SCIS 2018 2018 Symposium on Cryptography and Information Security
Niigata, Japan, Jan. 23 - 26, 2018
The Institute of Electronics, Information and Communication Engineers

脆弱性データベースを使用した脅威分析方法の提案
A Proposal of Threat Analyses using Vulnerability Databases

梅澤 克之 *† 三科 雄介 * 田口 研治 ‡ 寶木 和夫 *
Katsuyuki Umezawa Yusuke Mishina Kenji Taguchi Kazuo Takaragi

あらまし 重要インフラを支える制御情報システム，実世界情報を収集・分析する IoT システムに対するサイバー攻撃が増加している．セキュリティバイデザインのコンセプトに基づいて，セーフティとセキュリティを両立するシステム設計支援技術を開発することを目的とする．既報のサイバー攻撃が拡散・再発している現状を踏まえて，脆弱性事例データベースとシステム設計情報を活用した脅威分析方法を提案する．具体的には，分析対象システムの設計段階に実施が可能で，かつ実運用中の既存システムで発見された複数の脆弱性の組み合わせで生じる攻撃事例を活用することが可能な脅威分析方法を提案する．

キーワード 脅威分析，脆弱性情報，アタックツリー

1 はじめに

従来より脅威分析システムとして，米国マイクロソフト社が発表した脅威分析ツール “Microsoft Threat Modeling Tool”（以下，MS 脅威分析ツールと呼ぶ）がある [1]．MS 脅威分析ツールは，分析対象である情報システムに含まれるフローを組織間のインタフェースで分割して，フローの両端の要素の相互作用の特性情報を抽出し，あらかじめ STRIDE 分類軸でまとめた脅威分類表と比較して，発生する可能性がある脅威を提示できるようにしている．この MS 脅威分析ツールは，システムのデータフローダイアグラムを用いる．データフローダイアグラムはシステム設計の詳細設計段階で使用されることが多く，概念設計段階での脅威分析に適用が難しいという問題がある．また，脆弱性事例情報については，米国 MITRE 社が CWE（Common Weakness Enumeration）と呼ぶ脆弱性事例情報データベースを収集・蓄積し公開している [2]．しかし，CWE は実運用中の既存システムで発見された脆弱性事例を記述しているため，システム設計の概念設計段階に適用することが困難である．また，情報システムが高度化するにつれ，単一の脆弱性を突くだけでは攻撃が成功せず，複数の脆弱性を組み合わせて初めて攻撃が成立する事例が増加しつつある．上述の MS 脅威分析ツールでは，フロー断面でのスナップショット的な脅威分析は可能であるが，こうした複数の脆弱性を組み合わせた攻撃の可能性について脅威分析することは困難であった．組み合わせ的爆発については文献 [3] でも指摘しており，そこではランダムテストで対応する案が提案されている．また，攻撃の事例が増加しつつある中，NIST SP800-53 Rev.5[4] では「経験的攻撃データに基づく新しい最新のコントロールを組み込む」と記述されており，「経験的」な知識の有用性が指摘されている．

本論文では，分析対象システムの設計段階に実施が可能で，かつ実運用中の既存システムで発見された複数の脆弱性の組み合わせで生じる攻撃事例を活用することが可能な脅威分析システムを提案する．また，情報・制御システムにおいて脅威分析を実施する際に，システム全体に内在する複数の脆弱性を利用した一連の攻撃により，特定の脅威が発生する可能性がある場合に，この一連の攻撃の連鎖を効率的に発見する脅威分析システムを提案する．具体的には，伝統的に信頼性解析に使用されているフォールトツリー解析と，悪意のある攻撃パターンを検討するために導入されたアタックツリー解析とセキュリティ脆弱性データベースを組み合わせて，複雑なシステムのリスクを評価する新しい方法を提案する．提案するアプローチは，制御システムの文脈のなかで，脆弱性データベースの参照手順およびシステムフォールト確率の計算のための数学的モデルを提示する．

* 国立研究開発法人産業技術総合研究所, 東京都江東区青海 2-3-26 Advanced Industrial Science and Technology (AIST), 2-3-26 Aomi, Koto-ku, Tokyo 135-0064, Japan
† 湘南工科大学, 神奈川県藤沢市辻堂西海岸 1-1-25, Shonan Institute of Technology, 1-1-25 Tsujido-Nishikaigan, Fujisawa, Kanagawa 251-8511, Japan
‡ 国立研究開発法人産業技術総合研究所, 大阪府池田市緑丘 1-8-31 Advanced Industrial Science and Technology (AIST), 1-8-31 Midorigaoka, Ikeda, Osaka 563-8577 Japan

2 脆弱性データベース

米国 MITRE 社はいくつかの形式の脆弱性データベースを提供している．CVE（Common Vulnerability and Exposure:共通脆弱性識別子）[5] では，個別のソフトウェアの脆弱性がデータベース化されている．また，CWE（Common Weakness Enumeration : 共通脆弱性タイプ）[2] では，脆弱性が発生する原因部分に焦点を当て，共通の脆弱性がカタログ化されている．さらに，CAPEC（Common Attack Pattern Enumeration and Classification:共通攻撃パターンタイプと分類）[6] は，攻撃パターン別に分類されたデータベースとなっている．

