受動的かつ不完全なセキュリティログ情報を用いたネットワーク構成情報検証手法

上川 先之[1,a)]　尾上 勉[2]　塩治 榮太朗[1]　芝原 俊樹[1]　秋山 満昭[1]

概要：マネージドセキュリティサービスは，顧客からプロキシログやIDSログに代表されるセキュリティログを受け取り，高度な分析により，その中に潜むセキュリティ脅威等を発見するサービスである．しかし，分析官が顧客のネットワーク構成に関して誤った情報を持っていることが多く，そのことが分析の効率や正確性を低下させる要因となっている．そこで，本稿では，受動的かつ不完全な情報であるセキュリティログのみを基に，分析官の想定するネットワーク構成を検証する手法を提案する．検証のアイディアは，セキュリティログの情報を正しいものとして論理による推論を行い，分析官の想定情報に含まれる間違いをセキュリティログとの矛盾として導き出すことである．提案手法では，不完全な情報を扱うこと，および推論規則を柔軟に表現する必要があることから，解集合プログラミングによって推論を行う．また，提案手法の応用により，ネットワーク構成情報の間違いを見つけられるだけでなく，ネットワーク構成の変化を検知することも可能になる．提案手法を検証システムとして実現するにあたって，基本的な実現可能性の検証のために実ログデータを用いた評価を行い，その結果について報告する．

Validation Method for Network Structure Information using Passive and Incomplete Security Logs

Hiroyuki UEKAWA[1,a)]　Tsutomu Ogami[2]　Eitaro Shioji[1]　Toshiki Shibahara[1]
Mitsuaki Akiyama[1]

Abstract: Managed Security Service analyzes security logs received from a client's network to search for hidden security threats. However, an analyst often makes incorrect assumptions about a client's network structure, negatively impacting the efficiency and accuracy of analysis. To solve this problem, we propose a novel method for validating the network structure assumed by an analyst, solely based on incomplete and passive security logs. Its key idea is to conduct logical inference based on the assumption that security logs are correct, and derive errors in the assumed network structure as contradictions against security logs. To meet the requirements for handling incomplete information and expressing flexible inference rules, we adopt answer set programming. Our method enables the discovery of not only errors, but also temporal changes, in network structure. Towards implementing our proposed method as a validation system, we conduct a basic feasibility evaluation using real logs and report its results.

1. はじめに

高度化するサイバー攻撃に対抗するため，検知・解析・対策などの一連のセキュリティ機能を集約することによって継続的かつ包括的に対象組織を監視・保護する部署であるセキュリティオペレーションセンタ（SOC）が多くの組織で導入されている．SOCでは，監視対象組織のネットワークや機器のログを収集し，未然防止のための対策を講じるとともに，常時監視とインシデントのリアルタイムな分析によって被害状況の分析および被害を最小化する取り組みが行われている．

多様な機器からログを収集して統合的に管理および分

[1] NTTセキュアプラットフォーム研究所
NTT Secure Platform Laboratories
[2] NTTセキュリティ・ジャパン株式会社
NTT security (Japan) KK
[a)] hiroyuki.uekawa.zb@hco.ntt.co.jp

析ための仕組みである SIEM（Security Information and Event Management）が SOC で活用されており，多様なログから攻撃の検知精度を向上させる研究が行われている [1–3]．しかし，SOC において実施されている高度な分析および対応の手順やその際に解決すべき技術的課題については十分に明らかになっていない．

そこで本研究では，我々は国内外の SOC に対して，組織に入り込んで行うフィールドワーク（実態調査）を実施した．このフィールドワークを通じて，SOC の現場で行われている脅威対処に際して監視対象のネットワーク構成情報が必須であることがわかった．また，ネットワーク構成情報は情報管理上の問題から提供組織から提供されない場合や，提供された情報が正確でない場合が多く，分析官が不完全なネットワーク構成情報から脅威の対処をしていることやセキュリティログからネットワーク構成を手動で推測していることがわかった．

さらに，フィールドワークによって明らかになった課題である分析官が行うネットワーク構成情報の推測をサポートするため，論理による推論をベースとしたネットワーク構成情報の検証手法を作成した．また，検証システムの実現に向けて，基本的な実現性を確認するために，いくつかの推論規則を考え，実際のセキュリティログを用いた評価を行った．評価結果として，期待通り推論されること，および期待通り検証できることを確認した．

本研究の貢献は以下の通りである．

- フィールドワークを通じて，SOC におけるネットワーク構成情報を得る際の課題を明らかにした．
- 分析官が想定するネットワーク構成情報が，セキュリティログと矛盾しないことを検証する手法を確立した．

2. フィールドワークを通じた SOC の理解

SOC は，社内 SOC とアウトソース SOC の 2 種類に大別される．社内 SOC は，保護対象の組織自身が内部で運用する SOC である．一方，アウトソース SOC は保護対象の顧客組織とは独立したマネージドセキュリティサービスを提供する組織が持つ SOC である．本研究では，マネージドセキュリティサービスとして広く普及しているアウトソース SOC を対象とする．以降では，特に説明がなければアウトソース SOC を単に SOC と呼ぶ．

