大規模組織における Splunkを用いた脆弱性管理

- 情報収集と対応状況の可視化

株式会社NTTデータグループ
情報セキュリティ推進室　NTTDATA-CERT
2024年4月11日

目次

序章 Splunkと脆弱性管理	- 全体像	
1章 Splunkと脆弱性情報収集	- 情報取得編	#脆弱性とSplunk #Add-on Builder
2章 Splunkと脆弱性対応管理	- 可視化編	#メトリクス #Dashboard Studio
3章 Splunkと安定稼働	- 運用編	#AWSとSplunk #SplunkOps

自己紹介 – Daisuke Yamashita

- 山下 大輔
  - 株式会社NTTデータグループ（2018.04 -- ）
    - サイバーセキュリティ技術部
      NTTDATA-CERT
  - CISSP
- 経歴
  - 2018.04 入社
    - OSINT活動（2018 – 2020）
      - 「サイバーセキュリティに関する
        グローバル動向四半期レポート」
        ↑よかったら読んでみてください！
    - 脆弱性対応管理（2018 - ）
      - BlueKeep/DejaBlue(2019.05 -)
      - 複数のSSL-VPN製品の脆弱性(2019.09)
      - 緊急性高対応同時多発(2020.07)
        Microsoft, PaloAlt, F5, Citrix Apache . ..
      - Log4jは別PJ支援でチームに不在(2022)

自己紹介 – Mikiko Kikuchi

- 菊地美紀子
  - 株式会社NTTデータグループ（2022.04 -- ）
    - サイバーセキュリティ技術部
      NTTDATA-CERT
- 経歴
  - 2022.04　入社
    - 脆弱性対応を主に担当
- 学生時代と社会人1,2年目（現在3年目）
  - 大学：機械工学を専攻
    - 半分はコロナ渦
    - ITに関する知識はほぼゼロ
  - 1年目: 脆弱性対応チームに配属
    - Splunkはほぼ経験なし
  - 2年目: Splunkを用いた可視化(→今日の内容)

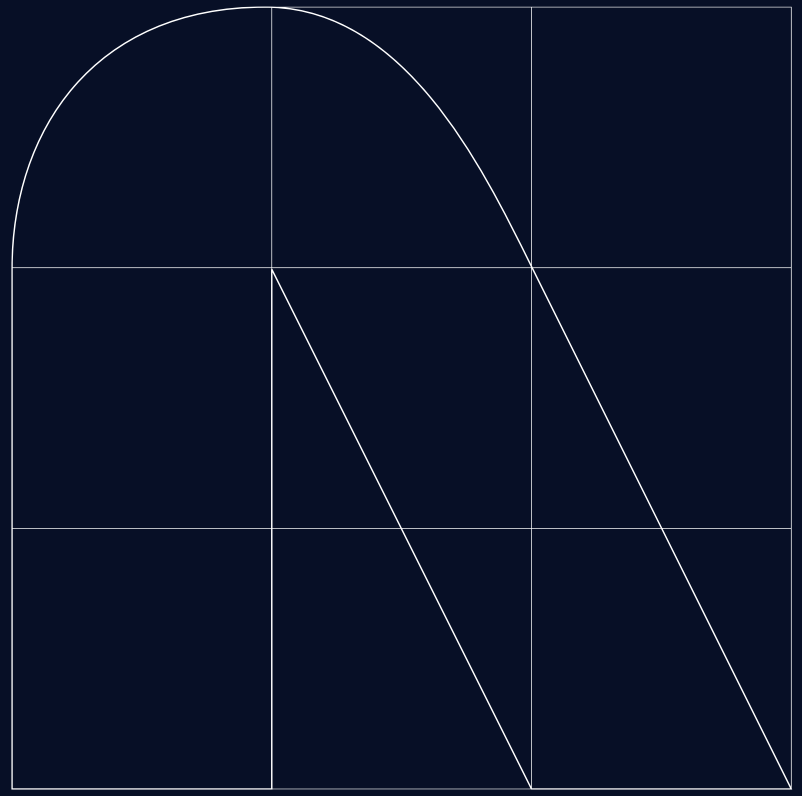
自己紹介 – Ryota Okuzawa

- 奥澤亮太
  - 株式会社NTTデータグループ（2021.11 -- ）
    - サイバーセキュリティ技術部
      NTTDATA-CERT
  - CISSP, GPEN, RISS,
    Certified SIM3 Auditor
- 経歴
  - 某自治体
    - (ほぼ)一人CSIRT
    - インシデント対応、ログ分析、セキュリティ規程
  - 某ベンチャー企業
    - Splunkを使ったサービス設計、デリバリ
      - 内部不正検知
      - データ分析全般
  - 2021.11より現職
    - 脆弱性対応を主に担当
    - 脆弱性情報収集システム(Splunk)の管理
    - Splunk BOTS v7 は10位でした

序章

Splunkと脆弱性管理

システム全体像



はじめに

- 大規模組織のCISRTによるログ分析以外のSplunk活用の話
  NTTDATA-CERT　　脆弱性管理

  - 脆弱性管理：**一般製品の公開脆弱性情報**からグループ内の対応方針決定、周知、管理
    ※ 自社開発システム/サービスの脆弱性を管理する話ではありません

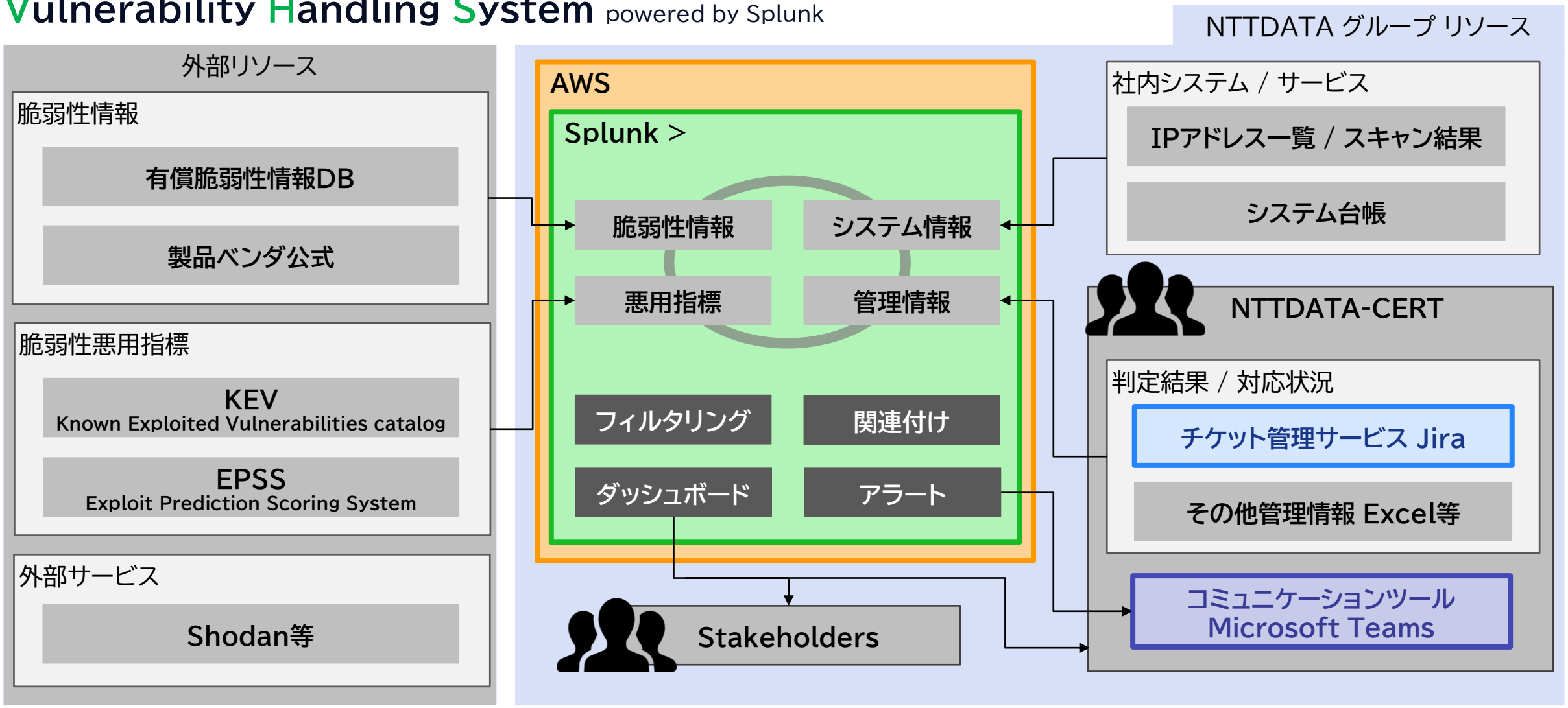
  - 組織の特徴
    - 組織はたくさん、グループ会社もたくさん、管理するシステムは盛りだくさん…
    - CISRT（自分達）、プロジェクト、統制組織、関係者もたくさん…

  - 脆弱性のトレンド
    - 報告件数は年々増加傾向、2024年は28,000件以上 ※
    - 脆弱性公開から攻撃発生までの短期間化

※ 出典：NVD（National Vulnerability Database）

システム全体図

Vulnerability Handling System powered by Splunk



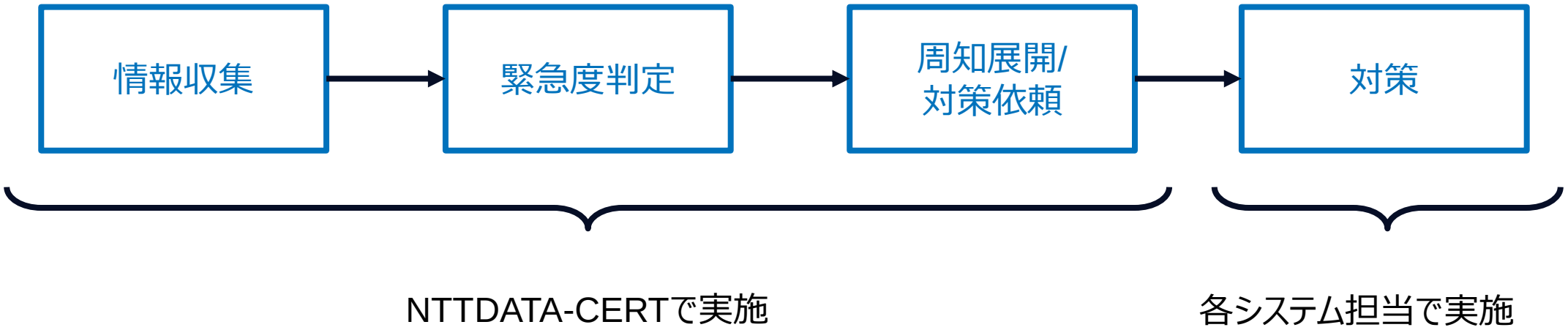
第1章

Splunkと情報収集

公開脆弱性情報の取得

どんなチーム？

- 公開された一般製品の脆弱性対応における
  - 情報収集
  - 緊急度判定
  - 周知展開/対策依頼
  - 対応状況管理
  - 技術支援
  - 外部通報窓口

