Vol.2019-IOT-46 No.2
2019/6/14

# セキュリティ事案における知見の蓄積・活用を可能とする対応フローの提案と実装

森 健人[1,a)]　石井 将大[1,b)]　松浦 知史[1,c)]　金 勇[1,d)]　北口 善明[1,e)]　友石 正彦[1,f)]

**概要**：昨今 CSIRT において対応すべきセキュリティインシデントは絶えることなく発生し，担当者は日々膨大なアラートの処理に追われながらイベントの調査・解析，連絡対応，報告書作成など多岐にわたるタスクを迅速に処理していくことを強いられている．このような対応の負荷軽減のためには，日々のインシデント対応内容とその結果を知見として継続的に蓄積及び共有し，類似性・関連が見い出せる将来の事案の対応に活かす仕組みが必要である．一方で，実際のインシデント発生時は対応に追われがちであり，知見の蓄積を行うための時間を別途確保する事が難しい．したがって，現場の負荷を抑えた上で効率的にインシデント対応の知見を蓄積・共有する仕組みを適切に対応フローに組み込むことには価値がある．
本稿では東工大 CERT で行なっているインシデント対応をアラート収集，トリアージ，ハンドリング，文書化の 4 つのフェーズとして抽象化し，各フェーズのタスクを担当者が順次処理することで現場の作業負荷及び心理的負荷を軽減しながらも，継続的な知見の蓄積を可能とするインシデント対応フローを提案する．さらに東工大 CERT では，提案する対応フローに沿って知見を蓄積および活用可能とするシステムを OSS である GitLab を用いた実装および運用に取り組んでおり，システムの実装方法に加えて実運用で得られた経験も踏まえ，インシデント対応時の知見の活用方法の可能性に関して報告する．

## 1. はじめに

CSIRT 業務における負荷については，セキュリティ機器からの大量のアラートの処理等，体制を整えるまでが高いものと考えられがちであるが，実は，体制が整うにつれ，予防措置，詳細な処理，小さな粒度の情報収集，組織内外関係各所への報告など，より適切に業務を進めるための項目が急増するため，体制が整う前より，限界・飽和を急激に向かえ易くなる場合が多い．このため，体制から逆算した時間や処理数などでアラート・タスクの足切りを早期に導入すべきとの提案もある．

限られた人員・予算で実効性の高いインシデント対応を提供するには，可能な限りルーチンワークを自動化し，担当者の能力が最も必要なタスクに集中できる組織的な対応体制の構築が必要である．この実現のためにはインシデント対応業務におけるワークフローの管理が重要となるが，対応中の事案を相手に応じてどの粒度で文書化すべきかなどを適切に判断しながら，迅速な緊急対応を両立させることは難しい．

そこで，本稿ではインシデント対応の現場において緊急対応に求められる正確さ，迅速さを維持しながら，継続的にインシデント対応の知見の蓄積を可能とするインシデント対応フローの提案を行う．既に東工大 CERT [*1] ではオンプレミスで実現する業務効率化のための OSS 基盤環境構築 [5] にて CERT チームの基盤サービスの効果的な導入及び業務効率化に取り組み，そして東工大 CERT におけるインシデント対応の分析とその自動化に関する考察 [6] にてインシデント対応体制及び学内インシデント対応における必要情報とその依存関係については整理を行なった．これらを踏まえ，本稿で提案するインシデント対応フローの GitLab [*2] を用いた実装例とその活用の可能性を議論する．

本稿の構成は次の通りである． 2 章で本研究の背景および目的について述べ， 3 章で提案するインシデント対応フローについて説明する． 4 章では GitLab を用いた東工大 CERT での実装例を紹介し，それに基づく具体的なインシデント対応の流れを示す． 5 章では蓄積した知見の活用例と今後の発展の可能性について述べる．最後に 6 章において纏める．

[1] 東京工業大学 Tokyo Institute of Technology, Meguro, Tokyo 152–8550, Japan
[a)] mori@cert.titech.ac.jp
[b)] mishii@gsic.titech.ac.jp
[c)] matsuura@gsic.titech.ac.jp
[d)] yongj@gsic.titech.ac.jp
[e)] kitaguchi@gsic.titech.ac.jp
[f)] tomoishi@noc.titech.ac.jp

[*1] http://cert.titech.ac.jp
[*2] https://gitlab.com/

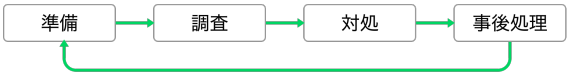


図 1　一般的なインシデント対応サイクル

## 2. 背景および目的

インシデント対応の現場では迅速な判断と対処が常に求められ，日々大量の処理すべきタスクを抱える担当者はインシデント対応の知見の蓄積・活用に積極的に取り組む余裕がない場合も多い．

取り扱ったインシデントの対応結果を整理し，記録・報告するタスクは時間のかかる作業であるため，調査過程で深刻度の判断理由となった情報や，対処の妥当性を示す情報などは報告書に整理されないまま破棄される情報も多い．限られた時間の中で試行錯誤を行う必要がある初動調査等では特に収集したデータを適切に蓄積・共有しにくい傾向があり，もし適切な知見の管理が行われない場合は，同じ調査を繰り返し行ったり，同じ判断に至るまで時間を取られがちである．その結果，インシデント対応において慢性的に時間的な余裕がない状況を招く．

