IPA Better Life with IT

ホワイトハッカー勉強会 初級編
～脆弱性調査ツールの解説と法の動向～

独立行政法人情報処理推進機構（IPA）
セキュリティセンター

2018年9月26日(水)

目次

1. 脆弱性発見手法の学習
2. 脆弱性の評価方法
3. 脆弱性情報の取扱いについて

1. 脆弱性発見手法の学習

独立行政法人情報処理推進機構（IPA）
セキュリティセンター

本プログラムの概要

「**ソフトウェア製品**」を対象とした脆弱性の発見手法の学び方を解説し、以下3点について理解を深める

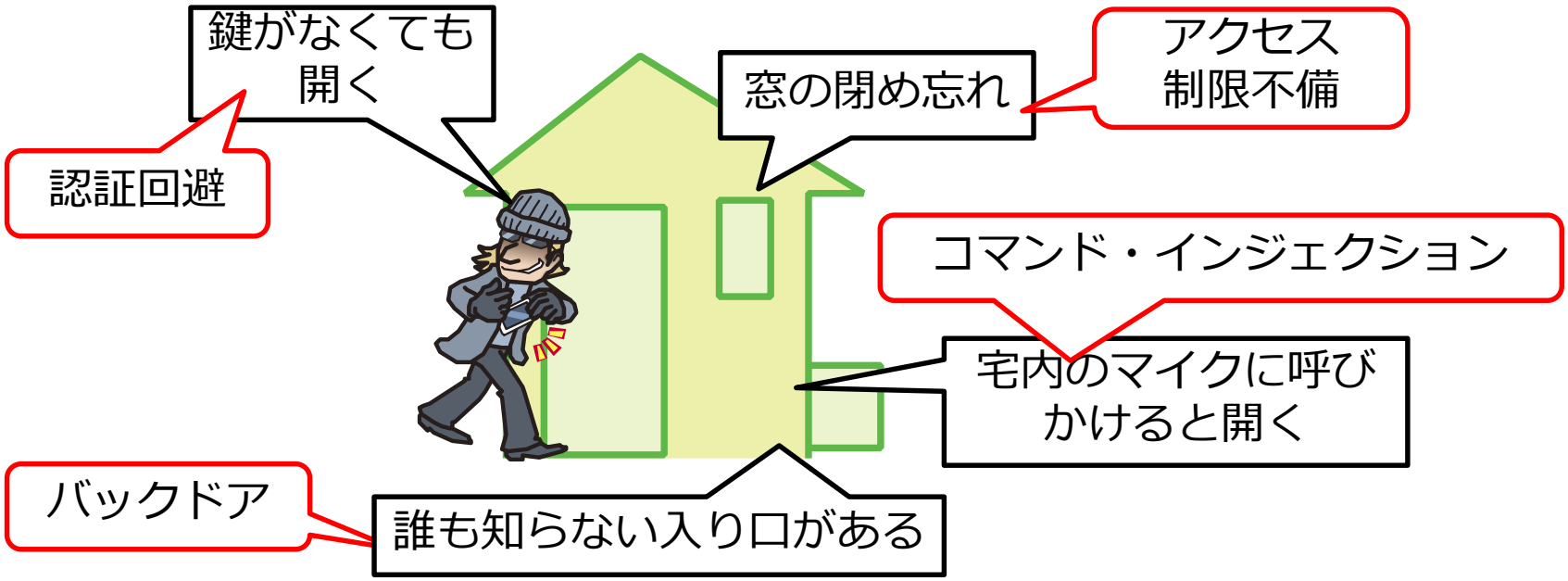
- 検証環境の作り方
- 既知の脆弱性の調べ方
- 脆弱性の検証方法や学習方法

本プログラムの内容はソフトウェア製品を対象としたものです。解説した内容を決してウェブサイトに対して実施しないでください。

目次


脆弱性ってどこにあるの？
どうやって見つけるの？

1.1 脆弱性を見つけるためには①



あらかじめ、どのような攻撃が可能か調査し、
攻撃方法を想定する必要がある

1.1 脆弱性を見つけるためには②

脆弱性はどこにあるのか？

外部API？

動的表示箇所？

例：検索欄、URLパラメータ 等

どうやったら発見できる？

ソースコードの解読？

手あたり次第に値を入力？

例：<script>alert(“1”)</script> 等

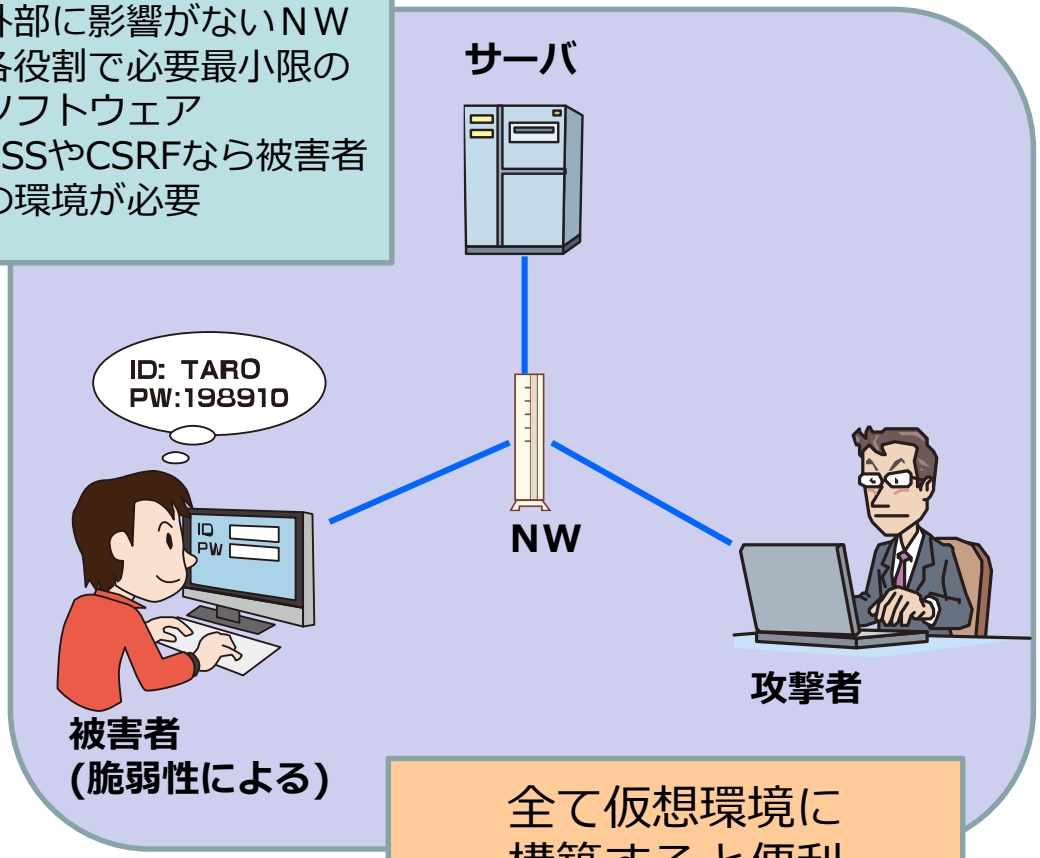
ご自身で既知の脆弱性を検証し、検証の経験を積むことで勘所を養いましょう！


脆弱性を見つけるには
どんな環境を用意するの？

1.2 検証環境の構築
検証環境（イメージ図）

- 検証対象以外に影響が生じない環境が必要
- できれば、自由に既定の状態に巻き戻したい

・外部に影響がないＮＷ
・各役割で必要最小限のソフトウェア
・XSSやCSRFなら被害者の環境が必要



