Vol.2015-SE-187 No.45
2015/3/13

トレーサビリティに基づく 仕様・設計におけるセキュリティ分析支援

谷津 弘一[†1] 松並 勝[†2] 澤田 壽實[†3] 平川 剛[†4] 野田 厚志[†4] 小幡 直也[†4]
安藤 崇央[†1] 久住 憲嗣[†1] 孔 維強[†5] 福田 晃[†1]

現在のシステム開発で実施されるセキュリティ分析においては，実装工程では，ソースコードの静的解析や，バイナリコード上での動的な脆弱性判断等，ツールによる分析支援が可能であるが，それ以前の工程では体系立てられた分析方法はなく，分析を支援するツールも見られない．本稿では，システムの実装工程以前に作成される仕様や設計モデル上で実施可能な，トレーサビリティに基づくセキュリティ分析方法を提案する．

Support for Security Analysis of Design Models based on Traceability

HIROKAZU YATSU[†1] MASARU MATSUNAMI[†2] TOSHIMI SAWADA[†3]
GO HIRAKAWA[†4] ATSUSHI NODA[†4] NAOYA OBATA[†4] TAKAHIRO ANDO[†1]
KENJI HISAZUMI[†1] WEIQIANG KONG[†5] AKIRA FUKUDA[†1]

Usually, security analysis is mainly done at the implementation phase of system development. In this paper, we propose an universal approach of security analysis at the design phase based on traceability established among security threats, measures for threats and SysML diagrams.

1. はじめに

情報社会の基盤に組み込まれているソフトウェアシステムには，セキュアであることが求められる．システムがセキュアであるための要件（以下本稿では，これをセキュリティ要件と呼ぶ）は，システム開発の上流の工程で適切に認識され，システムの仕様や設計に的確に反映されるべきものである．しかしながら，現在のシステム開発では，システムがセキュアであるか否かを判断するセキュリティ分析は，ソースコード上での静的解析やバイナリコード上での脆弱性判断等，システムの実装工程で作成されるコードに対してツールを用いて行われるものが専らである．実装工程以降で検出されたセキュリティの脆弱性を解消するためにシステムの仕様や設計を修正しなければならないこともある．その場合，修正に伴う手戻りの負荷は大きい．実装工程前にシステムの仕様や設計に対して行うセキュリティ分析（以下本稿では，これをセキュリティ設計分析と呼ぶ）の必要性は，このようなところにも現れる．

セキュリティ設計分析を行うには，暗号や OS に関する知識に加えて，システムに対してどのようなセキュリティ脅威が考えられ，それらの脅威にどのように対応していくのか，といった，セキュリティに係わるシステムの状況（以下，セキュリティ状況と略記する）の把握が必要である．残念ながら，前者に比べて，後者の必要性・重要性はあまり認識されていない．本稿では，ソニーデジタルネットワークアプリケーションズ株式会社(以下，SDNA 社と略記する）で試みられたセキュリティ設計分析[1 を例に取り，セキュリティ設計分析を支援するための仕組みについて議論する．

†1 九州大学
Kyushu University
†2 ソニーデジタルネットワークアプリケーションズ株式会社
Sony Digital Network Applications. Inc.
†3 株式会社 考作舎
Kosakusha, Ltd.
†4 株式会社 ネットワーク応用技術研究所
Network Application Engineering Laboratories, Ltd.
†5 大連理工大学
Dalian University of Technolog

2. セキュリティ設計分析

本節では，セキュリティ設計分析の例として，[1]で試みられている分析方法を紹介する．この方法は，脅威モデリング[2]を参考にしている．脅威モデリングにおける記述対象は，システムが直面するセキュリティ脅威であるが，[1]における記述対象は，システムの資産がセキュリティ脅威から保護されている状態である．さらに，[1]では，資産が保護されている状態の表現を，できるだけ資産に対する操作を用いて行うようにすることで，状態の表現から属人性を排除し，客観性を与えている．

