されていません。

GORMOSでは、2層アーキテクチャを採用しています。メインシャードがあり、いくつかのローカルシャードがあります。ローカルシャードは、いくつかの特定のトークンペアの取引を処理しますが、メインシャードはローカルシャードからの結果を収集し、集約します。ベース通貨（ETH、BTC）はメインのシャードとローカルのシャードに存在することができますが、他のトークンは1つの指定されたシャードにしか存在しません。このアプローチは、異なるペア間の取引が関連していないため意味があります。したがって、1つのチェーン内のすべての取引を保存して処理する必要はありません。これは、中央集権的取引所でのシャーディングから発想を得ています。つまり、ほとんどの中央集権的取引所では、異なるサーバーで異なる取引ペアを処理しています。この設計はスケーラブルなアーキテクチャを提供するだけでなく、リスクを明確に分離します。たとえば、何らかの理由でトークンが機能しない場合（つまり、最近のインシデントのようなトークン契約のバグ）、このシャードは取引を一時停止することができますが、他のシャードは引き続き通常どおり機能します。

ユーザーがルートチェーンからトークンを入金すると、対応するローカルシャードは対応するトークンバランスを更新します。ユーザーがベース通貨を入金すると、メインシャードが処理します。ユーザーがベース通貨（ETH、Bitcoin）を別のローカルシャードに移動したい場合は、コインをメインシャードからローカルシャードに移動するトランザクションを送信する必要があります。この設計により、クロスシャード通信の複雑さが軽減されます。具体的には、クロスシャード通信は、ユーザが基本通貨を移動したい場合にのみ必要です。例えば、 ETHをローカルシャード i から別のローカルシャード j に移動する場合です。これには、2つの異なる取引、すなわち、ローカルシャ

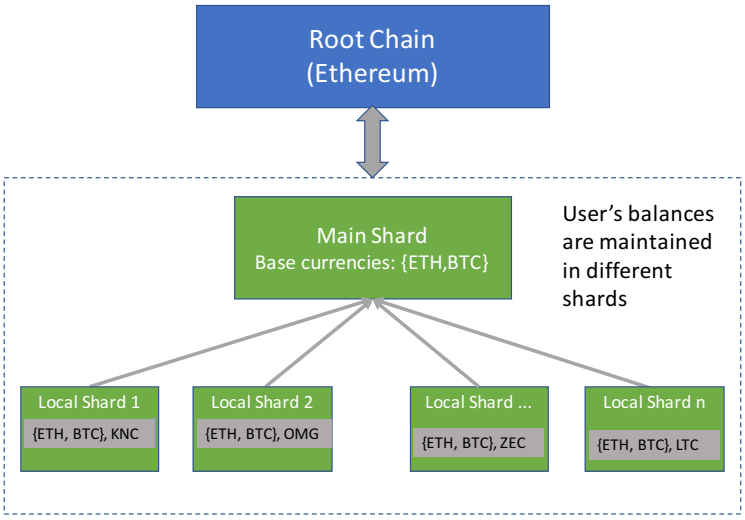

Fig. 2: GORMOS's design architecture

ード i からメインシャードへ、およびメインシャードからローカルシャード j へのETHの移動が必要です。

シャード内のバリデーターは、フォールトトレラントなコンセンサスプロトコルを実行し、定期的にシャードのトークンペアの最新の取引に同意します。具体的には、4 5秒ごとにシャードバリデーターは、前のブロック以降のユーザーの取引注文を表す新しい取引を含む新しいデータブロックを発行します。取引注文は、新規の売買注文や既存の注文のキャンセルがあります。シャードに記録されたすべての取引要求に基づいて、バリデーターは、シャード内のこのトークンペアの完全な注文書を保持することができます。バリデーターは、どの取引注文がマッチしているかを知ることができ、ユーザの残高を更新するために決済を行うこともできます。下の図は、ブロックデータの例と、ブロックが作成された後のユーザーの残高の変化を示しています。第 4章では、取引注文フォーマットの詳細、シャード内のブロック構造について議論します。

