ゼロトラストネットワークにおける 利便性を考慮したアクセス制御に関する研究
VDM ++を用いた妥当性確認

2020SE034 三輪晴輝 2020SE089 下林大祐
指導教員：張漢明

1 はじめに

近年、ゼロトラストネットワーク [1] を用いた IoT ネットワークシステムが普及している．ゼロトラストネットワークとは, すべてのトラフィックを信用しないアクセス制御である．ゼロトラストネットワークではセキュリティと利便性の間のトレードオフの問題がある．セキュリティを高めるためには，リソースにアクセスする毎に認証する必要がある．利便性を高めるために認証を省略すると利便性が低下する．セキュリティと利便性は対象とするアプリケーションの特性に応じて設計する必要がある．

本研究の目的は，ゼロトラストネットワークにおける利便性を考慮した IoT ネットワークシステムの設計と妥当性の確認である．アクセス制御の妥当性をプログラミングの段階で確認するためには，その実現のコストが高く，その振る舞いを網羅的に検証することは一般的に困難である．設計の段階でアクセス制御の妥当性を確認することが望まれる．設計段階でのコストは高くなるが, 実現のコストと比較した場合, 開発全体のコストは低くなることが期待できる．

上記の目的を達成するための研究課題は

- ゼロトラストネットワークと攻撃およびアクセス制御の関係の整理
- 設計段階の構成要素の抽象化と形式化

である．アクセス制御を定義するためには，前提となるゼロトラストネットワークと攻撃およびセキュリティの関係を明確にする必要がある．設計段階でアクセス制御の妥当性を確認するためには，ゼロトラストネットワークの構成要素と攻撃およびセキュリティを抽象化して形式化する必要がある．抽象化して形式化することにより, システム全体の理解を容易にし, 設計上の問題点が特定しやすくなる．

問題解決のための基本的なアイデアは，

- エージング期間の導入による攻撃の状況に応じたアクセス制御の変更
- 設計段階における形式手法の導入

である．

「エージング期間」とは, 一定期間認証なしでアクセスできる期間である．設計段階において形式手法を導入し, VDM++ を用いてテストの妥当性を確認する．

2 関連技術

本節では，関連技術と関連研究について説明する．

2.1 ゼロトラストネットワーク

ゼロトラストネットワークとは, すべてのトラフィックを信用しないアクセス制御を用いたネットワークである．文献 [2] ではゼロトラストネットワークの構造について説明している．図 1 は文献 [2] で示されているゼロトラストネットワークの論理コンポーネントを表している．. ゼロトラストアーキテクチャのコンポーネントとして, PDP(Policy Decision Point) 及び PEP(Policy Enforcement Point) がある．PDP は, アクセスの評価を行い, 認証, 認可を決定する役割を担う．PEP は, PDP からの指令を受け取り, 実際に IoT デバイスやサーバなどのリソースへアクセス制御を行う．

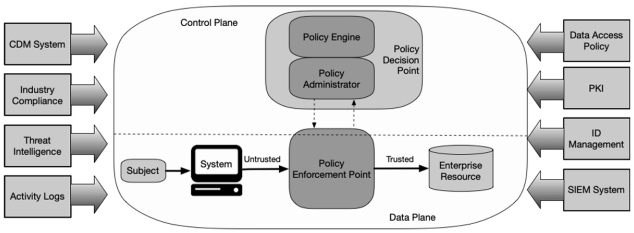

図 1 ゼロトラストネットワーク

2.2 アクセス制御

アクセス制御とは, システムやネットワークへのアクセスを管理・制御するための仕組みである．アクセス制御の主な目的として, リソースのセキュリティの維持や適切な利用があげられる．文献 [3] では基本的なアクセス制御の概要について説明している．アクセス制御は, 認証, 認可, 監査の 3 つのプロセスから構成されている．

認証 アクセスしてきたユーザーが, サイト内で操作する権限を持つかの判断

認可 操作できる範囲の制御を指す．認証によって許可されたユーザーが, そのサイトでどこまで操作して良いかを制限

監査 設定したルールや条件によって, 不正なアクセスを防止する設定したルールや条件によって, 不正なアクセスを防止

2.3 時間的推移を考慮したアクセス制御

時間的推移を考慮したアクセス制御とは, 過去のアクセス履歴である「場所や時間帯などの関係」と現在の状態を

考慮したアクセスを制御することである．文献 [4] では，過去にアクセスした履歴と現在のアクセス状態やアクセスした場所をポリシーとして，アクセス制御を行う方法について説明している．図 2 では，文献 [4] を参考にしたアクセス制御の例を示している．過去にアクセスした履歴がある場合，ポリシーによって，11:00 から 11:30 まで一時的に認証が不要であることを示している．