このような悪循環を避けるには，インシデント対応における業務サイクルに適切な形で必要な知見を必要な粒度で蓄積するフェーズやタスクを組み込む必要がある．そして組織の規模や体制に見合った具体的なワークフローの構築を行い，知見の蓄積を行いながら対応業務に取り組む基盤体制を整える必要がある．

### 2.1 インシデント対応サイクルにおける知見蓄積

様々なインシデント対応サイクルが提案されているが，多くは準備，調査，対処，事後対応の 4 つのフェーズに分けられるものが多い．このインシデント対応サイクルとして一般的な例を図 1 に示す．

インシデント対応サイクルの一般的な流れは次の通りである．第 1 フェーズでは，検知機器等の機器等の整備をはじめ，情報収集・連絡窓口などの体制整備を行い，第 2 フェーズでは，集められた情報の分析と対処すべき端末の特定および対処方法の調査を行う．そして第 3 フェーズで，端末の管理者に連絡を行い，封じ込め・根絶・修復といった技術的な対応を行う．最後の第 4 フェーズで報告書の作成，教育・啓発，関係者への情報共有を行う．

また，知見の蓄積に着目した対応サイクルもいくつか存在する．実践 CSIRT プレイブック [1] では，Cisco CSIRT で実践されている調査手段（検知ルール）と対応方法のセットであるプレイブックの管理を含むサイクルが紹介されている．またインテリジェンス駆動型インシデントレスポンス [3] では脅威インテリジェンスの活用・整理・共有を含むサイクルである F3EAD モデル [2] が紹介されている．これらは上記で述べたインシデント対応サイクルの 4 フェーズを細分化したものに，知見の監査・活用フェーズを加えたサイクルとなっている．

### 2.2 インシデント対応の知見蓄積における課題

前節で紹介したインシデント対応サイクルにおいて，知見の蓄積作業は第 4 フェーズに含まれる．しかしこの段階で整理される情報は報告書レベルの整理された情報であり，第 1 から第 3 フェーズで収集された様々な知見は整理されにくい傾向にある．インシデント対応で最終的にまとめる報告内容の情報のみならず，対応現場で生成されるインシデント調査等の有用な情報も記録するため，インシデント対応サイクルを再考する必要がある．

また，各フェーズにおける知見の蓄積を行うルールまたはポリシーを決めたとしても，実際に実効性のある対応フローを構築することは難しい．その理由として，時間的余裕の無さから知見の蓄積まで手が回らないこと以外に，知見の蓄積タスクにおいて以下に挙げる項目が担当者にとって不明瞭であることが挙げられる．

- どのような情報を貯めるべきか．
- どのタイミングで貯めるべきか．
- どのようなフォーマットで貯めるべきか．

継続的な知見の蓄積を可能とするには，これらの点を現場の担当者が都度判断して作業する必要がある．この判断にも作業コストがかかり，同時に蓄積作業への担当者のモチベーションにも悪影響を与える．つまり，知見の蓄積作業を円滑に行うには，これらがワークフローの中で自然に対応の副作用として明瞭化された上で，インシデント対応サイクルの各フェーズに知見の蓄積作業が適切に組み込まれることが必要となる．

## 3. 知見の蓄積のためのインシデント対応フロー

前節では知見の蓄積タスクをインシデント対応フローに組み込む必要性と，その際に考慮すべき課題を示した．本節では知見蓄積タスクにおいて明瞭化すべき事項とそれらの実現のために求められる対応フローの形についてさらに整理を行う．それに基づき実際に知見の蓄積タスクを含むインシデント対応フローの提案を行い，有効性について議論する．

### 3.1 セキュリティ事案における知見

まずは，本稿で述べるセキュリティ事案における知見は何かを定義する．インシデント対応の一連の作業の中には，情報収集と判断の作業が何度も発生する．現場の担当者は，アラートの詳細を調査すべきか否か，担当者へ連絡すべきか否か，または対応内容は妥当であるか否かなど，

対応過程で様々な判断を迅速に下すことが求められる．判断内容とその根拠のセットは，次回似たような作業を行う際に有用な情報であり，これらは報告書レベルで整理される情報には含まれないものも多い．本稿ではこの判断内容と判断の根拠およびその際に収集した情報の組み合わせを集積したものを，インシデント対応における知見と呼ぶことにする．

### 3.2 セキュリティ事案の知見蓄積における 5W1H

セキュリティ対応組織 (SOC/CSIRT) 強化に向けたサイバーセキュリティ情報共有の「5W1H」[4] では，サイバーセキュリティ情報共有における 5W1H を明瞭化し，それらを軸にして CSIRT における情報共有に取り組むべきかが示されている．セキュリティ事案における知見の蓄積においても，どの情報をどのタイミングでどのフォーマットで貯めるべきかが不明瞭であることが，知見蓄積タスクを組み込む上での課題であると 2 章で述べた．よってこのサイバーセキュリティ情報共有の 5W1H に倣って，セキュリティ事案の知見の蓄積における 5W1H が指し示すものを提示し，そして対応フローの中でそれらを明瞭化するために留意すべき事項を以下に整理する．