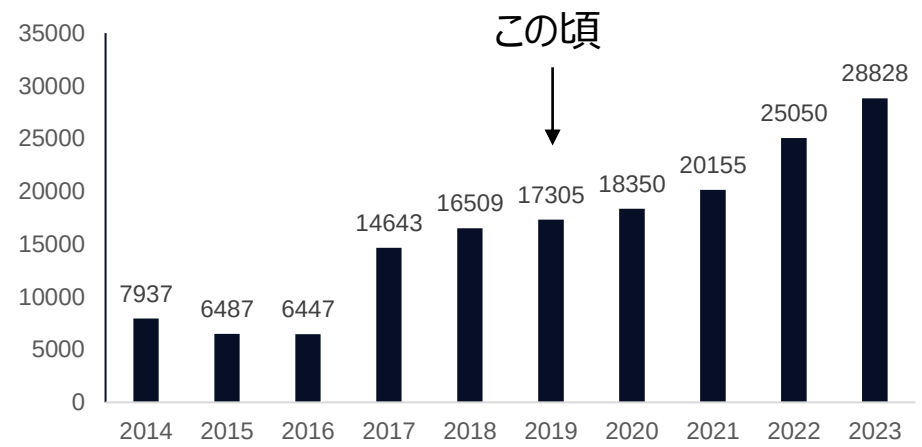


Splunk導入前の課題

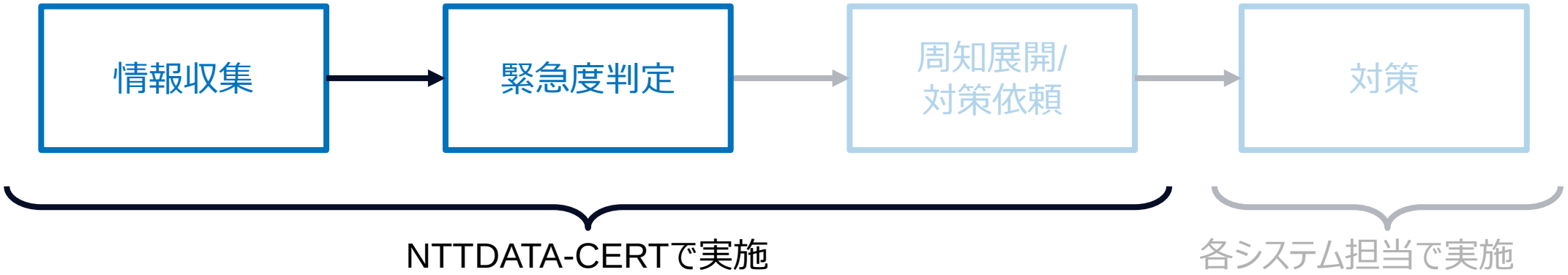
- 情報収集および緊急度判定の効率化が必要

  - 課題① 脆弱性報告件数の増加
    ⇒ 確認する脆弱性の件数(作業量)が増加

  - 課題② 情報公開から攻撃発生までの短期間化
    ⇒ 従来より早い対応が必要



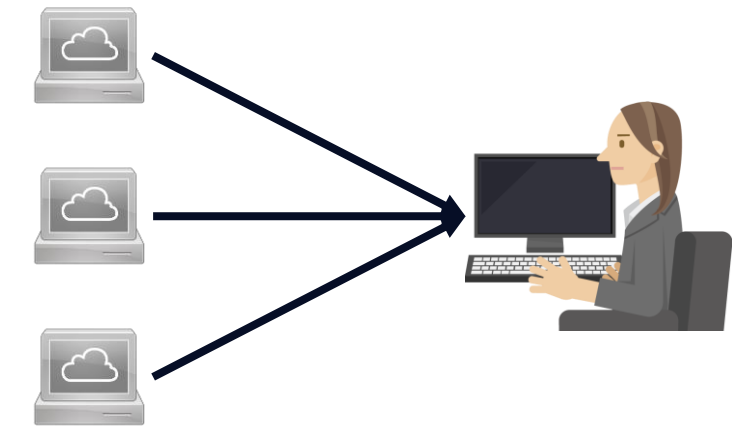
課題①参考 年間脆弱性報告件数
NVD登録データより作成 https://nvd.nist.gov/vuln/search/statistics



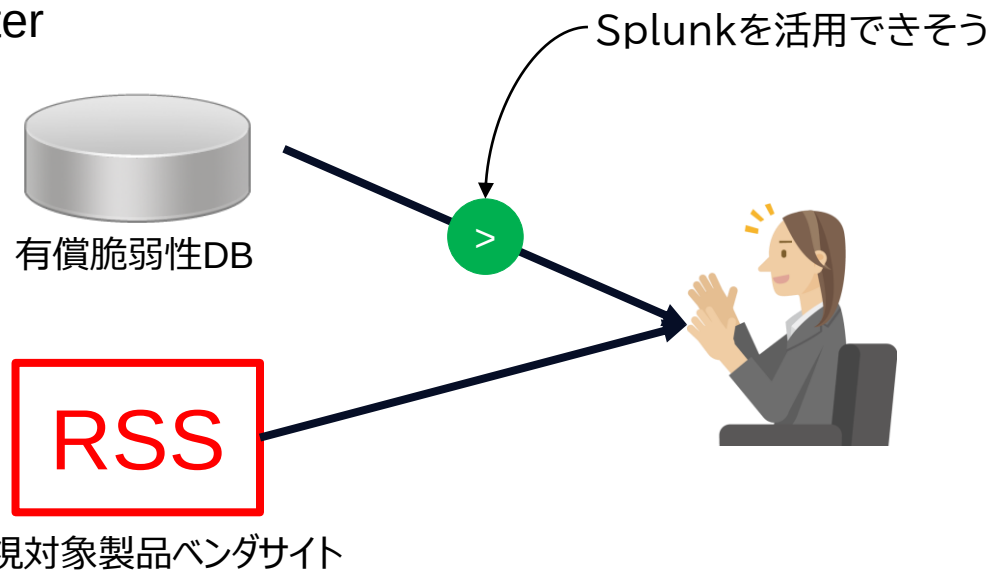
情報収集

- 従来の脆弱性確認手段
  - 注視製品の公式ベンダサイトを巡回
  - ベンダごとに異なるフォーマットから必要な情報を目視で確認
- 改善方針
  - 有償脆弱性DBの利用 ⇒ 規定フォーマットで複数製品を網羅
  - RSSの利用 ⇒有償脆弱性DBでカバーできない製品を補強

Before



After



緊急度判定

- 参照情報(集めた脆弱性情報)
  - 対象製品/ベンダ
  - CVE(共通脆弱性識別子)
  - CVSS(共通脆弱性評価システム)
  - 攻撃コード/悪用情報

- 判断ポイント
  - 外部から攻撃可能か(AV:N)
  - 前提条件が必要か(UI:N, PR:N)
  - 深刻な影響を与えるか(C:H/I:H)
  - 有効なPoC/攻撃情報の有無
  - 当社環境への攻撃は有効か
    - プロトコル/ポート
    - デフォルト設定での有効性
    - 利用環境
    - 想定攻撃方法

Splunk (SPL)
で自動化できそう

すぐに自動化は難しい

有償脆弱性DB CVSSのJSONサンプル


```
{
  "confidence": "High",
  "basescore": "5.5",
  "tempscore": "5.3",
  "basevector": "AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:N/A:H",
  "tempvector": "E:X/RL:O/RC:C",
  "baseseverity": "Medium",
  "tempseverity": "Medium",
  "av": "L",
  "ac": "L",
  "pr": "L",
  "ui": "N",
  "s": "U",
  "c": "N",
  "i": "N",
  "a": "H",
  "e": "X",
  "rl": "O",
  "rc": "C"
}
```

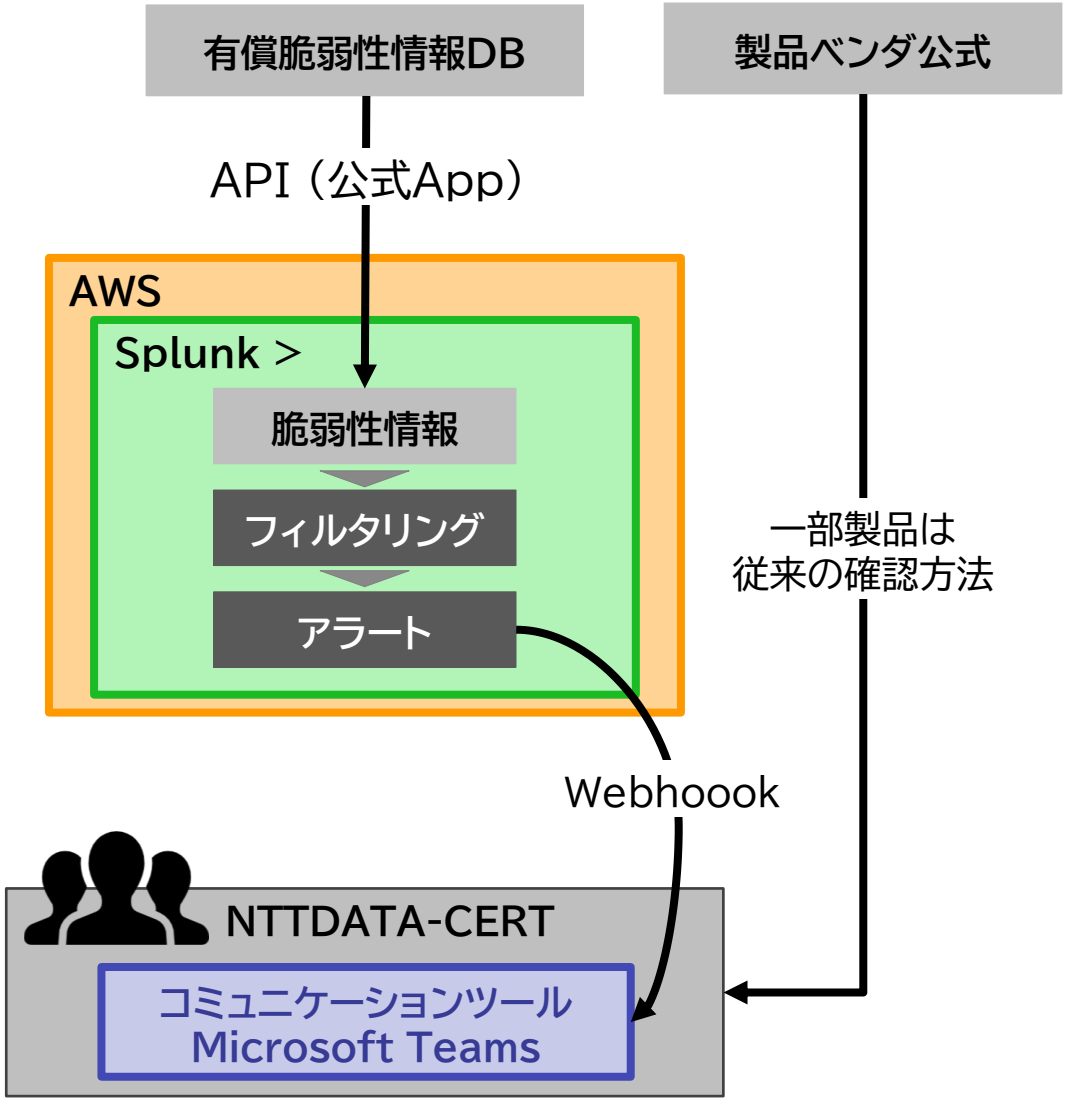
Splunk導入 脆弱性情報収集と判定の自動化

Vulnerability Handling System

- 外部から情報を収集する
- 脆弱性情報を貯める
- 緊急度が高い脆弱性をアラートする
  - 緊急度が低い脆弱性を機械的にフィルタ
  - ノウハウの判断部分も一部実装

処理の流れ

- API(公式App)経由で定期的に脆弱性情報取得
- Splunk アラート処理を定期実行
  - SPLで対象製品/ベンダでフィルタ
  - SPLで危険度の高いCVSSでフィルタ
  - 通知用に情報整理
  - アラートアクションでTeams通知



（参考）脆弱性情報通知サンプル

CVE-2020-5902

F5 BIG-IP Traffic Management User Interface Remote Code Execution

---

- Time : 2020/07/02 22:57
- Vendor : F5
- Software : BIG-IP
- Class : privilege escalation
- CVSSv3 : 9.8 AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
- Exploit : highly functional
- Link : [Vendor] [VulDB] [Splunk]

A vulnerability classified as very critical was found in F5 BIG-IP up to 11.6.5.1/12.1.5.1/13.1.3.3/14.1.2.5/15.1.0.3 (Firewall Software). Affected by this vulnerability is an unknown part of the component Traffic Management User Interface. Applying the patch 11.6.5.2/12.1.5.2/13.1.3.4/14.1.2.6/15.1.0.4 is able to eliminate this problem. The bugfix is ready for download at support.f5.com. A possible mitigation has been published immediately after the disclosure of the vulnerability.