3.4 Shard inspectors: Guardians of the chain

パブリックブロックチェーンを中央集権的システムよりも魅力的にする優れた特性は、すべてが公に監査可能で検証可能であることです。公開鍵暗号のおかげで、不正または悪意のあるすべてのアクティビティについて、悪質なアクターに説明責任を持たせることができます。すなわち、悪質なアクターは、暗号学的証明を条件に何も間違っていないことを否定できません。そのため、誰でも、いつでも不正な行為が発生しているかどうかを検出するために公開ブロックチェーンのデータを観察し、悪質なアクターを特定することができます。 Plasmaでは、Plasmaチェーンを観察し、バリデーターによる不正行為があるかどうかを検出するためには、ユーザーがオンラインでなければならないという懸念があります。そうであれば、ユーザーは、fraud proofを提出してexitすることができます。これにより、自分の資産を取り戻し、バリ

データーにペナルティを科すことができます。しかし、私たちの見解では、データが一般に公開されているので、この行動は誰でも行うことができます。第三者オブザーバーは、他のオブザーバーの代わりにexitすることができます。私たちは、この機能を拡張して、シャードインスペクタと呼ばれるすべてのシャードに新しいロール/プレイヤーを導入します。彼らの責任は以下の通りです。

シャード・インスペクタは、シャードのローカルコンセンサスに参加しておらず、単にシャードを観察して奇妙なことが起こっていないかを検証し、メインチェーン上にfraud proofを報告したり、シャードバリデータを罰したりします。シャード・インスペクタはデータの可用性も保証します。彼らは、シャード内で生成されたすべてのデータブロックを保存し、ユーザーにデータを提供することができます。これにより、シャード内のデータ可用性が向上します。シャードインスペクタは、EthereumとBitcoinにおけるフルノードと考えることができます。ただし、これらのフルノードは上位機関に報告することで、マイナー（つまり、Plasmaチェーンのバリデータ）にペナルティを科すことができます。以下のセクションでは、GORMOSで実際にこの役割をどのように実施するか、そしてインスペクタに参加するようインセンティブを与える方法について詳しく説明します。

3.5 Interoperability: bridging the many blockchains

既存の分散型取引所の大きな問題点の1つは、異なるコインと暗号通貨をサポートする能力です。たとえば、BitcoinをEthereumに移行するためのシームレスな解決策が存在しないため、現在のユーザーは既存の分散型取引所でBitcoinと他のERC20トークンとの間で取引することはできません。GORMOSがEthereumのエコシステム外で異なる暗号化通信をサポートするために使用すると考えるいくつかのソリューションとアプローチについて議論します。

- PeaceRelay: ユーザーはEVMベースの通貨をEthereumのメインチェーンと行き来することができます。 PeaceRelayは、2つのチェーン間の双方向リレーを構築し、ネイティブコインを1つのチェーンに固定して、Ethereum上のコインをトラストレスな方法で表現する新しいトークンを作成することを可能にします。同様に、ユーザはEthereum上で前記トークンをバーンして、他のチェーン上のコインを交換することもでます。もちろん、PeaceRelayを使ってRootstock経由でBitcoinをEthereumに移動したり戻したりすることもできます。
- Trustless Custodian approach: この最近提案されたソリューション [13]は、ユーザーが簡単にBitcoinトークンを生成できるようにします。それは、Bitcoinをデポジットするパブリックなウォレットを持っている保管人（必ずしも信頼できるとは限らない）です。しかし、保管人はETHやEthereum上の他のトークンを担保にして、ユーザーは、Bitcoinを戻すためにバリデータを報告してペナルティを科すことができます。

[13] https://blog.kyber.network/bringing-bitcoin-to-ethereum-7bf29db88b9a

- MakerDao’s approach: MakerDaoは、価値が1米ドルに固定された分散型ステーブルコイン（DAIなど）を構築するソリューションを設計しています。ユーザはETHを担保にしてDAIをミンティングすることができ、担保付資産は常にDAIの全流通以上です。 MakerDaoは、担保付資産の総額を評価するためにETHの価格供給に依存しています。同じメカニズムを使用して、私たちはETHをBitcoinトークンを作成するための担保として使用し、ETH：Bitcoinの価格フィードに依存することができます。

3.6 GORMOS: high performance decentralized exchange

これらのコア技術コンポーネントに基づいて、GORMOSを高性能分散型取引所として構築します。GORMOSが以下のようにすべての理想的な特性をどのように保管しているかを示します。