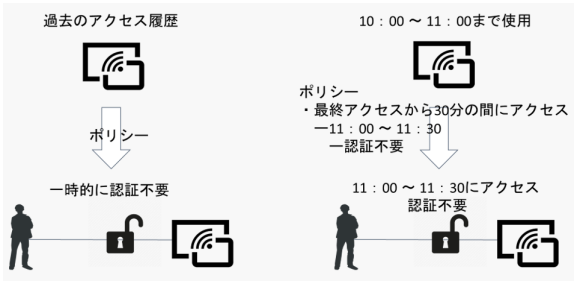

図 2　時間的推移を考慮したアクセス制御の例

2.4　セキュリティと利便性を考慮したアクセス制御

セキュリティと利便性を考慮したアクセス制御とは，ネットワーク内の状況に応じて，リスクを評価しそれに基づいたアクセス制御を行うことである．文献 [5] では，アクセスのし易さやし難さを動的に変える研究であり，デバイスのロケーションやコンピュータの脆弱性，サイバー攻撃などの動的な要因のリスクをアクセス制御に組込んだものである．

2.5　VDM++

VDM++[6] とは，仕様を数学の記法を用いて厳密に記述するための形式手法の 1 つである．ソフトウェアの品質や信頼性向上に役立つデータ型やクラス、操作などを定義しシステムの振る舞いをモデル化できる．作成したモデルに対して，形式的な検証や設計段階から要求を満たすかどうかの妥当性を確認できる．

3　エージング期間を用いたアクセス制御

本節では，対象とする IoT ネットワークの前提と，提案するアクセス制御の概要を説明する．

3.1　対象とする IoT ネットワーク

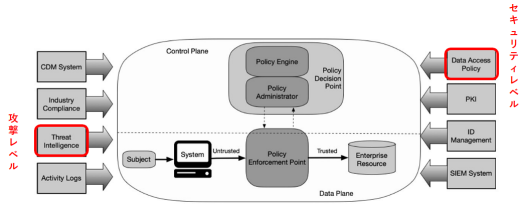

図 3　対象とする IoT ネットワーク

図 3 では，文献 [2] の論理コンポーネントの図に「攻撃レベル」と「セキュリティレベル」に相当したモデルである．

- 攻撃レベル
- セキュリティレベル

攻撃レベルは，外部からの攻撃頻度に応じてレベルをつける多種多様な攻撃を順序として段階化して抽象化したものである．デバイスのリソースにはセキュリティレベルがある．セキュリティレベルは，リソースの重要度に応じてそれぞれに順序としてレベルをつけて抽象化したものである．

3.2　エージング期間とエージング期間表

エージング期間とは，認証無しでアクセスできる期間である．この期間中は，アクセスした際に認証なしでアクセス可能である．図 4 では，サブジェクトがデバイスのリソースにアクセスするときの振る舞いを表している．サブジェクトが最初にリソースにアクセスした時，エージング期間のセッションが開始される．エージング期間のセッションが切れるまで認証は行わない．

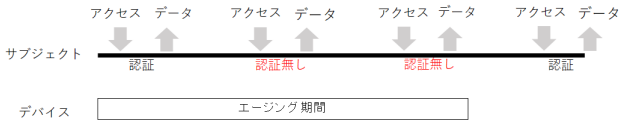

図 4　エージング期間

エージング期間表とは，攻撃レベルとセキュリティレベルに応じたエージング期間を示した表である．エージング期間表を図 1 に示す．エージング期間表は，「攻撃レベル」と「セキュリティレベル」の直積からエージング期間の写像とみなすことができる．具体例として，「攻撃レベル」と「セキュリティレベル」がそれぞれ 3 段階の場合の写像を 2 次元の表として示している．

表 1　エージング期間表

セキュリティレベル ＼ 攻撃レベル	1	2	3
1	30分	20分	10分
2	20分	10分	0分
3	10分	0分	0分

3.3　エージング期間表を用いたアクセス制御

攻撃を検出した場合，攻撃レベルが変化し，表 1 のエージング期間表に基づいてエージング期間が決まる．攻撃が変化した際の典型的な例を示す．

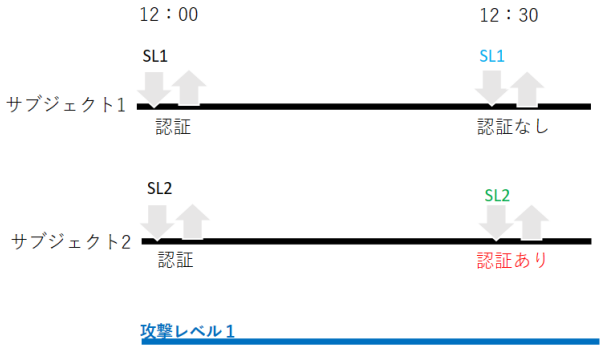

図 5　攻撃レベルが変化しない場合

図 5 は, 攻撃レベルが変化しない場合のエージング期間を示している. エージング期間開始時の攻撃レベルは 1 である. 一回目にアクセスする際は, すべてのリソースでは認証が必要. 二回目のアクセスは, 攻撃レベルとセキュリティレベルを考慮してエージング期間表に応じて, 認証か認証無しかどうかが決定する.