3 フォールトツリー (FT) ーアタックツリー (AT) アプローチ

セキュリティの脅威によるセーフティへの干渉/中断は，電力，情報通信，自動車，航空，鉄道，医療などの安全重視システムにおいて大きな課題として認識されている．EVITA プロジェクト [7] では，車両内通信のセキュリティに関して，リスク分析，セキュリティ要件，アーキテクチャ設計，FPGA による HSM の試作，デモの実施が行われた．この際のリスク分析にはアタックツリーが用いられた．また，セーフティ（ハザード）とセキュリティ（脅威）の因果関係を分析する方法の 1 つは，フォールトツリー (FT) とアタックツリー (AT) の組み合わせでその関係を表現することである [8]．

4 提案手法

4.1 提案手法の概要

FT 分析に関連する科学文献は，今日，成熟している．信頼性と安全性の分野では，多数の例と事例が存在する [8]．一方，セキュリティ分析では，問題の複雑度が格段に大きくなる．CWE 等に報告される脆弱性は年に 1 万件を超えること，さらに，巧妙な攻撃はそれら脆弱性の複数個の組み合わせで起き，かつ，その蓋然性は無視できない大きさであること，さらに，それらの可能性を網羅的に押さえた AT を作成することは容易でない等の問題があった．

本論文では，このような問題に着目し，現実的なアプローチとして，次の特性を踏まえた脅威分析方法を提案する．

(a). 守る側，つまり，設計者には，守る際に傾向がある．つまり，新しい脆弱性が発表されたとき，システム内に残存する脆弱性同士の組み合わせにまで含めて新たに攻撃につながるかどうかというチェックをしない傾向がある．

(b). アタッカーにはアタックする際の傾向がある．多くの攻撃は，既知攻撃の模倣，マイナー変更である．

(c). これらの傾向を FT-AT に含めることで，分析の有用な出発点とすることができる．

(d). 過去に起きた事例を FT-AT 上で表現することによって，設計者に関連攻撃を気づかせる（危険性を認識させる）ことができる．

(e). このアプローチを漸次，拡張適用することで，新規攻撃の発見，および，防止に役立てることができる．

上述の提案手法の全体構成を図 1 に示す．

4.2 脆弱性モデル情報の作成

米国 MITRE 社は，いくつかの形式の脆弱性 DB を公開している [5][2][6]．しかし，これらの DB を参照しただけでは具体的な対象に対する（例えばコネクテッドカーに対する）AT を作成することは困難である．既存の攻撃事例の文献，あるいは報告書などを参考に AT を作成することになる．このように既存の脆弱性 DB と既存のアタック報告より得られる AT を第 1 の AT と呼ぶことにする．この第 1 の AT は，頂上事象と複数の中間事象，それに，最下位事象に階層化されて描かれる．第 1 の AT は，各脆弱性毎に 1 個作成する．

4.3 コンポーネント DB の提案

自動車や IoT 機器などの組み込み系では既存ソフトウェアをそのまま搭載するのではなく必要な下位コンポーネントを必要に応じで組み込むということが行われる．これに対して，CWE のような脆弱性 DB は，あるソフトウェアに対する脆弱性情報は記載しているものの，そのソフトウェアに組み込まれている下位コンポーネントの情報まで記載されていない．そこで，表 1 に示すような，ソフトウェアのバージョンとそのソフトウェアが内部で使用している下位コンポーネントのバージョンの対応表を充実させることによって，IoT などの組み込み機器の製造段階で，脆弱性情報を容易に確認できるようになる．

具体的には，テスラの Browser ハッキングを例にとると，テスラのブラウザが示す UserAgent は「Mozilla/5.0 (X11; Linux) AppleWebKit/534.34 (HTML, like Gecko) QtCarBrowser Safari/534.34」である．これに対して，CVE のデータベースでは，例えばブラウザを例にとると「Google Chrome before 16.0.912.77」という記述になっている．テスラの場合は，直接 Chrome を使っていないが，Chome に組み込まれている（実際には過去に組み込まれていた）WebKit を使っている．よって，ある