全て仮想環境に構築すると便利

導入ソフトウェア例

〇攻撃者役
・Windows10
・ブラウザ
・パケット解析ツール
…etc

〇サーバ役
・Windows ServerあるいはLinux OS
・XAMPP
　- PHP
　- Apache
…etc

簡易にウェブサーバ環境を構築できる

〇被害者役
・Windows10

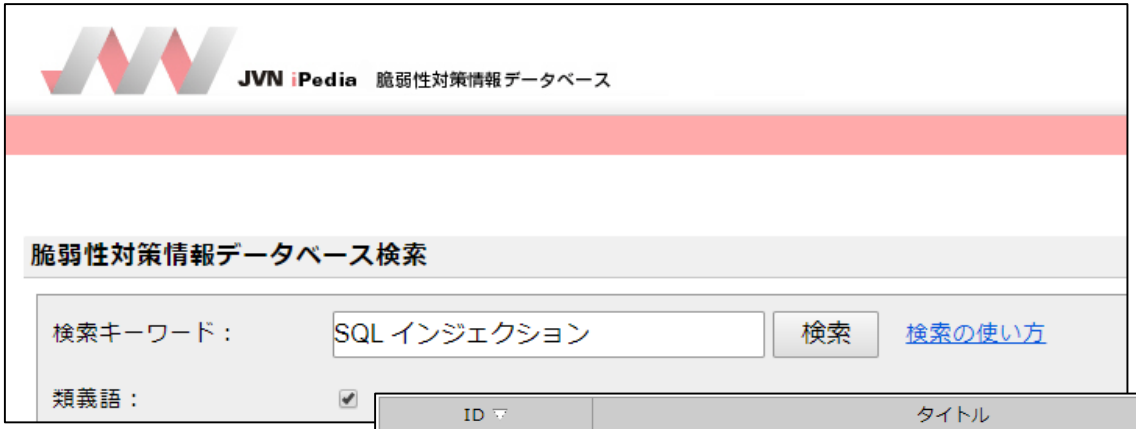
・クライアント端末やサーバにインストール
・構成変更が容易

〇その他
・VM(仮想環境)上にすべてを構築することも可能
例：vSphere、VirtualBoX


脆弱性の検証はどうやって
実施するの？

1.3 製品での検証
検証する脆弱性を探す

- 過去の脆弱性はJVN iPediaやCVEにて確認できる
  - JVN iPedia:https://jvndb.jvn.jp/
  - CVE　　:https://cve.mitre.org/index.html



ID	タイトル	CVSSv3	CVSSv2	公表日	最終更新日
JVNDB-2018-007249	Centreon および Centreon Web における SQL インジェクションの脆弱性	9.8	7.5	2018/06/25	2018/09/12
JVNDB-2018-007234	Schneider Electric U.motion Builder Software における SQL インジェクションの脆弱性	8.8	6.8	2018/04/05	2018/09/12
JVNDB-2018-006959	Schneider Electric U.motion Builder Software における SQL インジェクションの脆弱性	8.8	6.8	2018/04/05	2018/09/05
JVNDB-2018-006926	zzcms における SQL インジェクションの脆弱性	9.8	7.5	2018/07/03	2018/09/04