情報収集機能の追加

情報を追加してさらに業務効率化

- **注視ベンダの公式脆弱性情報**を取得して緊急度判定までにかかる時間を短縮
- 脆弱性の**悪用に関する情報**を取得して緊急度判定の参考に活用
- 緊急度判定結果等の**業務記録**を紐づけて次回以降の緊急度判定の参考に活用

Splunk Add-on Builder によりデータ収集機能を追加

- Add-on Builder：データ収集やアラートアクションの機能を自作可能、Splunk提供App
  - データ取得をPythonで実装可能 ⇒ 公式APIを用いてインプットAppを自作
- 作成したデータ取得機能
  - 追加機能① 個別ベンダ脆弱性情報収集機能
  - 追加機能② 脆弱性悪用指標収集機能
  - 追加機能③ 対応状況収集機能

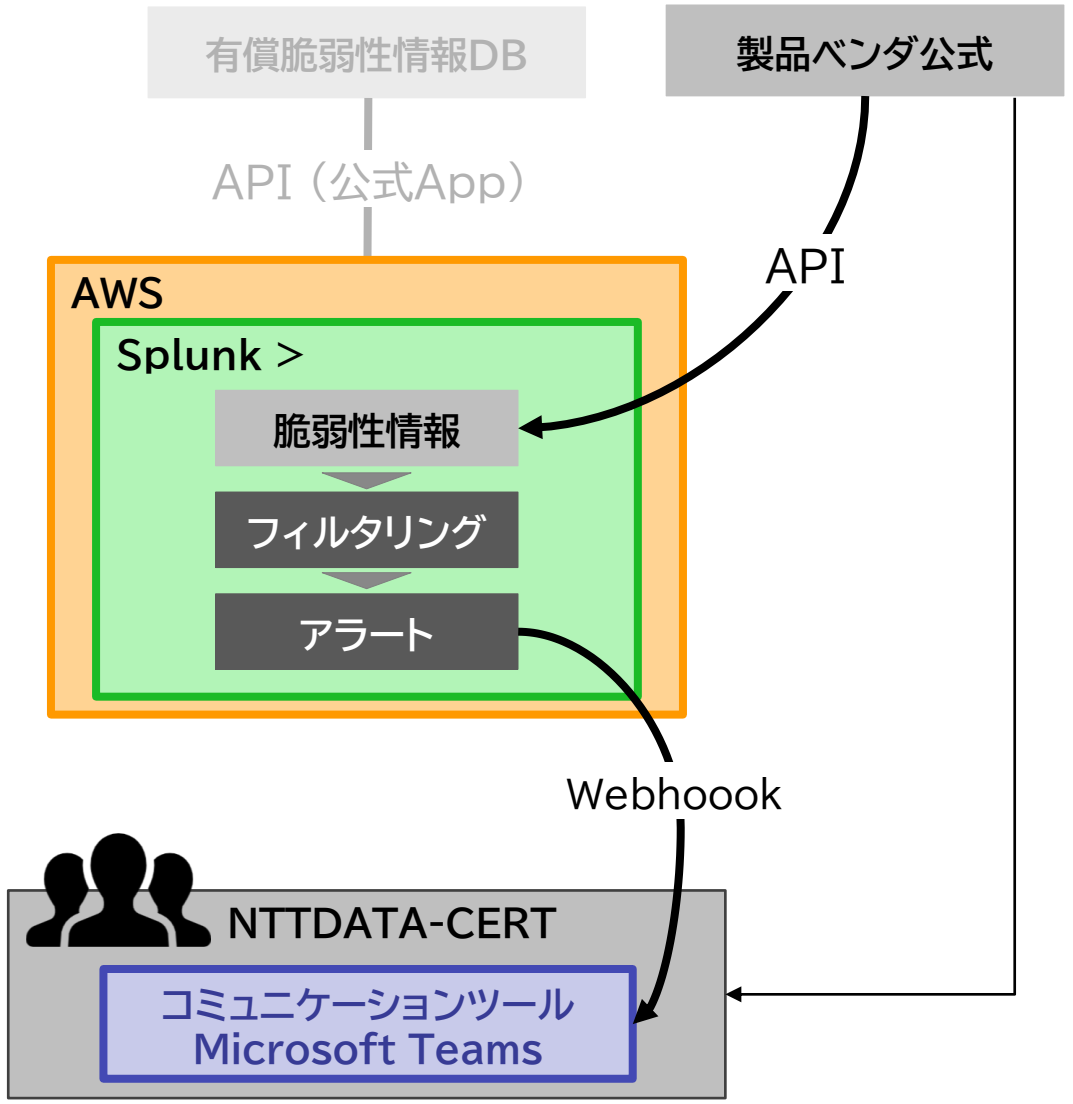
追加機能① 個別ベンダ脆弱性情報収集機能

- 公式API提供であればベンダから情報取得が最速
- 例: Microsoft製品
  - 月例更新時は100件以上の脆弱性を公開することも
    ⇒ 件数過多か有償脆弱性情報DBの更新の遅延も確認
  - Microsoft Security Response Center がAPIを提供

Powershellでレポート作成するコードサンプル:

```
Install-Module -Name MsrcSecurityUpdates -Force
Import-Module -Name MsrcSecurityUpdates -Force

$monthOfInterest = "2017-Mar"

$CVEsWanted = @(
        "CVE-2017-0001",
        "CVE-2017-0005"
        )
$Output_Location = "C:¥your¥path¥here"

$CVRFDoc = Get-MsrcCvrfDocument -ID $monthOfInterest -Verbose
$CVRFHtmlProperties = @{
    Vulnerability = $CVRFDoc.Vulnerability | Where-Object {$_.CVE -in $CVEsWanted}
    ProductTree = $CVRFDoc.ProductTree
    DocumentTracking = $CVRFDoc.DocumentTracking
    DocumentTitle = $CVRFDoc.DocumentTitle
}

Get-MsrcSecurityBulletinHtml @CVRFHtmlProperties -Verbose | Out-File $Output_Location
```

Microsoft公式GitHubより引用 https://github.com/microsoft

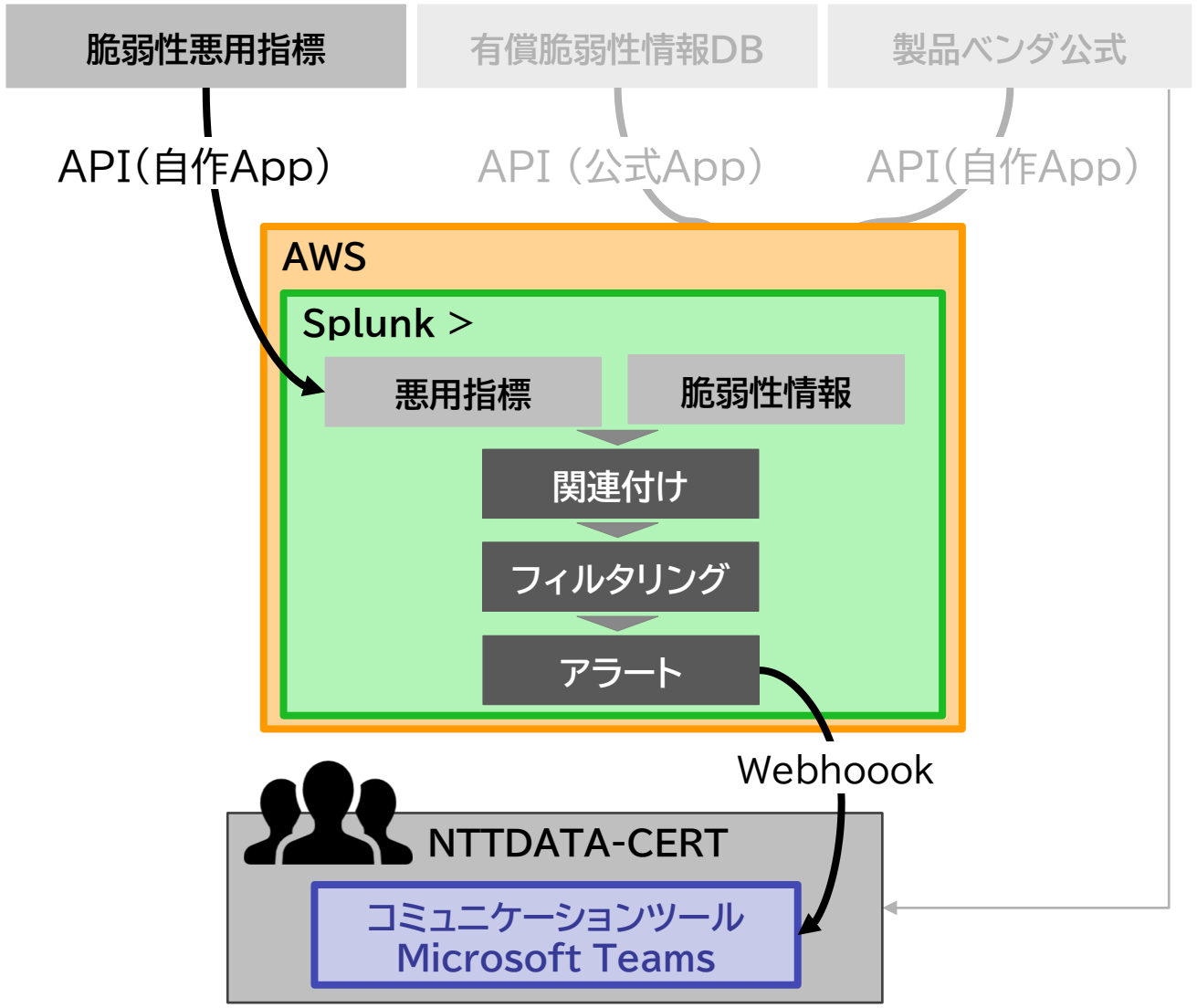


追加機能② 脆弱性悪用指標収集機能

- 脆弱性悪用指標
  - 攻撃発生有無、攻撃発生可能性を示唆する指標
  - 時間経過によって変化することもある
  - 対応判断のうえで重要な役割

- KEV（Known Exploited Vulnerabilities catalog）
  - CISA※1 が公開する悪用確認脆弱性リスト
  - すでに悪用が確認された脆弱性を確認可能

- EPSS（Exploit Prediction Scoring System）
  - FIRST ※2 が公開する悪用可能性を示す指標
  - 脅威情報と実際のエクスプロイトデータを利用
  - 0～1の確率を示す値で悪用可能性を確認可能



※1 Cybersecurity and Infrastructure Security Agency
※2 Forum of Incident Response and Security Teams