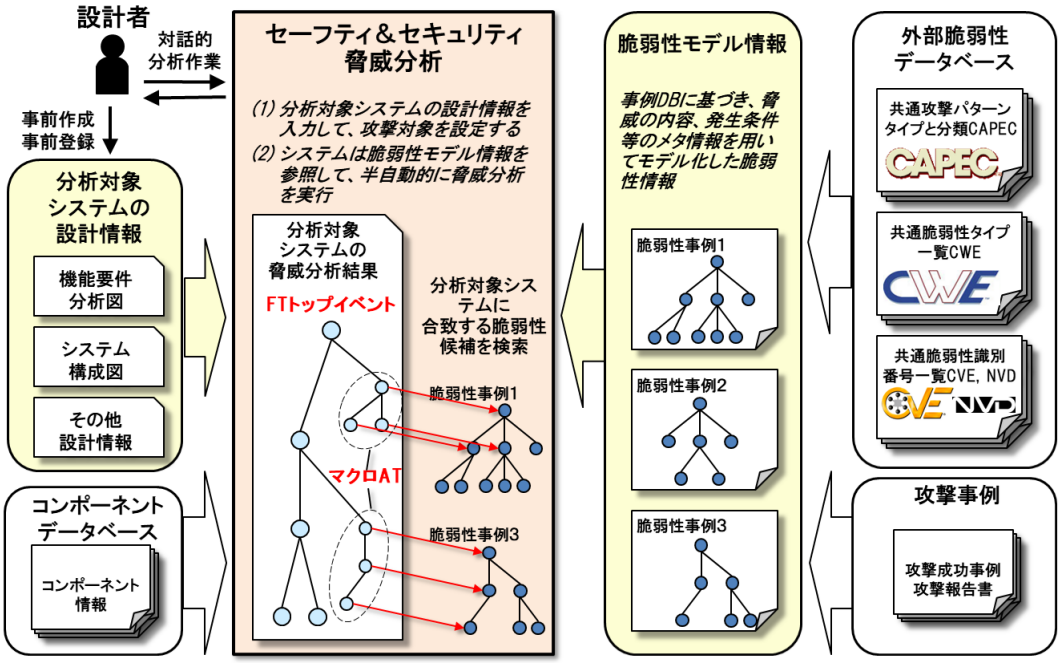

図 1: 提案手法の全体概要

ソフトウェアに組み込まれているコンポーネントのバージョンを示す対応表が必要となる．

表 1: コンポーネント DB の例

バージョン	リリース日	レイアウトエンジン
0.2.149	2008-09-02	WebKit 522
0.3.154	2008-10-29	WebKit 522
0.4.154	2008-11-24	WebKit 525
...	...	...
10.0.648	2011-03-08	WebKit 534.16
11.0.696	2011-04-27	WebKit 534.24
12.0.742	2011-06-07	WebKit 534.30
13.0.782	2011-08-02	WebKit 535.1
14.0.835	2011-09-16	WebKit 535.1
...	...	...
56.0.2924	2016-12-08	Blink 537.36

4.4 脅威分析アルゴリズム

4.2 節で示した脆弱性モデル，4.3 節で示したコンポーネント DB，および分析対象システムの設計情報をもとにした脅威分析アルゴリズムを示す．

(1). 評価対象システムに関する頂上事象（起きては困る事故，安全に関する事故でもよい）をトップノードとする第 2 の *AT* を作成する．この際に，評価対象システムに直接は含まれないコンポーネントであってもコンポーネント DB を参照することで関連があると判断されるコンポーネントを第 2 の *AT* に含める（図 2(2) の黒丸のノード）．第 2 の *AT* は，頂上事象と複数の中間事象，それに最下位事象に階層化されて描かれる．第 2 の *AT* は 1 個作成される（図 2(2)）．

(2). その後，各脆弱性毎に第 1 の *AT* と第 2 の *AT* を突合評価する（図 2(3)）．この突合評価には，自然言語処理と AI 処理[1]を用い，第 1 の *AT* に現れる頂上事象あるいは中間事象のうち，第 2 の *AT* に現れる中間事象に近いものがあるかどうかを判定する．もし，近いものがあれば，中間事象同士で論理和をとる（図 2(4)）．

(3). この論理和が取られたサブツリーに対して，第 1 の *AT* 側の最下位事象に，評価対象のコンポーネントがない中間事象に着目して再度分析を行い，追加すべきと判定されれば追加し，そうでなければその中間ノードを削除する．具体的には，第 2 の *AT* の構成コンポーネントに無関係（異なるコンポーネントやバージョンの違い）のノードを FALSE ノードとしたうえで，FALSE ノードの関係および FALSE ノードの直上の AND 関係の上位のノードの関係を削除する（図 2(5)）．

(4). このような処理を追加したすべての第 1 の *AT* について繰り返し，最終的に修正が終わったら，修正後の第 2 の *AT* を用いて，評価対象システムの

[1] 例えば後述の図 4 のトップノードの文言と図 5 の 3 段目最右の「Google Chrome before 13.0.782」という文言のマッチングを行うような処理

頂上事象（起きては困る事故）の真の発生確率を評価する．

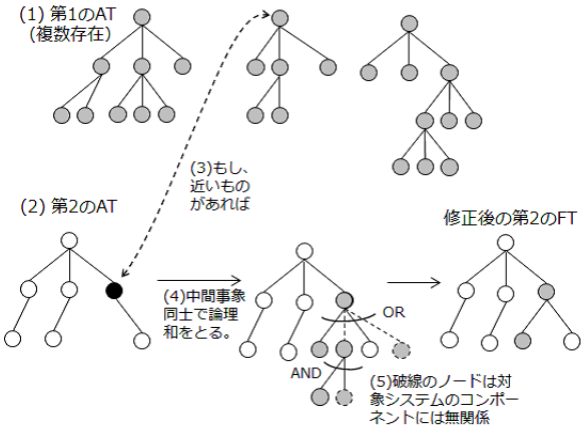

図 2: 脅威分析アルゴリズム

5 提案アルゴリズムの定式化

4.4 節で示したアルゴリズムを定式化する．

5.1 定義

文献 [8] に従い，アタックツリー AT の定義を下記に示す．また，AT の例を図 3 に示す．

$$\boldsymbol{G} = \{g_i\} : \text{アタックゴール} \tag{1}$$
$$\boldsymbol{O} = \{o_i\} : \text{オペレーション} \tag{2}$$
$$\boldsymbol{AS} = \{as_i\} : \text{アサーション} \tag{3}$$
$$\boldsymbol{V} = \{v_i\} : \text{脆弱性} \tag{4}$$
$$\boldsymbol{R} = \{r_i\} : \text{関係} \tag{5}$$

