Computer Security Symposium 2017
23 - 25 October 2017

分散台帳技術におけるインテグリティとプライバシー保護

吉濱佐知子 †1　齋藤新 †1

概要：分散台帳技術は，中央的な管理主体を持たずに複数組織間で取引データを共有し，そのインテグリティを保護することを可能とする技術である．一方で，台帳上に記録される取引の内容やその存在自体を，第三者から秘匿したいというニーズも非常に強い．分散台帳技術では，同じ台帳の複製を複数の参加者が持つため，データの機密性やプライバシーを保護するのは本質的に難しい．本論文では，分散台帳技術におけるインテグリティとプライバシーの保護の要件を定義し，具体的なユースケースにおける要件を議論する．また，代表的な分散台帳技術の実装におけるそれぞれの要件への対応状況に関する考察を述べる．

キーワード：ブロックチェーン，分散台帳技術，電子商取引

Protection of Integrity and Privacy on Distributed Ledger Technologies

Sachiko Yoshihama †1　Shin Saito †1

Abstract: Distributed Ledger Technology (DLT) enables sharing transaction data across multiple organizations and protecting data integrity without having a central authority. On the other hand, there are strong needs to protect privacy of transaction from third parties. Since DLT is a technology to share the replica of the same ledger data across multiple parties, it is essentially difficult to protect confidentiality and privacy of data. This paper first defines the properties of integrity and privacy requirements, and discuss requirements in typical use cases. Finally, the paper reviews how the requirements are addressed in existing implementations.

Keywords: Blockchain, Distributed Ledger Technology, Transaction Systems

1. はじめに

2008 年に Satoshi Nakamoto と名乗る人物により発表された論文[5]では，一連の暗号プロトコルにより，中央的な管理主体なしに転々流通可能な仮想通貨を実現する技術を提案した．これが後にビットコインという名の仮想通貨として運用され，大きな注目を集めている．ビットコインを実現している基盤技術は，ブロックチェーン，または分散台帳技術などと呼ばれ，この 技術を金融取引等の仮想通貨以外の業務上のユースケースで使用するための試みが世界の金融機関を中心に大きく広がっている．

一方で，分散台帳技術を業務使用する場合に問題となるのがプライバシーである．分散共有台帳の本質は「複数の参加者が取引台帳の複製を共有し，参加者間の合意をとりながら取引内容を追記していく」というところにある．その名の通り，同じ内容の台帳を複数の参加者が保有するため，一部の参加者が台帳を書き換えたとしても容易に検知が可能で，そのためにデータのインテグリティを保護することに長けた，改竄耐性の強い技術であると言われている．改竄を成功させるためには，コンセンサス方式にもよるが，原則的に過半数かそれ以上の参加者が結託する必要がある．一方で，同じ台帳を複数の参加者が持つため，データの機密性やプライバシーを保護するのは本質的に難しい．

本論文では，まず分散台帳技術の基本機能を説明したのち，分散台帳技術におけるインテグリティとプライバシーの保護の要件を定義し，いくつかの具体的なユースケースにおける要件の例を述べる．最後に，代表的な分散台帳技術の実装におけるそれぞれの要件への対応方針を議論する．

2. 分散台帳技術の基本機能

2.1 データ構造

分散共有台帳技術ではネットワークの参加者内でデータが共有されるが，このデータは大きく分けて２つの種類がある．

2.1.1 ブロック

過去に承認されたトランザクションの履歴を「ブロック」という塊にまとめて保持したもの．暗号的ハッシュ関数により各ブロックのハッシュ値を計算し，それをその直後のブロックの中に挿入しているため，過去のトランザクショ

†1 日本アイ・ビー・エム東京基礎研究所
IBM Research – Tokyo

© 2017 Information Processing Society of Japan

ンの改竄を検知することができる．また，最新のブロックのハッシュ値だけを比較することで過去の全データの同一性を確認することができる．ブロックを暗号的ハッシュ関数のハッシュチェーンによって連結していることが，ブロックチェーンという名称の由来となっている．

2.1.2 ステート

トランザクションを実行した結果の最新状態を管理する一種のデータベースである．

たとえば，分散台帳上でユーザの口座から入出金を行うアプリケーションを作った場合，各入出金の取引記録がブロックに記録され，最新の口座残高がステートに記録されるといった使い方が一般的である．

2.2 トランザクション処理方式

トランザクション処理方式としては，大きく分けて UTXO とスマートコントラクト方式がある．

2.2.1 UTXO

UTXO (Unspent Transaction Output) は Bitcoin を始めとする多くの分散台帳技術で採用されている方式である．例えば Bitcoin では, 新しいトランザクションを発生させる場合, 過去にコインを受領した際のトランザクションへのポインタを入力とし，送金先の公開鍵を解除条件としたものを出力とすることで，コインの所有権を移転する．移転する仮想通貨の情報以外にもメタデータを付与することが可能であり，資産移転や金融取引等の通貨以外のユースケースでも UTXO が使用されている．

2.2.2 スマートコントラクト

スマートコントラクトは，UTXO より柔軟にトランザクションの発生条件を定義することを可能とする技術であり，Ethereum[3]や Hyperledger Fabric[2]で採用されている．ここではチューリング完全なプログラミング言語によるロジックがステートに格納された状態変数を参照しながら処理を行う 複雑な判断ロジックによる状態遷移や, 複雑な計算を含む任意のアクションの実行が可能となる．