追加機能③ 対応状況収集機能

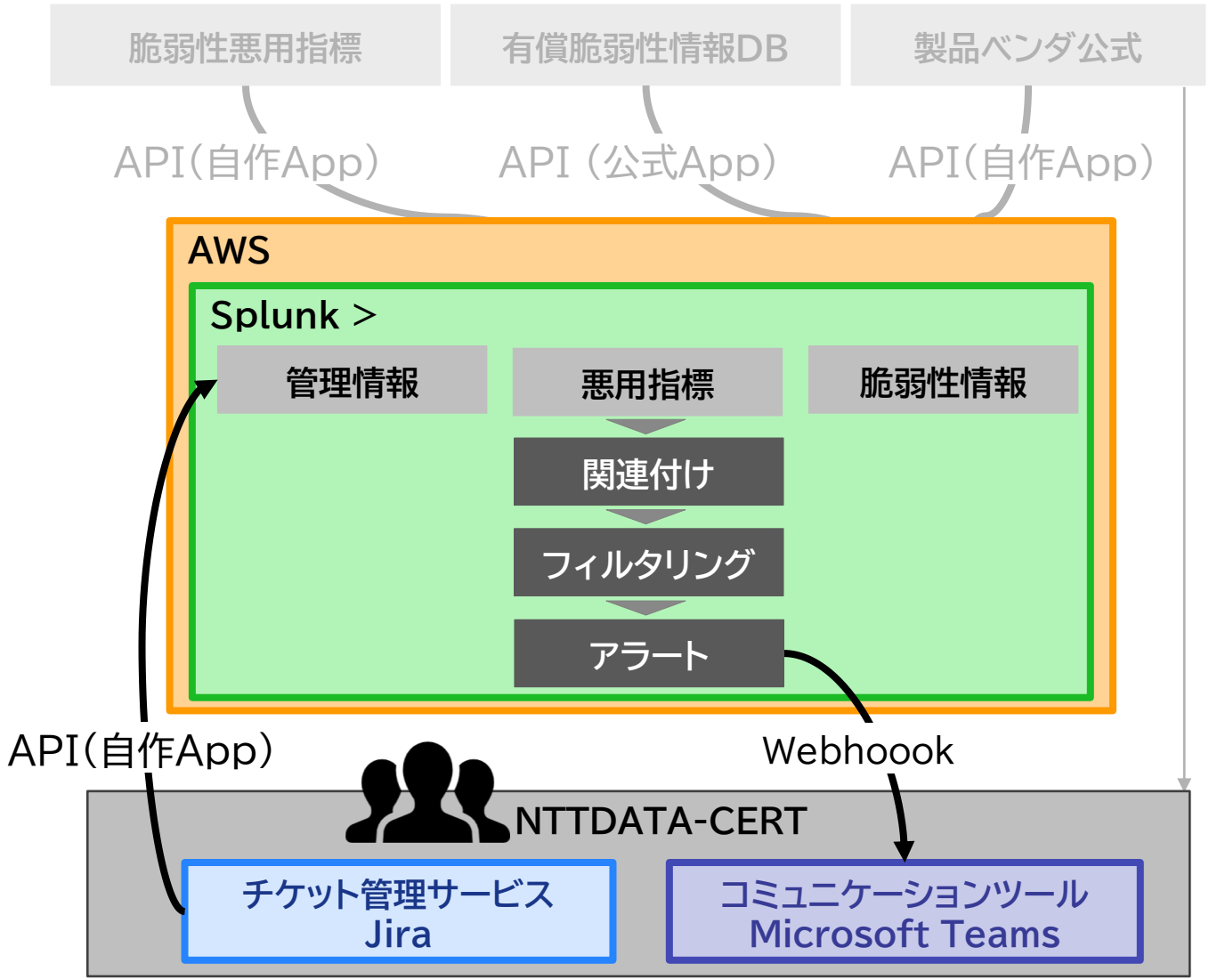
- 内部管理情報を参考にした判断
  - 脆弱性対応は時間経過で状況が変化
  - 過去の結果を判断に用いることも多数
  - 対応記録をSplunkに連携して効率化
- 脆弱性更新時にの過去の判断結果を参照
  - 更新情報はなにか？
  - いつどの脆弱性をどう判断したか

  ⇒ 対応方針が変わるかどうか判断

Jira上のイベントを取得するPythonコードサンプル

```python
import requests
from requests.auth import HTTPBasicAuth
import json

url = "https://your-domain.atlassian.net/rest/api/2/events"

auth = HTTPBasicAuth("email@example.com", "<api_token>")

headers = {
  "Accept": "application/json"
}

response = requests.request(
   "GET",
   url,
   headers=headers,
   auth=auth
)

print(json.dumps(json.loads(response.text), sort_keys=True, indent=4, separators=(",", ": ")))
```

Atlassian公式ドキュメントより引用 https://developer.atlassian.com



（参考）脆弱性悪用指標や対応結果を関連付けたアラート

CVE-2024-21762

Vendor: Fortinet

Software: FortiOS

過去に通知したCVE-IDについて、CISAによるKEVへの登録を検知しました。

KEV登録日: 2024-02-09

DueDate: 2024-02-16

[KEV URLリンク]

ランサムウェアキャンペーンでの使用実績： Unknown

CISA推奨アクション:

- Apply mitigations per vendor instructions or discontinue use of the product if mitigations are unavailable.

- 本脆弱性を判定した 2024/2/9 当時、重大度 1 として対応しています。
  - 脆弱性当番担当者は、当時の対応スレッドに、本アラートのTeamsリンクを投稿するとともに以下の依頼をしてください
    - 当該脆弱性や攻撃に関する最新の情報を調査収集してください（特に、Exploit状況やコード自体）。
    - 最新の情報を踏まえて、追加の対応要否を検討、対応を実施してください（周知情報の更新、未対応組織への催促、等）。
  - 当番として調査収集した情報があれば、同時に共有してください。

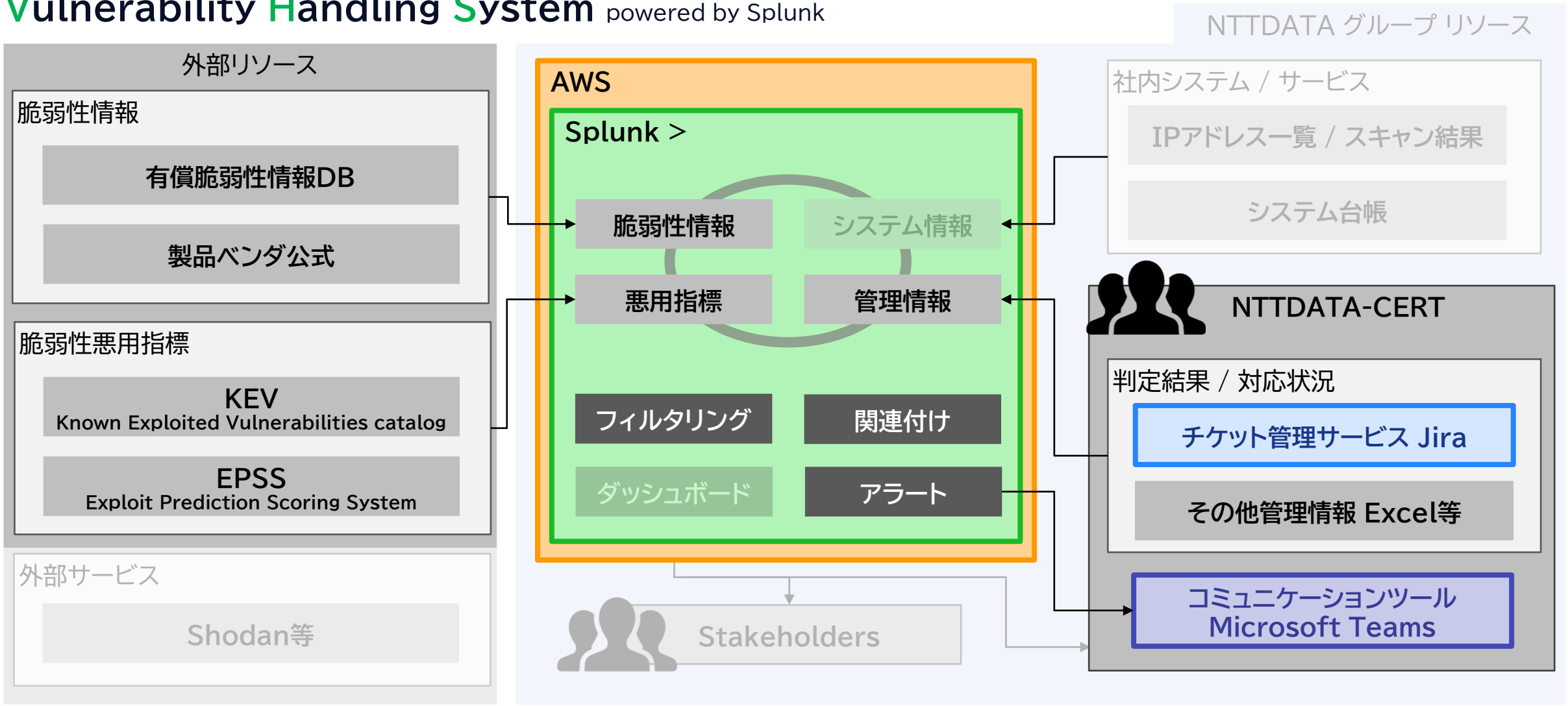
当時の判定理由(定常脆弱性確認記録」の判定結果備考欄):