ここで，アタックゴールとは，すべての潜在的なサイバー攻撃の目標であり，オペレーションとは，攻撃者またはシステムのオペレータのいずれかによって実行され得るすべての基本動作（読み込み，書き込みなど）を表す．アサーションとは，アタックツリーの実際の分岐を考慮するために「検証する条件」を表すステートメント（例えば，「Web サーバーにパッチが当てられていない」など）であり，脆弱性は，既知の脆弱性である．関係とは，アタックツリーを構成する要素，つまり前述のアタックゴール，オペレーション，アサーション，脆弱性の間に存在する関係である．

アタックツリー AT_k は下記のように定義される．

$$AT_k = \{g_i, \boldsymbol{O}_i, \boldsymbol{AS}_i, \boldsymbol{V}_i, \boldsymbol{R}_i\} \tag{6}$$

ここで，$g_i \in \boldsymbol{G}, \boldsymbol{O}_i \subseteq \boldsymbol{O}, \boldsymbol{AS}_i \subseteq \boldsymbol{AS}, \boldsymbol{V}_i \subseteq \boldsymbol{V}, \boldsymbol{R}_i$ は関係の集合である．

全ての AT は 1 つのメインゴール g を持ち，論理ゲートの出力（上側）は，アサーションとなる．

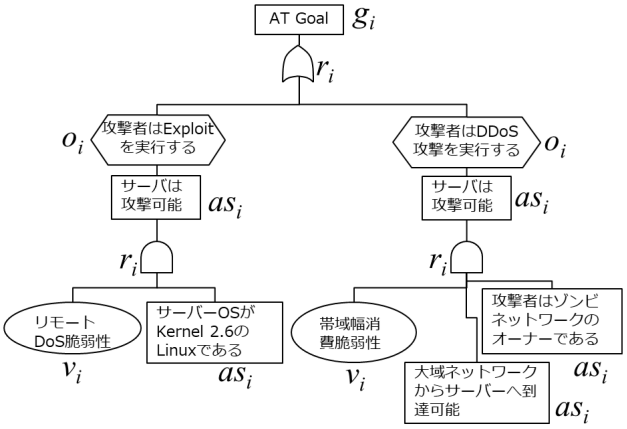

図 3: AT の例

5.2 提案アルゴリズムの定式化

第 1 のアタックツリー AT_k^1 と第 2 のアタックツリー AT^2 は以下のように定義される．

$$AT_k^1 = \{g_k, \boldsymbol{O}_k, \boldsymbol{AS}_k, \boldsymbol{V}_k, \boldsymbol{R}_k\} \tag{7}$$
$$AT^2 = \{g_j, \boldsymbol{O}_j, \boldsymbol{AS}_j, \boldsymbol{V}_j, \boldsymbol{R}_j\} \tag{8}$$

次に，$g_j = g_k$，または，$as_l \approx g_k$ なる k を探す．ただし，$as_l \in \boldsymbol{AS_j}$ である．さらに，$as_n \approx as_m$ なる n と m を探す．ただし，$as_n \in \boldsymbol{AS_k}, as_m \in \boldsymbol{AS_j}$ である．

ここで，as_n 以下のサブツリーを以下のように表す．

$$AT_n^{1_{sub}} = \{as_n, \boldsymbol{O}_n, \boldsymbol{AS}_n, \boldsymbol{V}_n, \boldsymbol{R}_n\} \tag{9}$$

次に，下記のように第 2 のアタックツリー AT^2 を更新する．

$$AT^2 = \{\ g_j, \boldsymbol{O}_j \cup \boldsymbol{O}_k \cup \boldsymbol{O}_n, \boldsymbol{AS}_j \cup \boldsymbol{AS}_k \cup \boldsymbol{AS}_n, \boldsymbol{V}_j \cup \boldsymbol{V}_k \cup \boldsymbol{V}_n, \boldsymbol{R}_j \cup \boldsymbol{R}_k \cup \boldsymbol{R}_n \backslash \boldsymbol{R}'\} \tag{10}$$

ここで，$\backslash$ は差集合を表す．また，$\boldsymbol{R}'$ は，$\boldsymbol{R}' = \boldsymbol{R}'_{OR} \cup \boldsymbol{R}'_{AND}$ であり，$\boldsymbol{R}'_{OR}$ は FALSE ノードの関係，$\boldsymbol{R}'_{AND}$ は，FALSE ノードの直上の AND 関係の上位のノードの関係である．FALSE ノードとは，AT^2 の構成コンポーネントに無関係（異なるコンポーネントやバージョンの違い等）である $o \in \boldsymbol{O}_k \cup \boldsymbol{O}_n, as \in \boldsymbol{AS}_k \cup \boldsymbol{AS}_n, v \in \boldsymbol{V}_k \cup \boldsymbol{V}_n$ を指す．

5.3 攻撃確率の計算 [8]

前節で定式化できたことにより，従来研究 [8] の計算方法で攻撃の確率計算を次の式を用いて行えることになる．

論理ゲートへの入力がそれぞれ独立である場合，i 番目の AND ゲートからの出力値の確率 $P_{out}AND_i$ と i 番目の OR ゲートからの出力値の確率 $P_{out}OR_i$ はそれぞれ下記の通りである．