- スケーラブル. GORMOSは、そのデザインに2つのスケーリングソリューション、すなわちPlasmaとシャーディングを実装しています。 Plasmaでは、高いスループットと安価（あるいは0）のトランザクション手数料を使用してルートチェーンからトランザクションを実行できますが、シャーディングでは、異なる取引ペアのアクティビティを分離することによってGORMOSをスケールアップできます。
- 低レイテンシ. GORMOSの新しいブロックが作成されるとすぐに注文を確認することができます。これは、シャード設定に応じて4〜5秒以内に行うことができます。ただし、取引が最終化される、すなわちルート・チェーンにコミットされるまでには時間がかかることがあります。
- セキュアで分散型. GORMOSでは、バリデーターが悪意のある行為を行った場合に、ユーザーはサイドチェーンを終了し、資産をルートチェーンに戻すことができます。さらに、不正なバリデータを報告し、罰則を科すfraud proofを提出することもできます。これにより、バリデータが不正行為をするのを防ぐことができます。さらに、すべてのシャード内のコンセンサスは複数のバリデーターによって実行され、バリデーターはシャード間で頻繁にローテーションされるため、特定のシャード上でシャードを妥協したり、検閲を実施することは困難です。
- 相互運用性. BitcoinやEthereumの他の暗号化ツールをEthereumに移行するためのさまざまなアプローチにより、ユーザーはGORMOS上で異なる通貨ペアを取引することができます。

シャーディング・アーキテクチャーの優れた特性の1つは、トレーディング資産の明確な分離です。その結果、異なるトレーディング資産のペアに対して専用のシャードを設けることができます。GORMOSが特別なシャードを設けることで、人々はNFT(Non-Fungible Tokens)を交換することができます。また、ユーザの参加に関する様々な要件を備えたセキュリティトークンの取引を容易にするために、特別な機関によって運営される別のシャードを設けることもできます。例えば、特別な機関のバリデータが運用するシャード上でセキュリティトークンをサポートし、そのシャード上で取引を行うために、ユーザーの情報登録やKYCチェックを必須にすることができるでしょう。

GORMOSのシャーディング設計は、取引活動のダイナミックで明確な分離を可能にし、異なるプラットフォームを同じプラットフォームで共存させることを支援します。

4 技術的な詳細

このセクションでは、バリデータの選択方法、シャードの形成方法、シャッフル方法、GORMOSでの取引の詳細について説明します。

4.1 バリデータの登録

GORMOSのバリデーターになるためには、KNCをルートチェーン（Ethereum）にデポジットしなければなりません。デポジットがPlasmaチェーン上で認識された後、バリデーターとして登録され、バリデーションプロセスを開始することができます。バリデータのデポジットの量を決定する方法はいくつかあります。 1つの方法は、量を固定し、いくらでもバリデータを受け入れる方法です。その結果、GORMOSには何万人ものバリデーターがいるかもしれません。もう1つのアプローチは、バリデーターになるためのデポジット額の決定は公開市場に任せて、固定数（例えば1,000）のアクティブバリデータのみを許可する方法です。この場合、新しいアクティブバリデーターになるためには、現在のアクティブバリデーターリストの最低デポジットより多くのデポジットを入れなければなりません。最低金額のバリデーターは、例えば24時間後に無効になります。ダイナミックデポジットは、バリデータスロットがどれだけ価値があるか（バリデータの主なコストは機会費用）とプラットフォームにどれくらいの自信を持っているかを市場が自由に決めることを可能にします。

4.2 シャード形成

アクティブなバリデータのリストから、GORMOSはバリデータをさまざまなサブセットに配布し、それぞれがシャードを担当します。一般に、Nをバリデータの数とし、nをシャード数とし、各シャードにcバリデータ（したがって$N = n * c$）を持たせます。より多くのシャード（より多くのn）がある場合、よりアクティブなバリデータを受け入れ（Nを増やす)、新しく作成されたシャードの異なる取引ペアをサポートすることができます。理想的には、VRF [7]やRandHound [16]なで実現できる偏りのないグローバル乱数ジェネレータに依存する一様分布を作りたいと考えています。しかし、不正なバリデーターが行うことができる最も大きな損害は、シャードで検閲を行うこと、またはファイナライズされていない取引をなかったことにすることであるため、潜在的なバイアスが小さい乱数のソースを用いることができます。GORMOSの最初のイテレーションでは、ルートチェーンのブロックハッシュをランダムシードとして使用することを提案します。