一般的に SOC で取り組まれているセキュリティオペレーションやその課題を理解することの難しさとして以下が挙げられる．

- SOC では顧客の機密性が高い情報を取り扱うため，SOC 内部の情報を公開することが難しい．
- SOC における業務は高度で複雑なため，必ずしもドキュメント化されていない．

表 1 フィールドワーク概要

SOC	拠点	内容	期間・人数
SOC-A	国内	脅威分析業務の補助	3 ヶ月
SOC-A	国内	分析官へのヒアリング	4 人
SOC-B	海外	脅威分析作業	2 週間
SOC-B	海外	分析官へのヒアリング	3 人

そこで，我々は SOC に対して組織に入り込んで行うフィールドワークを実施することで，SOC の現場で行われている脅威対処のワークフローに加えて，脅威対処時の制約条件や課題を把握した上で，SOC で求められている技術を明らかにする．

2.1 フィールドワーク

人類学的研究（Anthlopology/Ethnograhic study）は，人間が組織内の様々なコンテキストに応じて行う活動についての本質的な理解を得ることを目的としており，代表的な手法として組織に長期間入り込んで調査を行うフィールドワークが知られている．また SOC に対してもフィールドワークに基づいて分析官の行動をモデル化する試みが行われている [4]．

本研究では，SOC における脅威への対応手順やツール活用など，SOC 外部から見ると暗黙的な “文化” について，正確かつ深く理解して現状の課題を明らかにすることを目的として，著者の一人は複数の SOC に対してフィールドワークを実施した．具体的には，国内外の二つの異なる SOC（以降，SOC-A と SOC-B と呼ぶ）に対して，具体的な脅威分析業務の補助を行うことで SOC における基本的なワークフローを理解し，また分析官と複数回にわたって議論を行うことで SOC の現場における共通的な課題の抽出を行なった（表 1）．フィールドワークは 2018 年 4 月から 2019 年 9 月の期間中に実施した．

SOC-A では，著者の一人は脅威が発生した際の分析・対応業務を行う分析チームに参画し，分析官が行うログ分析の補助を行った．SOC-B では，著者の一人は分析チームが分析するセキュリティログを共有され，分析・対応業務とは独立して模擬的にログ分析を実施した．

フィールドワークを実施した SOC および分析官に対しては，研究の主旨を理解してもらうとともに，知見を論文として公開することに対して了承を得ている．

2.2 SOC のワークフロー

フィールドワークを行った SOC の基本的なワークフローを説明する．アウトソース SOC では，顧客ネットワークの監視を行う．具体的には，IDS やプロキシに代表されるセキュリティ機器で観測されるトラヒックを監視する．一般的に，セキュリティ機器は，攻撃の検知漏れを防ぐため

および攻撃に関する通信ログを網羅的に記録するために，複数設置されている [5].

SOC では，トラヒックの監視を 24 時間体制で行っており，インシデントが発生した場合即座に検知して，顧客に通知する．監視の起点となるのは，IDS や SIEM のアラートである [6]．監視しているトラヒックからシグネチャに合致する通信が検知されると，アラートが SOC に送信される．SOC では，専門的な知識をもつ分析官が，アラートを起点にインシデントの全容を調査し，レポートを作成して顧客に送付する．マルウェア感染の場合，マルウェアの感染経路（メール，Web 等）や攻撃の進行度合い（攻撃失敗，マルウェア感染，C&C サーバとの通信確立等）を特定する．つまり，攻撃の影響範囲を把握するために関連する一連の通信内容およびホストを特定する必要がある．調査結果はレポートとして顧客に送付され，顧客によるインシデントへの対応とセキュリティ対策の見直しに活用される．

2.3 ネットワーク構成情報に基づくアラート分析と課題

アラート分析は，監視してるネットワークの構成を意識して行う必要がある．例えば，マルウェア感染の場合，迅速なインシデント対応のために感染したホストの IP アドレスを特定することが重要である．しかし，顧客のネットワークでプロキシが使用されていた場合，アラートとして検知された通信の IP アドレスは，プロキシの IP アドレスの可能性もある．このため，プロキシが顧客ネットワークで使用されているかを意識しながら分析する必要がある．さらに，IP アドレスがプロキシであった場合，ホストとプロキシの間の通信を記録している機器があるかが，ホストの IP アドレスを特定できるかに影響する．つまり，ネットワークのどの位置にどのような機器が設置されているかによって分析の手順が異なるため，効率的に分析を行うためには顧客のネットワーク構成に関する情報が必要である．これらの理由から，SOC では，アセット管理（監視しているネットワークに接続されている機器やネットワーク構成情報の把握）を行っており，アラート分析時には，アラートやセキュリティ機器のログとアセットの情報とを組み合わせて脅威の影響範囲を特定する．