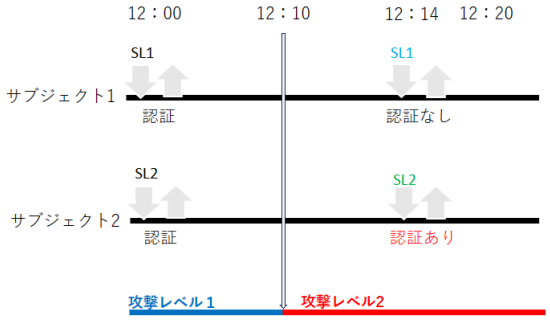

図 6　攻撃レベルが変化した場合

図 6 は, 攻撃レベルが変化した場合のエージング期間を表している. エージング期間開始時の攻撃レベルは 1 である. 攻撃により攻撃レベルが高くなった場合攻撃レベルが低い状態 (1) から高い状態 (2) に変更され, エージング期間表に基づき, エージング期間が変更される. エージング期間は, セキュリティレベルによって異なる. 認証が不要の場合と必要な場合を示している.

4　エージング期間を用いた IoT ネットワークシステムの設計

提案するエージング期間を用いた IoT ネットワークシステムの設計を UML と VDM++ を用いて記述する.

4.1　UML 記述

4.1.1　IoT ネットワークシステム

ゼロトラストを前提とした IoT ネットワークシステムのクラス図を図 7 に示す. このネットワークシステムは Zero Trust Architecture の figure1[2] を参考に作成したクラス図である. ポリシーエンジン (PE) クラスがサブジェクトの認証, ポリシー実施ポイント (PEP) クラスが認可を行っている.

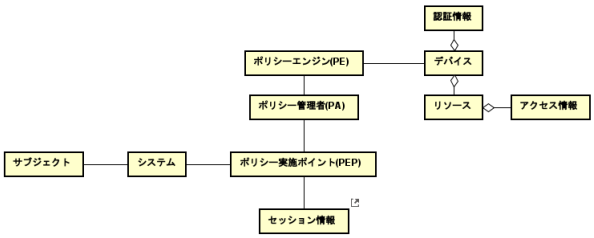

図 7　IoT ネットワークシステム

4.1.2　エージング期間による拡張

図 7 をエージング期間を用いて拡張した IoT ネットワークシステムのクラス図を図 8 に示す. このクラス図ではリソースに対してセキュリティレベルが設定されている. また, エージング期間クラスには以前にサブジェクトが認証された時刻が保持されている. エージング期間は認証された時刻とキュリティレベル, 攻撃レベルを図 1 のエージング期間表に適応した結果によって決定される.

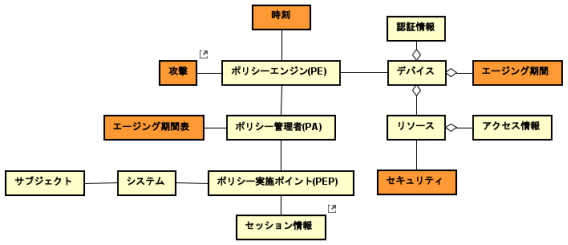

図 8　エージング期間を用いた IoT ネットワークシステム

4.2　VDM++ 記述

VDM 記述の例として認証情報クラスの照会の操作の VDM++ での記述を以下で説明する.