より良い決済レイテンシを達成するためには、シャード内のコンセンサスは、小さな数c_c以内のバリデーターによってなされなければなりません。例えば、シャード内のローカルブロック時間を2秒にするには、以前の実験 [11]に

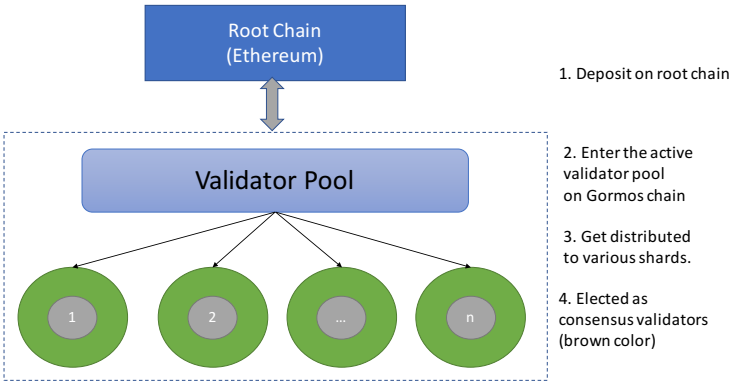

Fig. 3: Steps to be validators in GORMOS

基づいて、c_cは約20でなければなりません。しかし、シャード内のバリデータの数cは、数百に上げることができます。すなわち、シャード内のバリデータには2つの異なるロールがあります。

- シャードコンセンサスバリデータ: シャードには、新しいトランザクションの検証と新しいブロックの生成に積極的に関与するc_c個のバリデータが存在します。
- シャード・インスペクタ: 前述したように、残りのバリデータはシャード・インスペクタであり、シャード内のフルノードのように動作します。これらは、シャードの完全な状態を保存し、データの可用性を保証します。

バリデータに対する標的攻撃を軽減するために、GORMOSは定期的にシャード・インスペクタをシャード内の新しいコンセンサス・バリデータに昇格させ、既存のコンセンサス・バリデータをシャード・インスペクタとして機能する新しいシャードに再配置します。昇格と再配置のプロセスも、前述のグローバル乱数ジェネレータに依存する統一された方法で行われます。

4.3 取引の実行とブロックフォーマット

取引フォーマット. GORMOSのユーザは、トランザクションをシャードシャード内のトランザクションは、以下の情報で構成されます。

$$(\texttt{addr}, \texttt{shard\_id}, \texttt{src\_token\_id}, \texttt{des\_token\_id}, \texttt{order\_type}, \texttt{src\_amount}, \texttt{fee}, \texttt{metadata}, \texttt{nonce}),$$

ここで、

- `addr`: ユーザーのアドレス
- `shard_id`: ユーザーが取引をしているシャードのID
- `src_token_id`: ユーザが取引元にしたいトークンのID
- `des_token_id`: ユーザーが取引先にしたいトークンのID

- order_type: リミットオーダーやマーケットオーダー、既存オーダーのキャンセルリクエストなど
- src_amount: ユーザーが取引したいソーストークンの量
- fee: ユーザーが支払おうとした手数料（%）（トランザクション手数料と同様）
- metadata: 注文に関する詳細なデータ。たとえば、order_typeがリミットオーダーの場合、ユーザは価格を含める必要があり、注文タイプがキャンセルリクエストの場合、metadataはorder_idを持ちます。これは、以前の注文のハッシュです。
- nonce: Ethereumのトランザクションnonceに似ています。これは、これまでにユーザーが作成したオーダーの数です。

ブロックフォーマット. GORMOSには、ローカルデータブロックとメインブロックという2つのブロックがあります。シャード内のローカルデータブロックは、生の新しいトランザクションデータと、次のフィールドを持つデータブロックヘッダーで構成されます。

$$(\texttt{shard\_id}, \texttt{block\_number}, \texttt{prev\_block\_hash}, \texttt{new\_order\_merkle\_root}, \texttt{balance\_merkle\_root}, \texttt{avail\_balance\_merkle\_root}, \texttt{open\_order\_MMR\_root}, \texttt{closed\_order\_MMR\_root}, \texttt{cancelled\_order\_MMR\_root}, \texttt{timestamp}, \texttt{signatures}),$$