2.3 コンセンサス

分散台帳技術において，ネットワークに参加する複数のノード間で台帳の同一性や健全性を保持するためには，参加ノード間でなんらかの方法で合意をとる必要がある．本論文では，合意を形成する行為を「コンセンサス」と呼ぶ．コンセンサスの方式は大きく分けて２種類ある．

2.3.1 Proof-of-Work **およびその亜種**

Bitcoin に代表される proof-of-work (PoW)は， 参加者が特定の条件を満たす値を見つける一種の計算競争である「マイニング」を行い，最も早く値を見つけたノードが新しいブロックを追加するとともに報酬として仮想通貨を受け取るという仕組みである．Proof-of-work においては，新しいブロックを追加する際に，そのブロックに格納する一連のトランザクションと，任意のナンス値をあわせたものに対する暗号的ハッシュ値を計算し，このハッシュ値が指定されたターゲット値以下となるようなナンスが，特定の条件を満たす値となる．暗号的ハッシュ関数では一般に出力値から入力値を推測する効率的な方法は存在しないため，総当り方式で計算を行う必要があり，そのためにより大きな計算資源を保有するほど計算競争に勝つ確率が高くなるが, 消費エネルギーも増大する．計算競争に勝つ確率を計算資源量以外の属性に依存して変動させる，proof-of-stake (PoS) や proof-of-importance(PoI) のような亜種が存在する．

PoW の課題として上げられるものとして，安全性とファイナリティの欠如が上げられる．安全性とは，ネットワーク内に悪意のある参加者がいた場合に，不正なトランザクションを承認して有効としてしまうような攻撃に対する耐性を指す． 例えば攻撃者がある時点でマイニングに成功し，不正なトランザクションを含む新規のブロックを追加し，そのブロックを他のノードに伝搬する可能性がある．Bitcoin 論文[5]では, そのような場合でも攻撃者が連続的にマイニングに成功する確率は低く，他の健全なノードは不正なトランザクションを含むブロックを棄却するため，ブロック確定後に時間が経過するほど安全性は高まる，としている．しかし複数の参加者が結託してネットワーク内の 50%以上の計算資源を使用可能となった場合に，任意の不正なトランザクションを承認可能となる．

ファイナリティとは金融用語で，決済が完了し，後から絶対に取り消されないこと，というような意味である[4]．PoW やその亜種の場合には，マイニングに成功したノードで新規ブロックが追加されても，そのブロックがネットワーク上で伝搬される前に他のノードがマイニングに成功し, 事実上ネットワーク内に異なる内容の台帳が存在してしまうことを回避できない．このような状態は，ブロックチェーンが枝分かれしたとして「フォークする」と呼ばれる．フォークが発生した場合，多数の参加者が支持してマイニングを続けたほうの枝に新たなブロックが追加される確率が高いので，枝が長く伸びた方が正となり，短い方の枝に含まれるブロックは棄却される．そのため，短い方の枝にだけ記録されていたトランザクションは，一旦確定した後に無効となる 可能性があり, ファイナリティが保証されない．また，P2P ネットワークを分断してノード間の通信を阻害することで，一部のノードが最新ブロックを受け取れない状態を作り, 故意にブロックチェーンの分岐(fork)をひきおこすエクリプス攻撃[12]の可能性も指摘されている．PoW やその亜種は, 不特定多数の匿名のノードが参加するパブリックなネットワークにおいて比較的少数の攻撃者を仮定しつつ，コンセンサスをとることができるという意味で優れているが，安全性やファイナリティの欠如が業務使用の場合の課題となると考えられている．

2.3.2 分散コンセンサス形成アルゴリズム

分散台帳技術の業務使用 が検討される上で, 参加者に対する認証と認可を行う参加許可制のネットワーク

© 2017 Information Processing Society of Japan

(permissioned network) に対する興味が高まっている．参加許可制のネットワークでは，ノードの数や参加資格をある程度制御することができるので，メッセージ交換による分散コンセンサス形成アルゴリズムを採用することが可能となる．Paxos[13], View Stamp Replication (VSR)[14], Raft[15]や，ビザンチン障害耐性のある PBFT[6]などが知られている．これらの分散コンセンサス形成アルゴリズムは一般に eventual-synchrony network model という非同期型のメッセージ通信を前提とし，一連のメッセージがネットワーク内に発信 (broadcast) された後，コンセンサスを経て一定時間内に健全なノードに配信 (deliver) され，この際にノード間でメッセージの全順序が保証される．このような仕組みは「アトミック・ブロードキャスト(atomic broadcast)」と呼ばれ， 以下の性質を持つ[9]:

- 有効性 (Validity): 健全なノードがメッセージ m を発信した場合，そのメッセージ m はいずれそのノード上に配信される．
- 合意 (Agreement)：ある健全なノードにメッセージ m が配信される場合，そのメッセージは全ての健全なノードに配信される．
- 完全性 (Integrity)：健全なノードは各メッセージを最大１回しか配信しない．
- 全順序 (Total Order)：任意の 2 つの健全なノードの間で，常に配信されたメッセージの順序が同一となる．