2.1 脅威モデリング

脅威モデリングでは，攻撃木（Attack Tree）という，FTA（Fault Tree Analysis）で作成される失敗木（Fault Tree）と同様の木構造に従い，セキュリティ脅威が出現するための必要条件を洗い出していく（図 1）．

Vol.2015-SE-187 No.45
2015/3/13

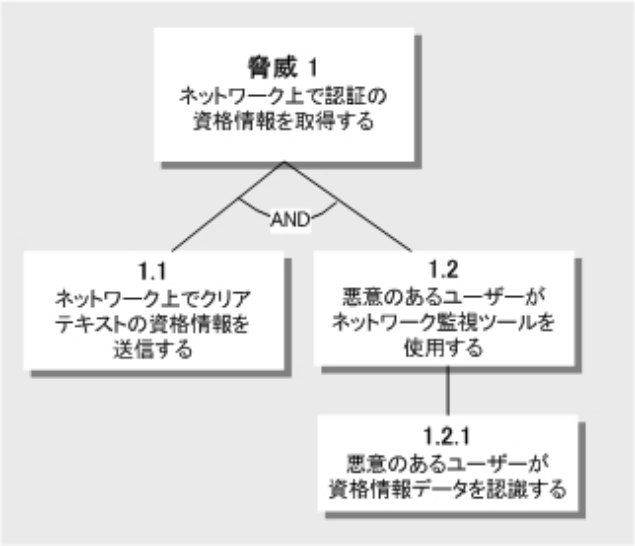

図 1 攻撃木の例

Figure1 An Example of Attack Tree

しかしながら，攻撃木で記述される必要条件やその出現順は属人的になりやすい．このため，セキュリティ知識の豊富な実務者であっても攻撃木を作成するには困難が伴う．

2.2 セキュリティ設計分析

[1]では，セキュリティ要件として，システムに蓄えられる情報や機能が保護されている状態を記述する．情報や機能等，セキュリティ上システムで保護されるべきものを資産と呼び，システムが許容あるいは看過できない，攻撃者の行為を次の 2 つの観点から洗い出している．

- 資産へのアクセス
- 利用者の資産へのアクセスに対する妨害

そして，洗い出した行為を攻撃者が実行できない状態を，システムが保護されている状態として記述する．

ここでは，セキュリティ脅威は，その双対（否定）をとる形で記述される．すなわち，セキュリティ脅威を直接記述するのではなく，その双対である，システムがセキュリティ脅威にさらされていない状態が記述される．

[1]では，記述の状態の表現や分解から属人性を排除するための工夫としてセキュリティ要件記述の雛形（表 1）を用意，これを利用することで，上で挙げたシステムが保護されている状態を表す文の文体を統一し，記述の属人性を排除している．

表 1 セキュリティ要件の雛形

Table 1. A Template for Describing Security Requirements

資産種別	操作	セキュリティ要件
情報	READ	【攻撃者】は【情報】を READ できない
		【攻撃者】は【被害者】が【情報】を READ することを妨害できない
	WRITE	【攻撃者】は【情報】を WRITE できない
		【攻撃者】は【被害者】が【情報】を WRITE することを妨害できない
機能	EXECUTE	【攻撃者】は【機能】を EXECUTE できない
		【攻撃者】は【被害者】が【機能】を EXECUTE することを妨害できない

2.3 セキュリティ要件の分解

セキュリティ要件は，システムの構成に応じて分解することができる．にここで，セキュリティ要件の例を紹介しよう．図 2 は，あるシステムの中でどのように資産が流れていくかの一例を表すものである．