ネットワーク構成情報はアラート分析に重要であるが，SOC でアセット情報の管理を完璧に行うことは非常に難しい．その結果，分析の効率や正確性を低下させる場合がある．アセット管理が難しいのは，アウトソース SOC が他組織のネットワークを監視していることが主な要因である．顧客はマネージドセキュリティサービスを利用する際，監視対象のセキュリティ機器の情報を契約に従って SOC に共有するが，セキュリティ機器以外の顧客ネットワークに接続されている機器の情報やネットワーク構成情報までは共有されない．また，SOC は顧客ネットワークを遠隔で

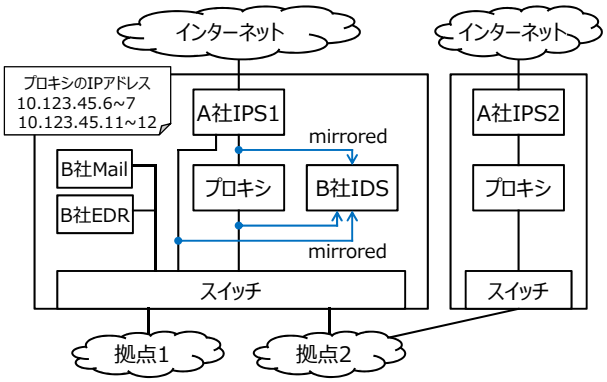

図 1 分析官が作成するネットワーク図のイメージ

監視しているため，現場に赴いてネットワーク構成の実態を確認することもできない．そのうえ，SOC では，能動的に顧客ネットワークの端末を操作したりスキャンしたりすることが契約上できないため，能動的なネットワーク構成推定技術 [7] を適用することもできない．

2.4 SOC での取り組み

SOC の現場では，顧客のネットワーク構成を把握するために，2 つの取り組みを定期的に行っている．さらに，アラートが発生してから初めて確認することができる機器もあるため，アラート分析時に新たに確認できた機器の情報もアセット管理にフィードバックしている．

顧客からの情報提供 一つ目の取り組みは，顧客に問い合わせてネットワーク図等の情報を提供してもらうことである．大抵の場合，顧客が運用しているネットワークに関する物理構成図や論理構成図が存在する．それらを入手できれば，どのような機器がネットワークに接続されているかや，セキュリティ機器のネットワークのどこに設置されているかを知ることができる．

この取り組みでも，分析官が必要としてるネットワーク構成情報が入手できない場合がある．主な理由は，このような図や情報は社外秘として扱われているため，契約やルールに厳しい顧客からは提供してもらえないからである．入手できた場合でも，ネットワーク図はネットワーク設計者の目線で作られているため，冗長構成が省略されてる場合や，セキュリティ機器が記載されていない場合がある．

顧客から入手したネットワーク図等の情報は，正確性に問題があることもある．具体的には，入手したあとにネットワーク構成が変更された場合や，入手したネットワーク図が古かった場合である．つまり，ネットワーク構成情報を顧客から入手できても，情報が必ずしも正しいとは限らない．

ログ情報からの推測 二つ目の取り組みとして，分析官は入手可能な情報であるセキュリティログを活用して，独自にネットワーク構成を推測して図を作成している．**図 1** に

簡略化したイメージ図を示す．SOC での分析では，各種通信がセキュリティ機器をどのように経由するかが重要なため，セキュリティ機器の配置や接続関係が主として書かれている．

セキュリティログに基づいてネットワーク構成を正確に推測することは簡単ではない．1 つ目の理由は，セキュリティログにネットワーク構成の推測に必要な情報がすべて含まれているとは限らないからである．セキュリティログに記録されていない機器があった場合，その機器に関する情報を推測することは不可能である．つまり，セキュリティログは基本的に情報として不完全で，すべてのネットワーク構成情報を正確に推定するには不十分である．2 つ目の理由は，実際のネットワーク構成が，分析官の想定するような単純な構成でない場合があるからである．単純な例としては，多段プロキシが挙げられる．顧客企業の運用上の都合やプロキシに持たせる役割によって，多段構成になる場合がある．しかし，プロキシは，もたせる役割を考慮しなければ多段にする理由がないため，顧客側の都合や理由を知らない分析官によるネットワーク構成の推測を難しくしている．3 つ目の理由は，IDS などの IP アドレスを持たないセキュリティ機器の存在である．このような機器は，ログに記録されている IP アドレスから，ネットワークにおける位置を特定することができず，ネットワーク構成の推測を難しくしている．これらの理由から，分析官が正確にネットワーク構成を推測することは非常に難しい．

2.5 本研究で取り組む課題

前節で説明した通り，SOC の分析官は顧客から入手するかセキュリティログに基づいて推測することで，ネットワク構成情報を入手することができる．しかし，これらの方法で入手された情報が，必ずしも正しいという保証はない．分析官がアラート分析に使用するネットワーク構成情報が間違っていた場合，分析の効率低下・顧客でのインシデント対応の遅れの原因となる可能性がある．例えば，感染端末の通信経路を誤って想定して分析することで，感染経路の特定に非常に時間がかかったり，プロキシの IP アドレスを感染端末の IP アドレスとして顧客に通知することで，感染端末の隔離に時間がかかったりする可能性がある．そこで，本稿では，誤ったネットワーク構成情報に基づく分析を防止するために，SOC が管理しているネットワーク構成情報が妥当か判定する手法を検討する．

