Ethereum ブロック・チェーンに基づいた IoT のための柔軟な分散型属性ベース・アクセス制御フレームワーク

研究代表者　　張　元玉　　奈良先端科学技術大学院大学・助教

１はじめに

インターネット技術の急速な発展に伴い，私達の身近にはインターネットやネットワークに接続可能な Internet of Things (IoT)デバイスが溢れている[1]．PC やスマートフォンはもちろん，ゲーム機やテレビ，エアコンや照明まで，様々な電化製品がインターネットに接続することができる．IoT によって生活が便利になる一方で，これらのデバイスを対象にしたサイバー攻撃が増えている．IoT デバイスを利用したサイバー攻撃の代表に Mirai がある[2]．Mirai は 2016 年に話題になった，IoT デバイスに不正アクセスをして遠隔操作可能にするマルウェアである．これにより多数の IoT デバイスが悪用され，様々な巨大サーバへの DDoS 攻撃が行われた．また，テレビなどに備えられたウェブカメラを遠隔操作して，その個人のプライバシーが侵害されるサイバー攻撃も報告されている[3]．PC やスマートフォンといったデバイスへのセキュリティ対策は一般に普及しているが，IoT デバイスに対するセキュリティ対策の重要性は認知度が低いのが現状である．これにより，IoT デバイスはサイバー攻撃の対象になる傾向が高く，不正アクセスや個人情報の流失などに繋がるリスクが大きい．また，不正アクセスを行う攻撃者は，その時のアクセスログを削除し，自らの痕跡を残さないことが広く知られている．したがって，アクセスが許可されていないユーザからの通信を遮断するだけでなく，万一，不正アクセスが生じたとしてもアクセスログの書き換えができないようなアクセス制御システムを構築することが急務である[4]．

現在，IoT のアクセス制御では中央集権型のアクセス制御方式が広く使用されている[5-7]．中央集権型は，一つの中央サーバを介しシステムにおける全てのアクセス要求を制御する方式である．単独サーバのため，その管理は容易である一方で，多数の IoT デバイスが中央サーバを介してアクセス制御を行うため，サーバに対する負荷は大きくなる．また，中央集権型ではアクセス制御に関するデータ（例えば，エンティティに割り当てられたアクセス権限やアクセス履歴など）が全て中央サーバに保存される．中央サーバが故障した場合や攻撃者により権限や履歴が改ざんされてしまった場合，アクセス制御が機能しなくなる可能性が高くなる．そのため，中央集権型のアクセス制御方式は大規模かつ分散的な IoT システムに適していない．

単一故障や負荷集中の問題を緩和する方策としては，アクセス制御の処理を複数のノードに分散化する分散型アクセス制御方式が挙げられる．ただし分散型方式では，アクセス制御に参加する全てのノードが同じアクセス制御に関するデータを保持し，処理結果に対して合意を形成することでアクセス制御を行うことから，悪意があるノードによるデータおよび合意形成の仕組み自体の改ざんに対して堅牢性の高いアクセス制御方式が求められる．

ビットコイン[8]に代表される暗号通貨の核となるブロック・チェーンは，耐改ざん性が高いため，分散型アクセス制御を実現するために有効な方法のひとつと考えられている．ブロッ ク・チェーンは，複数のブロックの連結によって形成された台帳で，システムに参加している全てのノードで共有される．ブロックは，送金の情報（例:送金元，送金先，金額）を表す取引データ（トランザクション）や直前のブロックのハッシュ値などを含め，マイナーと呼ばれる特別なノードにより生成される．自分が生成したブロックは，他のマイナーによって承認される必要があり，その際，作成したブロックはそのハッシュ値が一定の条件を満たすようにする必要がある．このプロセスはマイニングと呼ばれ，高難度のハッシュ計算を伴う．さらに， あるブロックに含まれるトランザクションを改ざんした場合，それ以降に続く全てのブロックに対してもハッシュの再計算が必要となる．その結果，ブロック・チェーンの高い耐改ざん性が実現される．

ビットコインのようなブロック・チェーンは，静的なトランザクションのみを格納しているため，分散型データベースと見なされている．近年，スマート・コントラクト[9]と呼ばれる実行可能なプログラムをブロック・チェーン上に格納することができる Ethereum ブロック・チェーン[10]が非常に多くの注目を集めている．スマート・コントラクトには，状態を表す変数と状態の参照・変更のための関数（Application Binary Interface: ABI）が含まれている．スマート・コントラクトの ABI にトランザクションを送ることにより，全てのマイナーがこの ABI を実行し，状態を変更できる．したがって，Ethereum のブロック・チ