3. 分散台帳技術におけるインテグリティとプライバシー保護要件

分散台帳技術の利点の１つとして，複数の参加者が台帳の複製を保持するため，一部の攻撃者による台帳の改竄を検知可能となり，データのインテグリティが担保できるというものがある．特にビザンチン障害耐性を考慮したコンセンサスの仕組みを採用した場合，一部の参加者が不正なトランザクションの承認行為を含む任意の振る舞いをしても，残りの参加者でコンセンサスが形成される限り，台帳のインテグリティを保護しつつ処理を継続することが可能となる．

一般に，ビザンチン障害耐性のあるアルゴリズムであれば，コンセンサス形成に N 個のノードが参加している場合，そのうちの f 個のノードに障害があっても N-f 個のノードが健全である限りコンセンサス形成が可能である．

一方で，台帳が共有されることにより全ての参加者に同じ情報が共有されるため，取引のプライバシーを守るのは本質的に難しい．例えば証券取引を分散台帳化した場合に，取引内容を売り手・買い手の二者間で共有する必要があるが，それ以外の第三者には開示しないというのは，業務上および法規制上の要件である．

アルゴリズムによっては，ネットワーク内の全ノードではなく，取引によって影響を受ける N_{cp} 個のカウンターパーティ（例えば売り手と買い手 の場合 N_{cp}=2）だけで取引の情報を共有し，コンセンサス形成する場合もある．しかし実際に二者間でだけその取引の情報を共有している時に，どちらかがビザンチン障害を発したとすれば，どちらの言い分が正かを決定することが困難となり，インテグリティを保護することは難しくなる．直感的には，コンセンサス形成に参加する参加者が多いほど，インテグリティ保護は向上する．

以下に，典型的なユースケースにおいて求められるインテグリティとプライバシーの性質を説明する．

3.1 インテグリティ

あるネットワーク上に N 個の分散台帳ノード $n_1, n_2, ...n_N$ が存在し，それぞれのノードが台帳 dlt_x を保持するものとする． この時，i 番目のノードの持つ m 個のトランザクションが記録された台帳の持つ情報は以下のように表現できる．

$$dlt_i = \{ h(tx_1), h(tx_2), \ldots h(tx_m)\}, \{ tx_1, tx_2, \ldots tx_m \}$$

ただし tx_j は j 番目のトランザクション，$h(tx_j)$ はその暗号的ハッシュ値とする．この時，前述のアトミック・ブロードキャストの性質に加えて，以下のインテグリティ要件が考えられる．

- **[I1]トランザクション内容の正当性**：インテグリティ要件の最も高いものとして，アプリケーション・ロジックに応じたトランザクション内容の検査がある．例えば仮想通貨における二重払いの検知や，残高の不足している口座からの支払い，などが例として上げられる．すなわちあるノード n_i においてトランザクション tx_p の内容が正当であることを $n_i.V(tx_p)$ とすれば，健全なノードの任意のペア n_i, n_j において $\forall$tx= $\{ tx_1, tx_2, \ldots tx_m \}$, $n_i.V(tx) = n_j.V(tx)$ が成り立った場合にトランザクションの正当性が確保されているものとする．
- **[I2] 改竄耐性**：台帳の内容が改竄されておらず，同一であることが求められる．トランザクションの改竄を検知するには，そのハッシュ値の同一性が確認できればよいので，$h(tx)$ をトランザクション tx のハッシュ値としたとき， 健全な任意のノード n_i, n_j において，$\forall h(tx)= \{ h(tx_1), h(tx_2), \ldots h(tx_m)\}$, $n_i.h(tx)=n_j.h(tx)$ が成立する場合に，トランザクションの改竄への耐性があるものとする．
- **[I3] ファイナリティ**：前述の通り，一旦確定されたトランザクションが後から取り消されないという性質を指す．すなわち任意の健全なノード n_i において，ある時点 T_t で $tx_p \in dlt_i$ のとき，T_{t+1} の時点でも常に $tx_p \in dlt_i$ となる．

3.2 プライバシー

簡単のために二者間取引を例として考えた場合，あるネットワーク上に N 個の分散台帳ノード $\{n_1, n_2, \ldots n_N\}$ が存

© 2017 Information Processing Society of Japan

在し，各ノードを経由して取引を行う，任意の数の取引主体 $S=\{s_1,s_2,\ldots\}$ が存在するものとする． その中の任意の二者 s_p, s_q,がそれぞれノード n_i, n_jを経由して取引 tx を行う時，各取引は $\{\mathrm{tx}, s_p, s_q, n_i, n_j\}$ で表すことができる．ただし主体 s_p, s_q は匿名化されており，主体 s_i のアイデンティティは ID_i と表すものとする．この時，プライバシー保護要件としては以下のようなものがある．