SOC におけるアセット管理に関する課題は，SOC の課題をインタビューを通じて調査した研究 [8] でも報告されており，フィールドワークを実施した SOC だけの課題ではなく，多くの SOC に共通の課題である．この研究では，課題の抽出は行っているが，その原因の調査や解決方法の検討は行われていない．一方，本研究では，フィールドワークを通じた課題の本質的な原因および SOC の現状の調査と，調査結果に基づいた効果的な解決方法の検討を行った．

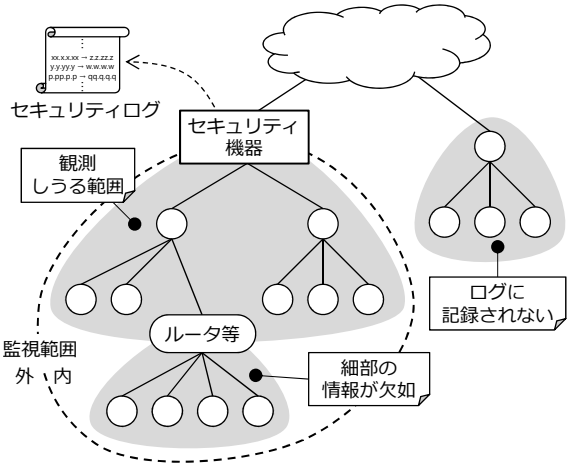

図 2 ネットワーク構成情報の範囲

3. ネットワーク構成情報の検証手法

本稿では，2 章で明らかにした SOC が管理しているネットワーク構成情報が必ずしも正しいとは限らないという課題を踏まえ，ネットワーク構成情報の検証手法を提案する．提案手法で分析官の想定しているネットワーク構成情報（以降，想定情報）の間違いを見つけることで，SOC が管理するネットワーク構成情報の正確性を向上させることができる．その結果，誤った想定に基づく分析が減少し，分析効率の向上が期待される．

SOC では，2.3 節で説明した通り，スキャン等の能動的な手法を適用できないため，受動的なアプローチを採用する．具体的には，SOC で入手可能なセキュリティログを用いて，想定情報に誤りがないかの検証を実施する．

3.1 検証対象のネットワーク構成情報

提案手法では，セキュリティログを用いて検証を行うため，ログに含まれている機器の構成情報が検証対象となり，それ以外は検証対象外となる．対象となる構成情報の範囲について，**図 2** を用いて説明する．まず，セキュリティ機器の監視範囲外は，そもそもログに記録されることがない．そのため，監視範囲外のネットワーク構成情報は対象外として扱う．セキュリティ機器の監視範囲内でも，情報の欠如が原因で検証ができない場合もある．例えば，マルウェア感染等のセキュリティイベントが発生した箇所とセキュリティ機器の間にプロキシや NAT を構成するルータがある場合，セキュリティ機器側から観測される IP アドレスはプロキシやルータのものになる．このように，プロキシやルータによって欠落した構成情報は検証不可能である．

上述の通り，提案手法では，セキュリティログに記録されているネットワーク構成情報を対象として，検証を実施する．対象の構成情報を検証できれば，SOC でのアラート分析に必要な構成情報の検証としては十分である．まず，

セキュリティ機器から観測できない範囲は，SOC の監視対象外であるため，分析官がネットワーク構成を知る必要がない．次に，欠落する情報については，アラート分析でも対象外となるため，この範囲のネットワーク構成が分析に影響することはない．

3.2 論理に基づく検証

提案手法では，ネットワーク構成情報の検証をセキュリティログを用いて実施する．SOC で正確なネットワーク構成情報を入手することができれば，それを教師データとして機械学習を適用することも可能である．しかし，2 章で説明した通り，SOC で正確なネットワーク構成情報を入手することはできないため，機械学習を適用することはできない．さらに，機械学習では構成情報の誤りを検知できても，検知の根拠を分析官が理解しやすい形式で提示することができないという問題もある．

本稿では，ネットワーク構成情報とログに食い違いがあるか確認することで検証を行う．ネットワーク構成情報とログは情報の粒度が異なるため直接比較することはできない．このため，既知の情報から新たな情報を導出しつつ，矛盾が存在するか判定可能な論理に着目する．新たな情報の導出は，ネットワークに関する常識を定式化することで実現できる．この定式化によって，セキュリティログからネットワーク構成情報を導出することで，想定情報との比較が可能となる．この定式化は，分析官によるネットワーク構成情報の推測と同等であが，論理に基づく検証では大量のセキュリティログを入力できるため，分析官の推測した構成情報とログとの矛盾を導出できる．ただし，提案手法では，入力されたセキュリティログとネットワーク構成情報に矛盾が存在するか判定しているため，検証する構成情報に関連するセキュリティログが不十分な場合矛盾が発生しない．