公益財団法人電気通信普及財団
研究調査助成報告書　第35号　2020年度

ェーンはデータベースだけではなく，耐改ざん性のある分散型計算プラットフォームとしても機能する．

このことから，近年，ブロック・チェーンを用いたアクセス制御が注目を集めている[11-28]．ただし既存研究の多くでは，IoT システムの特性（大規模，動的かつ分散型）に合う柔軟にかつ細かく制御できる動的なアクセス制御はまだ検討されていない．そこで，本研究はブロック・チェーン技術を用い，リソース（オブジェクト）の属性，リソースをアクセスしようとするユーザ（サブジェクト）の属性とアクセスが発生した際のコンテキスト（時間，場所など）を包括的に考慮し，ブロック・チェーン上で機能する動的かつ高信頼な分散型属性ベース・アクセス制御（ABAC）フレームワークの構築を目指し，その有効性とコストをシミュレーション実験を通して検証する．

2 提案 ABAC フレームワーク

スマート・コントラクト・システム

本研究では，Ethereum ブロック・チェーン[7]のスマート・コントラクト機能を適用し，ポリシー・マネージメント・コントラクト（PMC），サブジェクト属性マネージメント・コントラクト(SAMC)，オブジェクト属性マネージメント・コントラクト（OAMC），およびアクセス制御コントラクト(ACC)，の四つのコントラクトで属性ベースのアクセス制御を実現する（図１）．ポリシー，サブジェクト属性，オブジェクト属性に対するマネージメント・コントラクトは，それぞれ，アクセス制御ポリシー（誰がどのような条件下で何にアクセスできるかを記したステートメント），サブジェクト（リソースにアクセスするエンティティ）の属性，およびオブジェクト（アクセスされるリソース）の属性を格納および管理する機能を持つ．また，アクセス制御コントラクト ACC は，サブジェクトからアクセス要求を受信すると，対応するポリシー，サブジェクト属性，およびオブジェクト属性を対応するマネージメント・コントラクトから取得し，属性ベースのアクセス制御を実行する．以下でそれぞれのスマート・コントラクトについて説明する．

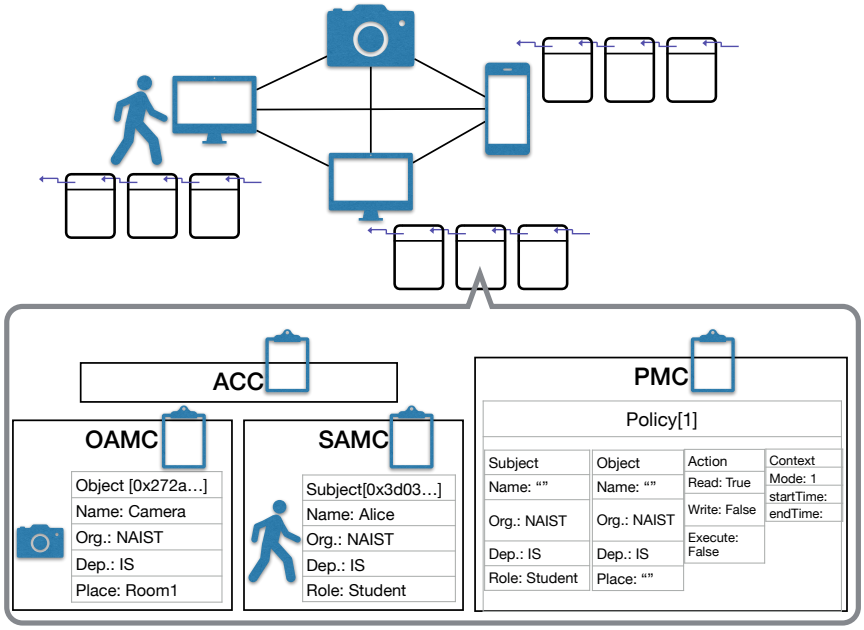

図１ ：提案の全体像

(1) Subject Attribute Management Contract (SAMC)

SAMC はサブジェクトの管理者権限を所持しているユーザのみが実行可能な，サブジェクトの属性を管理するためのスマート・コントラクトである．サブジェクトは一つのユニークな ID（Ethereum アカウントのアドレス）と複数の属性を所持し，属性情報は ID により一意に表される．表 1 に示すように，本研究ではサブジ

ェクトの所属する組織，その組織内で所属する部署，与えられている役割等の属性情報を想定する．これらの属性を管理するために，SAMC は subjectAdd()，subjectDelete()の ABI を提供することによって，属性情報の追加(更新)と削除機能を実現する．また，getSubject()の ABI を提供することにより，サブジェクトの属性を取得する機能を実現する．

表１：属性情報の例

SubjectList[0x3d03...]	ObjectList[0x272a...]
名前：Alice 所属機関：NAIST 部署：IS 課：LSM 役職：student その他：	名前：Camera 所属機関：NAIST 部署：IS 課：LSM 場所：Room 1 その他：

(2) Object Attribute Management Contract (OAMC)

OAMC はオブジェクトの管理者権限を所持しているユーザのみが実行可能な，オブジェクトの属性を管理するためのスマート・コントラクトである．オブジェクトもサブジェクト同様に一つのユニークな ID と複数の属性を所持している．表１に示すように，本研究では各オブジェクトの所属する組織，デバイスの種類や配置場所などの属性情報を想定する．OAMC は objectAdd()，objectDelete()の ABI を提供することによって，属性情報の追加(更新)と削除機能を実現する．また，getObject()の ABI を提供することにより，オブジェクトの属性を取得する機能を実現する．