- **[C1] 第三者に対する匿名性**：トランザクションを実行した主体の身元を，取引当事者以外の第三者から秘匿する要件である．すなわち，n_i, n_j, s_p, s_q,は互いに身元 $s_p=ID_p$, $s_q=ID_q$ を知りうるが，第三者の $s_r \neq s_p$, s_q, は知り得ない，という要件である．
- **[C2] 取引当事者同士の匿名性**: 取引当事者同士であっても匿名性を保持する要件である．つまり n_i, s_p は $s_q=ID_q$ を知り得ず， n_j, s_q は $s_p=ID_p$ を知り得ない． 例えば匿名のアドレスへの仮想通貨の送金は基本的には匿名性が保護されていると言える．
- **[C3]トランザクション内容の秘匿**：取引の詳細，たとえば取引金額や，証券や資産の名称・数量などの情報を取引当事者以外から秘匿する要件である．すなわち，$\{\mathrm{tx}, s_p, s_q, n_i, n_j\}$ の時，$s_r \neq s_p$, s_q となる s_r, および $n_k \neq n_i$, n_j となる n_k は tx の内容を知ることができるべきでない，という要件である．（ただし取引内容のハッシュ値 $h(tx)$ など内容の分からない情報はこの要件には含まないものとする）
- **[C4]トランザクションの存在の秘匿**：トランザクションが発生した事実そのものを $s_r \neq s_p$, s_q となる s_r および $n_k \neq n_i$, n_j となる n_k から秘匿するという要件である．
- **[C5] ステートの秘匿**：トランザクションを実行した結果の最新状態を管理するのがステートである．任意の主体 s_p が n_i のみ知りうる状態変数 v を持っている場合，これを $n_k \neq n_j$ となる n_k から秘匿するという要件である．

4. 代表的なユースケースにおけるインテグリティとプライバシー保護要件

本章では，代表的ないくつかのユースケースにおける，インテグリティとプライバシー保護要件を説明する．

4.1 仮想通貨

仮想通貨は，ブロックチェーンのユースケースの最も代表的なものといえる．Bitcoin を始めとして，本稿執筆時点で数百種類の仮想通貨がブロックチェーン上に実装されている．単純な価値の移転を行うユースケースのため，殆どは UTXO によるトランザクション処理を行い，パブリック・ネットワーク上で PoW またはその亜種をコンセンサスの仕組みとして用いている．

仮想通貨においても，トランザクション内容の正当性確認(I1)，改竄耐性(II)など一般的なインテグリティ要件が求められる．仮想通貨による決済が行われる以上，ファイナリティ(I3)に対する要求もあるはずでるが， PoW またはその亜種を前提としているために保証できない．ブロック確定後に一定時間待つことでトランザクションの棄却されるリスクを下げるなど，運用でカバーするのが一般的である．

プライバシーに関しては，匿名の公開鍵から導出したアドレスを用いて送受金を行うことで，匿名性要件 C1,C2 については充足する．ただし PoW の過程で参加者がトランザクション内容の検査を行うため，トランザクション内容(C3)やその存在(C4)自体は秘匿できない．ステート DB 内容の秘匿(C5)に関しては一般には仮想通貨ではステート DB を必要としないので該当しないが，パブリック・ネットワークである限り秘匿性を守るのは難しい．

4.2 資産管理

自動車や不動産などの有形資産，またはデジタルコンテンツなどの無形資産の所有権を分散台帳上で取引する試みが始まっている[16]．取引される資産をトークンとして表現し，そのトークンの所有者を UTXO で移転することが可能である．資産管理においてもトランザクション内容の正当性確認(I1)，改竄耐性(II)，ファイナリティ(I3)など一般的なインテグリティ要件が必要となる．プライバシー要件は，取引される対象資産によって異なる．例えば日本では不動産登記情報は一般に公開され，誰でも申請すれば取得できる[17]．その前提であれば，C1～C5 までの全てのプライバシー要件は不要といえるが，ただし個人情報に該当する内容は秘匿する，などの要件が存在すると考えられる．

4.3 証券ポストトレード処理

証券取引において，売買注文がマッチングした後に行う清算・決済などのポストトレード処理を分散台帳化する試みは，日本取引所グループによる実証実験[7]などが実施されている．清算・決済処理では複数の非同期な外部イベントにまたがった証券や資金の状態遷移や，ネッティングなどの複雑な計算処理が必要であるため，スマートコントラクトによる処理や，ステートによる状態管理が必要となる．

大量の資金や証券の決済が行われるため，トランザクション内容の正当性確認(I1), 改竄耐性(II)は重要であり， また，特に金融システムとして考えた場合ファイナリティ(I3)の保証に関する高い要求がある．ただし， 参加許可制ネットワークで証券会社や清算機関等の信頼度の高い金融機関同士が取引を行うという前提に立てば，コンセンサス形成を行うノードが N_{cp} 個の当事者同士のみ などと少数であっても，実用上問題がないと考えられる．

一方で，トランザクション内容の正当性確認のためには，取引当事者以外との間で情報フローが発生する．例えば，ある参加者 A が複数の別の参加者と個別に二者間取引を行っているとき， その参加者 A が売ろうとしている数の証券を実際に保有しているかといった口座残高情報は，複数の相手との取引をまたがってチェックする必要がある．そ