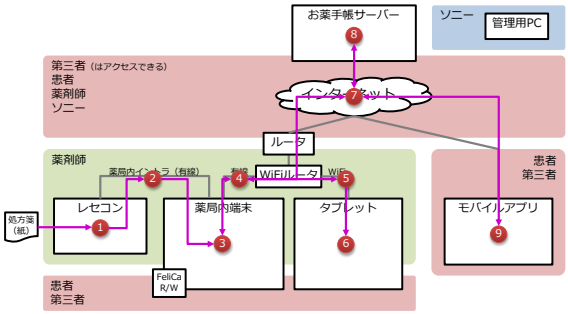

図 2 資産が存在する場所

Figure2 Places where Some Information Exists

セキュリティ要件の 1 つを「○○が第三者に READ できないこと」とすると，資産○○は，それが存在するシステムのあらゆる場所で，第三者が READ できないようにしておかなければならない．この要件は，図 3 のように分解される．

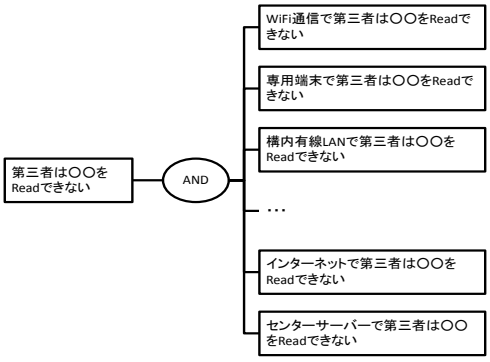

図 3 セキュリティ要件の分解

Figure3 Decomposition of a Security Requirement

3. トレーサビリティに基づくセキュリティ設計分析支援

本節では,

3.1 トレーサビリティに基づくセキュリティ設計分析支援結果の整理

前節で紹介したセキュリティ設計分析で作成されるセキュリティ要件は，分析の対象であるシステムの構成に従い分解される．そして，システムがセキュアであることは，現在発見されているセキュリティ脅威から資産が保護される，という形で宣言せざるを得ない．従って，セキュリティ要件はセキュリティ脅威と関連付けて記述されるべきものであると言える．以上を鑑みて，筆者らは，セキュリティ要件は，システムの構成やシステムがさらされるセキュリティ脅威と関連付けて把握されるべきものと考える．

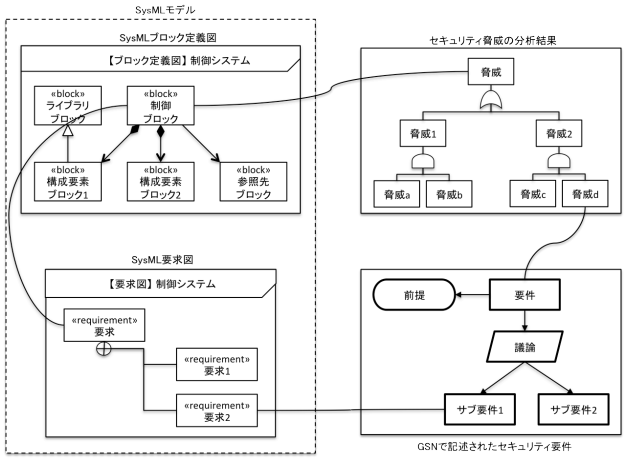

図 4 設計・脅威・要件の間のトレーサビリティ

Figure4 Traceability on Design, Threats and Requirements

図 4 は，システムの構成，セキュリティ脅威，そしてセキュリティ要件の間に確立されるべきトレーサビリティを表している．この図の左上，右上，右下，左下は，システムの構成を表す SysML ブロック定義図（図の左上），セキュリティ脅威を分析した攻撃木，GSN（Goal Structuring Notation）で表されるセキュリティ要件，SysML 要求図を各々表している．そして，ブロック定義図の中のブロック，攻撃木の中の脅威，GSN で書かれた図の中の要件，要求図の中の要求の間には，各々トレーサビリティが確立されている．各トレーサビリティが表すものは以下の通りである．

- ブロック B と脅威 T の間のトレーサビリティ
  「ブロック B が脅威 T にさらされている」
- 脅威 T と要件 S の間のトレーサビリティ
  「要件 S が満たされていれば，脅威 T から資産が保護される」
- 要件 S と要求 R の間のトレーサビリティ
  「要求 R が満たされていれば，要件 S も満たされている」
- 要求 R とブロック B の間のトレーサビリティ
  「ブロック B は要求 R を満たしている」