RCE脆弱性。攻撃コードの存在が示唆されており、悪用可能性が高い

NTT DATA

第1章　まとめ

Vulnerability Handling System powered by Splunk



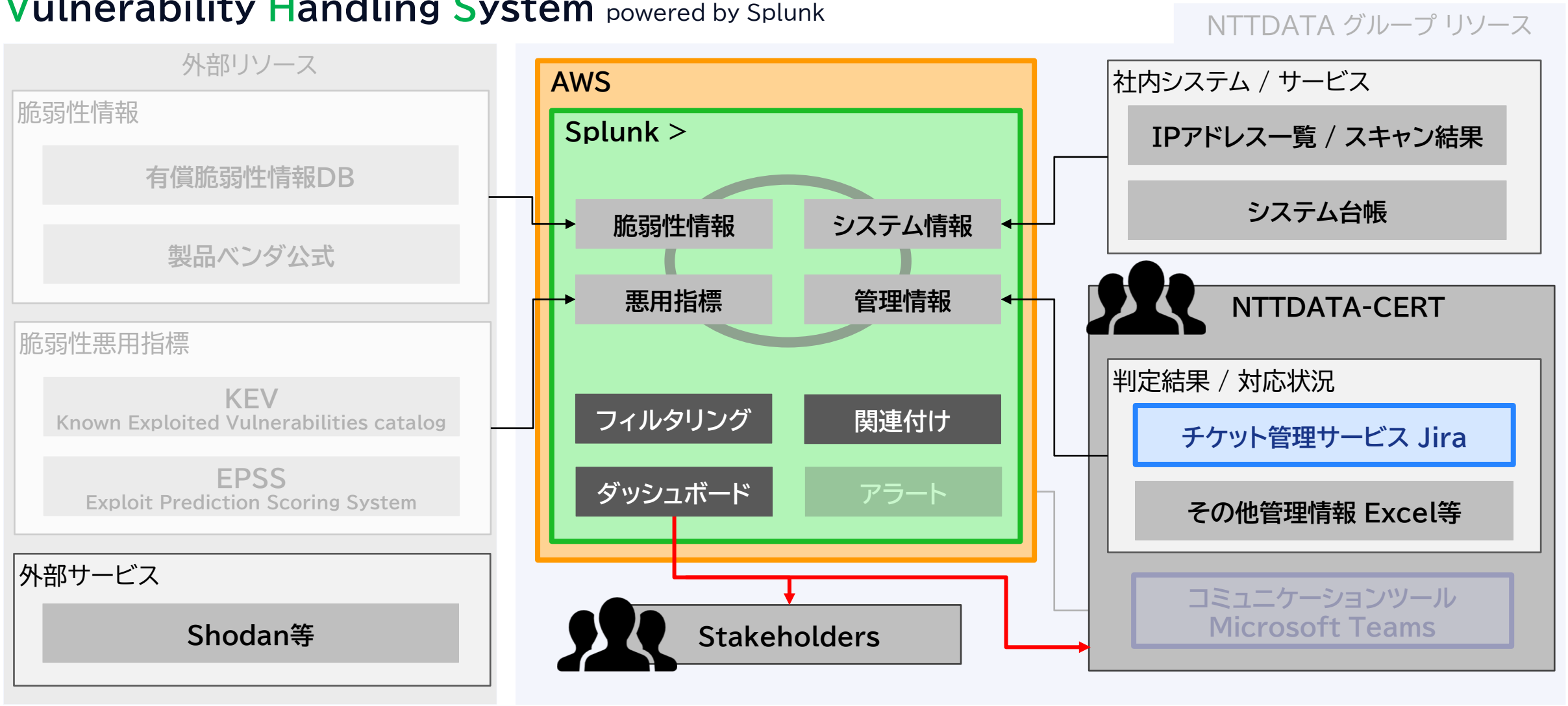
第2章

Splunkと脆弱性対応管理

対応状況の可視化

第2章　概要図

Vulnerability Handling System powered by Splunk



はじめに：なぜダッシュボードを構築したのか

改善活動の実施範囲が限られた範囲だったが、全体における取り組みを開始






平常時の対応/管理 | **緊急脆弱性周知** | **緊急対応管理**

- 平常時の情報管理
- 情報収集 → 重大度判定 → 緊急周知
- 緊急対応時の記録/情報連携

いままでの主な取り組み（～FY2022）

はじめに：なぜダッシュボードを構築したのか

改善活動の実施範囲が限られた範囲だったが、全体における取り組みを開始

平常時の対応/管理 | **緊急脆弱性周知** | **緊急対応管理**

平常時の情報管理 | 情報収集 → 重大度判定 → 緊急周知 | 緊急対応時の記録/情報連携

いままでの主な取り組み（～FY2022）

現在の取り組み（FY2024～）

SANS VMMM（Vulnerability Management Maturity Model）

- 組織全体の脆弱性対応力を評価するモデル
- 脆弱性対応の活動を12の項目に分割
- 各項目ごとに5段階の成熟度（レベル）を設定

Splunkを活用した成熟度向上

ダッシュボードを構築することで成熟度(レベル)の向上が可能

注力したい項目(2024年4月時点)

- 資産・構成管理
- 分析
- **メトリクスの可視化**

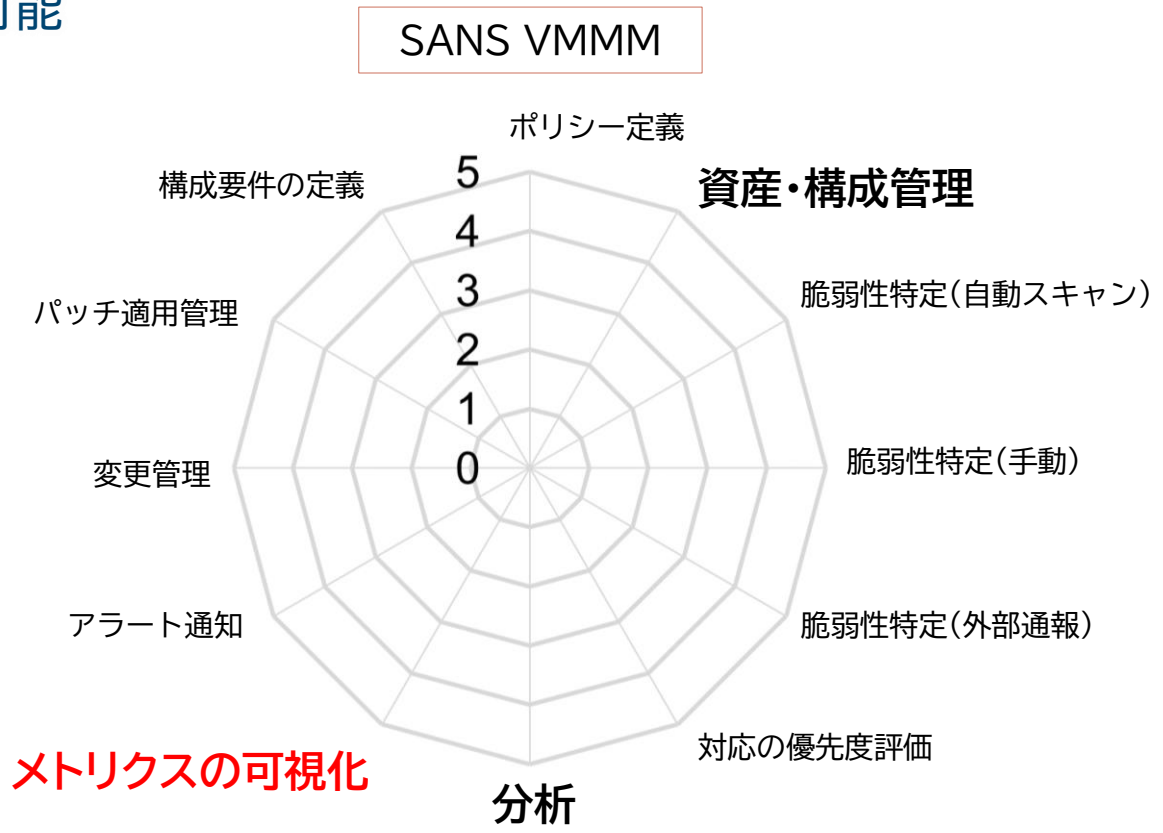
FY2024はメトリクスの可視化に注力

メトリクスの可視化とは？

脆弱性対応に関わる**データを収集、可視化**し、
リスクへの対応行動や運用改善に活かす取り組み

レベルを上げるには？

・**定期的にデータが可視化**されている
・可視化する項目が絞られている



Splunkでダッシュボードを構築することで達成できそう

ダッシュボード構築において不安だったこと

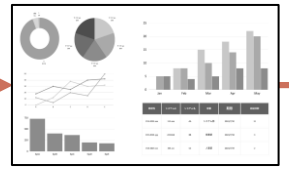
分散された情報を集め２種類のダッシュボードを構築

脆弱性情報
脆弱性情報データベース
ベンダサイト
JPCERT/CC …

外部スキャン
スキャン結果

外部サービス
ASM
Shodan …

脆弱性管理機能
VHS(Splunk)
Jira



ダッシュボード

チームメンバが担当

管理/分析用

対象：NTTDATA-CERT

- 分散された情報を活用し運用改善につながる
- リスクを継続して把握できる

自分が担当

情報連携用

対象：分野統括組織

- 必要な人への情報連携が可能になる
- 現場に近い人たちに危機感を持ってもらう

- ■ 様々な情報源のデータが必要になりそう
- ■ 何の情報をダッシュボードへ掲載すればいいんだろう
- ■ 短い時間で確認できるようにしたい
- ■ Splunk初心者の私が1人でダッシュボードを構築できるのか

ダッシュボード構築の取り組み

項目を選定してから、データを収集し、ダッシュボードを構築

項目の選定	■ 様々な資料を参照し、実装する項目を選定 ■ 必要なデータとその情報源を整理 **対象者とともに選定を進めることで「使える」ダッシュボードにつながった**
データ収集	■ 情報源からデータを収集（収集方法の検討） ■ 収集したデータをインデックス化 **分散された情報源から半自動的に定期的な収集が可能になった**
ダッシュボード構築	■ パネルのレイアウトを検討 ■ データを可視化するためのクエリの作成 ■ ダッシュボード同士を連携 **Dashboard Studioを使用することで視覚的にカスタマイズできた**

ここまでのまとめ –Splunkを用いたデータの可視化-

脆弱性管理の成熟度(レベル)を向上させるためにダッシュボード構築に取り組んだ

結果

ダッシュボード構築を完了し、レベルの向上につながった
継続して数値を監視でき、運用改善につなげられる状態になった

- 効率的に進めるには、対象者や有識者で可視化すべき項目を先に選定する必要があった
- 分散された情報源から定期的なデータ収集を半自動的に実施できるようになった
- 初心者でもほぼ独力で1からダッシュボードを構築することができた

