「マルチメディア，分散，協調とモバイル
(DICOMO2021) シンポジウム」 令和3年6月-7月

大規模ネットワークへのゼロトラスト適用を想定したセキュリティ検証機能の動的割当方式に関する一検討

風戸 雄太 [1]　仲川 宜秀 [1]

概要： サイバー攻撃の脅威・攻撃手法の変化やリモートワーク等の業務の多様化により，従来の境界型セキュリティの限界と，新たなセキュリティモデルであるゼロトラストネットワークが提唱されている．ゼロトラストネットワークでは，『何も信用しないこと』を基本原則とし，NW 上での継続的なセキュリティ検証と動的なアクセス制御を実施するセキュリティモデルである．しかし，多数の IoT 機器や NW を構成する機器などのエンティティが存在する大規模ネットワークにゼロトラストセキュリティを適用するためには，セキュリティ検証機能をソフトウェア動作するために必要となるコンピューティングリソース，通信リソースが不足することが課題となる．本稿では，エンティティに対するセキュリティ検証機能の割当，検証スケジューリング頻度を動的に制御することで，セキュリティ検証に必要となるリソースの削減，セキュリティ検証機能の最適割当を実現する新方式を提案する．提案手法に関して，既存手法と比較した定性評価，大規模 NW 適用を想定した基礎シミュレーションのフレームワークを検討の上，提案手法のリソース消費量削減への有効性・実現可能性を確認した．

A Study on Dynamic Allocation Method of Security Functions Toward Zero Trust-based Large Scale Network

YUTA KAZATO [1]　YOSHIHIDE NAKAGAWA [1]

1. はじめに

1.1 背景：ゼロトラストセキュリティ・ネットワーク

近年，サイバー攻撃に関連する脅威・攻撃手法の変化，クラウドの利活用促進，リモートワーク普及等による業務の多様化により，従来の境界型セキュリティモデルの限界が指摘されている．境界型セキュリティモデルでは，ネットワークの境界において，信頼できる内部ネットワーク（例：社内ネットワーク）と信頼できない外部ネットワーク（例：インターネット）に分割し，信頼できるネットワーク内の情報資産を保護することが目的である．しかし，巧妙な標的型攻撃・マルウェアによる内部ネットワーク侵害，ソフトウェアアップデートに対するサプライチェーン攻撃等により，攻撃者が内部ネットワークに侵入し，情報資産への不正アクセスや情報漏洩の被害が発生している．さらに，前述のクラウドサービス利活用，リモートワーク普及により，ネットワーク境界はより曖昧に変化しており，その結果，従来の境界型セキュリティモデルでは十分なセキュリティレベルを担保することが困難であるといえる．

そこで，ネットワークにおける新たなセキュリティモデルとしてゼロトラストネットワークが提唱されている．本セキュリティモデルの基本原則は『何も信用しないこと』であり，NW に接続されたエンティティ（例： ユーザ，機器等）に対して，継続的なセキュリティ検証を実施の上，検証結果に基づき，最小限のアクセス権限付与，情報資産への動的なアクセス制御を行うセキュリティモデルである．

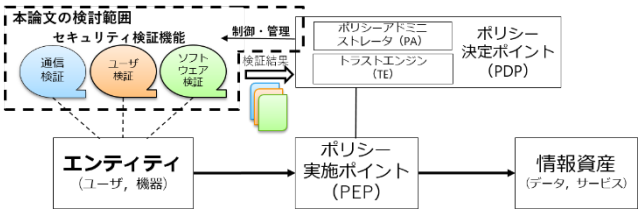

図 1. NIST ゼロトラストアーキテクチャ[1]と本論文における検討範囲

図 1 に NIST (National Institute of Standards and Technology) SP800-207 [1] におけるゼロトラストアーキテクチャの概要と本論文の検討範囲を示す．ゼロトラストアーキテクチャの論理コンポーネントは，ポリシー決定ポイント (PDP) とポリシー実施ポイント（PEP）がある．ポリシー決定ポイントでは，セキュリティ検証機能を管理・制御するポリシーアドミニストレータ（PA），検証結果を基にトラストスコアを計算するトラストエンジン（TE）／ポリシーエンジン（PE）の機能を提供することが特徴である．また，ポリシー実施ポイントではエンティティの情報資産に対するアクセス制御を実施する機能を提供する．

ゼロトラストネットワークにおけるエンティティを対象としたアクセス制御の動作フローは次の通りである．初めに，アクセス制御対象であるエンティティから取得可能な情報・ログより継続的にセキュリティ検証を実施する．次に，対象エンティティのセキュリティ検証結果を入力情報として，ポリシー決定ポイント内のトラストエンジンに