ここで、 shard_id: ユーザーが新しい注文で取引をしているシャードのID new_order_merkle_root: このブロックのすべての新しい注文をコミットするMerkleルート balance_merkle_root: このブロック内のすべての新しい注文を受け入れた後、シャード内の全員の残高をコミットするMerkleルート avail_balance_merkle_root: このブロック内のすべての新しい注文を受け入れた後、シャード内の全員の利用可能な残高をコミットするMerkleルート open_order_MMR_root: このシャードのすべての未処理注文をコミットするMerkle Mountain Rangesルート closed_order_MMR_root: このシャード内のすべてのクローズ注文をコミットするMerkle Mountain Rangesルート cancelled_order_MMR_root: このシャードのすべてのキャンセル注文をコミットするMerkle Mountain Rangesルート

一方、メインブロックはローカルシャードのすべてのローカルブロックヘッダを格納し、その結果を単一のメインブロックヘッダに合成します。例えば、メインブロックのブロックヘッダは、以下のようにすることができます。

$$(\texttt{block\_number}, \texttt{prev\_block\_hash}, \texttt{new\_local\_headers\_merkle\_root}, \texttt{new\_order\_merkle\_root}, \texttt{balance\_merkle\_root}, \texttt{avail\_balance\_merkle\_root}, \texttt{open\_order\_MMR\_root}, \texttt{closed\_order\_MMR\_root}, \texttt{cancelled\_order\_MMR\_root}, \texttt{timestamp}, \texttt{signatures}),$$

すべての新しいMerkleルーツはローカルブロックのヘッダーからのMerkleルーツに基づいて計算されます。

Merkle Mountain Rangeの活用. Merkle Mountain Rangeを活用して、閉鎖されたすべての注文、発注、キャンセルされた注文などのブロックごとにMerkleツリーを効率的に更新できるようにします。 GORMOSは、最近提案されたデータ構造（Merkle mountain range（MMR））を利用して、軽いクライアントが最新のブロックのみでブロックチェーン上の任意のイベントを効率的に検証できるようにします。 MMRを使用すると、すべての前のブロックを1つのハッシュで最新のブロックヘッダーに効率的にコミットできます。もちろん、元のMerkleツリー構造をそのまま使用して同じ目標を達成することはできますが、Merkleツリーを新しいブロックヘッダーで更新することは効率的ではありません。 Merkleツリー全体を再構成するか、または$\log n$ハッシュよりもはるかに大きい証明サイズをもたらす「アンバランス」ツリーに苦しむ必要があります。 MMRは、より効率的な更新プロセスを可能にする元のMerkleツリーの変形であるため、ブロックを処理する際の完全なノードのオーバーヘッドは無視できる程度になります。

4.4 シャード内のコンセンサスプロトコル

Plasmaでは、Minimal Viable Plasma [3]やPlasma Cash [4]で示唆されているように、Proof of Authorityのような集中型コンセンサススキームに頼ってセキュリティを犠牲にすることはありません。しかし、フォールトトレランスとセンサー耐性を向上させるために分散化が必要なため、合理的に多数のバリデーターを実行するために各シャードで効率的なBFTプロトコルを選択します。例えば、非同期ネットワークで高スループット（64ノードで$\approx 1.2 * 10^4\ tx/s$）を達成するHoneyBadger BFT [13]が良い候補です。

Practical asynchronous BFT in sporadic layer-2 network. シャード内のバリデーターは、ルートチェーン内の基礎となるネットワークから分離された別のP2Pネットワークで動作しなければならなりません。私たちの目標は、ネットワークが許す限り速く、平均で2秒のブロック時間で、シャードが新しいブロックを提案できるようにすることです。異なるシャード内のブロックは、非同期方式で生成されます。すなわち、異なるシャードは、異なる速度でブロックをファイナライズし、メインシャードは、メインブロックを生成するときに、利用可能なすべてのローカルブロックをみます。そこで、HoneyBadger BFT [13]で提案された最適化を用いた非同期アトミックブロードキャストプロトコルと非同期共通サブセットのより効率的なインスタンス化を目的とした公開鍵暗号化を利用する予定です。