© 2017 Information Processing Society of Japan

のため，信頼モデルとしては不正や改竄のリスクが小さいとしても，システム的に各参加者でデータを共有する必要がある．

一方で金融取引においては，プライバシー保護に関する法規制上の制約は非常に強い．特に金融機関同士が取引を行う場合に，その取引の内容や，特定の二者間で取引が存在したこと自体を秘匿する必要がある．例えば株式の売買 において，その売買の内容が事前に漏れることにより株価が変動する可能性がある．また売買成立後に予想される株価の変動を利用して利益を得る，といった不正が可能になる．そのため，第三者に対する匿名性(C1)，トランザクション内容の秘匿(C3)，ステートの秘匿(C5)などは必須の要件となる．前述のように，口座残高情報が取引相手をまたがって共有される必要がある場合に，プライバシー要件との両立が困難である．

取引当事者が既知の金融機関同士と仮定すれば取引当事者同士の匿名性(C2)は要件から除外されるが， 個々の金融機関に所属する投資家の情報などは C2 にも該当する可能性がある．トランザクションの存在(C4)に関しては完全に秘匿するのが望ましいと思われるが，十分に大きな K 個の参加者のうちの誰が取引を行っているか特定できない，すなわち K-匿名性を保証できる状態であれば，許容できると考えられる．

4.4 金融契約

相対によるデリバティブ取引や，デリバティブを証券化した金融商品は一種の金融契約であり，スマートコントラクトにより実現可能な代表的なユースケースの一つである．

例えば株価連動社債 (ELN: Equity Linked Note)は，株式とオプション取引を組み合わせた金融商品である．ELN はまず元本より安い価格で販売され，予め決めた評価日に，連動している株式の価格が契約上の基準値より高い場合には元本が支払われ，低い場合には投資家は損をする，というようなものである．このようなデリバティブ取引の場合は，対象となる株式自体は取引されることがない．あらかじめ一定の契約条件が決まっており，証券の価格の変化という外部イベントにより，支払い条件等が変化する．このようなユースケースの場合は，スマートコントラクトによる条件の判断や， ステートによる状態管理が必要となる．

金融契約においてトランザクション内容の正当性確認(I1)，改竄耐性(I2)は重要であり，また，特に金融システムとして考えた場合ファイナリティ(I3)の保証に関する高い要求がある．一方で，契約であるという性質上， コンセンサス形成を行うノードが N_{cp} 個の当事者同士のみであっても問題がないと考えられる．また証券と異なり残高管理すべき実体がないので，データの共有範囲が当事者同士のみであっても問題ないが，監査や争議解決などの必要がある場合はその限りではない．プライバシー保護に関しては前項と同じく，金融取引におけるプライバシー保護要件として C1, C3, C4, C5 の充足が必要と考えられる．

5. 代表的な分散台帳技術

以下に代表的な分散台帳技術における実装方式と，インテグリティおよびプライバシーの実現可否について考察を述べる．

5.1 Bitcoin

Bitcoin[5]では分散台帳として保持されるデータはブロックのみであり，トランザクション処理方式は UTXO である．コンセンサス方式は PoW であり，確率的なものであるため，任意の時点におけるインテグリティを保証できない．障害への耐性はノードの絶対数ではなく計算能力で表現されるが，障害を持つノードの計算能力が $f<2/N$ である限り時間の経過とともにインテグリティ要件の充足は向上するが， 確率的でありファイナリティがない． プライバシー要件については，取引を行う主体が匿名の公開鍵を元にしたアドレスで指定されるため，匿名性 C1,C2 は保持される．しかし，取引内容は全てのノードに共有されるため，匿名性以外のプライバシー要件 C3,C4,C5 は満たすことができない．

5.2 Ethereum

Ethereum[3]では分散台帳として保持されるのはブロックおよびステートである．トランザクション処理方式はスマートコントラクトであり，チューリング完全な Solidity 言語により実装されたアプリケーション・ロジックにより，トランザクションの正当性を判断する．しかしコンセンサス方式は PoW であるため，インテグリティ要件の充足も Bitcoin と同様に確率的となり，ファイナリティがない．スマートコントラクトの実装方式にもよるが，匿名アドレスを使用すれば匿名性 C1,C2 は充足されるが，Bitcoin 同様匿名性以外のプライバシー要件は満たされない．

5.3 Quorum

Quorum[10]は Ethereum を許可制ネットワーク向けに改造したもので，PoW の代わりに投票に基づくコンセンサスの仕組みを取り入れ，プライバシー機能やパフォーマンスを強化している．Quorum では，各参加者は通常の Ethereum ノードと同等の Quorum Node に加えて，Transaction Manager ノードを持つ． パブリックなトランザクションは Ethereum と同様に処理されるが，プライベート・トランザクションは暗号化して Quorum Node から Transaction Manager に送られ，その情報を参照する権限のある参加者の Transaction Manager 間だけで共有される．Quorum Node 間では，トランザクションのハッシュ値だけが共有される．トランザクションを処理する際は，当該プライベート・トランザクションの実体を持つ参加者だけがこれを復号し，Ethereum Virtual Machine (EVM) 上でスマートコントラクトを実行してトランザクション正当性の判定を行う．

Quorum のコンセンサスアルゴリズムとしては，

© 2017 Information Processing Society of Japan

QuorumChain と Raft との 2 種類がサポートされているが，本稿では QuorumChain を対象とする．QuroumChain では，ノードは投票権を持つ voter と，新しいブロックを作成する権利を持つ maker の, 最大２つの役割を持つことができる．何も役割を持たないノードは受動的な observer とみなされる．新しいブロックは以下の手順で作られる:

1) Maker は新しいブロックを作って署名し，P2P プロトコルで他のノードに伝播する. ブロックの中にはトランザクションのリストと，直前のブロックに対する vote が含まれる.
2) Voter ノードはブロックが権限を持つ maker から送られてきたことを検証し, ブロックに含まれるトランザクションをスマートコントラクトにて実行する. また, トランザクション実行結果のステートのハッシュ値と, 自身の持つステートのハッシュ値との整合性をチェックする.
3) Voter は検証が完了したら投票を行う．この時，BlockVoting という特殊なスマートコントラクトに対して, vote トランザクションを発生させることで投票を行う．コンセンサスを形成するための投票自体が vote 対象となるトランザクションとして処理されるため, 投票の結果が判明するのは次のブロック生成時となる.
4) Maker はタイムアウト前に必要な閾値以上の vote が得られていればブロック成立とし, そうでなければ上記 1)〜3)の処理を繰り返す.

なお必要な maker の数は 1 つ以上とされており，複数の maker が存在する場合は，ブロック生成のタイミングが衝突しないように, 各 maker はランダムな時間ウェイトした後に新しいブロックを作るようになっている．Voter はブロックの検証に成功する限り vote を行い，同一番号のブロックに対して複数回投票することが許される. 各ノードは必要数以上の vote の付与されたブロックを許容し，自身の持つブロックのデータに付け加える. 同じ番号のブロックが複数存在する場合，最も多く vote を得ているブロックが正となる.

このような仕組みであるため QuorumChain では maker が 1 つの場合には単一障害点となりうる上，maker がビザンチン障害であればチェーンが枝分かれ（fork） する可能性がある．また，maker が複数存在する場合，それらの maker の間で通信遅延が発生した場合にも複数の個なるバージョンのブロックが作成される可能性があり, fork が発生しうるため，ファイナリティを確保できない.

プライベート・トランザクションでは，トランザクション内容の正当性は権限のある任意のノード集合により検証される． カウンターパーティがコンセンサス形成に参加する必要性を考えると典型的には $N_{cp} \leqq x \leqq N$ となるような x を閾値として投票を行い, コンセンサス形成されると考えられる．一方で，権限を持たないノードもトランザクションのハッシュ値を持つのでインテグリティ検証には協力できるため，$f<2/N$ となる最大 f 個のノードの障害に耐性がある.

プライバシーに関しては, プライベートなトランザクションは暗号化されるため，C1，C3 は充足される．ただし当事者同士は相手を識別できるので C2 は該当しない. また, 第三者にもハッシュ化されたトランザクションが伝播されるため，C4 も充足しないと思われる．ステートに関しては，当事者だけがトランザクションを復号して実行し，結果としてステートを更新するので，C5 は充足する.

5.4 R3 Corda

Corda[11]はブロックデータを持たない．ステートがデータの最小単位である．個々のステートは取引される資産などを表しており, spent または unspent の状態を持つとともに，添付ファイルなどの付加情報を持つことができる．トランザクション処理方式は UTXO の拡張であり, 個々のトランザクションの中で，入力ステートと出力ステート，および notary（公証人）を指定する．また， Java バイトコードで定義されたロジックをステートに紐付けておき，トランザクションが発生した時にスマートコントラクトして実行して，トランザクションの正しさを検証することができる.

Notary はトランザクションの全ての入力ステートが unspent 状態である場合（すなわち二重使用が発生していない場合）ことを保証する． 個々のトランザクションは同一の notary に属するステートだけを入力とすることができるので，一つの notary でトランザクションの正しさを検証することができる．また，notary をまたがる取引のために，notary を変更するという特殊なトランザクションがある.

また，複数参加者間で複数のステップにより実行されるような複雑なトランザクションは flow と呼ばれる状態機械にエンコードされて実行される.

トランザクションは全てのノードに配布されるのではなく，トランザクションの検証のために，依存関係にあるトランザクションを参照する必要のあるノードに対して配布される．また，ブロックも持たないため，トランザクションの全順序は定義されない．依存関係のあるトランザクション間の半順序関係のみが定義され，notary がその順番を保証することとなる. なお notary は特定のコンセンサスプロトコルに依存せず， 1 つのノードが中央集権的に notary を務める場合と，複数の notary のノード間で Raft や BFT-SMaRt などのプロトコルによりコンセンサス形成する仕組みがサポートされている.

Corda ではコンセンサス形成に参加するのは Notary と N_{cp} 個のノードであり，これらのノードのビザンチン障害を許容しない．特に notary が中央集権的な場合，単一障害点となるとともに，fork を発生させることができる.

© 2017 Information Processing Society of Japan

プライバシーに関しては，あるトランザクションを処理するとき，個々のノードは依存関係のある全てのトランザクションを取得して，検証を行う．そのため，原則的には当該取引に関係のあるノードだけがトランザクションの内容を見ることができるはずではあるが，複雑な依存関係を持つトランザクションについて，どのノードが参照権限をもつかということを正確に判定するのは難しい．ただし，署名鍵を匿名性のあるものにすることで，匿名性を持たせることは可能である[11].