1 日本電信電話株式会社　NTT ネットワークサービスシステム研究所

© 2021 Information Processing Society of Japan

おいて『トラストスコア』を計算する．トラストスコアは，対象のエンティティに対する信用・信頼（例：高スコアは正常な挙動・動作を示す，低スコアは異常な挙動・動作を示す等）を定量化するスコアであり，トラストエンジンにおいて任意のトラストアルゴリズムにより定義，計算を行う．その後，ポリシー施行ポイントでは，トラストスコア値，およびセキュリティ検証結果を入力情報として，スコア値に基づくエンティティの動的なアクセス制御を実施する．この時，スコアの値がアクセスを認可できる閾値以上であれば，最低限のアクセス権がエンティティに付与され，情報資産へのアクセスが可能となり，スコア値が閾値以下の場合には，アクセス権を付与・許可されない．

1.2 セキュリティ検証機能

本論文では，ゼロトラストネットワークにおける重要な技術コンポーネントの 1 つであるセキュリティ検証機能について，その管理・制御方法に注目し，図 1 に検討範囲を示した．セキュリティ検証機能は，エンティティの動作・挙動に基づき，エンティティの完全性，真正性，信頼性等を検証すること，悪意のあるエンティティを検出することを目的とし，ソフトウェアの形態で構成され，自動検証可能であることを特徴とする．

ゼロトラストネットワークにおけるセキュリティ検証を実施する対象のエンティティは主に（1）ユーザ，(2）機器である．(1) ユーザに対するセキュリティ検証は，知識認証（パスワード認証，ロケーション認証等），所有物認証，生体認証等がある．ユーザ認証では，これらの認証を複数組み合わせた多要素認証が広く利用され，各認証のセキュリティ検証結果からユーザのトラストスコアを算出し，スコア値に基づきユーザの情報資産へのアクセス要求に対するアクセス可否を決定，制御する．次に，(2) 機器に対するセキュリティ検証では，文献[2]において多くのセキュリティ検証方法・機能が列挙されている．例えば，静的な検証手法であるバイナリ解析・ファームウェア解析等による完全性検証，動的な検証手法であるネットワークスキャン等による通信検証，脆弱性スキャン等によるソフトウェア挙動検証がある．静的／動的な検証手法を組み合わせることで，機器に対するセキュリティ検証を実現する．

セキュリティ検証機能の分類には，対象エンティティの内部で動作するエージェント型，対象エンティティを外部から検証を行うエージェントレス型が存在し，ソフトウェアによる動作では，コンピューティングリソース（CPU, メモリ, ストレージ等), 通信リソース (データ量, 帯域等）を必要とする．しかし，エージェント型のセキュリティ検証機能は，リソース制約のあるセンサ等の低性能 IoT 機器上では動作させることが難しい．一方，エージェントレス型ではセキュリティ検証機能を動作するクラウドやエッジコンピューティング等の計算基盤を構築することで，低性能な IoT 機器に対しても検証を実施することが可能である．

表 1. セキュリティ検証方式と消費リソース分類

検証方式		コンピューティングリソース	通信リソース
エージェント型		エンティティのリソースを消費	【低】検証結果の送付
エージェントレス型	パッシブ型	計算基盤リソースを消費	【低】検証結果の送付（エッジ分析・検証） 【大】キャプチャデータ・ログ転送（クラウド分析・検証）
	アクティブ型	計算基盤リソースを消費	【中】検証用パケット・データの転送

さらにエージェントレス型の分類には，対象エンティティから得られる情報を基に受動的に検証を実施するパッシブ型，対象エンティティに対して能動的にスキャン等を実施するアクティブ型の検証方法がある．

各セキュリティ検証方式での消費リソースの分類・量を表 1 に示す． エージェント型, エージェントレス型の両方式とも検証機能を動作するためには，コンピューティングリソースと通信リソースを必要とする．特に，エージェントレス型では，検証機能を対象エンティティに近いエッジ側に配置するか，クラウド側に配置するかで通信リソースの消費量が変化するため, 最適配置を検討する必要がある．

以降では，ゼロトラストセキュリティモデル，大規模 NW におけるセキュリティ検証の効率化に関する関連研究を示し，セキュリティ検証機能の消費リソースの効率化・最適化を考慮したセキュリティ検証機能の制御方式を検討する．