今回は二種類の脆弱性の再現検証について紹介

1.3 製品での検証
検証する脆弱性

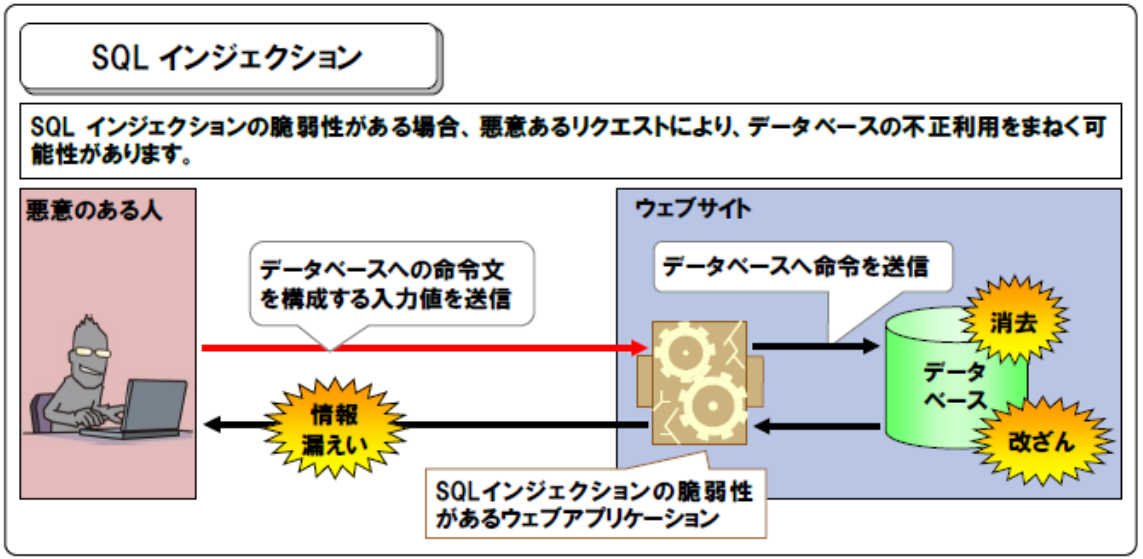
今回は下記の2つの脆弱性の検証について紹介します。

- SQLインジェクション
- ディレクトリ・トラバーサル

1.3 製品での検証①
SQLインジェクションとは

- SQL文の組み立て方法に問題がある場合、外部から攻撃によってデータベースの不正操作ができてしまう

被害の具体例：
- ログインフォームに悪意あるリクエストを送信されることで権限がない人にログインされてしまう
- データベースにある非公開情報を閲覧されたり、データベースを改ざん、消去されたりしてしまう



1.3 製品での検証①
紹介する検証シナリオについて

SQLインジェクションの検証は下記のシナリオと仮定して解説します。

- JVN iPediaでSQLインジェクションの脆弱性があったCMS製品を発見
- 詳細な再現手順がJVN iPediaの記載内容からは確認できない
- 脆弱性が存在する箇所を調査する
- 検証環境の構築から始める

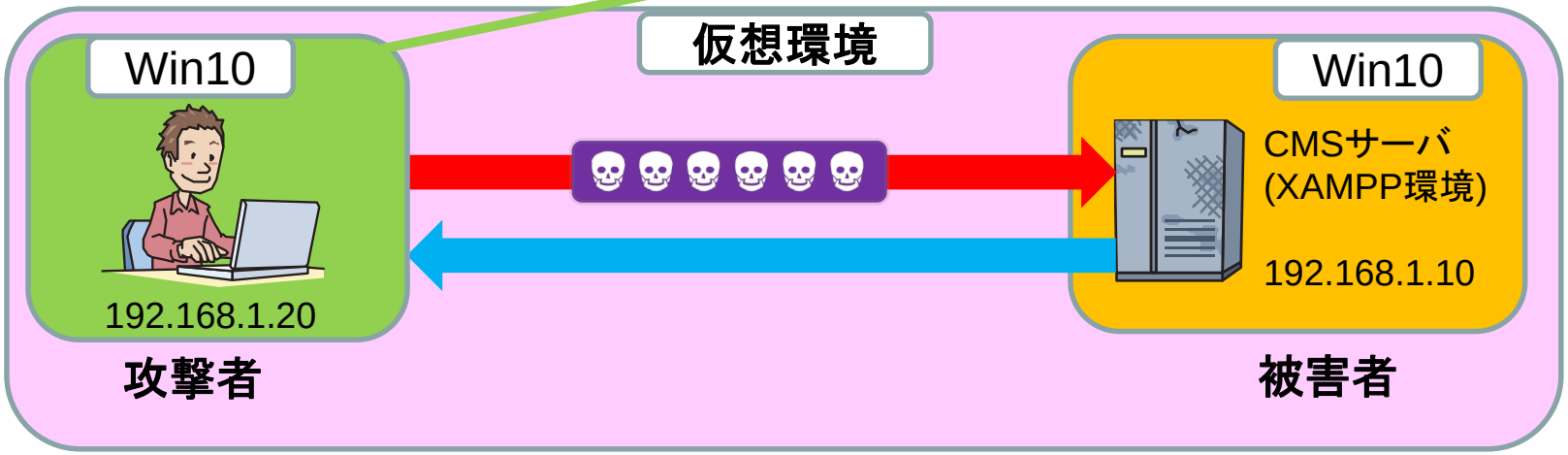
1.3 製品での検証①
検証環境の作成

前提：
- 対象はCMS製品
- XAMPPを使用して環境を構築
- 脆弱性がある過去のバージョンをダウンロード可能

JVN iPediaで公開されている
脆弱性はすでに修正済み

検証環境：
- 仮想環境上に被害者環境と攻撃者環境を構築する
- 被害者環境として仮想環境にWin10をインストール
- XAMPPをWin10にインストール
- XAMPP環境にCMSをインストール
- 攻撃者環境としてWin10をインストール

SQLインジェクションの検証は一台でも可能だが、実際の攻撃を想定した環境を構築



1.3 製品での検証①
発見手順

基本動作の確認：

1．http://192.168.1.10/cms/にアクセス
　→テストページが表示される
2．検索欄でtestと検索してみる
　→下記のURLパラメータを持つページが表示される

(省略)/kensaku?cf=&kw=test&s=0

検索ワード

- SQL文の組み立て方法に問題がある場合、不正な処理が実行されるのが、SQLインジェクション

検索ワードがSQL文に組み込まれていないだろうか。

1.3 製品での検証①
発見手順

脆弱性の調査：

1．検索欄で‘と検索してみる。もしくは、URLパラメータの kw の値に‘を入力する
→データベースからのエラーが表示された

データベースに異常なSQLクエリが送られている

2．以下のようなSQL文を上記と同様に入力する
‘ OR 1=1
‘ union select unhex(hex(version())), ¥0 – ‘x‘=‘x
→データベース内の情報やバージョン情報が表示された

実際にどのようなSQL文ならば実行できるかについては、ソースコードを分析したり、様々なSQL文を入力して確認するしかない

どのようなSQL操作が可能かは、アプリケーション等の実装による

1.3 製品での検証①
再現に失敗したら

1．’を入れてもエラーが出ない。
→ウェブアプリケーション側でエラーを表示させない仕様の可能性

2．SQLインジェクションが成功しない
→ウェブアプリケーションや、プログラムの実行環境等のバージョンで対策されているために失敗している可能性

3．etc...