**WHEN（どのタイミングで）** 対応過程で様々な判断を下すタイミングがこれに当たる．多くの場合これと同時に判断結果の報告及び情報共有が同時に行われる．重要な判断が必要なタスクは優先順位を付けて対応可能であるべきであり，それに付随する知見蓄積タスクがその優先順位付の判断に影響されない様にすべきである．また優先順位を付けてプールされたタスクはチーム内で見える化され，定期的に再評価・監査し易くすべきである．

**WHERE（どの場所に）** 現場の担当者が情報共有を行える場であればどこでも良い．ただし入出力が容易で検索可能なデータベースである必要があり，また実際に各タスクを行う際に使用する場に蓄積できることが好ましい．よって実際にはケース追跡ツールやコミュニケーションツールを拡張したものでも良い．

**WHO（誰が）** インシデント対応における各タスクの直接的な作業担当者がこれに当たる．小さなチームでは様々な分野・種類にまたがるタスクを少数で回す必要があり，担当者が曖昧になりがちであるため，各タスクに担当者をアサインする手順を含むことが好ましい．

**WHAT（どんな情報を）** これを明瞭にすることは難しい．これは基本的には各判断の根拠となる情報であり，技術的な情報に限らず，基本的に流動的であってスキーマとして定義しにくいものである．よってテンプレート等を用いて作業コストと蓄積される知見の質のバランスを取ることが求められる．またテンプレートを含み，情報の蓄積ポリシーは短いサイクルで監査可能であることが求められる．

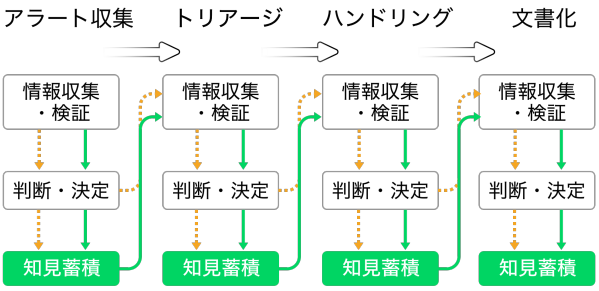


図 2 インシデント対応フロー

**WHY（どんな目的で）** 前節で述べた知見の定義の通り，インシデント対応過程の判断結果に対してその判断理由およびその関連情報を整理することが目的に当たる．つまり各判断が蓄積タスクにとって明示的に分解される対応フローであることが好ましい．これを満たす対応フローにおいて，判断に基づいてフェーズが遷移したとき，その遷移理由を整理する作業が，知見の蓄積に相当する．

**HOW（どのフォーマットで）** WHAT と同様にこれも明瞭化することは困難である．再利用を容易にするためには情報の正規化作業が必要である一方で，どのような正規化が適切であるかを判断することは難しく，また正規化そのものの作業コストも大きい．よって自由記述を許容し，必要に応じてテンプレート等を用意することでこの作業コストと実際に蓄積される情報のバランスを取る等の手順を考える必要がある．

### 3.3 インシデント対応フローの提案

前節で，セキュリティ事案の知見蓄積おける 5W1H を軸にインシデント対応フローに求められる留意点を整理した．対応フローに知見の蓄積作業を組み込むにあたり，これらの留意点に注意することで，これらの 5W1H が明示的に示されるような仕組みを備えるワークフローの構築を目指す．本節では図 2 のようなインシデント対応フローを提案する．

#### 3.3.1 基本設計

3.2 節の WHY で述べた様に，重要な判断を軸に対応フローのフェーズを分解することが望ましい．そこで東工大 CERT のインシデント対応の現場における経験から，インシデント対応業務を，アラート収集，トリアージ，ハンドリング，文書化の 4 つのフェーズに分けて抽象化を行い，その上でインシデント対応フローに知見の蓄積タスクを組み込みを行う．各フェーズにおける具体的なタスク内容は以下の通りである．

**アラート収集** 連絡窓口への情報提供やセキュリティ機器からのアラート情報，または脅威ハンティング等によ

表 1　対応フェーズと判断すべき事項

| 対応フェーズ | 判断すべき事項 |
|---|---|
| アラート収集 | 情報ソースの中でどの情報を初動調査すべきか，何をトリガーにすべきか． |
| トリアージ | 収集されたアラートは重大か，詳細調査すべきか． |
| ハンドリング | 対処内容は妥当か，危険性・原因は排除されたか． |
| 文書化 | 報告書の内容は妥当か，適切に整理されているか． |

り捕捉された脅威情報をまとめて，対応案件/インシデント候補としてリストアップを行うフェーズがこれに当たる．この対応案件を本稿ではアラートと呼称する．

**トリアージ**　収集されたアラート全体を表層分析（即時分析）し，対応プランへの分類および対応優先順位付けを行うフェーズがこれに当たる．誤検知であればホワイトリストへ追加し，情報不足で時間を要する案件はウォッチリストに追加する．一方ですぐに対処が必要な案件はインシデント扱いへ昇格し，次のハンドリングフェーズへ渡す．