2. 関連研究

2.1 ゼロトラストセキュリティモデル

ゼロトラストセキュリティモデルは, 2010 年に Forrester Research, Inc の John Kinderva により初めて提唱された[3]. それ以降, ゼロトラストセキュリティ実現に向けて産業界, 学術分野での研究開発や, NIST による関連レポートの公開が行われている[2], [4]. Google, Inc はエンタープライズ向けゼロトラストネットワークのコンセプト・セキュリティソリューションとして， BeyondCorp [5]を提案した．BeyondCorp では，アプリケーションを利用するユーザの環境（デバイス，OS，ソフトウェア，ロケーション等）や利用中の振る舞いを常時監視し，脅威の検出，動的なアクセス制御を行う．他にもゼロトラストセキュリティを実現するセキュリティソリューションが複数登場している[6] ．

学術分野におけるゼロトラストネットワークの先行研究では，トラストスコアの値に基づく動的アクセスポリシー制御方法が提案・検証されている[7]. Thomas ら[8]は，研究ネットワークを対象にゼロトラストのプロトタイプシステムを構築しており，デバイス種別，通信種別，二要素認

© 2021 Information Processing Society of Japan

証等の検証結果に基づきトラストスコアを算出，ユーザグループごとに閾値でのアクセス制御を実施している．

しかし，これらの先行研究のトラストスコアの算出方法は減算方式等の静的な計算中心であり，動的にセキュリティ検証機能を割当変更する仕組みはない．また，前述のエンタープライズ向けソリューションも含め，その対象はユーザから情報資産・サービスへのアクセス制御であり，文献[2]で提言された機器を対象としたセキュリティ検証，検証結果に基づくゼロトラスト適用は検討されていない．

2.2 大規模ネットワークでのセキュリティ検証

大規模ネットワークは，企業が利用するエンタープライズレベルのネットワークと比較して，ネットワークに接続するエンティティ（ユーザ，機器，デバイス），エンティティに関連する各種データ・情報資産，さらに，NW を構成するエンティティ数のオーダが数百万～億単位であるキャリア・テレコムネットワーク等が該当し，セキュリティレベルの担保に加え，高スケーラビリティ，効率的なセキュリティ運用を要求されることが特徴である．さらに近年，Internet of Things (IoT) やローカル 5G を活用した IoT 機器・センサのネットワーク導入も進んでおり，これらも機器数観点では大規模ネットワークに分類することができる．

大規模ネットワークでのセキュリティ機能を効率的に適用した先行研究では，文献[9]において，ネットワークフロー情報を活用した効率的な DDoS 攻撃検知手法を提案しており，大規模ネットワークでの全通信を対象としたトラフィック分析による攻撃検知が困難であることから，ネットワークフロー情報を攻撃検知に活用している．大規模ビッグデータの管理においてゼロトラストセキュリティを提案した[10]では，ビッグデータ管理におけるユーザ認証に着目したゼロトラスト適用を行っている． また，IoT 機器を対象としたゼロトラストの検討では，Zhang[11]らが電力 IoT 機器セキュリティに対するゼロトラストフレームワーク適用を提案している．さらに，文献[12]では指数関数的に増加する IoT 機器環境でのトラスト管理方法として，階層型 Block Chain の適用を提案しているが，大規模ネットワーク環境ではリソース不足によりゼロトラスト管理モデルの導入が難しいことを示唆している．したがって， 多数の IoT 機器が接続するネットワークやキャリア・テレコムネットワークを構成するエンティティを対象とした大規模なゼロトラスト基盤を構築する場合，セキュリティ検証機能のソフトウェア動作に必要なコンピューティングリソース，通信リソース等）の不足が大きな課題であるといえる．

2.3 先行研究の課題

ゼロトラストセキュリティモデルはエンタープライズでのユーザを対象にしたソリューションの導入，学術レベルでのプロトタイプ検証が進んでいる一方，大規模 NW 内のエンティティ・機器を対象としたゼロトラスト適用に関する事例や研究レベルでの取り組みは十分であるとはいえない．多くの研究では，データやアルゴリズムの工夫・改良に注力しており，大規模 NW の全体アーキテクチャを考慮した制御方式が検討されていない．特に，大規模 NW に接続した機器，NW 自体を構成する機器へのゼロトラスト適用の場合に大幅なリソースの不足が予想されるため，エンティティに割り当てるセキュリティ検証機能のコンピューティングリソース・通信リソースを効率的に管理・最適化する制御方式の適用が必要である．