4.5 Fraud proofの提出とexit

ここでは、何か悪意のある事態が発生した際に、ユーザーがどのようにfraud proofをルートチェーンに提出し、GORMOSチェインを正常にexitするのかにについて説明します。シャード・インスペクタによってデータの可用性が維持されていると仮定すると、2つのケースが考えられます。1つは、シャード内の共謀バリデーター（および機能しないインスペクター）が不正なトランザクション（有効なシグネチャまたは入力のミスマッチ、出力量）を取り込ん

だ場合、もう1つの可能性は、ネットワーク障害や攻撃者の意図的なメッセージのドロップにより、ネットワークの非同期が発生し、シャードが正常に進まなくなってしまった場合です。

ユーザーはフルノードにリクエストを送信することでfraud proofを取得します。フルノードは、シャード・インスペクタまたは自分自身（ユーザーがフルノードを実行している場合）になります。ユーザーの送信する要求では、ユーザーが正しいと合意する特定のブロックを指定します。つまり、そのブロック以降の他のブロックは無効と見なされます。フルノードからの応答は、その特定のブロック高での状態ツリーのユーザーバランスのMerkle Proofです。ユーザーはインフォーマルに次のような主張を行います。「私の最新の残高としてこのブロックの残高に同意します。次のブロックに不正な署名が含まれているか、動作が停止しているためにexitしたいと思います。」

最初のシナリオでは、ユーザーは2つのMerkleパスをルートチェーン（Ethereum）に提出します。1つは状態ツリー内の最新の有効な残高のMerkleパスです。もう1つは悪意のあるバリデータによって偽造された無効な状態のパスです。Plasmaのメインシャードの定期的なコミットにより、ルートチェーンは提出されたMerkle Proofを検証し、証明が有効になるとチャレンジ期間を開きます。この期間中、悪意のあるバリデーターが、ユーザーからの正式な署名を使用して、前者から後者への状態遷移を示す有効なトランザクションを提供できない場合、バリデーターは処罰され、その預金は失効し、ユーザーは最後の有効な残高でexitすることができます。

2番目のケースでは、そのシャード内の各ユーザーが1人ずつ退出することもできます。これは、チェーン上のすべての決済であるため、非常に高額です。オンチェーンのメッセージ数を減らすために、Plasmaで提案されているmass withdraw手法を用いることができます [8]。GORMOSが提供する優れた特性は、1つのシャードからexitすることが残りのシャードに影響しないことです。ダメージが独立しているため、GORMOSの設計はより堅牢です。

4.6 バリデータのインセンティブ

バリデータになる主なインセンティブは、シャード内の処理済みの取引すべてから取引手数料を得ることです。これはコンセンサスラウンドに参加するバリデータにとってより明白です。続いて、そのシャードのフルノードデータを保存し、シャードの中に何らかの不具合が起きたかどうかを監視するシャード・インスペクタのインセンティブについて議論します。

まず、シャード・インスペクタは、同じシャード内の次のラウンドのバリデータの適格候補です。なぜなら、インスペクタが保持しているすべてのデータがなければ、新しい今後のトランザクションを検証することは不可能だからです。加えて、シャード・インスペクタだけが、シャードのコンセンサスに参加するために選ばれます。

第二に、シャード・インスペクタが不正行為を発見したら、ユーザーのためにfraud proofを提出し、失効した預金から報酬を得ることができます。ほとんどの場合、シャード・インスペクタはつねにオンラインであり、ユーザーよりも厳密にシャードを監視しているため、不正を発見できる機会は多いでしょう。さらに、インスペクター/フルノードとして、（コンセンサスバリデータと比較して）取引手数料のほんの一部を受け取ることもできます。