3.2 トレーサビリティに基づく脆弱箇所の検出

図 4 で示されているように，設計・脅威・要件の間にトレーサビリティが確立されていれば，SMT ソルバを用いてセキュリティ脆弱箇所を検出することが可能である．ここで言うセキュリティ脆弱箇所とは，SysML ブロック定義図の中にある，セキュリティ要件が十分でないブロックのことである．この検出方法の原理を図 5 に示す．

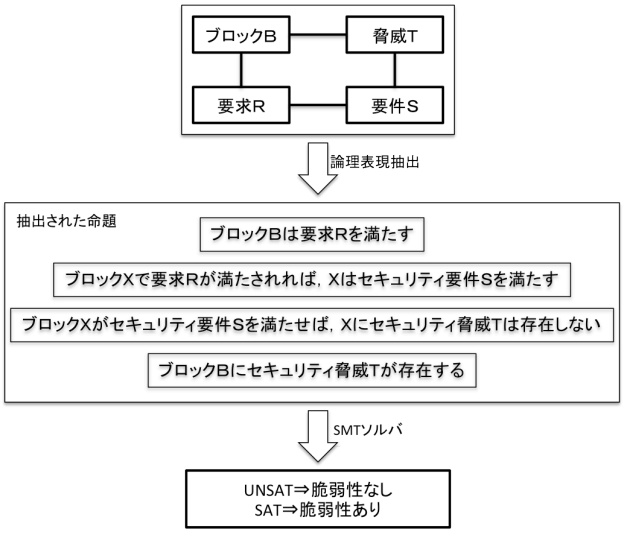

図 5 トレーサビリティに基づく脆弱箇所の検出

Figure5 Detecting Vulnerable Point based on Traceability

ブロック B がセキュリティ脅威 T にさらされているとし，脅威 T に対するセキュリティ要件を S としよう．この時，ブロック B が満たしている要求の中に要件 S が含まれていれば，ブロック B は脅威 T から保護される．反対に，ブロック B が満たしている要求の中に，脅威 T に対する要件が何も含まれていなければ，ブロック B は脅威 T から身を守ることはできない．ブロック B は脅威 T に対して脆弱な箇所であると言える．

ブロック B，脅威 T，要件 S，要求 R の間に確立されているトレーサビリティからは，以下を表す命題（あるいは述語）を抽出することができる．

1. ブロック B と要求 R の間のトレーサビリティ
   →「ブロック B は要求 R を満たす」
2. 要求 R と要件 S の間のトレーサビリティ

→「任意のブロック X に対し，X が要求 R を満たしていれば，X は要件 S を満たす」

3. 要件 S と脅威 T の間のトレーサビリティ
→「任意のブロック X に対し，X が要件 S を満たしていれば，X は脅威 T にさらされない」
4. 脅威 T とブロック B の間のトレーサビリティ
→「ブロック B は脅威 T にさらされている」

ブロック B，脅威 T，要件 S，要求 R の間に適切なトレーサビリティが確立されていれば，命題 1〜3 からは「ブロック B は脅威 T にさらされない」ことが導かれるが，これは命題 4「ブロック B は脅威 T にさらされている」ことと矛盾する．従って，命題 1〜4 を SMT ソルバに与えると，解が存在しない旨の答えが返される．

一方，ブロック B，脅威 T，要件 S，要求 R の間に確立されるトレーサビリティの中に漏れ･抜けがある場合，例えば，要件 S と要求 R の間のトレーサビリティが確立されておらず，ブロック B が要件 S を満たしていないような場合を考えよう．この場合，以下の 3 つの命題や述語が抽出される．