$$P_{out}AND_i = \prod_{k=1}^{n} P_{in}(k, i) \tag{11}$$

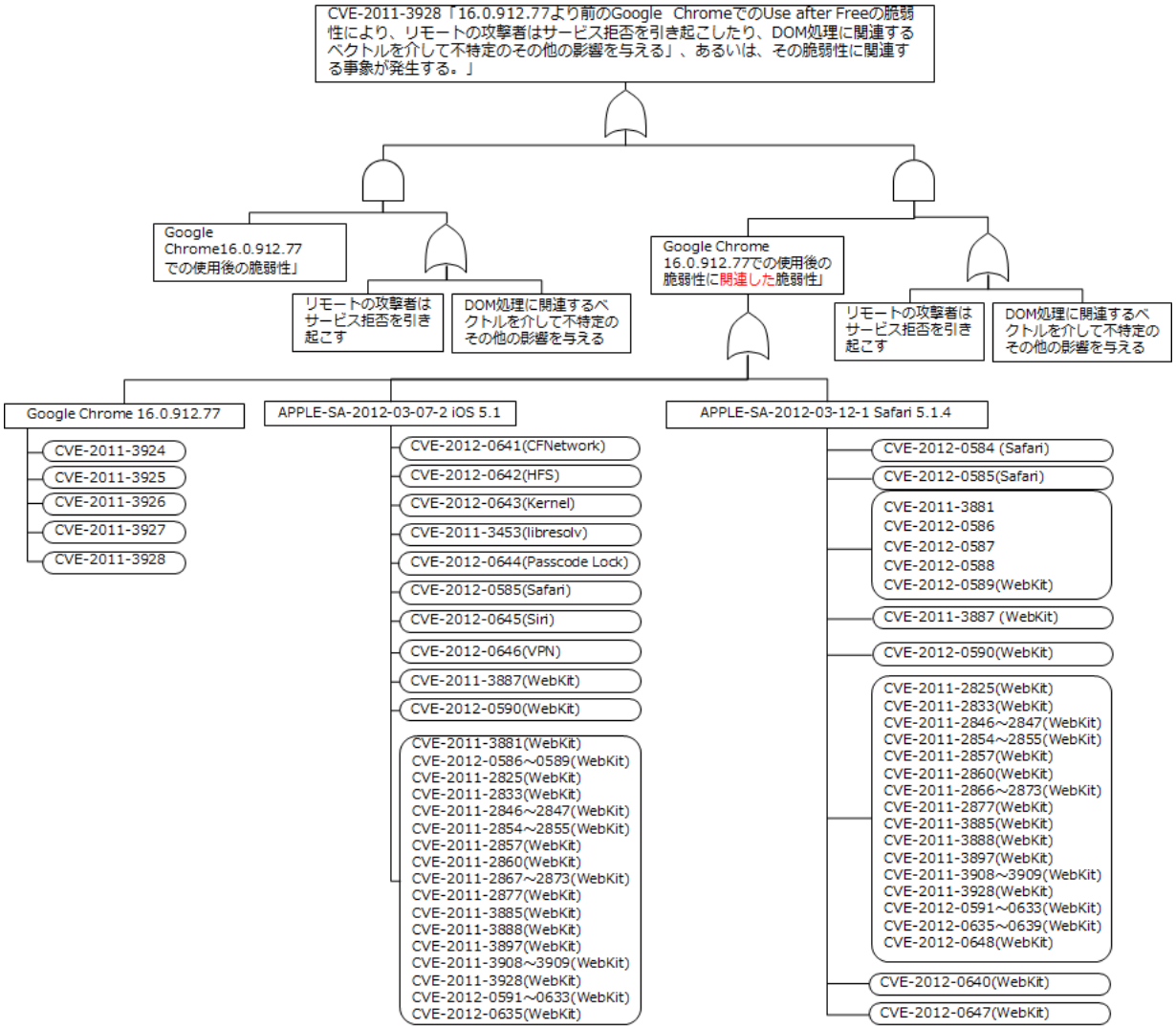

図 4: 第 1 の AT

$$P_{out}OR_i = \sum_{k=1}^{n} P_{in}(k,i) \qquad (12)$$

ただし，$P_{in}(k,i)$ は，n 個の入力を持つ i 番目のゲートへの k 番目の入力の確率である（$1 \leq k \leq n$）．

なお，文献 [8] では，論理ゲートへの入力がそれぞれ独立でない場合の計算式も示されている．

さらに，文献 [8] では，アタックツリーのトップイベント（アタックゴール）の確率を求めるために，オペレーションノードを AND ゲートとアサーションで書き換えることを提案している．具体的には，オペレーションノードを AND ゲートで置き換えて，元のオペレーションの記述をアサーションとして置き換えたうえで，AND ゲートへの入力とする．この置き換えによってアタックツリーからオペレーションの記述がなくなり，上述の式 (11) および式 (12) を順次計算することによってトップイベントの確率が計算できる．

6 テスラ事例への適用

4.4 節の提案アルゴリズムをテスラ社の具体的事例 [10] に適用する．

6.1 提案手法によるアタックツリーの作成

6.1.1 第 1 のアタックツリー AT^1 の作成

まず，脆弱性 DB およびテスラへの具体的攻撃事例 [10] をもとに第 1 の AT を作成する．第 1 の AT は全ての脆弱性について作成することになる．第 1 の AT の例を図 4 に示す．

6.1.2 第 2 のアタックツリー AT^2 の作成

次に，利用するアプリケーションに関連する脆弱性 DB および，コンポーネント DB を参照することによるコンポーネントに関連するアプリの脆弱性をもとに AT を構築する．第 2 の AT の例を図 5 に示す．利用している WebKit のバージョンをもとにコンポーネント DB を参照することで，Google Chrome の 13.0.782 以前の