4.7 他のユースケースのためのGORMOS

分散型取引所以外のアプリケーションでGORMOSを使用して、オンチェーンでより多くのトランザクションをスケーラブルかつ低レイテインシで実行できるようにする方法について説明します。第 1章で説明したように、GORMOSのスケーラビリティは、アクティビティを異なるシャードに分けることによってもたらされます。その結果、GORMOSは、別々のコンポーネントを持つアプリケーションに適しており、これらのコンポーネント間の相互作用はあまりありません。たとえば、Etheremonはポケモンに似たゲームで、ユーザーは”モンスター”を育成して成長させ、他の人と競うことができます。モンスターは、ゲームマップの異なる場所に属する異なる「ジム」に移動することができます。 GORMOSを使用すると、Etheremonはマップを異なるシャードに対応する異なるローカルマップに分割することができ、各シャードはゲーム内の地理的領域を表します。シャードは、それぞれのローカルマップ内のすべてのモンスターと活動を管理します。ユーザーは、メインのシャードにトランザクションを送信して、自分のモンスターに存在する新しいローカルマップを示すことによって、モンスターを異なるローカルマップに移動できます。 Etheremonのほとんどのアクティビティがジム内で起こるか、モンスターが同じ地域でのみ出会えるとき、GORMOSは間違いなくユーザーエクスペリエンスに影響を与えずにEtheremonをスケールアップする最良の設計です。

同様に、すべての旅行、支払い、格付けがチェーン上に記録されている分散Uberのような他のユースケースにはGORMOSを使用できます。 GORMOSベースのUberは、異なるシャードを持つことができ、それぞれが都市を担当します。シャード内のバリデーターは、都市内のすべてのトランザクションを検証し、ローカルブロックに組み込みます。一般的に、地理的に分散されているか、またはアーキテクチャ上異なるコンポーネントに分かれているアプリケーションは、GORMOSの恩恵を受けることができます。

5 ガバナンスとトークンユーティリティ

ステークベースの投票を通じてオンチェーンガバナンスの対象となるいくつかのパラメータがあります。これらの投票シナリオのそれぞれは、ルートチェーンまたはGORMOSのいずれかでスマートコントラクトを実装する一連の機能です。

- トークンペアの追加/削除: 新しいペアを追加するには、新しいシャードを作成し、バリデータプールを増やすか、シャードごとのバリデータの数を一時的に減らします。同様に、特定のシャードが非アクティブであるか、十分なアクティビティがない場合、Plasmaチェーンのステークホルダーで投票を行い、そのシャードをドロップするか他のシャードに統合します。
- バリデータの登録の閾値: この値は、コミュニティの信頼度に基づき柔軟に設定されるべきであり、保証金やバリデータの扱うトランザクション量などを比較する必要があります。（デフォルトでは、バリデータになるためには最小限の障壁が指定されているべきであり、ガバナンス上の決定により閾値を上げていくのが望ましいでしょう）

- バリデータプール: 各シャードに選ばれるバリデータの数を変更します。

トークンユーティリティ. KNCまたはGormosの根底にあるトークンは、以下のものをはじめ、いくつかのユーティリティーに使用することができます。

- Gormosバリデータになるためのステーキング. これは、ルートチェーン（すなわちEthereum）上におけるGormosのメインのコントラクトにKNCをデポジットすることでバリデータになれるという、基本的なステーキング機能です。
- KNCを使って取引手数料を払い割引を受ける. このユーティリティは、Binance、Huobiなどのいくつかの一般的な取引所で使用されています。これはGormosで簡単に実装できます。

6 Future Work

本稿では、Gormosのアーキテクチャー設計とシステム特性に焦点を当てました。Gormosは、複数の最先端のスケーラビリティソリューションを慎重に相互運用し、暗号通貨システムで高性能な分散型取引所を構築します。将来の仕事のために残されている未解決の問題は、フロントランニングに対処する方法です。マイナー/バリデータのフロントランニングは、非制限のブロックチェーンシステムでよく見られる問題であり、オークションシステムや交換市場では特に面倒です。特にGormos設計では、各シャードのバリデーターは、自身の取引を最初に挿入して元の受注者を「フロントランニング」し、流動性の変化により後で高額で再販することで利益を得ることができます。*Submarine Send* [2]のようなプロポーザルは、十分な匿名性の設定が与えられているため、コミット（submarine sendとも呼ばれる）が通常の取引とマイナーの視点からは区別できな「Commit-Reveal」パラダイムを利用しています。オークションの入札および注文の記入には実行可能なソリューションです。しかし、Submarineがバリデータを使って多くの偽の注文をしないようにする方法は明らかではありません。さらに、このソリューションは、遅延（コミットフェーズと公開フェーズの間の確認ブロック）を発生させるため、ユーザビリティに影響し、ユーザはプロトコルに参加するために追加の手順を踏む必要があります。中央集権的取引所では、フロントランニングが簡単であり、取引所が他のユーザに対してフロントランニングを行っているかを検出することさえ困難です。

7 Acknowledgment

We thank our advisors Prateek Saxena, Vitalik Buterin and Patrick McCorry for useful discussions and feedback on the early version of the paper.