ダッシュボードの紹介

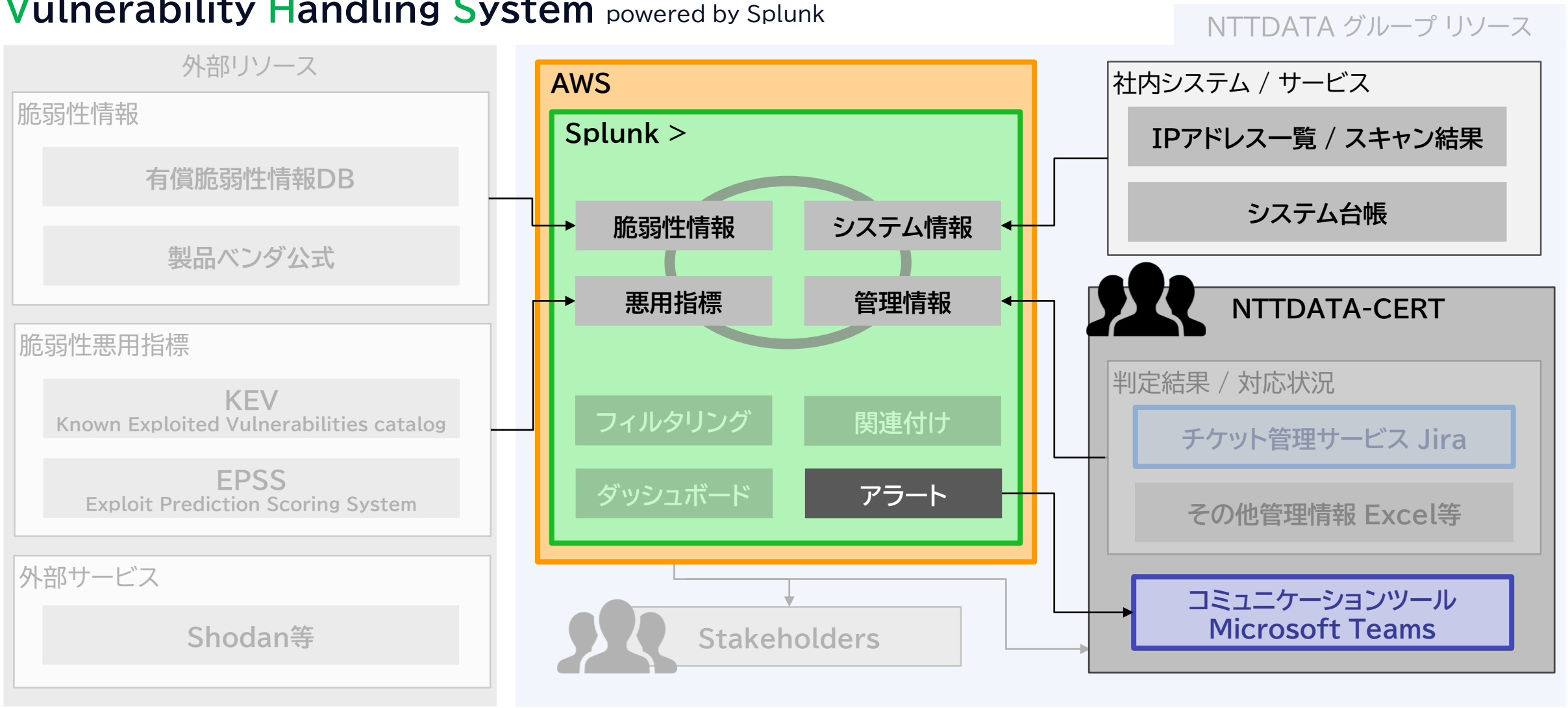
ダッシュボードのデモ画面を投影します

第3章

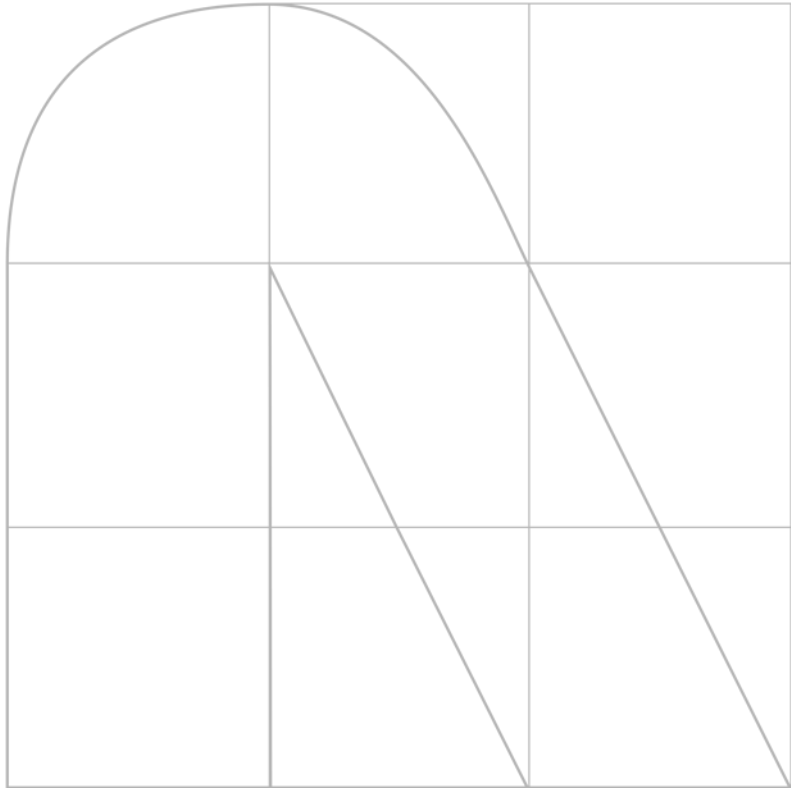
Splunkと安定稼働
仕組みとオペレーションの工夫

第3章　概要図

Vulnerability Handling System powered by Splunk



Splunkの安定稼働と
Splunk技術者の安定稼働



Splunk技術者が Single Point of Failure

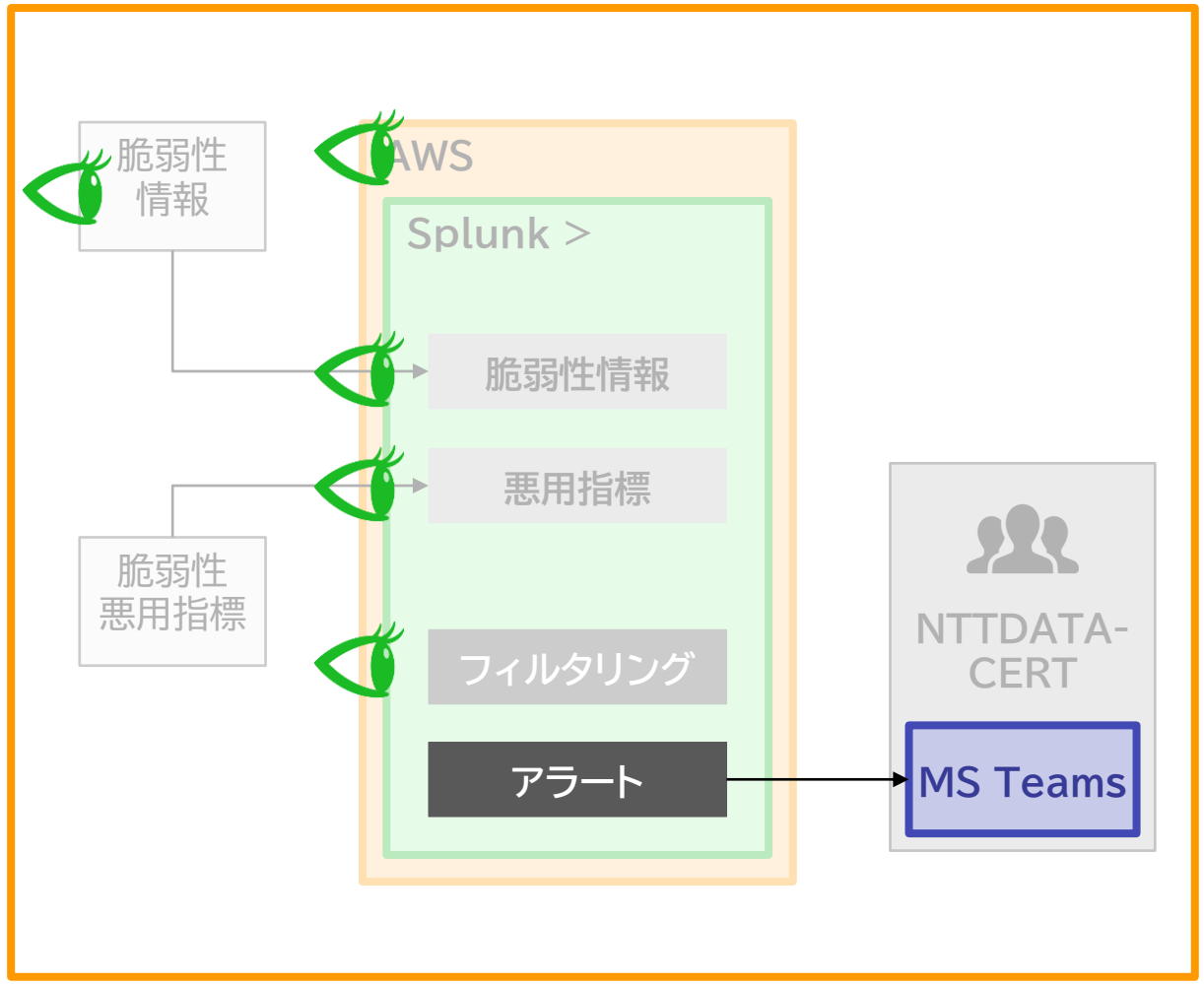
- 『このデータ、明日までに分析反映して欲しいんだけど、、、』
    - 今日は休暇で不在です
- 『データが反映されていません、、、すぐに原因調査と復旧をお願いします』
    - 原因調査と復旧を諦めて、明日に回す
    - 他のタスクを諦めて、こちらに専念
    - プライベートを諦めて、仕事に専念
    - 今の仕事を諦...

チームで起こったこと

- **新しいデータがインデックスされていない・・・**
  - 新規公開の脆弱性情報収集は、ミッションクリティカル
- **Splunk動かない・・・**
  - データ量の急増、インフラリソース許容量を超過
- **想定外のお金がかかる・・・**
  - 新機能の実装により、CPUリソース許容量を超過（AWS EC2のT系インスタンスでした）

まずは、稼働監視を実施

- **各データソースからのインデックスを監視**
  - _internal ログ中のエラーログ
    - Web API 経由で Get するデータ
  - 時間あたりのイベントインデックス数
  - APIクレジット残数
    - 商用の脆弱性DBに1日の取得上限がある
- **インフラリソースの利用状況を監視**
  - ロードバランサーと各Forwarderの疎通
  - EC2インスタンスのリソース（CPU、ディスク、etc.）
- **発生確率の高い事象が見えてくる**
  - クリティカルな監視ポイントに注力
  - 効果の薄い監視ポイントは切り捨て



アラートに対応を標準化し、負荷分散する

1. アラートごとに手順書を整備
    - 順番にやるだけで対応可能

2. アラートの文中に参照先URLリンクを掲載
    - 初見の人でも対応可能

3. Tier制を敷く
    - 負担が少数に偏らない

アラートに対応を標準化し、負荷分散する

1. アラートごとに手順書を整備する

- ローコンテクストな記載
  - できるだけ前提知識を不要に
  - 手順通りに進むだけで、調査や暫定対応を実施できる
- 条件合致で即クローズ
  - 数パターンの条件分岐を記載
- 縮退運用/緊急対応の手順も作成
  - 本来の運用が復旧できない場合でも、業務への影響を抑える

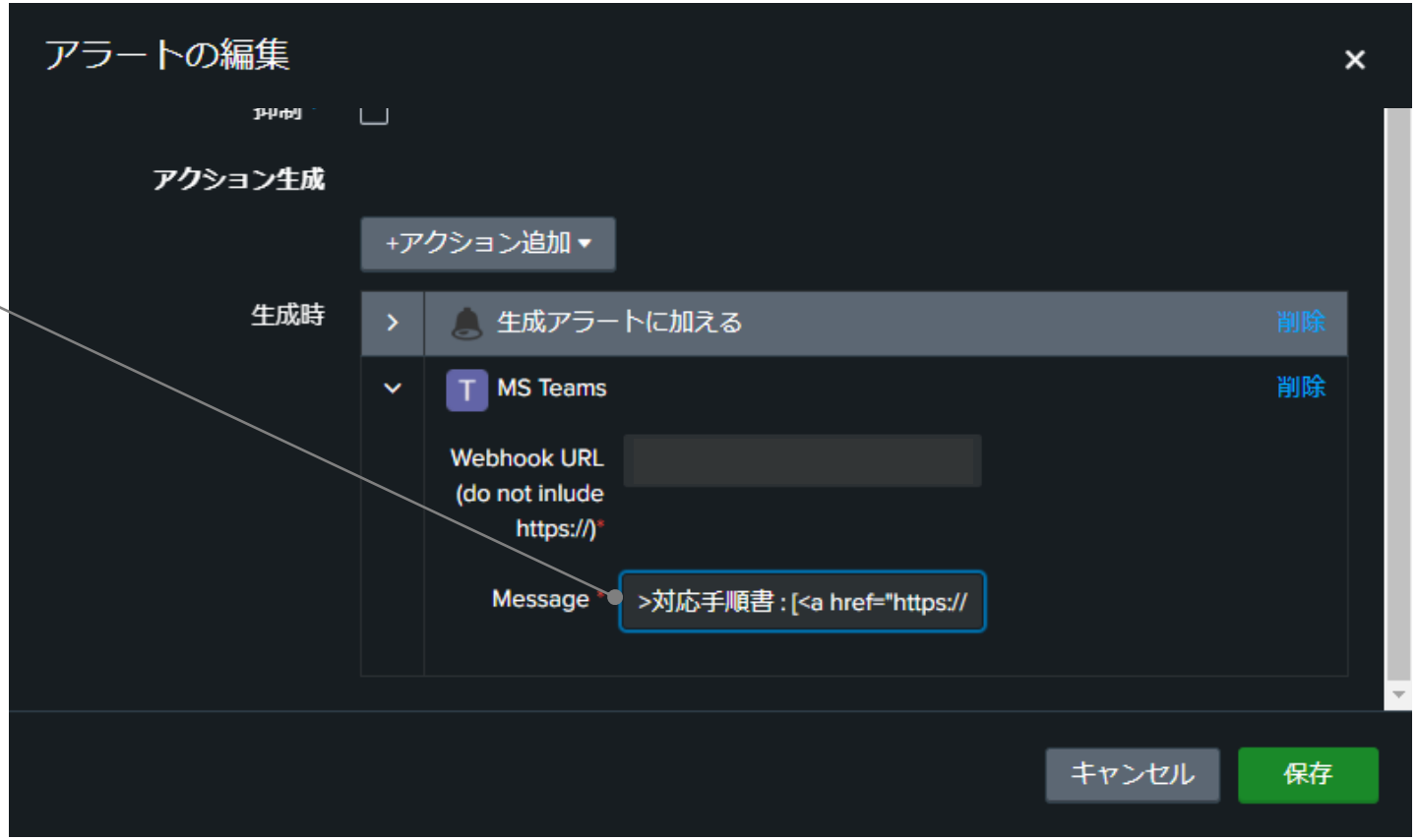
Splunk経験が浅くても対応可能

アラートに対応を標準化し、負荷分散する

2. アラートの文中に参照先URLリンクを掲載

- Splunk Alert Action WebhookでTeamsに通知
- Message をHTMLで記載
  <a> タグでURLリンクを挿入

対応手順書 : [手順_APIクレジットAlertへの対応]