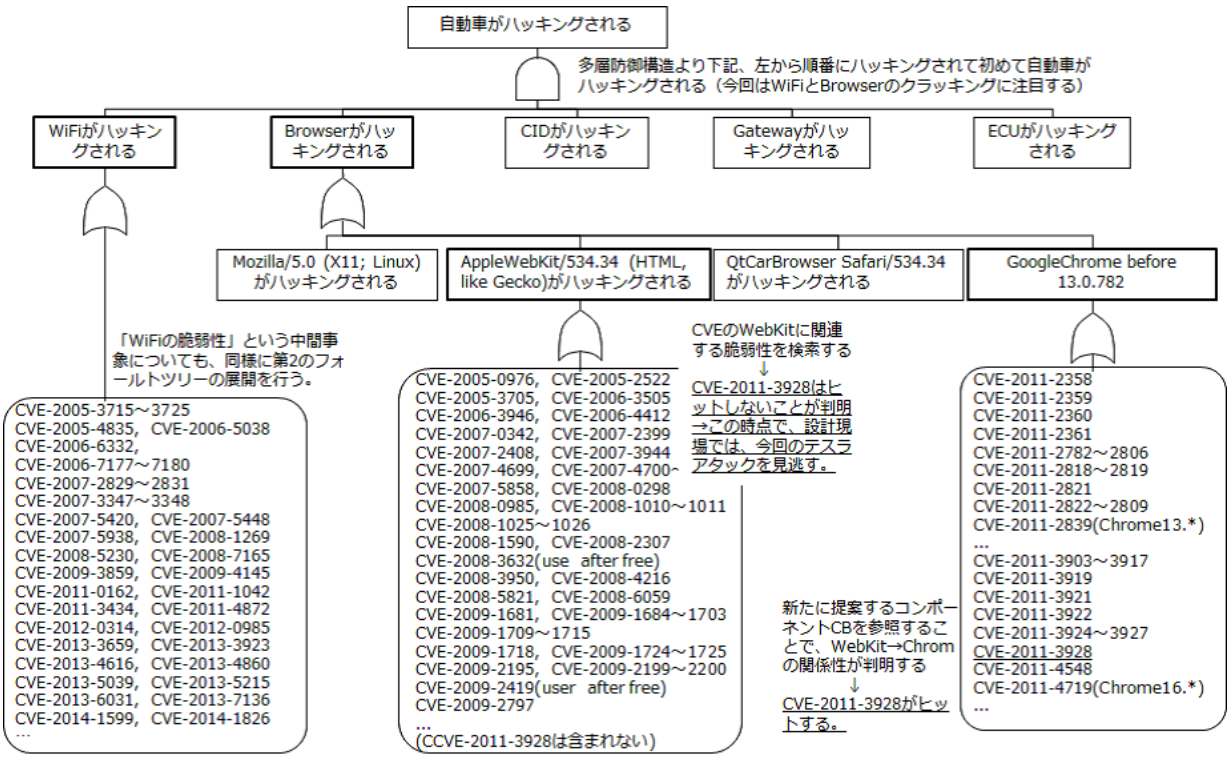

図 5: 第 2 の *AT*

バージョンに対する脆弱性も今回の *AT* の対象にすべきだということがわかり，そのノードを追加することで「CVE-2011-3928」の脆弱性が第 2 の *AT* 内に現れる．

次に第 2 の *AT* の「CVE-2011-3928」ノードと第 1 の *AT* のゴールノードを結合する．最後に今回のテスラのコンポーネントとして利用していないコンポーネントに対する脆弱性「CVE-20xx-yyyy」を第 2 の *AT* から削除する．

6.2 システムモデル記述

提案アルゴリズムを用いて作成した第 2 のアタックツリーのゴールノード「自動車がハッキングされる」の直下のノード（「WiFi がハッキングされる」から「ECU がハッキングされる」まで）におけるデータ（メッセージ）の入出力をシステムモデル記述を用いて分析する．

6.2.1 データフローダイアグラム

図 6 の上段にテスラの脅威分析対象システムのシステムモデル記述を示す．図 6 の上段はセーフティ検証で用いられるデータフローダイアグラムの記述結果である．分析対象のコネクティッドカーシステムは，4 個のハードウェアブロックと 5 個のソフトウェア機能モジュール，その他オペレーティングシステム（OS），Web ブラウザ，Web サーバ，各種ネットワークから構成される．ハードウェアブロックのうち，車両電子制御ユニット（ECU）と車両通信ゲートウェイ（GW），コンソールディスプレイ（CD）は，車両内部に搭載される．インターネット上の Web サーバ（WS）は，車両外部に設置され，Wi-Fi を介して車両のコンソールディスプレイと接続される．

まず，車両電子制御機能について説明する．コンソールディスプレイは，車両情報表示機能モジュールを実行する．車両情報表示機能モジュールは，運転者の指示を入力して，該当する車両制御コマンド（例えば，エンジンの動作状態のモニタリングや，ドアロックを解除するためのコマンドなど）を作成し，イーサネットで車両通信ゲートウェイに送信する．車両通信ゲートウェイの車両通信制御機能モジュールは，受信した車両制御コマンドを ECU コマンドにプロトコル変換して，CANBus を介して車両電子制御ユニットに送信する．車両電子制御ユニットの車両電子制御機能モジュールは，受信した ECU コマンドに対応するユニット内部制御ロジックを起動し，運転者が指示した車両制御を実行する．