3. 提案手法

3.1 セキュリティ検証機能の動的割当制御方式

先行研究，エンタープライズ向けゼロトラストネットワークにおけるセキュリティ検証機能の制御方法は，主にユーザの動作を対象とした静的なスコア計算もしくは脅威の可能性がある動作に対する追加検証を行う方法が採用されていた．一方，提案手法では，ネットワークを構成する機器や装置等のエンティティも対象とし，多数のエンティティに対するセキュリティ検証機能の効率的な管理を実現するために，検証結果・トラストスコアの値に基づき，動的なセキュリティ検証機能の割当制御（割当数，検証スケジューリング変更）を実施する．提案手法のアーキテクチャを図 2 に示す．提案手法では，ポリシー決定ポイントにおいてセキュリティ検証機能制御のために，情報収集機能，検証機能割当エンジン機能，制御コントローラ機能を有している．提案手法の動作フローは次の通りである．対象エンティティに割り当てられたセキュリティ検証機能の検証結果に基づき，トラストエンジンではトラストスコアをリアルタイムに計算する．ある任意のエンティティに対して N（$N > 0$）個のセキュリティ検証機能が割り当てられている場合，セキュリティ検証結果の集合を$R = \{r_1, r_2, \ldots, r_n\}$，トラストスコアをSと表すことができる．セキュリティ検証結果r_nは，セキュリティ検証機能により保持内容は異なるが，例えば 2 値による悪性検知結果を格納している．また，トラストスコアSには上限値S_{max}，下限値S_{min}が存在する．次に，事前にエンティティの挙動・動作，重要度等のパラメータから機械学習・深層学習を用いて設定した閾値（Th_{TRUST}, $Th_{UNTRUST}$）と，トラストスコアの値の変化に応じて，ポリシーアドミニストレータにおいてセキュリティ検証機能の割当検討・制御指示を実行する．

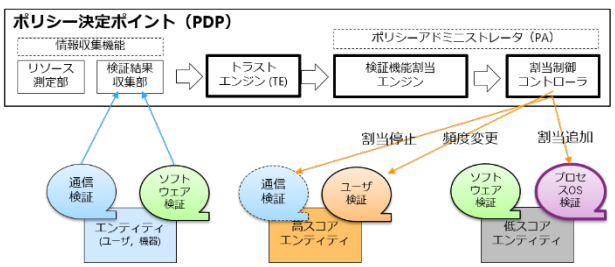

図 2. セキュリティ検証機能の割当制御アーキテクチャ

© 2021 Information Processing Society of Japan

（1）一定期間高トラストスコアであるエンティティに対する割当制御動作（任意の期間tにおいて$S \geqq Th_{TRUST}$の時）

Th_{TRUST}は，対象エンティティを信用・信頼する閾値としてエンティティの種別・グループごとに事前調整を行う．任意の期間tでトラストスコアSがTh_{TRUST}を上回る場合，検証機能割当エンジンにおいて，(A) セキュリティ検証機能の割当数削減，(B) 検証スケジューリングの間隔変更を実施可能か検討する．(A) セキュリティ検証機能の割当数削減では，検証ソフトウェアの割当を停止することで，ソフトウェア起動中に発生する定常的なコンピューティングリソース，検証動作時に発生する通信リソースを削減する．(B) 検証スケジューリングの間隔変更では，検証動作時に発生するコンピューティングリソース，通信リソースを削減することが可能である．ただし，割当削減，検証頻度変更を実行するセキュリティ検証機能の選定は，リソース量だけでなく，セキュリティレベルの低下，リスクの拡大等が発生しないように実施する必要がある．

（2）セキュリティ検証結果より低トラストスコアを示すエンティティに対する割当制御動作（$S < Th_{UNTRUST}$の時）

$Th_{UNTRUST}$は，対象エンティティを信用・信頼しない閾値として事前調整を行い，トラストスコア S が$Th_{UNTRUST}$を下回る場合，検証機能割当エンジンおよびトラストエンジンにおいて，（C）セキュリティ検証機能の追加割当，（D）追加検証結果に基づくトラストスコアの再計算を実施する．(C) セキュリティ検証機能の追加割当では，検証機能割当エンジンにおいて，計算基盤全体のリソース余剰情報を基に，セキュリティ検証機能の割当が可能であるかを判定の上，対象エンティティに対して検証機能の追加割当を決定，検証ソフトウェアの起動・割当制御を実施する．その後，追加したセキュリティ検証機能の検証結果を基に，（D）追加検証結果に基づくトラストスコアの再計算を実施する．

上記のように，トラストスコアのスコア値によりエンティティに対するセキュリティ検証機能の割当数・検証スケジューリングを動的に変更することで，セキュリティ検証機能の割当制御の最適化を行い，セキュリティ検証機能に必要なコンピューティング・通信リソースの削減を行う．