なぜ失敗しているかは、検証をこなして覚えていくしかない。

1.3 製品での検証①
検証結果と対策

- SQL文を含めたリクエストを送信することで、データベースの内容やバージョン情報が表示された
  →送信内容がSQL文として実行される
  →管理者以外が外部からデータベースを不正に操作
  →脆弱性と考えられる

- 検証では情報の出力を意図したリクエストだが、SQL文の記載次第でデータベースの改ざんや消去が発生する可能性がある

- 対策として送信する値をSQL文として解釈しない実装が必要

1.3 製品での検証②
検証する脆弱性

- 過去に届け出されて修正済みの
  ディレクトリ・トラバーサルの脆弱性を検証して確認

公開日：2018/08/06　最終更新日：2018/08/06

JVN#62121133
アタッシェケースにおける複数のディレクトリトラバーサルの脆弱性

概要
アタッシェケースには、複数のディレクトリトラバーサルの脆弱性が存在します。

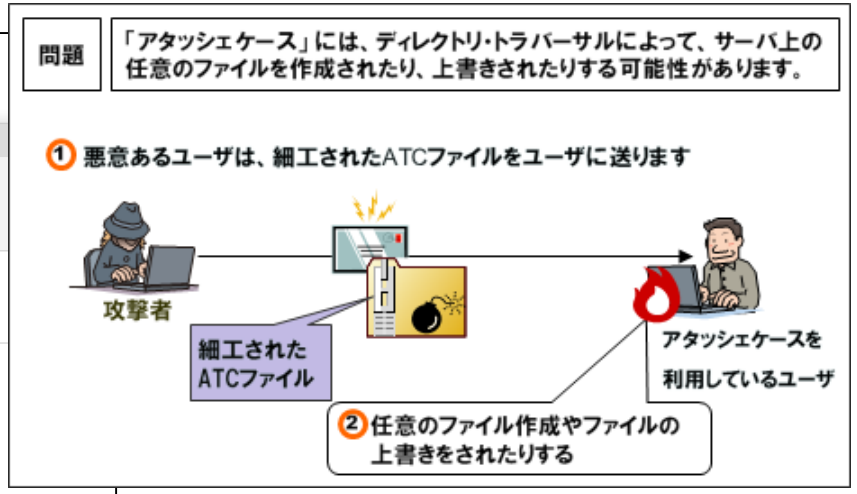
影響を受けるシステム

- アタッシェケース ver.2.8.3.0 およびそれ以前
- アタッシェケース ver.3.2.3.0 およびそれ以前

詳細情報
HiBARA Software が提供するアタッシェケースは、オープンソースのファイル暗号化ソフトです。アタッシェケースには、ATC ファイルに含まれるファイル名の処理に起因する、ディレクトリトラバーサル (CWE-22) の脆弱性が存在します。

想定される影響
想定される影響は各脆弱性により異なりますが、細工された ATC ファイルを復号することで、次のような影響を受ける可能性があります。

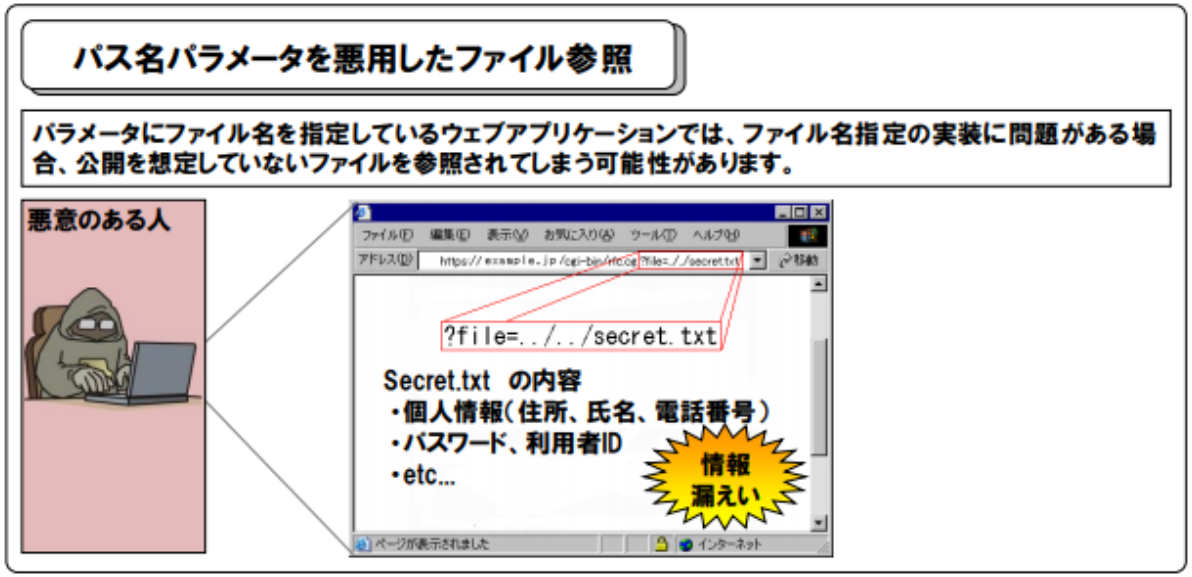
- 任意のファイルを作成されたり既存のファイルを上書きされたりする - CVE-2018-0659
- 任意のファイルを作成される - CVE-2018-0660