次に，Web 情報表示機能について説明する．Web 情報表示機能モジュールは，運転者の指示を入力して，必要な Web ページ要求コマンド（例えば，http プロトコルの get コマンド）を作成し，Wi-Fi ネットワークを介して Web サーバに送信する．Web サーバの Web 情報サーバ機能モジュールは，受信コマンドを従って該当する Web ページを検索し，検索結果であるページ内容（例えば，HTML コンテンツや，Javascript コード）を要求コマンドの応答結果として返信する．

この段階で，脆弱性がないと仮定した場合のセーフティ分析が行われる．つまり，危険 (hazard) から安全機

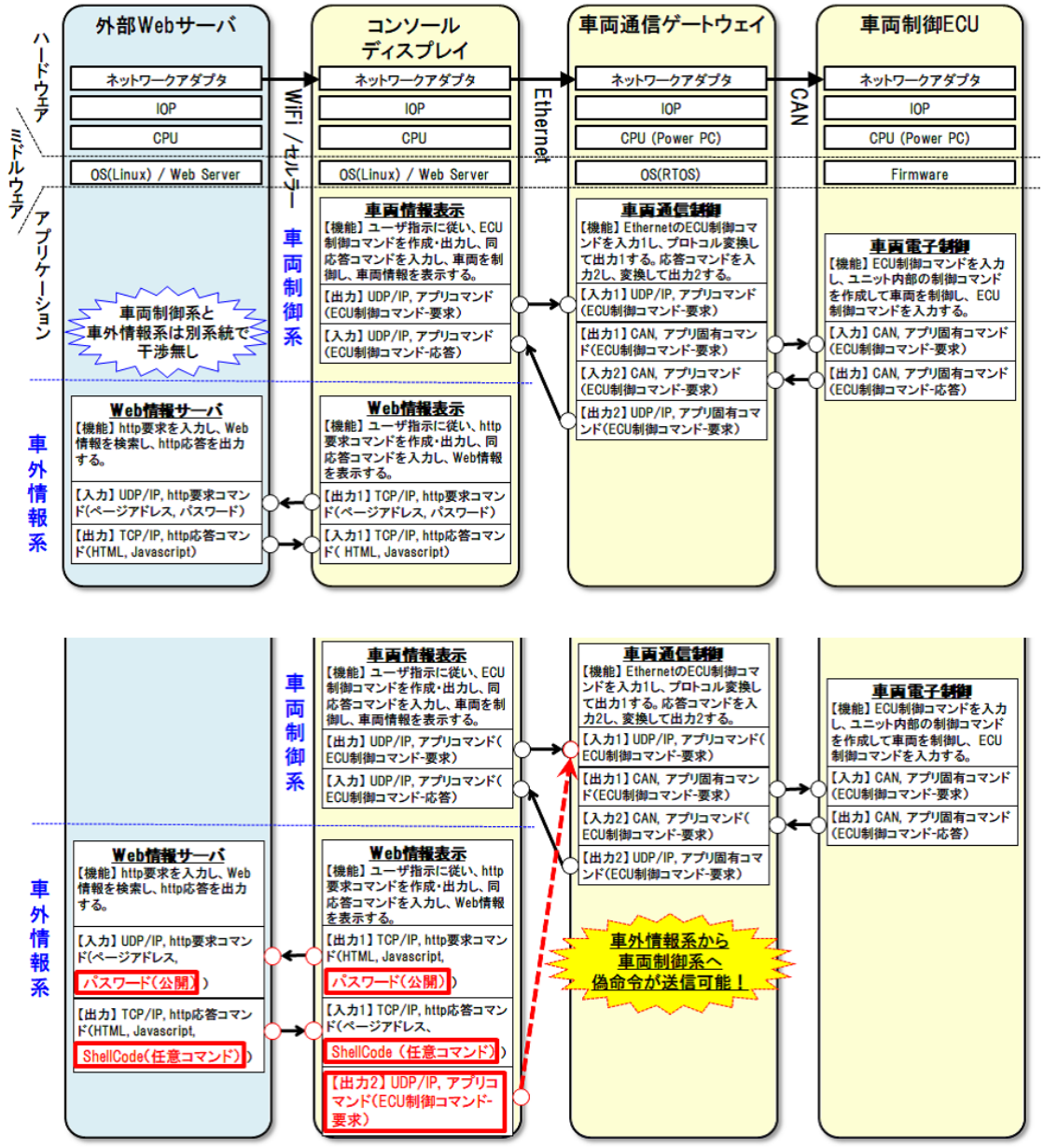

図 6: システムモデル記述

能を経由して検証までの追跡などセーフティの検証が行われる [9]．車両制御系と車外情報系は別系統で干渉はないことがわかる．

6.2.2 脆弱性の追記

次に，データフローダイアグラムに脆弱性による入出力を追記する．図 6 の下段は，上段の図に対して太枠四角で囲った属性と点線矢印を追加し，攻撃の可能性を含めた全体のデータフローダイアグラムである．

先ず,「Web ブラウザで任意シェルコード実行」に係る脆弱性モデル情報を説明する．脆弱性は，Web ブラウザで，アドレスリークと Use After Free の脆弱性により，任意のシェルコードが実行可能となる（CVE2011-3928 及び CVE2011-0154），という内容である．ここで CVE とは前述の通り，米国 MITRE 社が登録し公開している脆弱性事例情報データベースの採番であり，本件が実際に発生した脆弱性事例であることを示している [10]．