論理による検証の具体的な手順について述べる．まず，想定情報とセキュリティログをそれぞれ述語（論理式）に変換する．述語は，$\neg Proxy(10.1.2.3)$, $Located(10.1.2.4, DMZ)$ のような式で表現し， それぞれ「IP アドレス 10.1.2.3 はプロキシでない」「IP アドレス 10.1.2.4 は DMZ セグメントに位置する」という意味を持たせたものである．次に，それらの述語に対し，ネットワークの仕組みや常識等を推論規則として推論を行う．推論規則の例として，「ある IP アドレスのノードについて，TCP/8080 番ポートで HTTP リクエストを受けているならば，そのノードはプロキシである」といった例が考えられる．この推論規則を，「IP アドレス 10.1.2.3 の TCP/8080 番ポートへの HTTP リクエスト」を含むセキュリティログに適用すると，推論結果として，「IP アドレス 10.1.2.3 はプロキシである」が導出される．分析官が「IP アドレス 10.1.2.3 はプロキシでない」と想定していた場合，これも含めて推論を行うと矛盾が導出される．つまり，想定情報とセキュリティログから変換された述語をもとに推論を実施し，その結果矛盾が導出された場合，ログ情報を正しいものとすれば想定情報に間違いが含まれることがわかる．一方，矛盾が導出されなかった場合には，想定情報は妥当ということができる．

3.3 解集合プログラミング

提案手法には，セキュリティログの特性を考慮して解集合プログラミングを適用する．これは，不完全なデータの推論に適したデフォルト推論を行うことができる枠組みである．セキュリティログは，偶発的なセキュリティイベントを記録したものであり，情報として不完全である．つまり，セキュリティ機器から観測しうる範囲のすべてのネットワーク構成情報が出揃うことはまずない．デフォルト推論は，「クライアントであることが否定されなければ，クライアントとみなす」といった推論規則を考えることで，不完全なデータの推論を可能としている．より厳密には，「ある事柄が否定される根拠がなければ」という条件を推論規則で考慮している．デフォルト推論を実現するためには，非単調推論が行えることと，2 つの否定の区別ができることが必要である．解集合プログラミングは，これらの条件を満たす推論を行うことができる．

非単調推論 デフォルト推論では，新たな情報を加えた際に，それまでの知識の一部を否定するケースが考えられる．例えば，ある IP アドレスについて，それは「クライアントである」と考えているとする．ここで，新たに「TCP/8080 番ポートで HTTP リクエストを受けている」という情報を得た場合，それまでの「クライアントである」という知識を否定して「プロキシ」であると考え直すという状況が発生する．このように，新たな知識によってそれまでの知識が覆される可能性がある推論を，非単調推論と言う[*1]．

2 種類の否定の区別 非単調推論には，明示的な否定とデフォルトの否定の 2 種類がある．それぞれ「○○でない」と「○○とは言えない」と表現できる．これらを端的に説明すると，それぞれ「偽 (false)」と「真でない (not true)」であり，両者は等価ではない．「ある事柄が否定される根拠がなければ」というデフォルト推論で用いられる条件を実現するためには，これら 2 種類の否定を使い分けて定義する必要がある．

4. 検証システムの実現に向けて

4.1 検証手順の構築

3 章での検討を踏まえ，ネットワーク構成検証手法を実現する検証手順を具体的に構築した（**図 3**）．入力として，アイディアの通りセキュリティログと想定情報をそれぞれ述語に変換する．想定情報については，分析官など検証す

[*1] 厳密な定義ではない

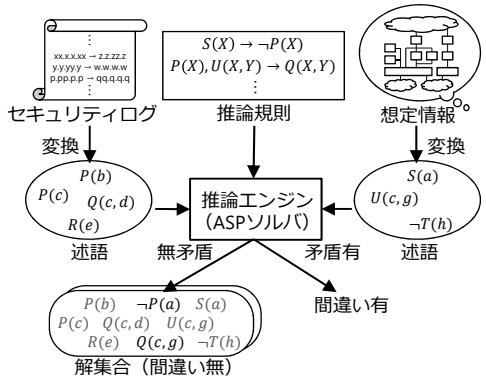

図 3 検証手順

る者が書き出す必要がある．

2 つの入力による述語とあらかじめ定義した推論規則を合わせて，これを推論エンジンで処理する．その結果，矛盾するか否か，および矛盾しない場合はいくつかの解集合が得られる．本研究では，推論エンジンとして解集合プログラミングのソルバである clingo [10] を使用する．

推論の結果として，矛盾の有無を根拠に，想定情報に間違いが含まれるか否かがわかる．また，無矛盾の場合は，ネットワーク構成のあり得るパターンがいくつかの解集合（述語の組み合わせ）として得られる．

4.2 ユースケース

前節で述べた検証手順を実現する検証システムとしてのユースケースについて述べる．本システムは，想定情報の間違いを見つけるという本来の目的どおり，既知の情報や不明確な情報を検証するためのツールとして活用できる．例えば，分析官が分析結果を報告する際，被害対象端末の IP アドレスを報告書に記述する．報告書の中でもこの IP アドレスは非常に重要な情報であり，分析官は細心の注意を払って記述している．このようなときに本システムを使い，その IP アドレスより下流の情報がないことを調べるなど，最終確認として検証するという使い方ができる．