1.3 製品での検証②
ディレクトリ・トラバーサル(DT)とは

- ファイル名指定の実装に問題がある場合、攻撃者に任意のファイルを指定され、意図しない処理を行ってしまう

被害の具体例：
- ../../等の記号を用いてディレクトリをさかのぼり、意図しないディレクトリにファイルを作成されたり、非公開ファイルを参照されたりしてしまう



?file=A.pdf

?file=../B.pdf

1.3 製品での検証②
アタッシェケースについて

- ファイル/フォルダ暗号化に用いるクライアントソフト
- フリーでダウンロード可能
- オープンソースソフトウェア

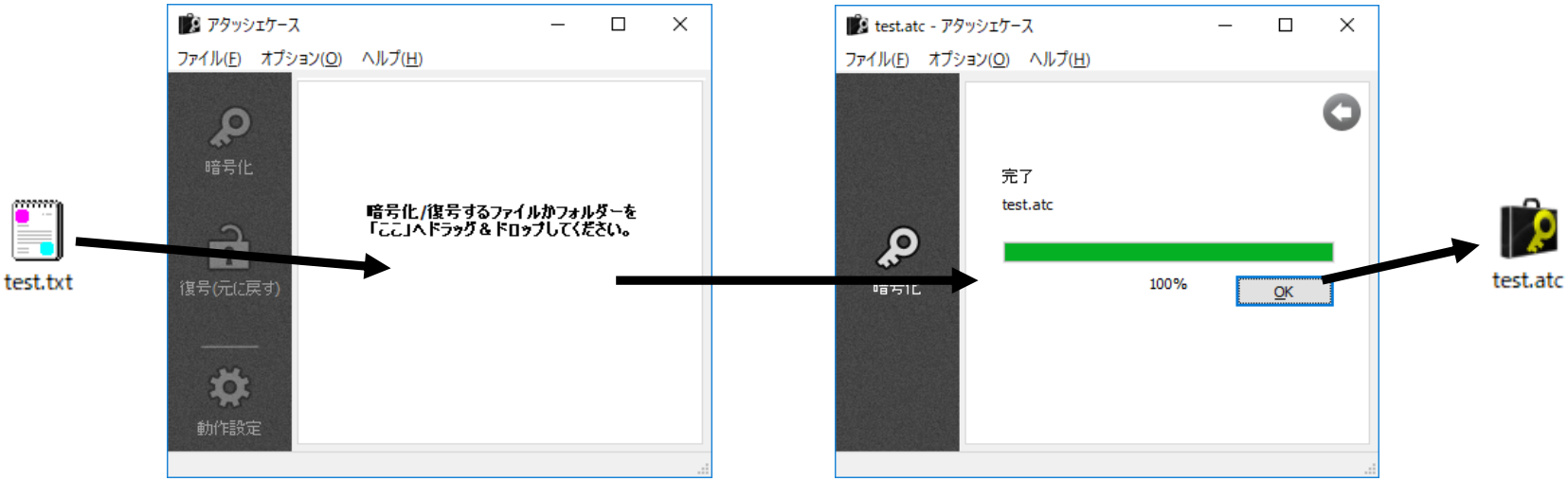
1.3 製品での検証②
開発者による脆弱性情報の公開

- 開発者が脆弱性の修正後、個人のブログで詳細を公開

https://hibara.org/software/attachecase/vulnerability/

1.3 製品での検証②
アタッシェケースについて

検証環境：
- デフォルト設定でWin10 Proにインストール
- 今回は脆弱性があるver. 2.8.2.8

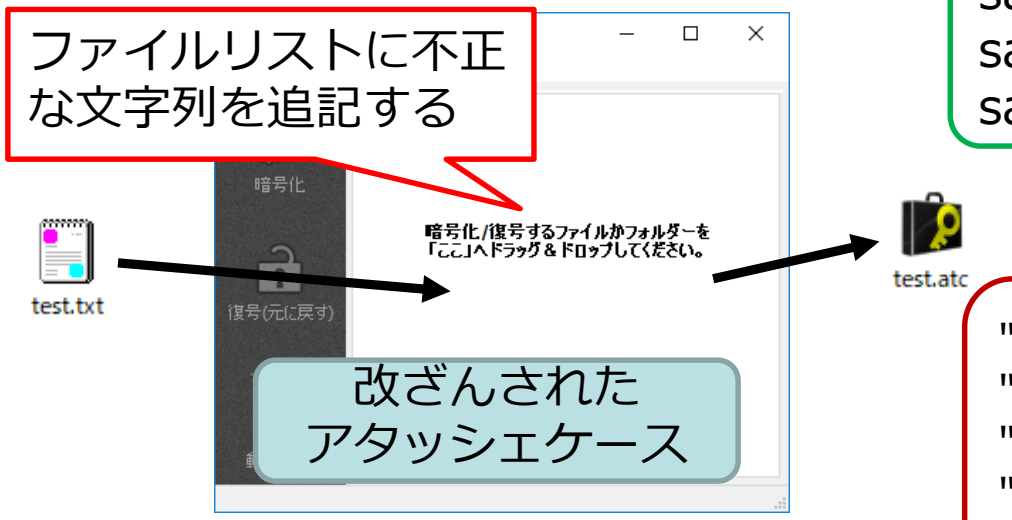


基本動作の確認：
- ファイルをドラッグ &ドロップで暗号化/復号できる

1.3 製品での検証②
検証

事前準備：

- アタッシェケースを改ざんし、細工したATCファイル（暗号化ファイル）を作成する。
  (オープンソースなので、自身で変更しコンパイルする)