アラートに対応を標準化し、負荷分散する

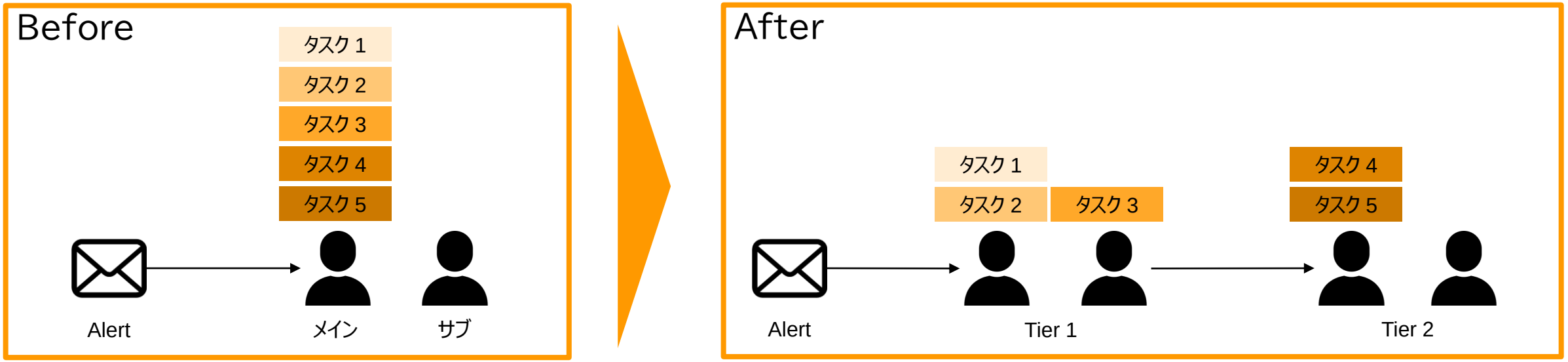
2. アラートの文中に参照先URLリンクを掲載

- 調査用のSplunkレポートを事前に作成、
  対応するURLリンクを埋め込み
- 作成した手順書は全てConfluenceにまとめ、
  対応するURLリンクを埋め込み
- 手順に沿って条件判定を実施、
  エスカレーション無しでもクローズ可能

アラートに対応を標準化し、負荷分散する

3. Tier制を敷く

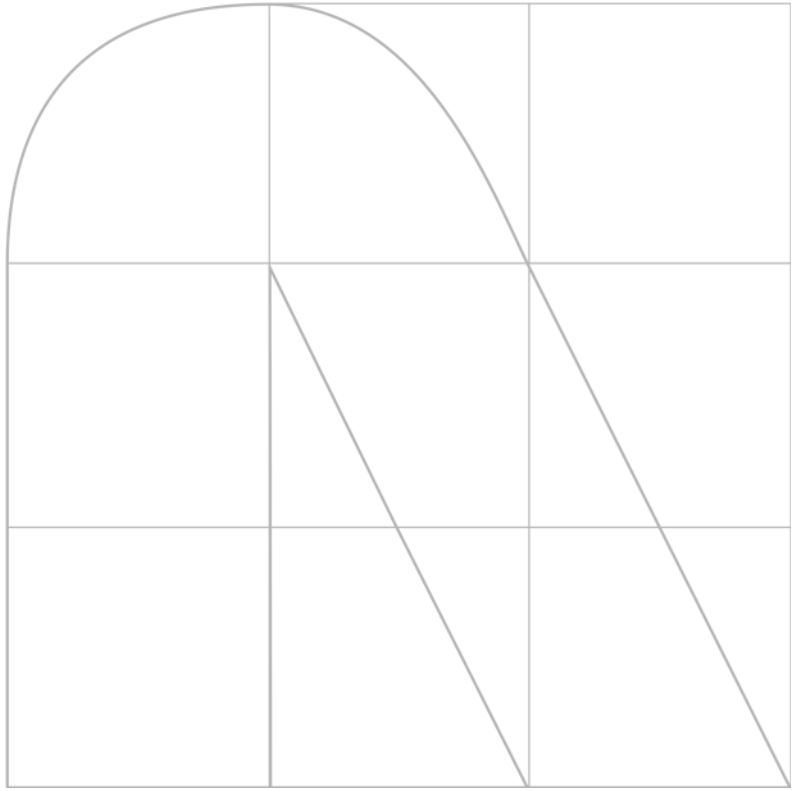
- 負担がSplunk管理者に偏らない
  - Tier 1： アラート受付、状況調査、対応要否の判断、ワークアラウンド実施、調査結果のエスカレ
  - Tier 2： 原因究明、暫定対処、脆弱性対応業務側との調整、恒久対応の計画



結局、オペレーションの設計が大事

- アラート対応の安定化を実現
    - 稼働監視アラートを実装
    - 手順書を整備
    - アラート中に、手順書や調査用SPLの直リンクを設置
    - Tier制を敷いて対応人数を増強
- 安定化のポイント： 属人化を排除
    - 一人で対応しない、仕事が一人に偏らない
    - 場を整える、プロセスを設計する
        - 業務経験、Splunk経験が浅くても対応できるように
    - ITIL® サービスオペレーションの考え方を参考
- 負担が集中しない安定した体制のためには、Splunk管理者の、もうひと頑張りが必要そうです。。。

データを手動でインデックスする煩わしさを軽減



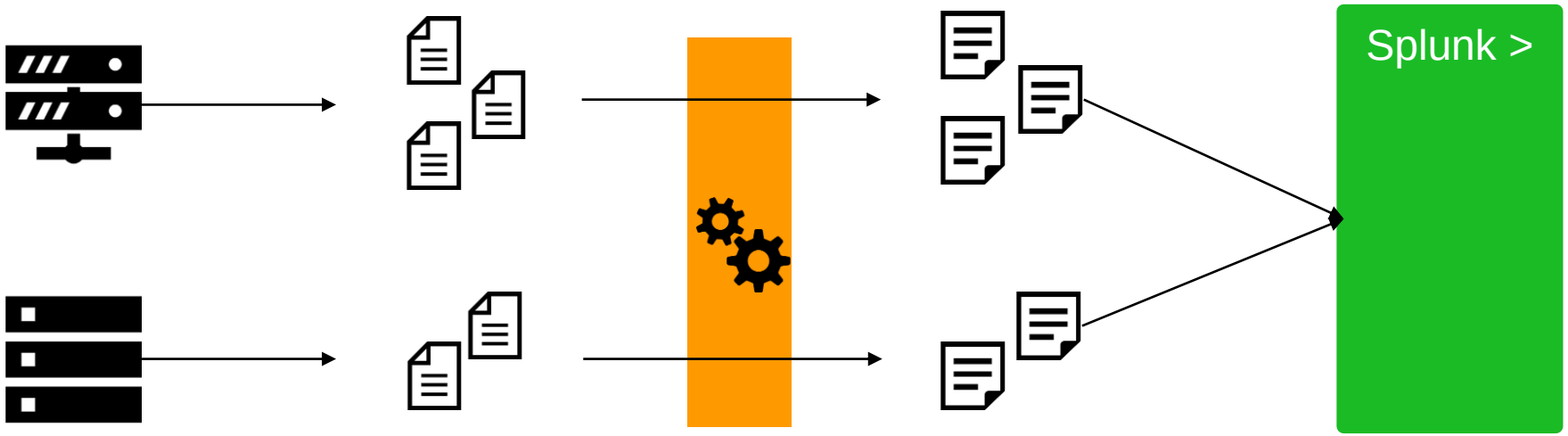
質問タイム

- 手動でインデックスしなければならないデータを扱っている/いた/扱いたい？
  （出力/転送/インデックス時に、毎回手動作業が発生するもの）

  - 自社開発の業務ツールのログファイル
  - マイナーなツールのログファイル
  - 他チームから受け取るファイル
  - 手動更新のスプレッドシート
  - 勤怠データ
  - etc.

チームの状況と課題

- 5種類のファイルを、前処理を実施した上で、Splunkへインデックスしたい
  - 2つの社内システムから、手動でダウンロード　→　　外部システムの仕様制約
  - 前処理とインデックス　→　　せめてこちらは省力化したい



データ手動連携の難しさ

- **データの受け渡し作業が発生する**
    - データの出力作業者と、Splunkへのインデックス作業者が異なる
    - Splunkの理解度にも差がある
- **前処理を実施したい**（自動連携できないデータは、データが綺麗に整形されていないことが多い）
    - CSVファイルのヘッダに日本語/数字/特殊記号が入っており、そのままフィールド名として扱えない
    - 不要な部分を Cut Off したい
    - 不正な改行が混入しており、インデックス時のイベント区切り（LINE_BREAKER）が機能しない
- **複数ファイルのインデックス**
    - 単純に繰り返し作業が面倒

データ手動連携の難しさ

- **誰が作業する？**
    - データ出力作業者： 権限の制約、Splunkに詳しくない
    - Splunk管理者： メンバーが少数に限定される
    - Splunk作業者： 実施内容やアクセス範囲の制約
- **どのようにインデックスする？**
    - Indexerで、Monitoring Directory
    - Forwarderを構築し、Monitoring Directory
    - 共有フォルダをMonitoring
    - Splunk Web UI からアップロード
    - 場合によっては、データ加工用スクリプトを作成し、毎回処理を挟む
- **毎月？毎週？毎日？ いつまで続ける？**