**ハンドリング**　対処が必要と判断された事案/インシデントの詳細な分析（深層分析）を行い，その結果に基づき該当する端末の管理者等に連絡し，対処完了までの流れをサポートするフェーズがこれに当たる．封じ込め・根絶・修復と言った技術的な対応もこれに含まれる．

**文書化**　対応が完了した事案/インシデントの情報を報告書として整理するフェーズがこれに当たる．この文書とは事案発生した組織の上層部向けの報告書，サポートを行った CSIRT の対応内容報告書などが該当し，対応情報を共有する対象ごとに作成が必要である．

#### 3.3.2　知見蓄積タスクの設計と 5W1H との対応関係

この 4 フェーズはインシデント対応における重要な判断に連動しており，各フェーズ間を遷移する作業がそれらの判断とその根拠を明示的に示すタスクに相当する．各フェーズにおける判断すべき事項の一例を表 1 に示す．

これらの判断結果とその根拠及び関連する情報を整理するには，各フェーズの遷移時に知見の蓄積タスクを組み込むことで実現できる．フェーズに分解したことにより，この知見の蓄積タスクにおいて次のことが明瞭化される．

**いつ知見蓄積タスクを行うか (WHEN)**　各フェーズの遷移時に重大な判断が下されるため，この判断結果と根拠の収集もまたフェーズの遷移に行うことと明示化できる．多くの場合対応状況の共有及び報告のためにこれらの情報が整理されるため，その作業そのものを知見の蓄積とすれば良い．これは 3.2 節の WHEN で示した留意点に対応している．

**誰が知見蓄積タスクを担うか (WHO)**　各フェーズのタスクごとに担当者がアサインすることにより，誰が知見の蓄積タスクを行うかが明示することが可能である．これは 3.2 節の WHO で述べた留意点に対応している

**どのような目的で知見蓄積タスクを行うか (WHY)**　上記のの各判断を下すために各フェーズでは情報収集・調査を行い，その結果判明した根拠を示してフェーズ遷移に至る．つまり，この対応フローにおいて表 1 で示す判断とその根拠は自然に整理されながら段階的に業務が進んでいく．よってこのフェーズ移行のタイミングで知見の蓄積タスクを組み込めば良い．これは 3.2 節の WHY で述べた留意点に対応している．

一方で下記の項目については対応関係が明確に示されておらず，実装の段階でより考慮が必要となる．

- どこに知見を蓄積するか (WHERE),
- どのような情報を蓄積するか (WHAT),
- どのようなフォーマットで蓄積するか (HOW).

これらについては次章における実装するシステム構築の過程で対応関係を示す．

## 4.　インシデント対応・管理のための Gitlab の活用

前章ではインシデント対応時の知見蓄積における留意点を整理し，それに基づいた対応フローの提案を行った．本章では，これを適えるインシデント対応フローを支えるシステムとして，東工大 CERT にて内部向けサービスとして導入している GitLab を活用した実装例について述べる．

### 4.1　GitLab を採用した理由

3 章で提案したインシデント対応フローは，BTS/ITS を含む一般的なワークフロー管理ツールの多くで実現可能である．今回インシデント対応フローの管理のため GitLab を採用した．これは今回提案する対応フローに沿ってデザインされたツールではないものの，多くの要件を満たしている．これを採用した理由は以下が挙げられる．

- Git の自由度と信頼性の高さ，
- 課題管理機能（カンバンボード機能），
- Issue/MergeRequest 作成時のテンプレート指定機能，
- WebUI のみで一連の作業が完結可能，
- CI（継続的インテグレーション）との連携のし易さ．

Git の学習コストの高さがチーム全体への導入の障害になりうるものの，GitLab はオンプレミスの OSS として運用可能でありながら，課題管理において優れたユーザインタフェースと十分な機能を備えていると判断して採用した．そして何より東工大 CERT は既に GitLab を内部向けサービスとして導入済みであり，運用するサービスが増えない点も採用の大きな理由である．

### 4.2 インシデント対応フロー管理のための GitLab 環境の構築

東工大 CERT では，3 章で述べた対応フローの実現のため，GitLab のワークフロー上に提案するインシデント対応フローの効果的な落とし込みを試みた．本節では GitLab のワークフローとインシデント対応フローがどのような対応関係にあるかを整理する．

#### 4.2.1 用語の整理

説明に先立ち，予め Repository，Issue，MergeRequest が今回のインシデント対応フローにおいて何を意味するのかを説明する．

**Repository** インシデント対応管理プロジェクトにおける Repository の主目的はインシデント対応記録をまとめることである．対応が完了したインシデントの報告書を含み，対応時に整理した情報をリポジトリに追加していく．また後述する Issue/MergeRequest のテンプレートをはじめ，この対応フローを管理していく上でのガイドラインやルールもこちらへ追加し，適宜適当であるかを監査して必要に応じて修正を行う．また取り扱ったインシデントの検知ルールまたはプレイブック等の情報も合わせて整理する．