脆弱性モデル情報内の機能モジュール種別は，本脆弱性を有するソフトウェア機能モジュールが，Web ブラウザアプリケーションであることを示している．さらにそのモデルは，本脆弱性の条件として，OS が Linux(Version) であること，Web ブラウザが Webkit(Version) であること，プロセスモードがユーザモードであることを記述している．入力に関する条件としては，1 以上の入力のうち一つについて，通信プロトコルが TCP/IP であることと，通信内容が http 応答コマンド（HTML, Javascript, Shell code（任意コード））であることが記述されている．この段階で，脆弱性の属性が追加されたデータフローダ

イアグラムに対して，セーフティ分析が行われる．つまり，脅威 (threat) から安全機能を経由して検証までの追跡などセーフティの検証が行われる．結果として，車外情報系から車両制御系へ偽命令が送信可能であることが，データフローダイアグラム上で導かれた．偽命令により，走行中の車両においてドアを開閉すること等が可能となり，危険な状態となることが示された．

7 考察

今回テスラの事例でみたように，攻撃者は多層防御の外側から順番に WiFi への攻撃 (V1)，ブラウザへの攻撃 (V2)，コンソールディスプレイへの攻撃 (V3) を行っている．これらの V1 と V2，V2 と V3 は “コネクティッドカーの車両装備機能を異常動作させる” という攻撃のユースケース[2]において、組み合わせで出現する頻度が高い攻撃（以下,「共起性を有する攻撃」と呼ぶ）である．また，今回の論文では詳しく述べなかったが一昨年の Jeep チェロキーへの攻撃 [11] も携帯電話網への攻撃 (V1')，コンソールディスプレイへの攻撃 (V3') を行っており，V1' と V3' は共起性を有する攻撃である．テスラの事例やチェロキーの事例をもとに未知の事例に活用できるように（V1 → V2 → V3）や（V1' → V3'）などの共起性を有する攻撃は上位の概念レベルで記述する必要があると考える．このように複数の脆弱性を利用した一連の攻撃にはそれぞれの脆弱性を突いた攻撃に共起性がある．

また，本論文では，異なる攻撃対象（自動車）に対して同種の脆弱性は攻撃され易いという仮説に基づいた分析手法を提案している．実際にチェロキーの事例の一連の攻撃の連鎖がテスラの事例の攻撃の連鎖として表れている．このように本提案は，過去の事例を次の分析のために活用する方法であり，共起性を有する一連の攻撃を効率的に発見できる脅威分析システムと言える．

8 まとめ

本論文では，脆弱性事例データベースとシステム設計情報を活用したた脅威分析方法を提案した．具体手には，第 1 のアタックツリーおよび第 2 のアタックツリーの作成方法とそれたの数学的定式化を行い，さらに実事例を基にした脅威分析を行った．これにより，脅威分析者が分析対象システムの設計段階において，攻撃者の視点で攻撃目標を設定すれば，提案方法により攻撃の有無を判断できることを示した．

一般に，脆弱性を突いた攻撃について，同類のシステムで適用可能な抽象的かつ適切なメタ情報を作成することは，セキュリティに関する深い洞察力と労力を要する．しかし，本方法は，テスラの事例の事例の第 2 のツリーが今後の分析の第 1 のツリーとして利用できるように，適用事例を増やせば増やすほど有用なデータベースが充実する．よって本方法は，多くの設計者がセーフティとセキュリティの分析を行う上で有用なものになると考える．

[2] 攻撃のコンテキストと言い換えても良い．

謝辞

本研究の実施にあたり，後藤 厚宏氏，大崎 人士氏より貴重なコメントをいただき感謝いたします．本研究の一部は，総合科学技術・イノベーション会議の戦略的イノベーション創造プログラム（SIP）「重要インフラ等におけるサイバーセキュリティの確保」（管理法人：NEDO）によって実施されました．

参考文献

[1] Microsoft, “The STRIDE Threat Model,” https://msdn.microsoft.com/ja-jp/ee823878

[2] MITRE Corporation, “CWE List - Common Weakness Enumeration,” https://cwe.mitre.org/data/

[3] Antti Levomaki, Olli-Pekka Niemi, Christian Jalio, “Automatic Discovery of Evasion Vulnerabilities Using Targeted Protocol Fuzzing,” Briefing, Black Hat Europe 2017, Dec. 2017.

[4] SP 800-53 Rev. 5 “Security and Privacy Controls for Information Systems and Organizations”

[5] MITRE Corporation, “CVE - Common Vulnerability and Exposure,” https://cve.mitre.org/

[6] MITRE Corporation, “CAPEC - Common Attack Pattern Enumeration and Classification,” https://capec.mitre.org/

[7] A. Ruddle, et al.,“Deliverable D2.3: Security requirements for automotive on-board networks based on dark-side scenarios,” Seventh Research Framework Programme of the European Community, July 2008.

[8] I. N. Fovino, et al., “Integrating cyber attacks within fault trees,” Reliability Engineering and System Safety 94 (2009) p.p.1394–1402.

[9] Geoffrey Biggs, et al, “A profile and tool for modelling safety information with design information in SysML,” Software & Systems Modeling 15, 1 (Jan 2016), p.p.147–178.

[10] Sen Nie, et al., “FREE-FALL: HACKING TESLA FROM WIRELESS TO CAN BUS,” Briefing, Black Hat USA 2017, July 2017.

[11] C. Miller and C. Valasek. “Remote Exploitation of an Unaltered Passenger Vehicle,” Briefing, Black Hat USA 2015, pp.1–91, 2015.