3.2 提案手法の定性評価

提案手法の技術優位性について，提案手法と既存手法を 4 つの定性評価観点で比較する．比較する既存手法は，ゼロトラスト NW での静的なセキュリティ検証を実行する[7]，[8]，大規模 IoT 環境でのトラスト管理を実施する[11]を対象とする．評価観点は，セキュリティ検証機能のリソース消費量，スコア精度改善に対する寄与，対象エンティティの種類，アーキテクチャ全体の制御性である．机上での定性評価による比較結果を表 2 に示す．提案手法は，既存手法と比較して，割当制御による消費リソースの抑制・最適化が可能であり，さらに追加割当の仕組みを活用することでスコア精度向上への寄与もあるといえる．また，

表 2. 提案手法と既存手法の比較評価結果

	提案手法	静的検証[7], [8]	大規模全検証[11]
リソース量	○抑制・最適化可能	△検証数に比例（削減不可）	×リソース不足
スコア精度	○精度向上可(加減算方式)	△追加検証なし，減算のみ	－ 検証機能・検証数に依存
対象	○ユーザ・機器	△ユーザのみ	△機器のみ
制御性	△全体の制御が必要	○動的な制御不要	△Block Chain による分散管理

ゼロトラスト NW の仕組みをユーザの動作だけでなく，ネットワークを構成するエンティティ（機器）まで拡張する点も差分である．一方，提案手法ではセキュリティ検証機能の動的割当制御のために，余剰リソース・検証結果の情報収集，アーキテクチャ全体で割当制御を細かく実施するため，制御性の観点に関して，実現性確認が必要である．

4. 基礎シミュレーション実験

4.1 基礎シミュレーションの概要

提案手法の有効性・実現可能性，有効な制御条件について，大規模ネットワークを想定したシミュレーションのフレームワークを検討の上，基礎シミュレーション実験，評価を行う．今回の基礎シミュレーションで検証するアルゴリズムは，検証機能割当エンジンにおいて，エンティティに割り当てるセキュリティ検証機能数を決定するセキュリティ検証機能配置アルゴリズム，トラストスコアから悪性エンティティへの脅威対処を実施するエンティティ対処アルゴリズムである．基礎シミュレーションの概要・フレームワークを図 3 に示す． 基礎シミュレーションでは，対象となる大規模ネットワーク部（エンティティ，セキュリティ検証機能）と提案システム部（検証機能割当エンジン：セキュリティ検証機能配置アルゴリズム，エンティティ対処アルゴリズム）から構成される．大規模ネットワーク部，提案システム部で利用する基礎シミュレーションのパラメータを表 3 に示す．また，シミュレーションの簡略化のために，4 つの仮定・前提のもと，実験を実施する．

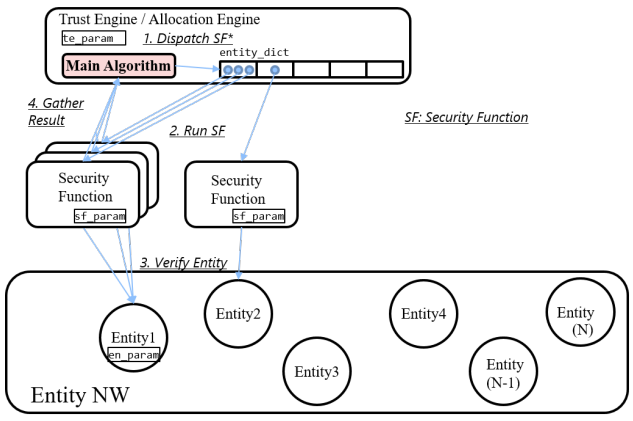

図 3. 基礎シミュレーションの概要図/フレームワーク

© 2021 Information Processing Society of Japan

表 3. 基礎シミュレーションで利用する主なパラメータ

パラメータ	概要
エンティティ数	NW 上にあるエンティティの数
悪性化確率	エンティティが悪性化する確率
検知率	セキュリティ検証機能が悪性のエンティティを悪性と正しく判別する確率
誤検知率	セキュリティ検証機能が正常なエンティティを悪性と誤判別する確率
脅威対処閾値	脅威を対処するトラストスコアの閾値
脅威対処率	発見された脅威が完全に対処される確率

[基礎シミュレーションの仮定・前提]