一方，検証システムは本来の目的と異なる使い方もできる．具体的には，ネットワーク構成が変化した際に，その変化を検知することができる．例えば，Web プロキシが 2 台だけ存在することがわかっているとする．このとき「Web プロキシの数は 2 台である」という既知情報を想定情報として入力する．ここで，Web プロキシの台数が増え，ログ情報から 3 台存在することが推論された場合，既知情報との矛盾が生じる．この矛盾をもって，ネットワーク構成が変化したと言うことができる．

また，既知情報を入力せずとも，推論規則によってはログ情報だけで矛盾することが考えられる．このような場合も，入力したログの期間内で構成が変わったと言える．

```
conn("172.20.7.18","10.252.129.20",1447500409).
conn("10.102.1.1","10.252.129.20",1447500410).
xff("10.200.1.11","10.102.1.1",1447500410).
conn("10.102.1.1","10.252.129.20",1447500411).
conn("10.102.1.1","10.252.129.20",1447500412).
xff("10.200.0.223","10.102.1.1",1447500412).
```

図 4 述語に変換したログ（一部）

```
1 xff(Client,Proxy) :- xff(Client,Proxy,SeqNo).
2 proxy_est(Proxy) :- xff(Client,Proxy).
3 proxy(Proxy) :- proxy_est(Proxy).
```

図 5 プロキシを導出する推論規則（XFF ヘッダ基準）

```
proxy("10.102.1.1") proxy("10.252.129.20") proxy
    ("10.101.1.1")
SATISFIABLE
```

図 6 XFF ヘッダ情報から導出されたプロキシ

4.3 評価：推論による導出

解集合プログラミングで矛盾の有無を検証するには，分析官による推論を推論規則として表現する必要がある．本節では，プロキシに関するいくつかの推論について具体的に推論規則を定義し，企業の実際のログに適用して，期待する述語が導出できることを確認する．使用するログは，ある企業の Palo Alto Networks 製ファイアウォール機器 1 台による 1 日分（3,132,384 行）の URL ログ[*2]である．なお，ログの URL 内に含まれるセンシティブな情報（企業名やメールアドレス等）は除外しており，また個人が一意に識別されるような分析は行なっていない．

まず，ある IP アドレスがプロキシであることを解集合プログラミングで導出できることを確認する．URL ログには，X-Forwarded-For（XFF）ヘッダ[*3]の情報が含まれるため，これを利用する．XFF ヘッダを持つ HTTP 通信の直接的な送信元は，通常はプロキシである．このことを踏まえて，**図 4** のようにログから必要な述語を作ることで，推論規則として**図 5** のように表現できる．これらを解集合プログラミングのソルバ clingo に入力することで，導出された結果が解集合として得られる（**図 6**）．3,132,384 行のログから 632 個の述語 `xff(C,P)` と 3 個の述語 `proxy(X)` が導出され，結果から 3 つの IP アドレスがプロキシとして推論されたことがわかる．XFF ヘッダ情報のあるログの送信元 IP アドレスをログから直接抽出したところ，結果が一致したため，期待通りに推論が行われたことが確認できた．この例では，XFF ヘッダを持つ通信の送信元 IP アドレスをログから直接述語 `proxy(X)`. に変換することもできるが，あえて推論規則で処理することで解集合プロ

*2 HTTP 通信を検知し，送信元・送信先の IP アドレス・ポート番号やパケットから再構築した URL，一部の HTTP ヘッダ情報などが記録される．

*3 プロキシが付加する HTTP ヘッダ．

```
1 dst(X) :- conn(_,X,_).
2 dst_cnt(X,N) :- N = #count{S:conn(_,X,S)}, dst(X).
3 threshold(1).
4 threshold(N) :- dst_cnt(_,N), N >= #max{P*5:
      dst_cnt(_,P), P < N} > 2.
5 proxy(X) :- dst_cnt(X,N), N >= #max{M:double_score
      (M)} >= 100, not -proxy(X).
```

図 7 プロキシを導出する推論規則 (通信数基準)

グラミングの柔軟な表現力を活かした推論ができるようになり，また，ログの変換処理が煩雑になるのを抑えることができる．

SOC の現場では，通信の量を考慮した推定を行う場合もある．ある特定の IP アドレスの Web 通信数が他の IP アドレスと比べて明らかに多い場合，その IP アドレスはプロキシなどのように複数端末の通信を集約しているものと考えて分析を行うことがある．解集合プログラミングでは，このような数を考慮する推論規則も表現できる．例えば，URL ログに記録されている通信先 IP アドレスごとにログ件数を数え，他の IP と比べて 5 倍以上多いとき，その IP アドレスをプロキシとみなすという推論規則を考える．この推論規則は，**図 7** のように表現できる．この推論規則によって，通信数上位 5 件は 2,102,221 件，16,728 件，15,413 件，12,626 件，12,462 件と数えられ，5 倍以上の差が開いた上位 1 件が `proxy("10.252.129.20")` として導出された．ただし，この推論規則は XFF の例と比較して確度が低く，プロキシであると言い切ると不都合な場合がある．このことを踏まえて，導出ルールの前提として 5 行目のように `not -proxy(X)` を指定することで，デフォルトでは導出するがプロキシであることを否定しても矛盾しないルールを表現できる．このように，解集合プログラミングでは，確度の低い推論を真偽がはっきりと決まる論理の枠組みの中で柔軟に扱うことができる．