5.5 Hyperledger Fabric V0.6

Hyperledger Fabric[2] V0.6 は, 許可制ネットワークを対象としたオープンソースの分散台帳基盤である．分散台帳として保持されるのはブロックおよびステートである．トランザクション処理方式はスマートコントラクトである．Go 言語等チューリング完全な言語を複数サポートし，スマートコントラクトの中でトランザクションの正当性を判断する．コンセンサス方式はプラッガブルとされているが，事実上 PBFT[6]が使用されている（図 1）．PBFT の性質により，$f \leqq (N-1)/3$ までの f 個のノードのビザンチン障害に耐性があり，その条件の下でインテグリティ要件 I1，I2 を満たすことができ，また PBFT の性質によりファイナリティ要件 I3 を保証する．ただし，スマートコントラクト実行前にコンセンサスを形成し，スマートコントラクトの実行結果については合意しないため，スマートコントラクトが完全に決定的な振る舞いをする必要がある．

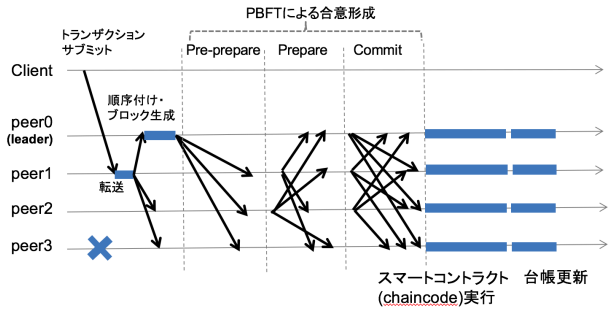

図 1 Hyperledger Fabric V0.6 コンセンサス

プライバシーに関しては，許可制ネットワークのため全ての取引主体が ECert (Enrollment Certificate)と呼ばれる証明書により身元を証明した上でネットワークに参加する前提となっているが，実際の取引に対する署名には TCert (Transaction Certificate) と呼ばれるワンタイムの匿名証明書を使用する．そのため，匿名性要件 C1, C2 は基本的に充足可能だが，スマートコントラクトの実装により，トランザクションの内容として名前やユーザ ID が含まれる場合はその限りではない．トランザクション内容やステート DB の内容は暗号化され秘匿されるが，悪意を持ったノード管理者であればノード上の暗号鍵を利用して復号される可能性がある．そのため，要件 C3, C5 の保護には，全てのノードを信頼できる管理化に置くか，もしくは管理者権限であっても攻撃のできないように，ハードウェアにより保護された Trusted Execution Environment のような安全な実行環境や，ハードウェアセキュリティモジュールによる鍵管理が必要となる．なおトランザクションの発生 C4 に関しては，主体 s_r に対しては秘匿が可能だが，全てのノードが同じブロックとステートを保持するため，ノード n_k に対する秘匿はできない．

5.6 Hyperledger Fabric V1.0

Hyperledger Fabric V1.0 は，V0.6 の短所を改善するためにアーキテクチャを大きく変更したオープンソースの分散台帳基盤である．V0.6 と同様に，分散台帳として保持されるのはブロックおよびステートである．Fabric V1.0 ではノードの役割が別れており，台帳を保持するノードを peer，トランザクションの順番付けを行うノードを orderer と呼ぶ．また，特にスマートコントラクトを実行する機能を持つノードを endorsing peer と呼ぶ．

コンセンサスは以下のような仕組みとなっている（図 2）．まずクライアントが endorsing peer に対してトランザクション提案を行う．Peer はスマートコントラクトを実行し，実行結果と，実行時に行われるステートへの読み書きの情報 (read-write set と呼ばれる)に対して署名をしたものを，endorsement （承認）としてクライアントに返却する．クライアントは複数の peer に対して同じ処理を繰り返し，endorsement policy に規定された必要な数の endorsement を取得した後に, orderer に対してトランザクションをサブミットする. Orderer は一連のトランザクションに対する全順序を定義し，1 ブロック分のトランザクションを peer に配信する．最後に，peer は配信されたトランザクションに付与された endorsement の署名を検証した上で，トランザクション間でステートの上書きによる不整合が発生していないかチェックして，問題のないトランザクションをブロックに書き込むと同時に，ステートを更新する．

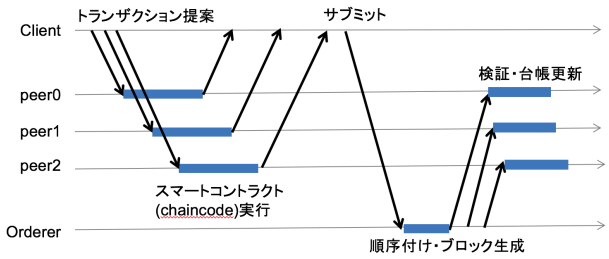

図 2. Hyperledger Fabric V1.0 コンセンサス

一部の endorsing peer のビザンチン障害に耐性があり，endorsement policy の定義によって異なるが, 多数決方式で決めるとすれば最大で $f<N/2$ となる f 個の peer の障害を許容する．ただし endorsement policy は peer の属する組織単