(3) Policy Management Contract (PMC)

PMC はポリシーの管理者権限を所持しているユーザのみが実行可能な，ポリシーを管理するためのスマート・コントラクトである．ポリシーは，サブジェクトの属性集合 SA，オブジェクトの属性集合 OA，許可される実行操作の集合 A とアクセスが発生した際のコンテキスト集合 C から構成されており，C に記録されているコンテキストで SA を満たすような属性を持っているサブジェクトが OA を満たすような属性を持っているオブジェクトに A に記録されている操作を実行することができることを意味する．例えば，表２に示すポリシーは，2019/5/24 12:00 から 2019/5/31 11:59 までの間に NAIST 機関の IS 部署の LSM 課に所属している学生さんが同じ所属に設置される全てのオブジェクトに read と write の操作を実行することができることを示している．全てのポリシーが policyList という名前のリストに保持されている．ポリシーを管理するために，PMC は policyAdd()，policyDelete()，policyUpdate()の ABI を提供することによって，ポリシーの追加，削除，更新機能を提供する．さらに，findMatchPolicy()，findExactMatchPolicy()の ABI を提供することにより，ポリシーの検索機能を可能にする．

表２：ポリシーの例

Subject Attributes (SA)	Object Attributes (OA)	Action (A)	Context (C)
名前： 所属機関：NAIST 部署：IS 課：LSM 役職：student	名前： 所属機関：NAIST 部署：IS 課：LSM 場所：	Read: True Write: True Execute: False	開始時間： 2019/5/24 12:00 終了時間： 2019/5/31 11:59

(4) Access Control Contract (ACC)

ACC はサブジェクト・オブジェクトの属性情報，アクセスが発生した際のコンテキスト情報（時間，場所など）とポリシーをもとに，accessControl()の ABI を提供することで，動的なアクセス制御を行うスマート・コントラクトである．サブジェクトからオブジェクトへのアクセス・リクエストを受け付け，SAMC と OAMC からそれぞれの属性情報を受け取る．その後，PMC よりポリシーを取得し，サブジェクトのオブジェクトに対するアクセス権を検証する．

主要な機能

本提案の ABAC フレームワークは以下の機能を提供する．

(1) 属性情報の登録，更新，削除

サブジェクトやオブジェクトの管理者は，それぞれ SAMC と OAMC を通して，属性情報の管理を行う．具体的には,subjectAdd()と objectAdd()の ABI にエンティティの ID,表１のような属性情報を引数として与え，この ABI を通してトランザクションを送信する．この時，サブジェクトとオブジェクトのリストに当該の ID がなかった場合はリストへの追加作業を，ID が存在する場合はその ID に対応する属性情報の更新作業をする．属性情報を削除する場合は，subjectDelete()や objectDelete()の ABI にエンティティの ID を引数として与え，リストに当該の ID が存在する場合は，リストから対応する ID の属性情報を削除する．

(2) ポリシーの検索

提案手法のポリシーは構造体の配列で実現され，配列のインデックスは 0 から始まる非負の整数のため，ポリシーを使用する際には，検索が必要になる．ポリシーの検索には，findExactMatchPolicy()の ABI で実現される完全一致と findMatchPolicy()の ABI で実現される部分一致の 2 種類がある．完全一致は，主にポリシーを削除する場合に使用され，入力で与えられた属性情報と内容が完全に一致するポリシーを検索する機能である．ポリシーの中には，表２の Subject Attributes の名前という属性のように，属性情報が空欄で定義されている場合がある．この時，空欄は空欄として検索することが完全一致の検索機能になり，部分一致との差になる．例えば，あるポリシーを削除したい場合，入力で空欄にした属性情報はそのまま空欄の属性情報という形で一致するポリシーを検索する．部分一致は，主に ACC がポリシーを取得する場合，あるいはポリシーの削除と更新の場合に使用される．部分一致の場合，サブジェクトとオブジェクトの属性情報を入力として受け取り，ポリシーに定義されている属性情報が入力の属性情報に含まれているかを検索する．ポリシーの属性情報が空欄で定義されている場合，どのような属性もこのポリシーに当てはまるということが部分一致の検索機能になり，完全一致との差になる．部分一致の場合，検索の終了時には，一致するポリシーに対応するインデックスの配列を返すため,この機能の実行者は返ってきた値を基に次の操作を始める．