本節の例では XFF ヘッダによる導出と通信数による導出で独立に推論を行ったが，両者は共存可能である．さまざまな推論規則を定義していくことで，分析官による推論のノウハウが蓄積され，また，複数の分析官の間でノウハウが共有されるという側面もある．

4.4 評価：想定情報の検証

企業の実際のログを用い，想定情報を提案手法で正しく検証できることを確認する．ログは，前節と同じ URL ログを使用する．

想定情報の検証ができるか否かを確認するには，制約としてのルールかまたは入力する想定情報を否定する述語を導出できるルールが必要である．例えば，前節で説明した XFF ヘッダ情報からプロキシであることを導出する推論規則（図 5）の場合，ある IP アドレスがプロキシでないという想定情報と矛盾する可能性はあるが，プロキシであるという想定情報と矛盾することはない．

```
1 proxy_cnt(N) :- N = #count { X:proxy_est(X) }.
2 :- proxy_cnt(N), proxy_cnt(M), N != M.
```

図 8 プロキシの台数で矛盾を導出する推論規則

そこで，まずは図 5 の推論規則でプロキシであると導出された IP アドレスについて，それを誤ってプロキシでないと想定した（`-proxy("10.252.129.20").`[*4]）として検証を行った．検証結果は矛盾であることを示す `UNSATISFIABLE` の出力となり，想定情報が間違いであることがわかる．この結果より，矛盾の導出による想定情報の検証が可能であることが確認できた．

解集合プログラミングでは述語の数を数えることができる．そこで，プロキシの想定台数と推論された数が異なる場合に矛盾を導出できることを確認する．推論規則は**図 8** のように表現でき，1 行目が `proxy_est(X)` の数を数えるルール，2 行目で数が異なる場合に矛盾を導出する制約ルールである．図 6 で示した通り，XFF ヘッダ情報から 3 つの IP アドレスがプロキシとして推論される．この推論規則を用いて，誤ってプロキシの台数が 2 台であると想定した（`proxy_cnt(2).`）として検証した．その結果，期待通り矛盾（`UNSATISFIABLE`）となり，数に関する推論による検証が可能であることが確認できた．

5. 議論

5.1 実用的な推論規則に向けて

本稿では，検証の基本的な仕組みの提案を中心に行なったため，それが利用する推論規則自体についての踏み込んだ議論や検証は範囲外とした．しかしながら，本提案手法の検証能力が蓄積された推論規則の質や量に大きく依存することは明らかであり，推論規則をどのようにして充実させていくかは，提案手法を実現する検証システムの実用性を高める上での重要な課題の一つである．前節で例示した単純な推論規則は分析官へのヒアリングに基づいて著者が作成したものであるが，分析官自身が規則を直接記述できるような仕組みを整備することで，推論規則を効率的に拡充することができる．それにあたり，例えば柔軟な推論規則を容易に記述するための記述言語を整備することが考えられる．また，ログや推論規則の増加・複雑化を踏まえた，検証システムとしてのパフォーマンスの検証も必要である．述語の数に応じて検証処理が重くなることが予想されるため，実際の実用的な目的に即した推論規則の性質の解明や，効率的な推定アルゴリズムの検討が必要である．

5.2 今後の発展

本検証システムの基幹部分となる推論システムは，ネッ

[*4] マイナス記号が否定を表す．

トワーク構成の誤りの有無の検出に限らず，多くの発展的な応用が考えられる．一例として，ネットワーク構成の間違っている箇所の特定が考えられる．分析官は，最終的に想定情報の誤りを修正する必要があるため，具体的にどの部分の情報が間違っているのかまで含めて提示できることが望ましい．本提案手法は推論に基づいているという仕組み上，矛盾の原因となった述語を自動的に特定することは難しくない．また，より発展的な例として，ある程度の網羅的な推論規則が蓄積されていることが前提となるが，具体的なネットワーク構成の仮定と検証を繰り返すことにより，ネットワーク構成を自動的に推定する方式についても今後検討を進める予定である．

6. 関連研究

ネットワーク構成情報に関する研究としては，構成情報の推定を目的とした研究が行われている．これらの研究は，推定対象（WAN または LAN）および推定方法（能動的または受動的）の観点から 4 つに大別される．