共通.vdmpp

```
types
    public 可否 = <可> | <否>;
    public id = int;
    public password = int;
```

認証情報.vdmpp

```
instance variables
    private 認証 : 共通`可否;
    private 登録id : 共通`id;
    private 登録password : 共通`id;
operations
public 照会 :  共通`id*共通`password==> 共通`可否
    照会(i,p) == (
        if {i,p} in set {{登録id, 登録password}}
        then
            認証 := <可>
        else
            認証 := <否>;
        return 認証;
    );
```

5 VDM++ による妥当性確認

テストケースの一例を以下に示す. セキュリティレベル 1 のリソースを持っているデバイス 1 にサブジェクトがアクセスし, 初回のアクセス時の攻撃レベルは 3 であり,12 分時点で攻撃レベル 1 に変化した際の攻撃レベル, セキュリティレベルによって適切にエージング期間表が適応されているか確認できるテストケースである. 期待値には攻撃レベルが変化した際に適切に図 1 のエージング期間表が適応された場合の結果を代入している. テストケースを実行し, その期待値通りの結果になることで妥当性を確認できる.

```
public テストケース : () ==> bool
    テストケース() == (
    時刻.時刻設定(0);
    攻撃.攻撃レベル設定(3);
    時刻.時刻設定(1);
    システム.獲得要求(デバイス1);
    システム.認証返答(デバイス1,1,1111);
    時刻.時刻設定(11);
    システム.獲得要求(デバイス1);
    時刻.時刻設定(12);
    システム.獲得要求(デバイス1);
    攻撃.攻撃レベル設定(1);
    時刻.時刻設定(31);
    システム.獲得要求(デバイス1);
    時刻.時刻設定(32);
    システム.獲得要求(デバイス1);
    期待値 := [<認証要求>,<データ獲得>,<データ獲得>,<認証要求>,<データ獲得>,<認証要求>];
    return 期待値 = ログ.表示();
    );
```

6 考察

6.1 妥当性の確認の意義

VDM++ では攻撃レベルの変化に応じてエージング期間が正しく変更されているのか確認することができるテストケースを用意した. 用意したテストケースは,

1. セキュリティレベルと攻撃レベルの変化なし
2. 攻撃レベルが低から高になった場合
3. 攻撃レベルが高から低になった場合
4. 認証が失敗した場合

の 4 種類 28 項目である. このテストケースを実行し, 期待値を満すことで妥当性確認を行うことができた. このテストケース以外にも攻撃レベルが激しく変化する場合などのより多くのテストケースを追加することによってより妥当性確認の向上が見込まれる.

6.2 VDM++ 記述の意義

テストを網羅的に行うためには, 不変条件の導入が重要である. 不変条件の導入を行うことで, 本研究で示したポリシーより厳密な定義が可能になり, さらに複雑なテストケースや広範囲に渡り妥当性の確認が可能になることが期待できる. VDM++ で設計したエージング期間を用いたネットワークシステムを記述したことによってクラス図やシーケンス図の誤りを設計段階で気付くことができる. このことから VDM++ での記述を行うことによって設計段階でのコストは上がるが実現段階でのコストが下がり, 開発の全体的なコストは低くなることが期待できる.

7 おわりに

本研究では, ゼロトラストネットワークにおける IoT ネットワークシステムの設計と妥当性の確認を目的として, エージング期間を用いたアクセス制御を提案し, VDM++ を用いて設計段階の妥当性の確認を検討した. VDM++ によるテストを実施することにより, 設定したアクセス制御のポリシーに対して, 妥当性のある適切な変更を確認することができた. アクセス制御のポリシーはアプリケーションの特性に応じて設計する必要がある. 設計の妥当性を確認するために, VDM++ を用いた設計の記述が有用であることを確認した.

今後の課題として, 文献 [4] での研究で示されているように, 「場所」や「アクセスの多い時間帯」,「使用者とデバイスの関係」などをアクセス制御に盛り込んだ複雑な制御を考慮した場合, コンテキスト指向を導入した設計による妥当性の確認があげられる.

参考文献

[1] 境野 哲: ”IoT への期待と課題 〜IoT システム開発者・利用者の心得〜” ,J-STAGE, 2017.

[2] Scott Rose, Oliver Borchert, Stu Mitchell, Sean Connelly : ”Zero Trust Architecture”, National Institute of Standards and Technology, 2020.

[3] 菊池 浩明, 上原 哲太郎 : ”ネットワークセキュリティ” 情報処理学会 編集, 2017.

[4] Chahal Arora, Syed Zain R. Rizvi, Philip W.L. Fong : ”Higher-Order Relationship-Based Access Control: A Temporal Instantiation with IoT Applications”, University of Calgary, 2022.

[5] Brian Lee, Roman Vanickis, Franklin Rogelio and Paul Jacob : ”Situation Awareness Based Risk Adaptive Access Control in Enterprise Network”, Software Research Institute, Athlone Institute of Technology, Athlone,Ireland, 2017.

[6] ジョン・フィッツジェラルド, ピーター・ゴルム・ラーセン, ポール・マッカージー, ニコ・プラット, マーセル・バーホフ, : ”VDM++ によるオブジェクト指向システムの高品質設計と検証 仕様の品質を飛躍的に高める手法 ”, 2010.