**Issue** インシデント対応管理プロジェクトにおいて，Issue はその Repository への修正検討の提案に相当する．具体的には対処が必要な可能性のあるネットワークイベント情報や脅威情報がこれに当たり，メンテナがトリアージを行い，その結果担当者のアサインとラベリング等される．本稿では，基本的には 3 章で言及したアラート情報が Issue として収集されるものに当たる．

**MergeRequest** インシデント対応管理プロジェクトにおいて，MergeRequest は Repository への修正要求に相当する．これは収集された Issue の中でインシデントとして緊急対応が必要であると判断された場合に作成し，解決後に整理した報告書及び関連情報を Repository へ追加要求がメンテナに対して発生する．

#### 4.2.2 IssueBoard によるタスクプールの可視化

GitLab には Issues 管理システムの一つとして，IssueBoard 機能を備える．これを用いると，Issue を指定した Issue のラベルをカラムとしたカンバンボードとして整理することで，タスクの可視化が可能となる．このカンバンボードは異なるステータスの多くのケースを並行して処理する必要がある現場では特に有効である．東工大 CERT で実際に使用している IssueBoard 画面の一部を図 3 に示す．

各ケースはそのステータスによって取り組むべきタスクが分類可能であり，このステータスを示す IssueBoard のカラムとして，東工大 CERT では次の 5 つを使用している．なお，このカラムは Issue のラベルの中から指定されたものであり，@が頭文字として付与されるラベルは東工大 CERT で独自に作成したものを指す．

**open (backlog)** 新規作成した Issue はこちらに格納される．またどのカラムにも属さない Issue もここに入る．こちらに Issue をプールしていく作業がアラート収集フェーズに当たり，そしてここにある Issue の緊急性を判断し，ラベル付けを行う作業がトリアージフェーズに当たる．また，これがトリアージフェーズのタスクプールに相当する．

**@onhold** トリアージの結果，様子見と判断した Issue はこちらに格納する．定期的にレビューを行う．

**@noticed** トリアージの結果，緊急性は低いと判断した上で，念の為注意喚起または情報提供のみを行った Issue はこちらへ格納する．

**@handling** トリアージの結果，緊急対応が必要と判断した Issue をここに格納する．ここを主として@onhold,@noticed と合わせて 3 つがハンドリングフェーズのタスクプールに相当する．

**@resolved** 緊急対応が完了した Issue をここに格納する．ここにが文書化フェーズのタスクプールに相当する．

**closed(done)** 文書化を含めて全ての対応が完了した Issue はここに格納される．こちらに情報を集積することが知見の蓄積に当たる．

#### 4.2.3 対応フェーズの遷移と Issue/MergeRequest のライフサイクルの対応

インシデント対応における各タスクは 3 章で述べた対応フェーズによって大きく整理される．各ケースはステータスごとにタスクを持ち，そのタスクは 3 章で述べた対応フェーズによって整理することができる．そして各フェーズの遷移がインシデント対応における重要な判断と連動する旨は 3 章で示している

今回実装したシステムにおいて，各フェーズの遷移は全て Issue または MergeRequest のラベリングを含むライフサイクルと連動しており，Issue/MergeRequest のステータス (ラベル) の遷移がインシデント対応フローにおける様々な判断事項に対応している．具体的な対応関係を表 2 に示す．

Issue/MergeRequest のライフサイクルはラベルによって整理され，IssueBoard のカラムはそのラベルにより整理される．各フェーズごとにカラムを整理することで左から右へ処理が進むと対応フェーズも遷移する．対応フェーズと Issue/MergeRequest のラベルの対応関係を図 4 に示す．

また 3 章で示した対応フローのフェーズごとの作業内容の概要を以下に整理する．

**フェーズ 1：アラート収集** セキュリティ機器からのアラートや目についた攻撃，外部からの情報提供などといったインシデントとして対応すべき可能性のある案件を Issue として書きだし，整理を行う．その際に情報ソースをはじめとする基本的な情報を Issue ページにて整理を行い，チーム内で情報共有を行う．