WAN の能動的な推定では，`traceroute` のデータを活用したルータレベルでのインターネットの接続関係の特定 [11] や，AS の接続関係の推定 [12] が行われている．WAN の受動的な推定では，BGP メッセージの観測に基づいた AS 内でのネットワークトポロジ推定 [13] や，IP パケットの観測に基づいたネットワークトポロジ推定 [14] が行われている．LAN の能動的な推定では，組織内のネットワーク環境を内部から能動的にスキャンを行い，ネットワーク図を作成する技術がすでに商用化される [7]．これらの研究は，本研究と推定対象または推定方法が異なっているため，SOC におけるネットワーク構成情報管理には適用することができない．LAN の受動的な推定では，セキュリティログを用いてネットワークトポロジの推定を行う手法が提案されている [15]．しかし，この研究の推定範囲は，ネットワーク的に下流の部分（図 1 の “拠点” とされている部分）であり，本研究の推定対象とは異なっている．

本研究では，ネットワーク構成の推定ではなく，ネットワーク構成情報の検証を実施したが，我々の知る限りでは，ネットワーク構成情報の検証を目的とした研究は行われていない．

7. おわりに

本研究では，フィールドワークにより明らかにした課題を解決するため，論理による推論をベースとしたネットワーク構成情報の検証手法を作成した．検証システムの実現に向けて，基本的な実現性を確認するための評価を行った．具体的には，プロキシに関するいくつかの推論規則を考え，それぞれ期待通り推論が行われることを確認した．さらに，いくつかの誤った想定情報を考え，それぞれ期待通り正しく検証できていることを確認した．

また，検証システムとしての実用性を高める上で必要な推論規則の充実について議論した．今後，誤っている情報の特定および提示を含め，ネットワーク構成の推定方式について検討を進める予定である．

参考文献

[1] Oprea, A., Li, Z., Norris, R. and Bowers, K.: MADE: Security Analytics for Enterprise Threat Detection, *Proc. ACSAC'18*, pp. 124–136 (2018).

[2] Yen, T.-F., Oprea, A., Onarlioglu, K., Leetham, T., Robertson, W., Juels, A. and Kirda, E.: Beehive: Large-Scale Log Analysis for Detecting Suspicious Activity in Enterprise Networks, *Proc. ACSAC'13*, pp. 199–208 (2013).

[3] Ho, G., Sharma, A., Javed, M., Paxson, V. and Wagner, D.: Detecting Credential Spearphishing Attacks in Enterprise Settings, *Proc. SEC'17*, pp. 469–485 (2017).

[4] Sundaramurthy, S. C., McHugh, J., Ou, X., Wesch, M., Bardas, A. G. and Rajagopalan, S. R.: Turning Contradictions into Innovations or: How We Learned to Stop Whining and Improve Security Operations, *Proc. SOUPS'16*, p. 237–251 (2016).

[5] Chen, S.-T., Han, Y., Chau, D. H., Gates, C., Hart, M. and Roundy, K. A.: Predicting Cyber Threats with Virtual Security Products, *Proc. ACSAC'17*, pp. 189–199 (2017).

[6] Roundy, K. A., Tamersoy, A., Spertus, M., Hart, M., Kats, D., Dell'Amico, M. and Scott, R.: Smoke detector: cross-product intrusion detection with weak indicators, *Proc. ACSAC'17*, pp. 200–211 (2017).

[7] SolarWinds: Network Mapping Software, available from ⟨`https://www.solarwinds.com/network-topology-mapper`⟩ (accessed 2019-06-05).

[8] Kokulu, F. B., Soneji, A., Bao, T., Shoshitaishvili, Y., Zhao, Z., Doupé, A. and Ahn, G.-J.: Matched and Mismatched SOCs: A Qualitative Study on Security Operations Center Issues, *Proc. CCS'19*, pp. 1955–1970 (2019).

[9] 坂間千秋, 井上克巳: 解集合プログラミング, 人工知能学会誌 特集「論理に基づく推論研究の動向」, Vol. 25, No. 3, pp. 368–378 (2010).

[10] clingo and gringo — Potassco, the Potsdam Answer Set Solving Collection, The University of Potsdam, available from ⟨`https://potassco.org/clingo/`⟩

[11] Govindan, R. and Tangmunarunkit, H.: Heuristics for Internet map discovery, *Proc. INFOCOM'00*, Vol. 3, IEEE, pp. 1371–1380 (2000).

[12] Chang, H., Jamin, S. and Willinger, W.: Inferring AS-level Internet topology from router-level path traces, *Scalability and traffic control in IP networks*, Vol. 4526, International Society for Optics and Photonics, pp. 196–207 (2001).

[13] Andersen, D. G., Feamster, N., Bauer, S. and Balakrishnan, H.: Topology inference from BGP routing dynamics, *Proc. IMW'02*, pp. 243–248 (2002).

[14] Eriksson, B., Barford, P. and Nowak, R.: Network Discovery from Passive Measurements, *Proc. SIGCOMM'08*, pp. 291–302 (2008).

[15] Azodi, A., Cheng, F. and Meinel, C.: Event Driven Network Topology Discovery and Inventory Listing Using REAMS, *Wirel. Pers. Commun.*, Vol. 94, No. 3, pp. 415–430 (2017).