・**前提 1**：セキュリティ検証機能は同一性能であり，検証種別，コンピューティングリソース消費，検証間隔も同一とする．

・**前提 2**：あるエンティティに割り当てたセキュリティ検証機能はそれぞれ独立の検証を実施し，2 値の検証結果を出力する．

・**前提 3**：セキュリティ検証機能の割当制御のうち，割当数の制御のみ実施し，通信リソース・検証頻度変更は対象外とする．

・**前提 4**：エンティティの種別，アタックサーフェスは考慮せず，どのエンティティも同一の脆弱性を有する．

表 3 のパラメータ，シミュレーションの仮定・前提に基づく，基礎シミュレーションのシナリオは次の通りである．“ネットワーク内には一定割合の悪性エンティティが存在し，情報資産への不正アクセスを試みる．悪性エンティティの割合は悪性化確率で示し，シミュレーション実行毎に変化する．セキュリティ検証機能は対象エンティティの検証を実施し，2 値（良性・悪性）で検証結果を出力し，トラストエンジンでは検証結果よりトラストスコアを計算する．今回は，検証結果の平均値からトラストスコアを算出する．その後，検証機能割当制御エンジンでは，下記のセキュリティ検証機能配置アルゴリズム，エンティティ対処アルゴリズムに基づき，セキュリティ検証機能の割当数削減，割当数追加，およびエンティティへの対処を実行する．”

[セキュリティ検証機能配置・エンティティ対処 アルゴリズム]

1. 初回のセキュリティ検証では，各エンティティに対して同数のセキュリティ検証機能を割り当て，検証を実行する．
2. トラストスコアが閾値$Th_{UNTRUST}$より低いエンティティに対して，検証機能の追加割当を実施する．
3. トラストスコアが閾値$Th_{UNTRUST}$より低いエンティティに対して，悪性検知を行い，脅威対処率に基づき，対象エンティティから悪性除去の対処を実施する．
4. 任意の期間tでトラストスコアが閾値Th_{TRUST}より高いエンティティに対して，検証機能の割当数削減を実施する．
5. 2〜4 の動作をシミュレーション実行期間内で繰り返す．

評価実験では，基礎シミュレーションのシナリオを単位時間T実行の上，下記 2 つの評価指標を用いて，リソース消費量削減観点，セキュリティレベルの維持観点での有効性を確認する．

・コンピューティングリソース消費量削減効率 R_r（数式 1）

ある単位時間$\mathrm{T} = [t_1, t_2, \ldots, t_n]$における本アルゴリズム未適用時（ベース手法：検証機能割当数を固定）のコンピューティングリソース消費量$C_u = \{c_{u1}, c_{u2}, \ldots, c_{un}\}$と，提案手法アルゴリズム適用時のコンピューティングリソース消費量$C_a = \{c_{a1}, c_{a2}, \ldots, c_{an}\}$の比率を示す．$R_r$の値が小さいほど，コンピューティングリソース消費量が削減されている．

$$R_r = \frac{\sum_{i=1}^{n} C_{ai}}{\sum_{i=1}^{n} C_{ui}} \quad (1)$$

・脅威見逃し率 R_m（数式 2）

ある単位時間$\mathrm{T} = [t_1, t_2, \ldots, t_n]$における本アルゴリズム未適用時（ベース手法）の悪性エンティティ残存数$E_u = \{e_{u1}, e_{u2}, \ldots, e_{un}\}$と，提案手法アルゴリズム適用時の悪性エンティティ残存数$E_a = \{e_{u1}, e_{u2}, \ldots, e_{un}\}$の比率を示し，基本アルゴリズムにおいて悪性エンティティの見逃し数と悪性の未除去数の合計数と対応する．R_mの値が小さいほど，提案手法によるセキュリティレベルの低下が少ない．

$$R_m = \frac{\sum_{i=1}^{n} E_{ai}}{\sum_{i=1}^{n} E_{ui}} \quad (2)$$

4.2 基礎シミュレーション結果

基礎シミュレーションにおけるパラメータレンジ，閾値の設定値を表 4 に示す．シミュレーションを実行する単位時間Tはセキュリティ検証回数 100 回分，検証機能の割当数削減を検討する任意期間tは検証回数 5 回分，評価指標は基礎シミュレーション 10 回分の実行結果の平均値で評価を実施した．基礎シミュレーションの評価結果の内，代表的なシミュレーション3パターンの評価結果を表5に示す．尚，各パターンにおける共通パラメータは表 4 の太字の値に設定し，各パターンの悪性化確率，検知確率，誤検知率はシミュレーション 3 パターンの設定詳細に記載する．

表 4. シミュレーションのパラメータと閾値の設定値

パラメータ	パラメータ設定レンジ
エンティティ数	1M, 10M, **100M**, 1000M
悪性化確率	1%, 5%, 10%, 30%
検知率	60%, 80%, 99%
誤検知率	0.1%, 1%, 5%, 10%
脅威対処閾値	0.1, 0.5, **0.9**
脅威対処率	20%, 40%, 60%, **80%**
検証機能初期割当数	3, **5**, 10, 20
閾値設定	$Th_{UNTRUST}$: 0.5, Th_{TRUST}: 1.0