(3) ポリシーの登録，更新，削除

ポリシーの管理者は，サブジェクトとオブジェクトの属性情報，許可される実行操作，コンテキスト情報の４種類の属性情報を組み合わせて， ポリシーを定義することが可能である．ポリシー内では，属性情報を細かく定義することが可能であり，これによって，より柔軟性の高いアクセス制御を実現する．新たなポリシーを追加する場合，PMC の policyAdd()を実行する．policyAdd()では始めに新しいポリシーと類似したポリシーが存在するかを部分一致で検索する．これは，ポリシーリストの肥大化とアクセス制御時の競合問題を防ぐためである．新ポリシーを入力に既存ポリシーの検索を実行した時，表２のような既存ポリシーが発見されたとする．この二つのポリシーは新ポリシーの方が既存ポリシーよりも属性情報の適用範囲が狭くなる．そのため，既存ポリシーを削除して新ポリシーを追加，もしくは新ポリシーを追加しないという判断をポリシーの追加者自身で行い，ポリシーの追加を終了する．ポリシーを更新する場合も同様に，類似ポリシーの検索を行う必要がある．ポリシーの更新者は，更新したいポリシーを入力に類似ポリシーの検索を実行し，ポリシーが一つだけ列挙された場合は，そのポリシーのインデックスと新しいポリシーの情報を引数に policyUpdate()を実行する．一方，ポリシーが複数列挙された場合は，発見された類似ポリシーと新たなポリシーが競合しないよう，適切にポリシーの更新，削除を行う．ポリシーの削除を行う場合は，PMC の policyDelete()を実行する．この ABI を実行する前に，削除したいポリシーと完全に一致するポリシーを検索する必要がある．このポリシーを見つけた場合に，ポリシーのインデックスを引数に，policyDelete()を実行する．

(4) アクセス制御

サブジェクトがアクセス・リクエストを送信した後，アクセス許可が返されるまでの流れは以下のようになる(図２)．

- ステップ１：サブジェクトは ACC にオブジェクトの ID と実行したい操作を入力に，アクセス・リクエストを送信する．
- ステップ２：ACC はサブジェクトとオブジェクトの ID をキーに，SAMC，OAMC へ属性情報を取得するためのトランザクションを送信する．
- ステップ３：SAMC と OAMC は該当するエンティティの属性情報を ACC に渡す．
- ステップ４：ACC は受け取った属性情報をキーに，PMC へ該当ポリシーを取得するためのトランザクションを送信する．
- ステップ５：PMC は受信した属性情報が含まれているポリシーの action の情報を ACC に渡す．

- ステップ 6：受け取った情報をもとに，サブジェクトがリクエストする実行操作のアクセス権を判定する．
- ステップ 7：アクセス権の判定結果をサブジェクトとオブジェクトへ返す．

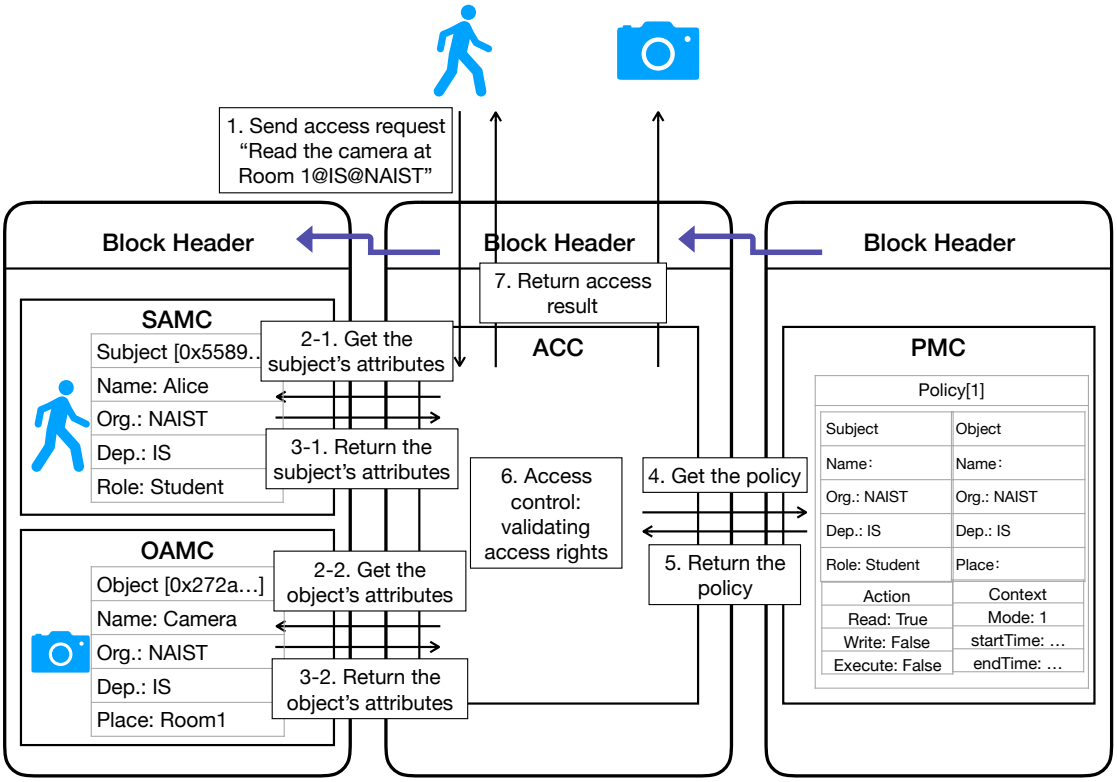

図２：アクセス制御の流れ

このような流れでサブジェクトがアクセス・リクエストを送信すると，分散型アプリケーションとなったアクセス制御が実行され，その処理結果はブロック・チェーン上で保管される．そのため，ブロック・チェーン・ネットワークに参加している全てのノードがアクセス制御サーバの役割を担うことから，ある一つのノードが故障した場合でも，アクセス制御システムは稼働し続けられる．また，処理結果がブロック・チェーン上で保管されるため，実行した内容などが改ざんされる可能性を削減でき，信頼性を向上できる．