sample.txt
sample1.txt
sample2.txt

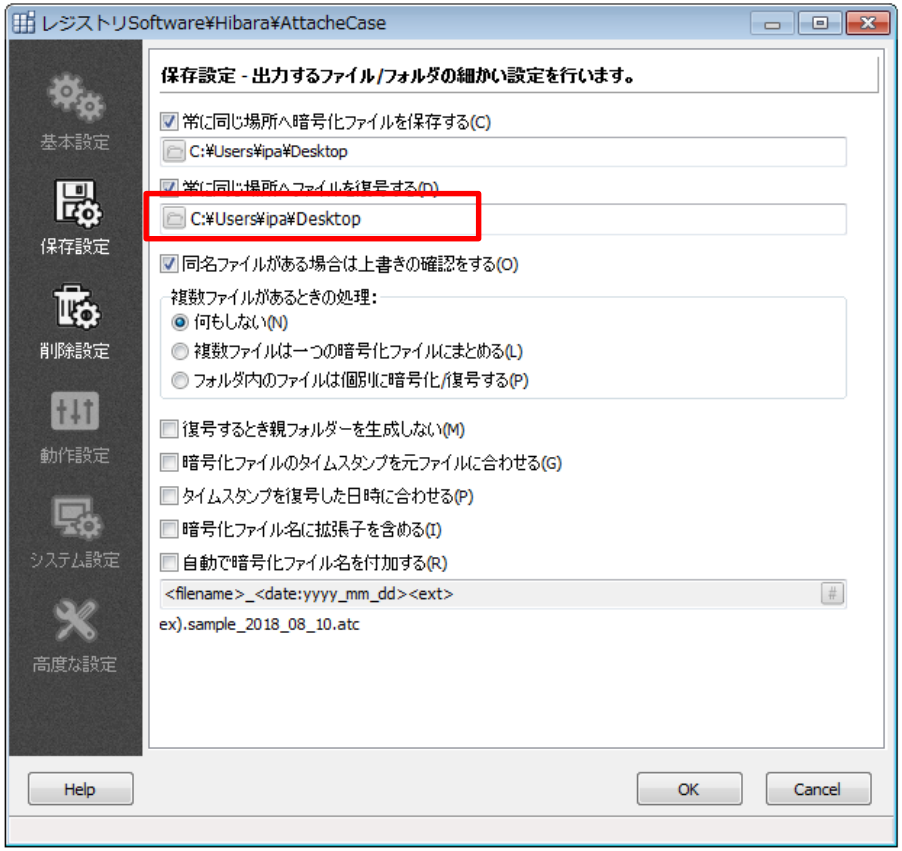
":¥Windows¥Temp¥sample.txt"
":　　¥Windows¥Temp¥sample.txt"
"　:¥Windows¥Temp¥sample.txt"
"z　:¥Windows¥Temp¥sample.txt"