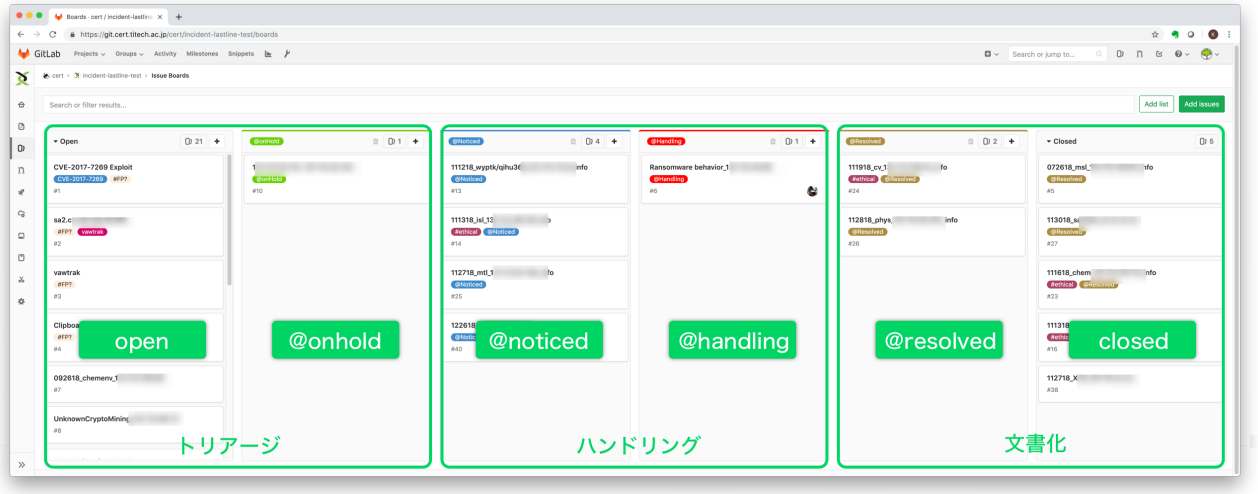


図 3　IssueBoard によるタスクプールの可視化

表 2　Issue/MergeRequest のステータスの遷移先と判断事項

| 対象とステータス | 遷移先 | 判断事項 |
|---|---|---|
| Issue | open | 調査/トリアージが必要である |
| | closed | 対応終了して良い |
| | @onhold | 経過観察が必要である |
| | @noticed | 注意喚起が必要である |
| | @handling | 緊急対応が必要である |
| MergeRequest | open | 緊急対応が必要である・<br>対応情報の整理が必要である |
| | closed | 対応を終了して良い・<br>対応情報の整理が必要ない・ |
| | merged | 対応を終了して良い<br>対応情報が適切に整理された |

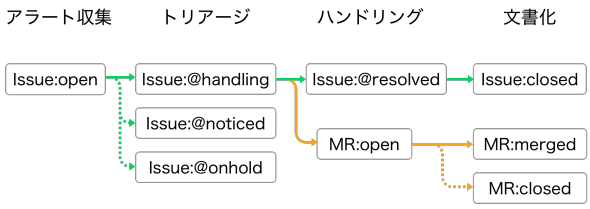


図 4　対応フェーズと Issue/MergeRequest(MR) のラベルの対応関係

**フェーズ 2：トリアージ**　Issue として収集されたアラート情報の緊急性を判断し，対応プランに応じて Issue のラベル付けを行う．ラベリングを行う際にはその判断を下した根拠となる情報を Issue ページに追記する．そして実際にハンドリングが必要と判断された Issue はフェーズ 3 へ進む．

**フェーズ 3：ハンドリング**　対応が必要と判断された Issue に対して，対応依頼先ごとに MergeRequest を作成する．MergeRequest ページに実際にハンドリング中のやり取りや判断をはじめとする情報をダンプと整理を行う．封じ込め，根絶，修復等の対処が完了した段階で解決済みとして resolved のラベル付けを行う．

**フェーズ 4：文書化**　resolved のラベルを付けられた Issue の MergeRequest ページにて，該当のインシデント対応全体の情報を整理し，それらを含む最終的な対応報告書をリポジトリへマージする．そして必要に応じて関連する Issue のクローズを行う．

#### 4.2.4　チャットツールとの連携

東工大 CERT ではチーム内コミュニケーションツールの基盤として Slack クローンの OSS である Rocket.Chat[*3] を利用している．チーム外部とのやりとりはメールまたは電話を用いて行うが，そこでやりとりした対応内容などのチーム内情報共有はチャットツール上に展開されることが多い．特にハンドリングフェーズではメンバー間でインタラクティブなやりとりを頻繁に行う必要があり，メインの情報共有が GitLab からチャットツール上に移る．そのため，ハンドリングフェーズに入るタイミングでチャットツール上にも該当のインシデント情報が展開する機能が必要となり，連携機能の実装に取り組む必要があった．

連携機能の目的は以下の通りである．

- GitLab で管理しているインシデントのステータスとチャットのチャンネルが連動する．
- GitLab で Issue をクローズするとチャンネルもアーカイブされる．
- GitLab で Issue に書き込むと連動したチャンネルにも書き込まれる．

具体的な実装方法はここでは割愛する．上記の機能の実装により，ハンドリングフェーズで収集される情報をチャットへの集約が実現できた．

#### 4.2.5　Issue/MergeRequest のテンプレート機能の活用

GitLab の Issue/MergeRequest にはタイトル以外に description を付与することができ，そしてその description

[*3] https://rocket.chat

表 3　Issue 用のテンプレート (Incident.md) の項目例

| | |
|---|---|
| SOURCES | アラート情報のトリガーとなったイベントとその種別（セキュリティ機器のログ/レポート，連絡元/情報提供元など）を記す． |
| SUMMARY | 調査内容の概要．調査が進むにつれ判明した事実を追記していき，最終的には情報を整理する． |
| TRIAGE | トリアージ結果とその根拠（内部向け）を記す．状況によって変更等があれば，その変更と理由等を追記していく．（誤検知と判明して Issue から直接クローズする場合など） |
| CONTACT | 調査時点での連絡先/報告先の情報を記す |
| REASON | 連絡/報告するにあたり，連絡する表向きの根拠を記す |
| LOGS | 連絡/報告するにあたり，先方の調査のために渡す参考ログを記す |