3 実証実験

計算機サーバ(Intel Xeon CPU E5-1620 3.60 GHz，32 GB メモリ)上に起動した三台の Ethereum ノードを使用して一つのプライベート・ブロック・チェーン・ネットワークを構築し，実証実験を行った．図３，図４は表２のポリシーが存在する際の，アクセス制御の実行結果である．サブジェクトとオブジェクトはそれぞれ表１の属性情報を保持しており，この２者の属性情報を基に検索されたポリシーが表２になる．対応したポリシーを参照すると，オブジェクト Camera に対して，サブジェクト Alice が実行可能な操作は Read と Write の２操作である．

図３はサブジェクト Alice が Read の操作を行いたいというアクセス・リクエストに対して，ACC がアクセス許可を出した結果である．一方，図４ではサブジェクト Alice が Execute の操作を行いたいというアクセス・リクエストに対して，ACC がアクセスを却下した結果を示している．以上のことから，ブロック・チェーン上に保存されているサブジェクト，オブジェクトの属性情報を取得し，対応するポリシーを検索した後に，実行操作の認可をするという動作を確認できた．

図３：アクセス制御実行結果（Action: Read）

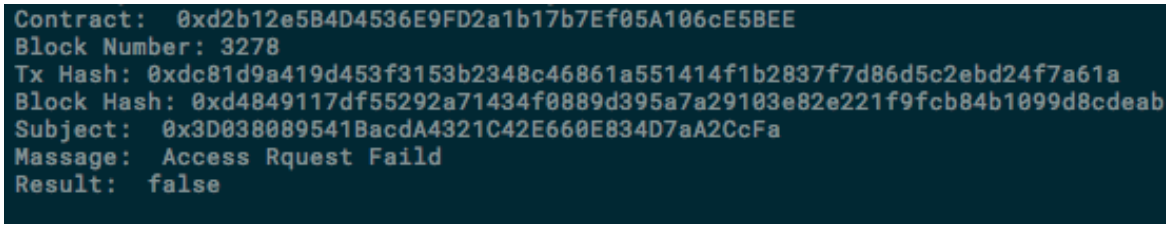

図４：アクセス制御実行結果（Action: Execute）

４コスト評価

スマートコントラクトの展開やABIの実行時に，実行者は手数料を支払う必要がある．Ethereumでは，gasと呼ばれる単位を使用し，トランザクション実行に使用したコストを計測する．一般に，トランザクションの内容が複雑になるほど，消費するgasの量が多くなる．また，ガソリンスタンドの燃料と同様に，Ethereumのgasにも価格があり，この価格は時間によって変化する．Gasの価格が高くなるほど，トランザクションが優先的にマイニングされる．本研究では，導入時と運用時のコストを分けて，提案フレームワークのコスト評価を行った．

導入時のコスト評価

提案手法のABACフレームワークがIoTシステムに導入される時，スマート・コントラクト(ACC，SAMC，OAMC，PMC)のデプロイに必要なコストが導入コストとして存在する．これは管理者によって負担されるコストである．また，比較するために既存手法としてACL（Access Control List)ベースの手法[18]の初期コストも評価した．提案手法と既存手法の初期コストと換算した日本円を表3に示す（2020/01/16 時点でのレート［円/Gas］を基に算出[29]）．提案手法と既存手法では，デプロイするスマート・コントラクトの数が異なるため，このように初期コストに差が出る結果となった．

表３：導入時の管理者の負担コスト

	Gas	日本円
提案手法	2,809,093	406
既存手法	4,943,332	715

運用時のコスト評価

ここでは，提案手法と既存手法の運用コスト，つまりアクセス制御システムの運用中に消費されるgasを評価する．運用コストはIoTシステムのサブジェクト・オブジェクトの数に依存する．運用コストの評価では，システムの運用開始後にサブジェクト・オブジェクトペアのmペアがシステムに追加された時のコストと，新しいシナリオを追加した際に必要なコストを評価する．

(1) サブジェクト・オブジェクトペアが増加する場合

サブジェクト・オブジェクトペアが増加する時，提案手法では，サブジェクトとオブジェクトの属性情報の追加を行う．また，これらのペアに対応するポリシーを追加する必要があるときにのみポリシーの追加が実行される．追加されるポリシーの数をnとすると，システムの運用開始時にサブジェクト・オブジェクトペアをmだけ追加するコストの上限は以下の図3と図4のようになる．