- 今回の検証では、指定したディレクトリの一つ上にcalc.exe を作成するように細工

1.3 製品での検証②
検証

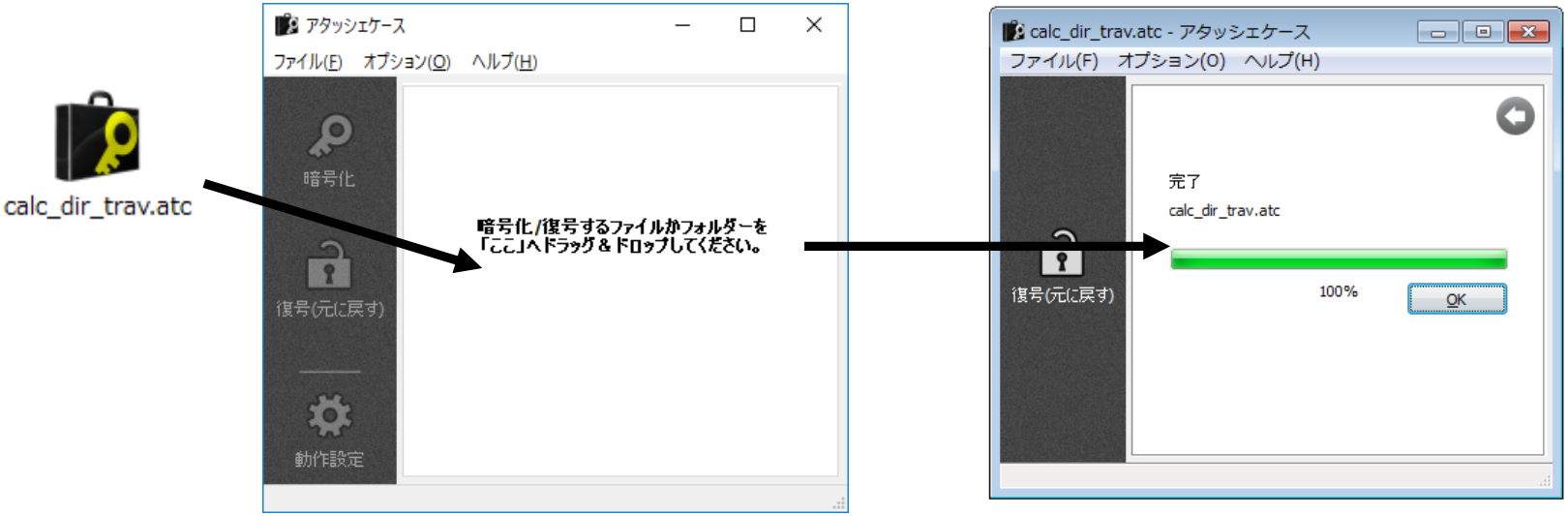
脆弱性検証：

1. 復号したファイルがC:¥Users¥IPA¥Desktopに保存されるよう設定



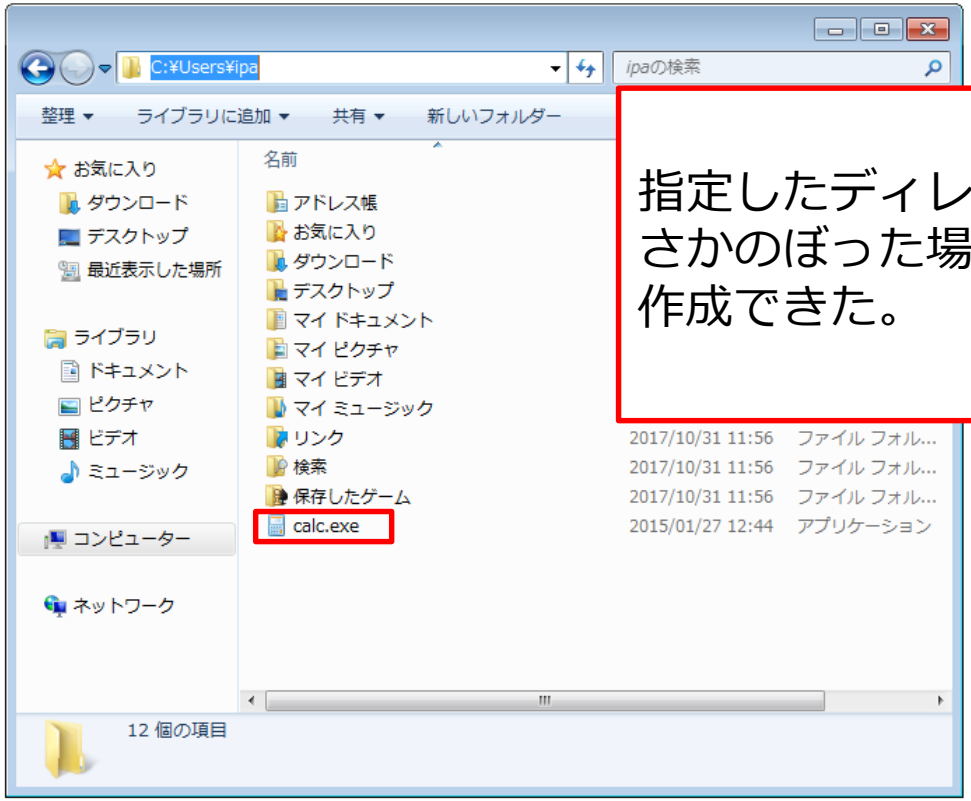
1.3 製品での検証②
検証

2. 細工されたATCファイルをアタッシェケースで復号
(今回の場合、細工されたファイルをウェブサイトからダウンロードさせたり、メールで被害者に送り付けたりするという想定)



1.3 製品での検証②
検証

3. C:¥Users¥IPA¥Desktopに復号されたファイルがないことを確認
4. C:¥Users¥IPAを確認し、calc.exeが作成されていることを確認



指定したディレクトリからさかのぼった場所にファイルを作成できた。

1.3 製品での検証②
検証結果

- ファイルを復号するだけで設定したディレクトリより上位のディレクトリにファイルが作成された
  →本来の意図した挙動とは違う挙動
  →悪用することで、任意のディレクトリにファイルを作成可
  →脆弱性と言える

- 実際に発生する被害は、脆弱性が存在する箇所の機能による

- 対策として「../../」といった文字列がファイル名等に含まれないか、検証する機能を実装する必要がある


全ての入力箇所や脆弱性の
種類に対して一つ一つ検証
をするの？

1.4 脆弱性検査ツール
概要

- セキュリティベンダの診断サービスでは検査ツール（自作、有償問わず）が用いられることが多い
- システム開発時にはリリース前に用いられることも
- 短期間での検査が可能となり、多くの脆弱性検証を行うことができる
- 検査ツールは主に以下の3タイプに分けられる

1.自動検査型
：自動的に検査コードの送信やレスポンスを解析

2.手動検査型
：リクエストをプロキシとして一旦保留、利用者が送信

3.通信監視型
：リクエストとレスポンスを監視、分析し脆弱性の有無を判定

1.4 脆弱性検査ツール
概要

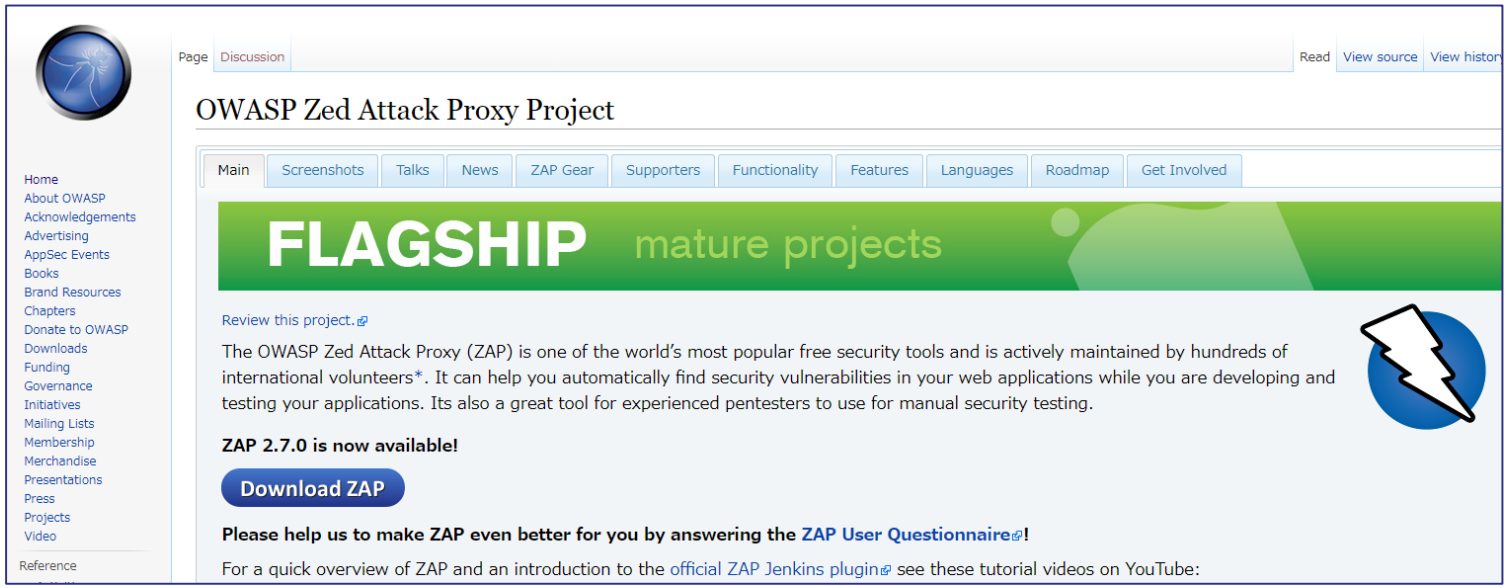
- 無償で公開されている代表的なツール

1. OWASP ZAP
→ボタン一つを押して調査対象のウェブアプリケーション内のリンク先も含め、自動で検査を行う
2. SQLMAP
→ウェブアプリケーション等に対し、SQLインジェクションの脆弱性検査を自動で行う。
3. Fiddler
→プロキシとして設定し、検査対象のウェブアプリケーションに検査コードを送付し、脆弱性を確認することができる