の編集時に独自に用意したテンプレートを表示させることが可能である．これは単純なテキスト（マークダウン）ファイルであって入力情報を厳密に縛るものではないが，担当者が Issue/MergeRequest において入力すべき情報のガイドラインを示して入力者をサポートすることが可能となっている．

知見の蓄積タスクにおいて何の情報を蓄積すべきか (WHAT), どのようなフォーマットで蓄積すべきか (HOW) の明瞭化は難しいと 3.2 節にて述べた．そこでこのテンプレートを活用し，ある程度方針を明示しつつバランスのとれた知見の蓄積タスクの実現を目指す．

また，このテンプレートも同じ Git Repository 内で管理されるため，必要に応じて適正であるかを監査し修正するサイクルを同じシステム上で構築することが可能である．東工大 CERT ではこのようなテンプレートを複数用意し，担当者が適宜状況に合わせて活用できるようにしている．

一例として東工大 CERT で実際に使用している基本的な Issue 用テンプレートで使用している項目を表 3 に示す．SOURCES，SUMMARY，TRIAGE の情報は多くの場合アラート収集にて整理される．そしてトリアージフェーズにてハンドリングの必要があると判断されたタイミングで残りの CONTACT，REASON，LOGS が整理される．

これら以外にアラート収集時のテンプレート (Alert.md) や，MergeRequest で主に使用されるテンプレート (Handling.md) や最終的な報告書用のテンプレート (Report.md) などを用意し，必要に応じて参照することで，情報の種類及びフォーマットをある程度揃えながらも知見の蓄積タスクを迅速に行えるようにしている．

### 4.3　知見蓄積タスクの実装/運用と 5W1H との対応関係

3.3.2 目で本稿で提案する対応フローが 3.2 節で整理した 5W1H の明瞭化にどのように寄与しているを示した．これと同様に，本節では今回実装したシステム上の対応フローが，3.2 節で整理した 5W1H の留意点にどのように対応しているかを整理する．Issue/MergeRequest のライフサイクルと連動していることを踏まえて以下に整理する．

**いつ知見蓄積タスクを行うか (WHEN)**

Issue/MergeRequest のステータス（ラベル）遷移時が発生する時がこれにあたり，その理由を description に追記していく形で整理していく．

**誰が知見蓄積タスクを担うか (WHO)**

Issue/MergeRequest でアサインされた担当者が知見蓄積タスクも担う．

**どのような目的で知見蓄積タスクを行うか (WHY)**

Issue/MergeRequest のステータス遷移に関する情報の記録・共有を行いつつ，今後似たようなタスクの対処の助けとなるような情報の蓄積を行うことが目的である．

**どこに知見を蓄積するか (WHERE)**

Issue/MergeRequest のデータベースおよび Git Repository がこれに相当する．Issue はアラート収集およびトリアージフェーズにおける知見が蓄積され，MergeRequest はトリアージおよびハンドリングフェーズの知見が蓄積される．そして Git Repository には文書化フェーズおよび全体の知見が蓄積されている．

**どのような情報を蓄積するか (WHAT)**

Issue/MergeRequest のテンプレートによって必要な情報は整理されている．Repository についても，テンプレートを用意して対応している．

**どのようなフォーマットで蓄積するか (HOW)**

Issue/MergeRequest, Repository のテンプレートにより，情報を記録する際のある程度のフォーマットを指針が示されている．

3.3.2 目の対応フロー構築時にはっきりと明瞭化ができていなかった WHERE，WHAT，HOW についても Issue/MergeRequest のテンプレートを適切に設定することである程度明瞭化することができている．ただし，Issue/MergeRequest のテンプレートの質を維持することが重要であり，それを監査するサイクル等が必要となる．

以上で今回の実装で 3.2 節で述べた留意点を考慮した対応フローの構築ができた．

## 5.　今後の展望と知見の活用方法について

ここまで，東工大 CERT におけるインシデント対応フローの管理の方法とそれを支えるシステムの実装について述べた．本節では，それらの基盤を用いて今後展開を検討している知見の活用例をいくつか挙げる．

### 5.1　CI ツールによる報告書作成の自動化

セキュリティ事案対応の報告書が Git のリポジトリとして管理されることで，CI（継続的インテグレーション）

ツールを用いて常に最新の報告書の統計情報等を自動的に生成させることが容易となった．これにより定期的に開催される委員会等の報告資料の作成等のルーチンワークの自動化を見込める．チームメンバーは業務負荷の少ないタイミングで文書化に取り組むことができ，また，必要な報告書に応じて，生成される資料のフォーマットの設定，あるいは，CI ツールによる柔軟な操作が可能であり，それらの設定を Git のリポジトリ内で管理することで，インシデント情報の入力及び出力を包括的に管理することも可能となる．さらに，GitLab はその内部に GitLabCI ツールと呼ばれるサービスをもっており，容易に導入及び運用が可能になっている．