© 2021 Information Processing Society of Japan

表 5. 基礎シミュレーションによる評価結果

	PT1	PT2	PT3
リソース消費量 削減効率 R_r	0.3202	0.4013	3.7399
脅威見逃し率 R_m	1.0105	1.3104	0.9727

（PT1：悪性少/検知精度高，PT2：悪性少/検知精度低，PT3：悪性多/検知精度中）

[シミュレーション 3 パターンの設定詳細]

- **パターン 1**：悪性エンティティ【少】，検証精度【高】
  NW 内の悪性エンティティ数が少ない，
  検証機能の検知精度が高い状況での有効性確認
  設定値：悪性化確率 1%，検知率 99%，誤検知率 1%
- **パターン 2**：悪性エンティティ【少】，検証精度【低】
  検証機能の検知精度が低い場合での有効性確認
  設定値：悪性化確率 1%，検知率 60%，誤検知率 10%
- **パターン 3**：悪性エンティティ【多】，検証精度【中】
  NW 内の悪性エンティティ数が多い場合での有効性確認
  設定値：悪性化確率 30%，検知率 80%，誤検知率 5%

表 5 の評価結果より，悪性エンティティ数の少ない環境（PT1）では，本提案手法によりセキュリティレベルを低下せずにリソース消費量を約 68%削減することができた．しかし，検証機能の検知精度が低い環境（PT2）では，リソース消費量を削減できるがセキュリティレベルも低下する，すなわち，セキュリティ機能割当数を削減した結果，悪性エンティティの脅威を見逃す可能性が高まるといえる．さらに，悪性エンティティ数が多い環境（PT3）では，トラストスコアの低いエンティティが多くなった結果，セキュリティ検証機能の割当数も多くなり，セキュリティレベルは改善した一方，リソース消費量はアルゴリズム未適用時と比較して約 3.74 倍に増加しており，セキュリティ検証機能配置アルゴリズムの更なる改良が必要であると考えられる．

5. 考察

5.1 基礎シミュレーション結果・提案手法の効果

基礎シミュレーションの評価結果より，NW 上の悪性エンティティ数，すなわち脅威や攻撃者・マルウェアが少ない環境では，提案手法のセキュリティ検証機能の割当数削減によるリソース削減効果が大きく，有効性が高いといえる．しかし，セキュリティ検証機能の検証精度が低い場合にはセキュリティレベル低下に影響を及ぼすため，セキュリティ検証機能自体の精度向上の必要性がある．また，セキュリティレベルを維持しつつ，どのセキュリティ検証機能を削減するか決定を行うアルゴリズムを検討する必要がある．さらに，悪性エンティティ数が多い環境においては，セキュリティ検証機能の割当数が増加することによりリソース消費量は増加してしまったが，一方でトラストスコアの低いエンティティに対して検証を追加で実施することで，脅威見逃し率が低下し，セキュリティレベルの向上に寄与する可能性があることが分かった．今後は，セキュリティ検証機能の最適配置を行うアルゴリズムを確立していくことで，NW 全体でのスコア・検知精度向上とリソース消費量の削減の両立を目指す．

今回実施した基礎シミュレーションの前提条件では，検証スケジューリング変更による通信リソース削減，セキュリティ検証機能，エンティティ，大規模ネットワーク構造等に関するより詳細な条件を組み込んだ実験は実施しておらず，基本コンセプトの実現可能性のみを確認した．次のステップとして，実環境に即したマルチエージェントシミュレーション実験による技術評価・実現性の確認を進める予定である．

5.2 提案手法のシステム基本機能要件

提案手法をシステムとして実現するための基本機能要件を検討する．ソフトウェアにより構成されたセキュリティ検証機能（エージェント型，エージェントレス型）に対して，ポリシーアドミニストレータでは下記の動作制御を想定する．

- **制御機能 1**：セキュリティ検証機能の起動・停止
- **制御機能 2**：セキュリティ検証機能の新規割当・割当変更
- **制御機能 3**：セキュリティ検証機能のスケジューリング変更

制御機能 1 では，対象エンティティにセキュリティ検証を動的に割り当てること，コンピューティングリソースを削減するために，ソフトウェアを実行する仮想マシン，コンテナの起動や停止を制御可能であること，さらにリソース量を測定し，実行可否の判定を行うことが機能要件である．次に，制御機能 2 では，エンティティと検証機能の NW 接続構成を自動設定することが機能要件である．制御機能 3 の検証スケジューリング変更に関しては，セキュリティ検証機能への個別設定が必要なため，外部装置との連携 IF（例: RestAPI 対応）を有することを機能要件とする．