© 2017 Information Processing Society of Japan

位で指定され， policy としては同一組織に属する複数の peer を識別して指定できないため，同一組織内の peer が同じ信頼レベルを有するという仮定を置く必要がある．すなわち，N はチャネルに参加する組織の数であり，$f<N/2$ となる f 個の組織の障害を許容する，と読み替えることができる．ただし，ある組織の障害とは，その組織に属するいずれかの peer が障害を発生している状態を指す．

また，N の数が大きい時に，全ての組織の peer から endorsement を取得することはパフォーマンス上好ましくないので，現実的にはより少ない f を前提とした上で policy を決定する必要があると思われる．また V1.0 では orderer として Apache Kafka を採用した実装が公開されている．この場合 orderer は多重化によりクラッシュ障害耐性を持たせることができるが， ビザンチン障害耐性はない．

Fabric V1.0 では multichannel という仕組みを備えており，ネットワーク内の一部の peer 群が，その peer 群の間に閉じたチャネルを作成することが可能となっている．データの共有やコンセンサスもチャネルに閉じた形で行われるため，プライバシーやインテグリティ要件の充足も，チャネルを分離するかどうかによって異なる．

まず，同一のチャネルに参加している peer 間という観点に着目すると，適切な endorsement policy が決められており，orderer にビザンチン障害が発生していないという前提の下で，インテグリティ要件 I1～I3 を充足可能である．

また，orderer のビザンチン障害への耐性を実現するには，PBFT に基づいた orderer の実装など，今後の改善が必要となる． プライバシー保護に関しては，C3～C5 は Fabric V0.6 と同等の条件となる．一方で，ワンタイムの匿名証明書である TCert がサポートされないため，要件 C1，C2 については充足されない．

一方で，Multichannel を用いた場合には，トランザクションの送受信のフローや台帳データはチャネルの単位で分離されるため，C1～C5 の全てのプライバシー要件を充足する．ただし同時に，任意の peer の障害への耐性を持つためには，チャネル単位で $f<N/2$ が成立するような endorsement policy の定義をするように注意する必要がある．

6. まとめ

本論文では，分散台帳技術におけるインテグリティやプライバシーといったセキュリティ要件を形式的に定義し，既存の実装における要件の充足状況を分析した．分散台帳技術の仮想通貨以外への適用は，現時点では実証実験的なものが多いが，今後様々な業務に使用されるようになるにつれ，ますます厳密な安全性の評価が必要になってくると考えられる．一方で， 分散台帳技術の実装には，技術的詳細やソースコードが公開されていないものも多く，厳密な評価が難しい．分散台帳技術を普及するためには，安全性の性質や評価基準を正確に定義し，オープンに安全性を議論するとともに，安全に使用するための ポリシー設計の指針などを整理していく必要があると考えられる．

謝辞 本稿の執筆にあたっては，株式会社日本取引所グループ総合企画部の山藤敦史氏，近藤真史氏，保坂豪氏，土井惟成氏，および日本アイ・ビー・エム株式会社東京基礎研究所の立石孝彰氏，上條浩一氏他各位に有用なご意見をいただきました．ここに感謝します．

参考文献

[1] Andreas M. Antonopoulos, “Mastering Bitcoin”, December 20, 2014. ISBN-13: 978-1449374044.
[2]Hyperledger Fabric, https://github.com/hyperledger/fabric.
[3]Ethereum, https://www.ethereum.org
[4]日本銀行「決済システムの概要」 https://www.boj.or.jp/paym/outline/kg72.htm/
[5]Satoshi Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System, 2008.
[6]M. Kastro, B.Liskov, Practical Byzantine Fault Tolerance, Proceedings of the Symposium on Operating System Design and Implementation (OSDI), 1999.
[7]JPX Working Paper
[8]C. Dwork, N.Lynch, and L.Stockmeyer. Consensus in the presence of partial synchrony. Journal of the ACM, 35 (2): 288-323, 1988.
[9]C. Cachin and M. Vukolić, Blockchain Consensus Protocols in the Wild, URL: http://arxiv.org/abs/1707.01873
[10]Quorum https://github.com/jpmorganchase/quorum
[11]Mike Hearn, “Corda: A distributed ledger”, Version 0.5, November 29, 2016.
[12]E.Heilman, A.Kendler, A.Zohar, S.Goldberg, “Eclipse Attacks on Bitcoin’s Peer-to-Peer Network”, proceedings of the 24th USENIX Security Symposium, Washington, DC., August 2015.
[13]L. Lamport. The Part-Time Parliament, ACM Transactions on Computer Systems, 16, 2 (May 1998), 133-169.
[14]B.M.Oki and B. Liskov. Viewstamped replication: Anew primary copy method to support highly- available distributed systems. In Proc. 7th ACMSymposium on Principles ofDistributed Computing (PODC), pages 8–17, 1988.
[15]D. Ongaro and J. K. Ousterhout. In search of an understandable consensus algorithm. In Proc. USENIX Annual Technical Conference, pages 305–319, 2014.
[16]Laura Shin, The First Government To Secure Land Titles On The Bitcoin Blockchain Expands Project, Feb 7, 2017, Forbes. https://www.forbes.com/sites/laurashin/2017/02/07/the-first-government-to-secure-land-titles-on-the-bitcoin-blockchain-expands-project/
[17]法務省，不動産登記の ABC, http://www.moj.go.jp/MINJI/minji02.html

© 2017 Information Processing Society of Japan