図3はサブジェクト・オブジェクトペアが増加したときに管理者が負担するコストのグラフである．提案手法では，ペアの数が増加した場合のポリシーの追加について，6つのケース($n = m$, $n = m/2$, $n = m/3$, $n = m/4$, $n = m/5$, $n = m/m$)を想定する．ここで，$n = m/p$は1つのポリシーがpペアのアクセス制御を処理することを意味する．mペアが1つのポリシーを共有している，つまりポリシーの追加が必要ないn =

m/mのケースが提案手法におけるベストケースである．対して，既存手法では 1 つのペアにつき 1 つの ACC のデプロイとポリシーの追加を行なう必要がある．ペアの数が 3 未満の場合，提案手法では導入コストとして 4 つのスマート・コントラクトのデプロイが必要になるため，既存手法よりも管理者の負担コストが大きくなる．しかし，ペアの数が 3 以上の場合，既存手法で行われる ACC のデプロイが大きく影響し，提案手法の方がより低コストで運用できる．また，1 つのポリシーを共有できるペアの数が増加すると，運用コストが減少する．これは追加する必要のあるポリシーが少なくなるため，よりベストケース($n = m/m$)に近づいていくためである．

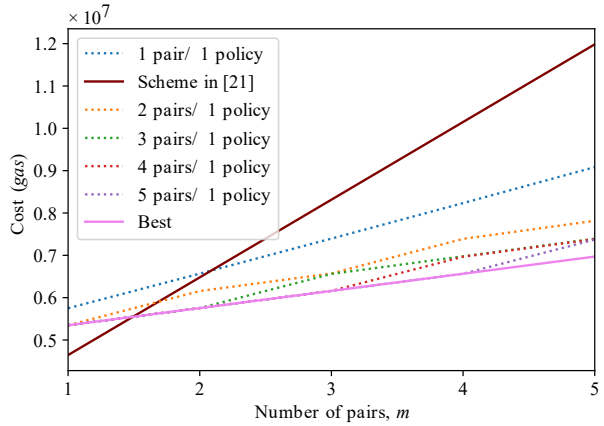

図３：運用時の管理者の負担コストの比較

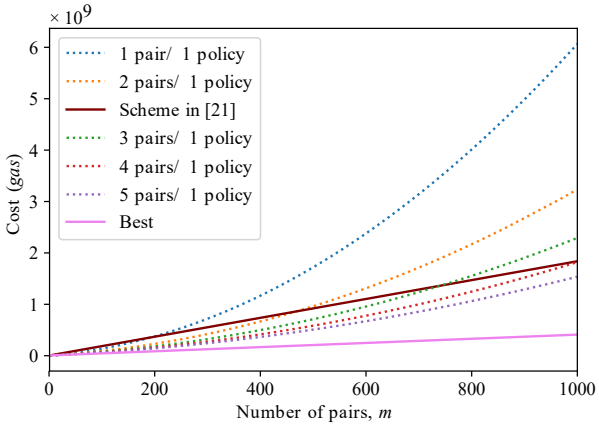

図４： 運用時の管理者の負担コストの比較（さらにペア数を増加させた場合）

図 4 は横軸を図 3 より広範囲に取った時のグラフである．提案手法ではコストが 2 次的に増加するため，ポリシーの集約状況によって，mがある値を越えると既存手法よりも負担コストが大きくなる．毎回ポリシーを追加するn=mの場合は約 214，2 ペアに 1 つポリシーを追加する$n = m/2$の場合は約 489，$n = m/3$の場合は約 761 の時に既存手法よりも負担コストが大きくなる．

図 5 は各ポリシーがより多くのペアで共有されている場合，つまり pの値が大きい場合の管理者の運用コストとペア数 mの関係を示している．pが増加すると，提案手法の管理者の負担コストは$p = m$に近づいていき，既存研究よりも低コストでシステムを運用できる．

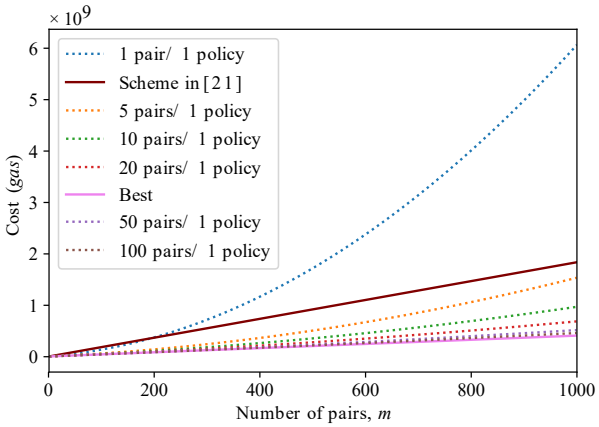

図 5：運用時の管理者の負担コストの比較（よりポリシーの集約が行われた場合）

（2）新しいサブジェクトを追加する場合

スタッフ，学生を含めたメンバーが 10 人の研究室に，新たに IoT 対応のライトが 2 つ設置されたとする．この時，提案手法で行う操作は，2 つのオブジェクトの属性情報の追加である．もし既存ポリシーでサブジェクトが各オブジェクトにアクセスできるのであれば，ポリシーの新たな追加は必要ない（これをベストケースとする）．しかし，詳細なポリシーを定義する必要のある状況が生じれば，サブジェクトとオブジェクトの積の分だけポリシーを追加する必要がある（これをワーストケースとする）．