以上の検討結果を提案手法におけるシステムの基本機能要件とし，今後の取り組みとして，基本機能要件を有する制御コントローラの選定，システム PoC の検討・評価，大規模ネットワーク環境で求められる性能要件（デプロイ時間，検証動作時間，スループット等）を明確化する．

5.3 ゼロトラスト基本原則との関係性，懸念事項

提案手法は，大規模ネットワークへのゼロトラストセキュリティモデル適用のために，過去のトラストスコアの変動を基に高スコアを示すエンティティに対するセキュリティ検証機能の割当数削減や検証頻度削減（検証スケジュール変更）を行うことにより，コンピューティング・通信リソースの削減，効率化を行う手法である．しかし，過去の検証結果やトラストスコアの情報を利用する点，検証頻度の削減を行う点は，ゼロトラストネットワークの基本原則[1]のうち，『過去のセキュリティ検証結果を信用しないこ

© 2021 Information Processing Society of Japan

と』,『アクセス制御毎にセキュリティ検証を実施すること』と異なった方針であるため，セキュリティレベルの維持と効率的なセキュリティ運用とのトレードオフ関係を技術的に解消する必要がある．本課題を解決するための検証最適化アルゴリズム検討を進め，セキュリティレベル維持と消費リソース削減を両立する制御方式の確立を目指したい．

6. まとめと今後の課題

本稿では，ネットワークにおける新しいセキュリティモデルであるゼロトラストネットワークを大規模かつ多数のエンティティが存在する環境へ適用するために，トラストスコアの値に基づきセキュリティ検証機能の割当を動的に制御することで，コンピューティングリソース，通信リソースの削減・最適化を実現する手法を提案した．大規模ネットワーク適用を想定した基礎シミュレーションのフレームワーク検討と評価実験を実施し，動的な割当数制御によるリソース管理の効率化，特に，提案手法適用によるコンピューティングリソース消費量削減に関して，60～68%削減できることを確認した．また，特定条件においてセキュリティレベルの維持・向上に対する有効性も確認した．

今後の課題として，セキュリティレベルを低下せずにセキュリティ検証を効率化するアルゴリズムの検討，実環境条件を反映したマルチエージェントシミュレーションの実施，基本機能要件／性能要件に基づくシステム PoC 構築・技術検証を進める．

参考文献

[1] S. Rose, O. Borchest, S. Mitchell, and S. Connelly, : SP800- 207: Zero Trust Architecture, NIST, 2020

[2] 経済産業省. : 機器のサイバーセキュリティ確保のためのセキュリティ検証の手引きを取りまとめました, 入手先 ＜http://www. meti.go.jp/press/2021/04/20210419003 /20210419003.html＞ (参照 2021-4-20), 2021.

[3] J. Kindervag. : Build Security Into Your Network's DNA: The Zero Trust Network Architecture, Forrester Research Inc: 1-26, 2010.

[4] Y. Xiangshuai, and H. Wang. : Survey on Zero-Trust Network Security, International Conference on Artificial Intelligence and Security (ICAIS), 2020.

[5] R. Ward and B. Beyer. : BeyondCorp: A New Approach to Enterprise Security, The magazine of USENIX & SAGE 39.6: 6-11, 2014.

[6] Gartner. : Market Guide for Zero Trust Network Access, 入手先 ＜https://www.gartner.com/en/documents /3986053/market-guide-for-zero-trust-network-access＞ (参照 2021-4-20), 2021.

[7] 河合 将隆, 妙中 雄三, 門林 雄基. : 社内システムを想定したゼロトラストネットワークの試作とセキュリティポリシーの検討, IEICE ネットワークシステム（NS）研究会: NS2020-111, 2020.

[8] L. Thomas, M. Halter, and F. Kargl. : Context-based Access Control and Trust Scores in Zero Trust Campus Networks, SICHERHEIT2020, 2020.

[9] D. Jisa, and C. Thomas. : Efficient DDoS flood attack detection using dynamic thresholding on flow-based network traffic, Computers & Security 82: 284-295, 2019.

[10] T. Yang, Z. Lei, and P. Ruxiang. : Fine-grained big data security method based on zero trust model, 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS), 2018.

[11] X. Zhang, et al. : Power IoT security protection architecture based on zero trust framework, 2021 IEEE 5th International Conference on Cryptography, Security and Privacy (CSP), 2021.

[12] S. Mayra, and R. Deters. : Zero-trust hierarchical management in IoT, IEEE international congress on Internet of Things (ICIOT), 2018.

© 2021 Information Processing Society of Japan