References

1. Bentov, I., Ji, Y., Zhang, F., Li, Y., Zhao, X., Breidenbach, L., Daian, P., Juels, A.: Tesseract: Real-time cryptocurrency exchange using trusted hardware. Cryptology ePrint

Archive, Report 2017/1153 (2017), https://eprint.iacr.org/2017/1153.pdf, accessed:2017-12-04
2. Breidenbach, L., Daian, P., Juels, A., Tramèr, F.: To sink frontrunners, send in the submarines. http://hackingdistributed.com/2017/08/28/submarine-sends/ (2017)
3. Buterin, V.: Minimal viable plasma. https://ethresear.ch/t/minimal-viable-plasma/426 (2018)
4. Buterin, V.: Plasma cash. https://ethresear.ch/t/plasma-cash-plasma-with-much-less-per-user-data-checking/1298 (2018)
5. Coleman, J., Horne, L., Xuanji, L.: Counterfactual: Generalized state channels. http://l4.ventures/papers/statechannels.pdf (2018)
6. Dziembowski, S., Faust, S., Hostakova, K.: Foundations of state channel networks. https://eprint.iacr.org/2018/320.pdf (2018)
7. Gilad, Y., Hemo, R., Micali, S., Vlachos, G., Zeldovich, N.: Algorand: Scaling byzantine agreements for cryptocurrencies. In: Proceedings of the 26th Symposium on Operating Systems Principles. pp. 51–68. SOSP '17, ACM, New York, NY, USA (2017), http://doi.acm.org/10.1145/3132747.3132757
8. Joseph, P., Vitalik, B.: Plasma: Scalable autonomous smart contracts. https://plasma.io/ (2017)
9. Kokoris-Kogias, E., Jovanovic, P., Gasser, L., Gailly, N., Syta, E., Ford, B.: Omniledger: A secure, scale-out, decentralized ledger via sharding. Cryptology ePrint Archive, Report 2017/406 (2017), https://eprint.iacr.org/2017/406
10. Luu, L.: Peacerelay: Connecting the many ethereum blockchains. https://medium.com/@loiluu/peacerelay-connecting-the-many-ethereum-blockchains-22605c300ad3 (2017)
11. Luu, L., Narayanan, V., Zheng, C., Baweja, K., Gilbert, S., Saxena, P.: A secure sharding protocol for open blockchains. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. pp. 17–30. CCS '16, ACM, New York, NY, USA (2016), http://doi.acm.org/10.1145/2976749.2978389
12. Miller, A., Bentov, I., Kumaresan, R., McCorry, P.: Sprites: Payment channels that go faster than lightning. https://arxiv.org/pdf/1702.05812.pdf (2017), https://arxiv.org/pdf/1702.05812.pdf, accessed: 2017-03-22
13. Miller, A., Xia, Y., Croman, K., Shi, E., Song, D.: The honey badger of bft protocols. Cryptology ePrint Archive, Report 2016/199 (2016), https://eprint.iacr.org/2016/199
14. Pagnia, H., Gärtner, F.C.: On the impossibility of fair exchange without a trusted third party. Tech. rep. (1999)
15. Poon, J., Dryja, T.: The bitcoin lightning network. https://lightning.network/lightning-network-paper.pdf (2016), https://lightning.network/lightning-network-paper.pdf, accessed: 2016-07-07
16. Syta, E., Jovanovic, P., Kogias, E.K., Gailly, N., Gasser, L., Khoffi, I., Fischer, M.J., Ford, B.: Scalable bias-resistant distributed randomness. Cryptology ePrint Archive, Report 2016/1067 (2016), https://eprint.iacr.org/2016/1067
17. team, T.Z.: The zilliqa technical whitepaper. https://docs.zilliqa.com/whitepaper.pdf (2017)
18. Wiki, B.: Atomic cross-chain trading. https://en.bitcoin.it/wiki/Atomic_cross-chain_trading (January 2018)
19. Zhao, C.: Binance q2 — recap. https://www.linkedin.com/pulse/binance-q2-changpeng-zhao/ (2018)