一方で，既存手法で行う操作は，メンバー全員がライトを使用できるように，サブジェクトとオブジェクトのペアの数，つまり，サブジェクトとオブジェクトの積の分だけ ACC のデプロイとポリシーの追加を行う．それぞれの実行操作にかかった gas の量

表４：運用時の管理者の負担コストの比較（新しいサブジェクトを追加する場合）

	Gas	日本円
既存手法	36, 701, 340	5, 307
提案手法(Best)	310, 136	45
提案手法(Worst)	16, 163, 226	2, 337

と日本円への換算を表４に示す．既存手法では ACC をサブジェクトとオブジェクトの積の分だけデプロイする必要があるため，管理者の負担コストが大きくなる．また，提案手法では，ポリシーの追加に必要なコストが管理者の負担コストに大きく影響するため，ポリシーをより多く追加する必要のある場面では，管理者の負担コストが大きくなる．

５まとめ

本研究では，ブロック・チェーン技術と属性ベースのアクセス制御モデルに基づいた IoT システム用の分散動的アクセス制御方式を提案した．提案の実現可能性を検証するため，プライベート・ブロッ ク・チェーンネットワークを構築し，料金の観点からコストを評価した．運用時の管理者の負担コストでは，より多くのサブジェクト・オブジェクトペアで１つのポリシーを共有できれば，既存手法よりも提案手法の方がより低コストで運用できるという結果になった．今後の課題としては，多くのサブジェクト・オブジェクトが存在する場合，提案手法が適切にトランザクションを処理できるかの評価を行いたい．

【参考文献】

[1] “Gartner Identifies Top 10 Strategic IoT Technologies and Trends,” available at https://www.gartner.com/en/ newsroom/press- releases/2018- 11- 07- gartner- identifies- top- 10- strategic- iot- technologies- and- trends/.

[2] “Mirai botnet linked to dyn DNS DDoS attacks,” avail- able at https://www.flashpoint-intel.com/blog/cybercrime/ mirai- botnet- linked- dyn- dns- ddos- attacks/.

[3] “ネットワークカメラシステムにおける情報セキュリティ対策要件に関する調査,” available at https://www.ipa.go.jp/ security/jisec/choutatsu/nwcs/refer_201705_nwc_report.pdf/.

[4] A.Ouaddah, H.Mousannif, A.A.Elkalam, and A.A.Ouahman,“Access control in the Internet of Things: Big challenges and new opportunities,” Computer Networks, vol. 112, pp. 237–262, 2017.

[5] A. Yavari, A. S. Panah, D. Georgakopoulos, P. P. Jayaraman, and R. V. Schyndel, “Scalable role-based data disclosure control for the internet of things,” in Proc. of 2017 IEEE 37th International Conference on Distributed Computing Systems, 2017, pp. 2226–2233.

[6] Q. Liu, H. Zhang, J. Wan, and X. Chen, “An access control model for resource sharing based on the role-based access control intended for multi-domain manufacturing internet of things,” IEEE Access, vol. 5, no. 2, pp. 7001–7011, 2017.

[7] E. Yuan and J. Tong, “Attributed based access control (ABAC) for web services,” in Proc. of IEEE International Conference on Web Services, 2005, pp. 561–569.

[8] “Bitcoin - open source p2p money,” available at https://bitcoin.org/en/.

[9] “A next-generation smart contract and decentralized application platform,” available at https://cryptorating.eu/whitepapers/Ethereum/Ethereum white paper.pdf.

[10] “An introduction to Ethereum platform,” available at http://ethdocs.org/en/latest/introduction/what-is-ethereum.html.

[11] C. Dukkipati, Y. Zhang, and L. C. Cheng, “Decentralized, blockchain based access control framework for the heterogeneous internet of things,” in Proc. of 3rd Workshop on Attribute Based Access Control, 2018, pp. 61–69.

[12] P. Wang, Y. Yue, W. Sun, and J. Liu, “An attribute-based distributed access control for blockchain-enabled IoT,” in 2019 International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), 2019, pp. 1–6.

[13] D. D. F. Maesa, P. Mori, and L. Ricci, “A blockchain based approach for the definition of auditable access control systems,” Computers & Security, vol. 84, pp. 93 – 119, 2019.

[14] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, “Blockchain for IoT security and privacy: The case study of a smart home,” in Proc. of IEEE PerCom Workshops, 2017, pp. 618–623.

[15] D. F. Maesa, P. Mori, and L. Ricci, “Blockchain based access control,” in Proc. of IFIP International Conference on Distributed Applications and Interoperable Systems, 2017, pp. 206–220.