試した方法

Amazon S3

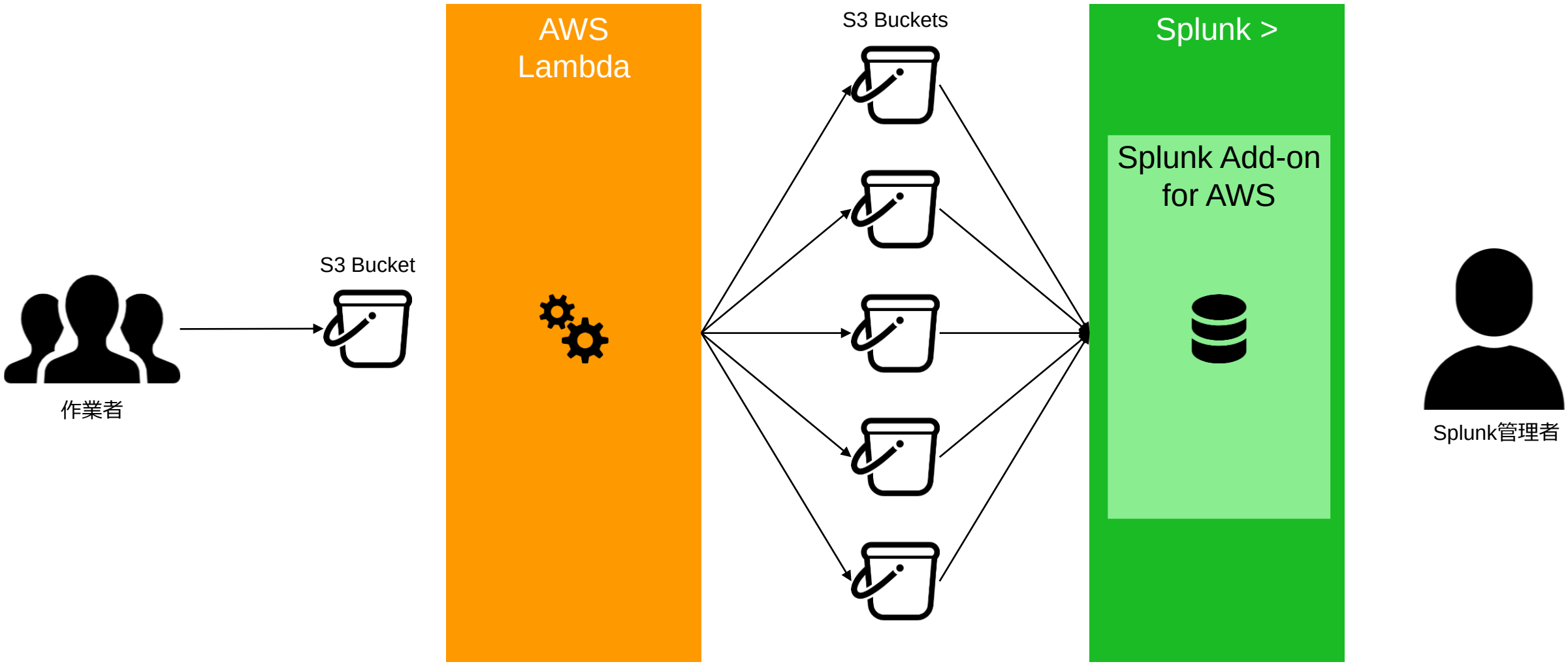
+

AWS Lambda

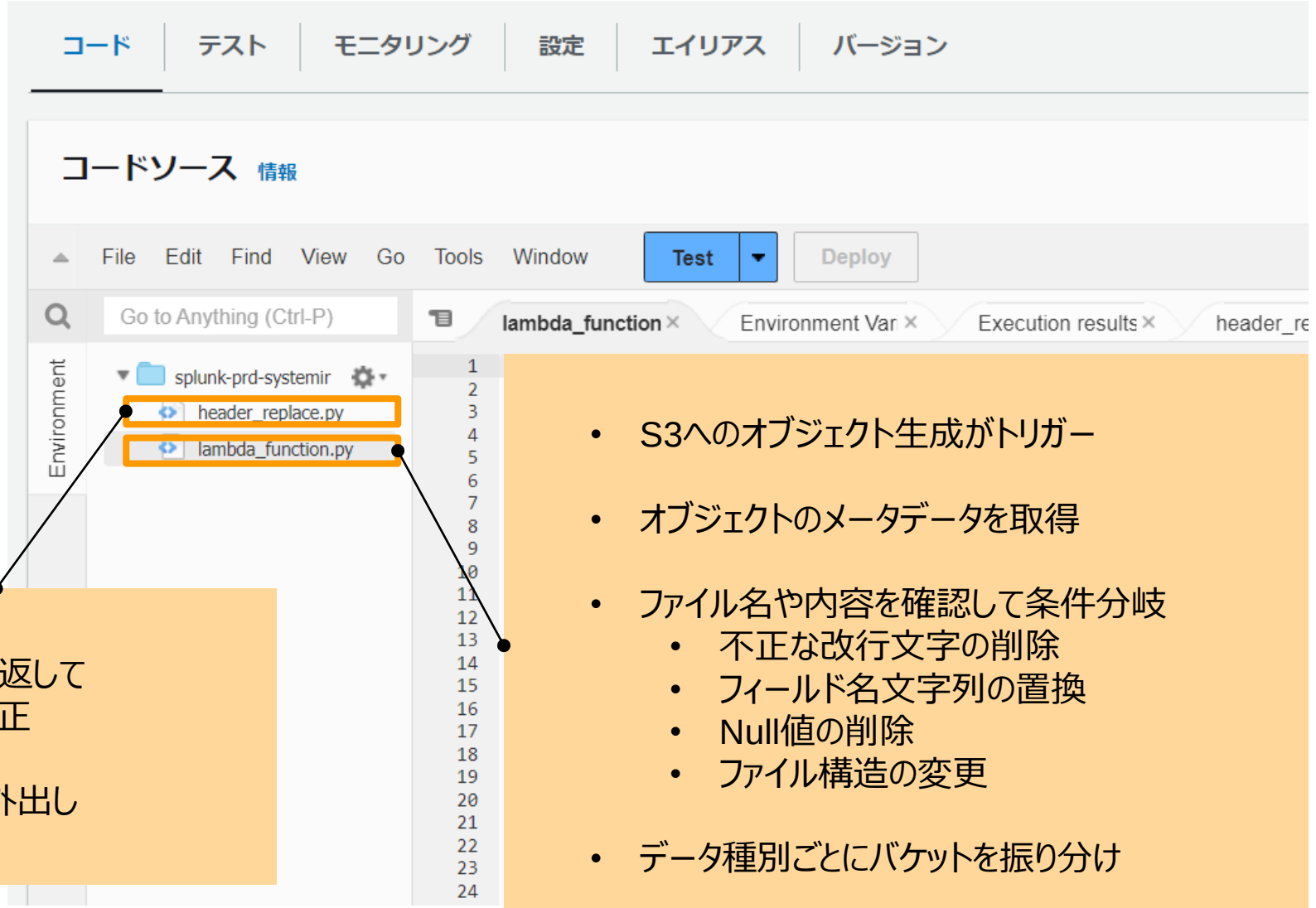
+

Splunk Add-on for
Amazon Web Services（AWS）

データフロー図



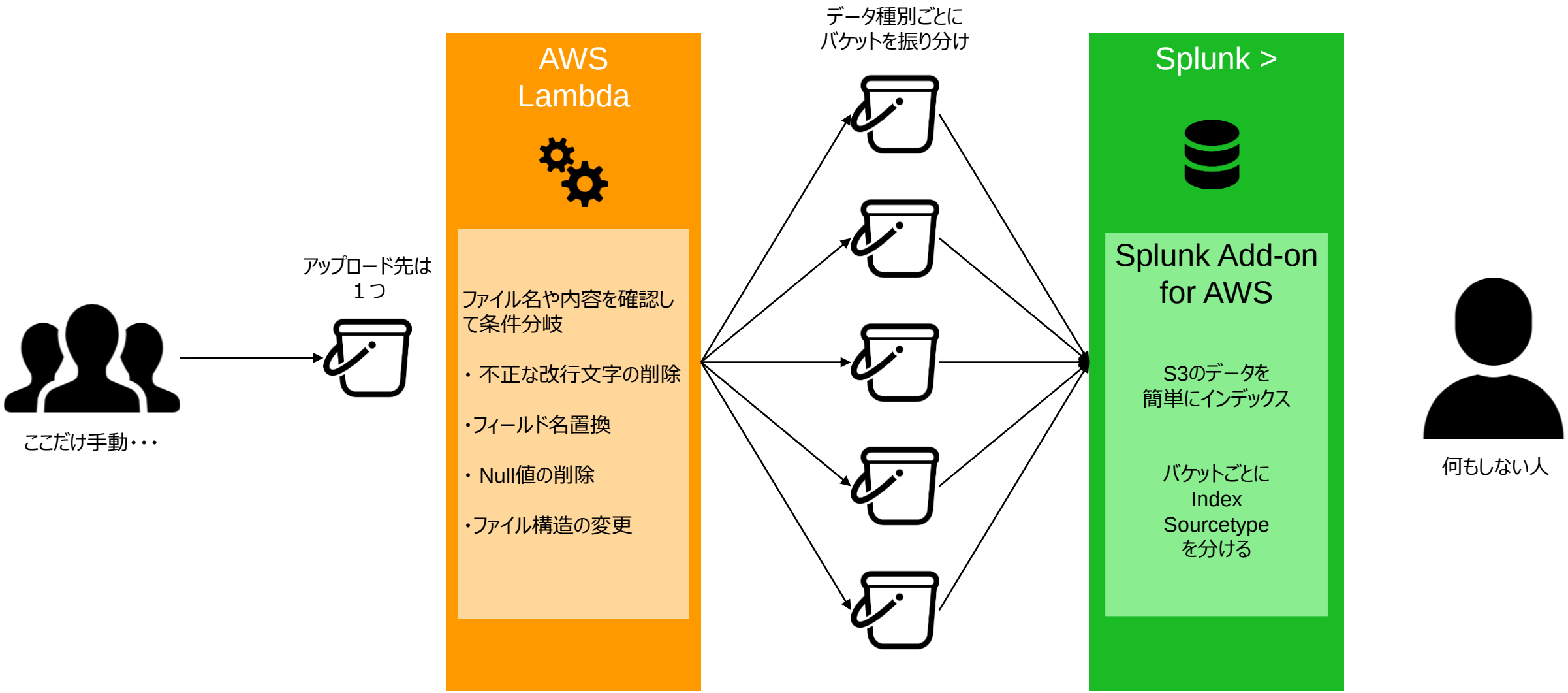
Lambdaの処理



- 置換処理を繰り返して
  フィールド名を修正
- 別ファイルとして外出し

- S3へのオブジェクト生成がトリガー
- オブジェクトのメータデータを取得
- ファイル名や内容を確認して条件分岐
  - 不正な改行文字の削除
  - フィールド名文字列の置換
  - Null値の削除
  - ファイル構造の変更
- データ種別ごとにバケットを振り分け

データフロー図（+ Lambda処理）



この処理の良い点

- 作業者からのデータアップロード先は、1つのS3バケットだけ
  - わかりやすい、間違えにくい
- 一度構築したら、Splunk管理者/作業者はその運用プロセスに入らない
- インスタンス(サーバー)本体へのログイン不要
- 処理設定が散逸しない
  - インスタンス本体(サーバー)に処理設定を直接置かない
  - データプレーンの設定はSplunk内に収める(Splunk Add-on for AWS)
  - 処理設定を置く場所をコントロールプレーンに寄せる(S3、Lambda)
    - コントロールプレーンの資産はタグ付けして管理できる

しかし、弱点もあり...
まずLambda関数を書いて、そのメンテナンスしなければならない...

終章

本日のまとめ

本日のまとめ

1. Splunkを活用した脆弱性情報の収集
    1. ベンダーサイト巡回を止め、データベース化＆通知を実装
    2. Addon Builderの活用

2. Splunkダッシュボードで脆弱性対応管理
    1. 既存のデータを統合し、対応状況を可視化、厳選した指標をモニタリング
    2. Dashboard Studioの活用

3. Splunkの運用効率を上げるための工夫
    1. オペレーションを設計してSplunkの管理負担を分散
    2. S3 + Lambda + Addon for AWS で、データ手動インデックスを楽に

質問や感想があれば、お寄せください！
脆弱性の話、可視化の話、Splunkの話、、、


NTT DATA