### 5.2 過去対応案件情報のサジェスト機能

緊急時における対応のあり方はサイバー攻撃の種類はもちろんの事，発生した組織や関連するデータの機密性など様々な要因によって変化する．そのため非常に多くの組み合わせが存在し，知見が蓄積された場合であっても適切な類似例を検索する事は困難な場合が存在する．そのために事案発生時の環境情報を基に AI や機械学習の技術を応用した検索手法を検討している．具体的にはセキュリティ機器のログ情報や該当組織や人の情報，受けたサイバー攻撃の種類等々，事案発生時に関連する収集可能な環境情報を収集し，それらの情報を知見と併せて蓄積する．検索時は事案発生時の環境情報を基にし，過去の類似例を AI や機械学習の技術を応用して検索し，得られた過去の知見（判断やその根拠など）を用い，緊急対応時の意思決定に活用する計画である．また，次項で述べる自然言語処理技術を活用し，様々な環境情報を出来るだけ自動的に収集・蓄積する事も併せて検討している．

### 5.3 自然言語処理技術による知見蓄積の半自動化

2 章および 3 章で述べたように，判断および行動が発生する箇所を基点に 4 つのフェーズに分けて知見を蓄積する枠組みを設計した．時間に追われるインシデント対応現場の負荷を考慮し，出来るだけ簡素化した枠組みとなっている，迅速なインシデント対応を行い，また担当者の入力負荷を下げるために GitLab 内で表示されるテンプレート等では自由記述を許している部分も多く，担当者間のやり取りは chat システムを通じで行われている．このような自然言語で書かれた部分から自動的に必要な情報を抽出し，蓄積すべき正規化された情報としてまとめる事が出来ればより現場の負荷は下がり，知見蓄積の質も向上すると考えられる．そのために，自然言語処理技術を利用し，メール，GitLab 上の文書，chat，報告書等を対象に新規に発生したインシデントの自動分類に関する研究を行っている．将来的には該当組織や該当情報に関する詳細，関連するログの自動収集などより粒度の細かいデータ等が自動的に分類されるように取り組みを進める計画である．

## 6. おわりに

本稿ではインシデント対応をアラート収集，トリアージ，ハンドリング，文書化の 4 つのフェーズとして抽象化し，各フェーズのタスクを担当者が順次処理することで現場の作業負荷及び心理的負荷を軽減しながらも，担当者の役割に応じた継続的な知見の蓄積を可能とするインシデント対応フローを提案を行った．そして，東工大 CERT における本稿で提案した対応フローを実現する GitLab を用いた実装例を紹介し，具体的な知見の蓄積タスクを組み込んだ対応手順を示した．

さらに，このシステムを用いた蓄積された知見の活用方法をいくつか検討しており，実際に CI ツールによる報告書作成の自動化，過去対応案件情報のサジェスト機能，自然言語処理技術による知見蓄積の半自動化の 3 点の概要の説明と，取り組み状況について述べた．

また，近年のセキュリティツール/システムには担当者のタスクやケース追跡・管理，プレイブック管理をはじめ機能の統合が進みつつあり，知見の蓄積タスクとしてより効果的な機能を備えるツールが登場する可能性は大きい．よって，これらのツールの動向にも注目しながら，今回実装したシステムを適切に評価，運用していく必要がある．

**謝辞** 本研究の一部は，JST, CREST, JPMJCR1783, また，JSPS 科研費 JP15K00115 の支援を受けたものである．

### 参考文献

[1] Bollinger, J., Enright, B. and Valites, M.: *Crafting the InfoSec Playbook*, O'Reilly Media (2015), 飯島 卓也，小川 梢，柴田 亮，山田正浩（監訳），谷崎 朋子（訳）: 実践 CSIRT プレイブック — セキュリティ監視とインシデント対応の基本計画, O'Reilly Japan, 2018.

[2] Faint, C. and Harris, M.: F3EAD: OPS/INTEL FUSION "FEEDS" THE SOF TARGETING PROCESS (2019), `https://smallwarsjournal.com/jrnl/art/f3ead-opsintel-fusion-%E2%80%9Cfeeds%E2%80%9D-the-sof-targeting-process`, Accessed: 2019-05-05.

[3] Roberts, S. J. and Brown, R.: インテリジェンス駆動型インシデントレスポンス - 攻撃者を出し抜くサイバー脅威インテリジェンスの実践的活用法，オライリージャパン (2018).

[4] NPO 日本ネットワークセキュリティ協会 日本セキュリティオペレーション事業者協議会（ISOG-J）：セキュリティ対応組織 (SOC/CSIRT) 強化に向けたサイバーセキュリティ情報共有の「5W1H」第 2.0 版 (2019), `https://isog-j.org/output/2019/5W1H-Cyber_Threat_Information_Sharing_v2.0.pdf`, Accessed: 2019-05-05.

[5] 森 健人，松浦知史，金 勇，友石正彦：オンプレミスで実現する業務効率化のための OSS 基盤環境構築，情報処理学会研究報告 (2016).

[6] 石井将大，森 健人，松浦知史，金 勇，北口善明，友石正彦：東工大 CERT におけるインシデント対応の分析とその自動化に関する考察，情報処理学会研究報告 (2018).