[16] Y. Zhu, Y. Qin, G. Gan, S. Yang, and W. C.-C. Chu, “TBAC: Transaction-based access control on blockchain for resource sharing with cryptographically decentralized authorization,” in Proc. of 2018 42nd IEEE International Conference on Computer Software & Applications, 2018, pp. 535–544.
[17] A.Ouaddah,H.Mousannif,A.A.Elkalam,andA.A.Ouahman,“Access control in the internet of things: Big challenges and new opportunities,” Computer Networks, vol. 112, pp. 237–262, 2017.
[18] Y. Zhang, S. Kasahara, Y. Shen, X. Jiang, and J. Wan, “Smart contract-based access control for the internet of things,” IEEE Internet of Things Journal, vol. 6, no. 2, pp. 1594–1605, 2019.
[19] T. Sultana, A. Ghaffar, M. Azeem, Z. Abubaker, M. U. Gurmani, and N. Javaid, “Data sharing system integrating access control based on smart contracts for IoT,” in International Conference on P2P, Parallel, Grid, Cloud and Internet Computing. Springer, 2019, pp. 863–874.
[20] R. Xu, Y. Chen, E. Blasch, and G. Chen, “BlendCAC: A smart contract enabled decentralized capability-based access control mechanism for the IoT,” Computers, vol. 7, no. 3, pp. 39–65, 2018.
[21] Y. Nakamura, Y. Zhang, M. Sasabe, and S. Kasahara, “Capability-based access control for the internet of things: An ethereum blockchain-based scheme,” in Proc. of IEEE GLOBECOM 2019, 2019.
[22] H. Albreiki, L. Alqassem, K. Salah, M. H. Rehman, and D. Svetinovic, “Decentralized access control for IoT data using blockchain and trusted oracles,” in In Proceedings of IEEE International Conference on Indus- trial Internet (ICII), Nov. 2019, pp. 248–257.
[23] Q. Lyu, Y. Qi, X. Zhang, H. Liu, Q. Wang, and N. Zheng, “Sbac: A secure blockchain-based access control framework for information- centric networking,” Journal of Network and Computer Applications, vol. 149, p. 102444, 2020.
[24] J.P.Cruz, Y. Kaji, and N.Yanai, “RBAC-SC:Role-based access control using smart contract,” IEEE Access, vol. 6, pp. 12240–12251, Mar. 2018.
[25] G. Hao, E. Meamari, and C.-C. Shen, “Multi-authority attribute-based access control with smart contract,” in Proc. of 2019 International Conference on Blockchain Technology, 2019, pp. 6–11.
[26] S. Ding, J. Cao, C. Li, K. Fan, and H. Li, “A novel attribute-based access control scheme using blockchain for IoT,” IEEE Access, vol. 7, pp. 38 431–38 441, 2019.
[27] G. Yu, X. Zha, X. Wang, W. Ni, K. Yu, P. Yu, J. A. Zhang, R. P. Liu, and Y. J. Guo, “Enabling attribute revocation for fine-grained access control in blockchain-IoT systems,” IEEE Transactions on Engineering Management, 2020.
[28] G. Suciu, C.-I. Istrate, A. Vulpe, M.-A. Sachian, M. Vochin, A. Farao, and C. Xenakis, “Attribute-based access control for secure and resilient smart grids,” in 6th International Symposium for ICS & SCADA Cyber Security Research 2019 6, 2019, pp. 67–73.
[29] “ETH Gas Station,” available at https://ethgasstation.info/index.php.

〈発　表　資　料〉

題　　名	掲載誌・学会名等	発表年月
Using Ethereum Blockchain for Distributed Attribute-Based Access Control in the Internet of Things	IEEE GLOBECOM	2019/12
Capability-Based Access Control for the Internet of Things: An Ethereum Blockchain-Based Scheme	IEEE GLOBECOM	2019/12
IoT のための柔軟な分散型属性ベース・アクセス制御の実現 ～ Ethereum ブロックチェーンベースのフレームワーク ～	電子情報通信学会技術研究報告	2019/3
IoT に向けた Ethereum ブロックチェーンを用いた Capability-Based Access Control	電子情報通信学会技術研究報告	2019/3

の実装 ～ アクションレベルでの Capability の構築		
Ethereum ブロックチェーンを用いた IoT のための属性ベース・アクセス制御の実現	インターネット技術第 163 委員会（ITRC）新世代ネットワーク構築のための基盤技術研究分科会（NWGN）ワークショップ	2019/9
IoT に向けた Ethereum ブロック・チェーンを用いた Capability-Based Access Control	インターネット技術第 163 委員会（ITRC）新世代ネットワーク構築のための基盤技術研究分科会（NWGN）ワークショップ	2019/9
Attribute-Based Access Control for Smart Cities: A Smart Contract-Driven Framework	IEEE Internet of Things Journal	2020/5 投稿，査読中
Exploiting Smart Contracts for Capability-Based Access Control in the Internet of Things	Sensors	2020/1