1. ブロック B と要求 R の間のトレーサビリティ
→「ブロック B は要求 R を満たす」
2. 要件 S と脅威 T の間のトレーサビリティ
→「任意のブロック X に対し，X が要件 S を満たしていれば，X は脅威 T にさらされない」
3. 脅威 T とブロック B の間のトレーサビリティ
→「ブロック B は脅威 T にさらされている」

この場合，命題 1〜3 を SMT ソルバに与えると，ブロック B で要件 S が満たされていることが偽となるような解が返されるので，ブロック B が満たす要求に要件 S が含まれるようトレーサビリティを確立する必要があることがわかる．

3.3 例

図 3 では，セキュリティ要件「資産〇〇が第三者に READ されない」を，図 2 で示したシステムの構成に応じて分解したものを挙げた．ここでは，分解した要件の 1 つである「インターネットで第三者は〇〇を READ できない」ことを，さらに以下の 3 つに分解してみよう．

- 通信はすべて HTTPS で行われる
- サーバの証明書は自己署名のものではない
- クライアントは証明書エラーを無視しない

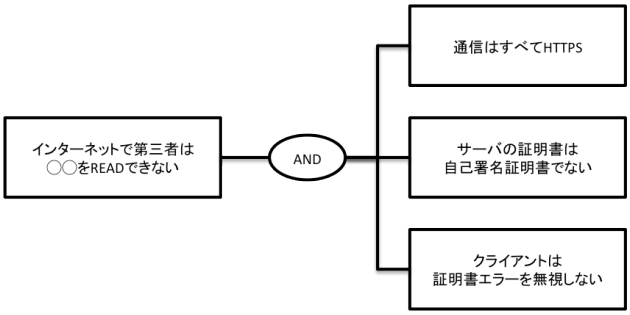

図 6 セキュリティ要件の分解
Figure6 Another Decomposition of a Security Requirement

ここで挙げたセキュリティ要件と関係するセキュリティ脅威は「インターネットで情報が漏洩する」ことである．そして，分解されたセキュリティ要件は，GSN を用いて次のように表現できる．

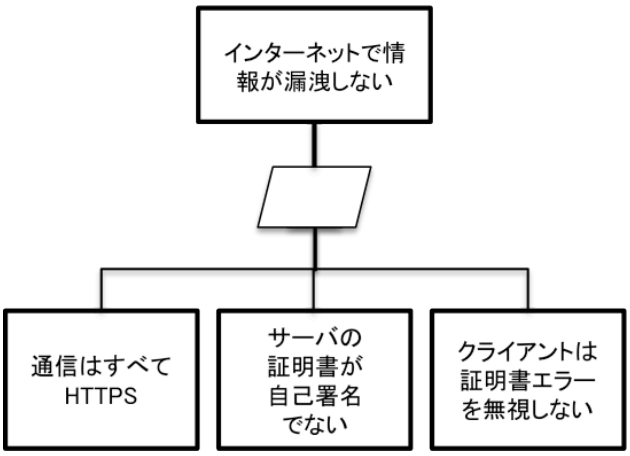

図 7 セキュリティ要件の GSN 表記
Figure7 A Security Requirement written in GSN

この GSN 表記では，ゴール「インターネットで情報が漏洩しない」ことは，3 つのサブゴール「通信はすべて HTTPS」「サーバの証明書が自己署名でない」「クライアントは証明書エラーを無視しない」ことに分解される．すなわち，この 3 つのサブゴールが達成されれば，ゴールは達成されることになる．

一方，図 2 で示されたシステムの構成を表す SysML ブロック定義図を図 8 に示す．

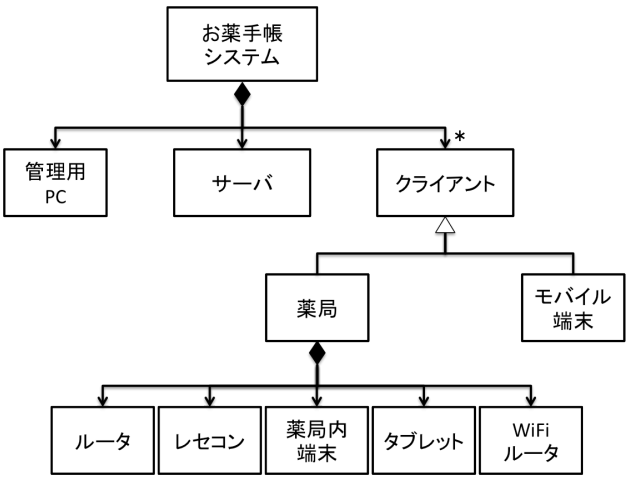

図 8 システムのブロック定義図
Figure7 A Block Definition Diagram of the System

システムは管理用 PC，サーバ，クライアントの 3 つから構成され，クライアントには薬局とモバイル端末の 2 種類がある．薬局は，ルータ，レセプトコンピュータ（レセコン），薬局内端末，タブレット，WiFi ルータから構成される．

以上を踏まえると，図 6 で挙げたセキュリティ要件は，SysML ブロック定義図，セキュリティ脅威，GSN で記述したセキュリティ要件の間に下図に示されるようなトレーサビリティを付けて整理される．

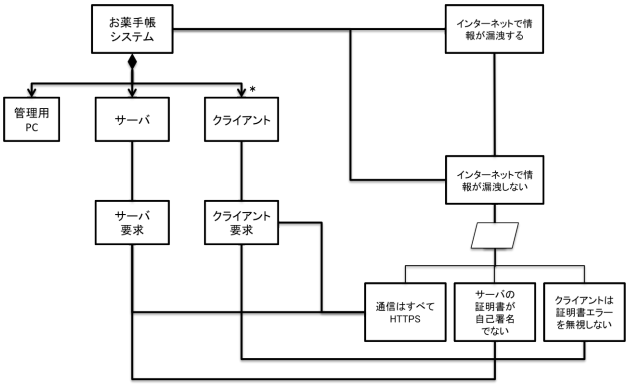

図 9　トレーサビリティに基づく要件の整理
Figure9 Arrangement of a System Requirement based on Traceability

システムのセキュリティ脅威として「インターネットで情報が漏洩する」ことがあり，それから資産を保護するために十分なセキュリティ要件として，「通信はすべて HTTPS」「サーバの証明書が自己署名でない」「クライアントが証明書エラーを無視しない」ことが挙げられている．そして，サーバに対する要求に「通信が全て HTTPS」「サーバの証明書が自己署名でない」ことが含まれ，クライアントに対する要求に「通信がすべて HTTPS」「クライアントは証明書エラーを無視しない」ことが含まれる．トレーサビリティが適切に付けられており，システムにセキュリティの脆弱性が存在しないことは，図 9 から抽出した命題を SMT ソルバに与えて，命題をすべて真にするような解が存在しない旨の回答が返されることで確認できる．

4. おわりに

トレーサビリティに基づきセキュリティ設計分析結果の整理や脆弱性の検出を行う仕組みを提案した．この仕組みのプロトタイプが現在実装されており，その有効性の確認を進めているところである．

今後の課題としては，システム設計の記述範囲と階層性を持つ対象の間の効率的なトレース付けが挙げられる．前者は，セキュリティ要件は，実装されたシステムだけでなく，そのステークホルダ達にも与えられるので，ステークホルダを含めたシステム設計の記述が求められるからである．後者は，対象の階層をトレース付けに活かすことが，処理の効率化にとって必要だからである．

謝辞　本研究は総務省 SCOPE「システム開発の設計工程におけるセキュリティ分析手法の
研究開発」の支援を受けて実施されました．ここに感謝の意を表します．

参考文献

1) 松並勝「ソニーの電子お薬手帳システムに適用したセキュリティ設計分析手法」The 12th Workshop on Critical Software Systems (12thWOCS^2), 2015
2) 松並勝「ねえねえやってる？仕様・設計のセキュリティ分析♪　面白いよぉ−」(ISC)2 Japan Chapter 発足記念イベント, 2014