※本番運用中のサービスに対して
ツールを実行しないこと

1.4 脆弱性検査ツール 使い方

- 今回は「OWASP ZAP」を用いた検証を解説
- 以下からフリーでダウンロード可能
  https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project
- 今回は初期設定通りにインストール
  ※設定手順を解説するブログ記事も多数ある



1.4 脆弱性検査ツール
デモ

前提：
- 調査対象はCMS等のソフトウェア
  →今回は脆弱性を体験できるソフトウェアを使用
- ローカルの環境に検証環境を構築
- 調査のため、OWASP ZAPを使用する

実際の画面を見ていきます。

1.4 脆弱性検査ツール
実行結果

- 実行すると様々なパラメータに対して、脆弱性を再現するための値を入力していく。
- 脆弱性と考えられる応答があった場合は、レポートとして入力した値と応答結果が表示される。
- 検査対象となる脆弱性やパラメータはあらかじめ設定できる。
  →自動で様々な脆弱性の検証ができる。

ただし、本番運用中のサービスに対しては実行してはいけない


脆弱性を見つけたらどうすれ
ばいいの？

1.5 脆弱性の届出先

適切な脆弱性情報の取り扱いと、届出先については
プログラム3にてご説明します。

1.6 まとめ

- ソフトウェア製品の脆弱性は仮想環境等の安全な環境で実施し、インターネットへの誤った攻撃に注意
- 脆弱性の発見技術の向上には、過去の脆弱性から勘所を養うことが必要
- 開発した製品を検査する場合は脆弱性検査ツールを活用することで効率良く発見できる

- 稼働中のウェブサイトの脆弱性を調査すると意図せずシステムを停止してしまったり、不正アクセスとして訴えられるといったリスクがある

絶対に、自分が管理するシステムやウェブサイト以外に調査を実施しないこと。

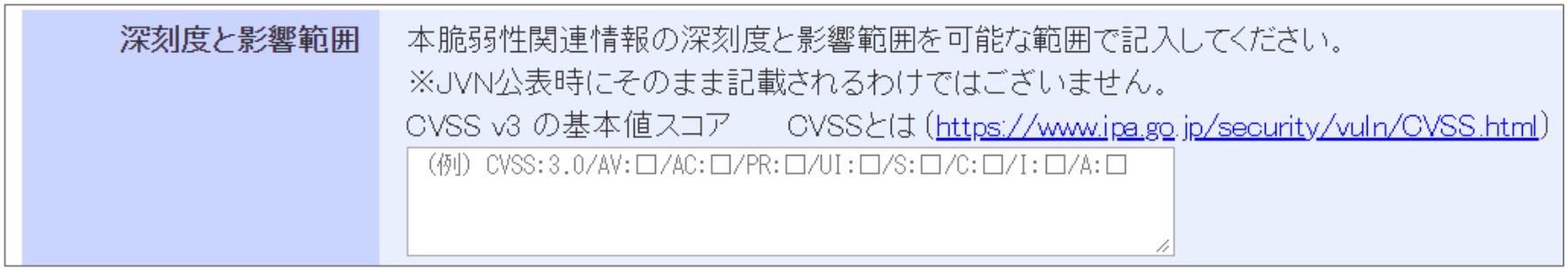
これで皆さんも快適な
脆弱性発見ライフを！

2. 脆弱性の評価方法
～ 脆弱性届出に CVSS v 3 を活用しよう ～

独立行政法人情報処理推進機構（IPA）
セキュリティセンター

2. 脆弱性の評価方法
本プログラムの概要

- IPAでは脆弱性の届出を受け付けています。
- 届出のフォーム上で、CVSSv3 の基本値の入力欄を設けています。



ソフトウェア製品脆弱性関連情報の届出
（https://isec-vul-form.ipa.go.jp/ipa-vul-main/software/vulnerability）

➡ IPA が上記の入力欄を設けた意図と、
CVSSv3 基本値の評価方法を解説します。

目次

2.1 CVSSとは

Common Vulnerability Scoring System
（共通脆弱性評価システム）

- 脆弱性に対する汎用的な評価手法
- IPAではCVSSv2 と CVSSv3 を用いてます
- 脆弱性の深刻度を 0.0 ～ 10.0 で評価
- CVSSv3では 3 つの基準で脆弱性の深刻度を評価します

I. 基本評価基準(Base Metrics)

II. 現状評価基準(Temporal Metrics)

III. 環境評価基準(Environmental Metrics)

2.1 CVSSとは
～CVSSの3つの評価基準～

基本評価基準 (Base Metrics)	【脆弱性そのものの評価】 • 悪用の条件は？ • 悪用された場合の影響は？
現状評価基準 (Temporal Metrics)	【脆弱性を取り巻く状況を評価】 • 既に悪用されている？ • 修正パッチは出ている？
環境評価基準 (Environmental Metrics)	【個別の環境における再評価】 • 自社環境での影響は？ • 影響を受けるシステムの重要度は？

JVNや JVN iPedia 上では、脆弱性の基本評価値を公開

CVSS v3 の値は以下の計算機で算出することができる

Common Vulnerability Scoring System Version 3.0 Calculator
https://www.first.org/cvss/calculator/3.0

2.1 CVSSとは
～CVSSの3つの評価基準～

IPA

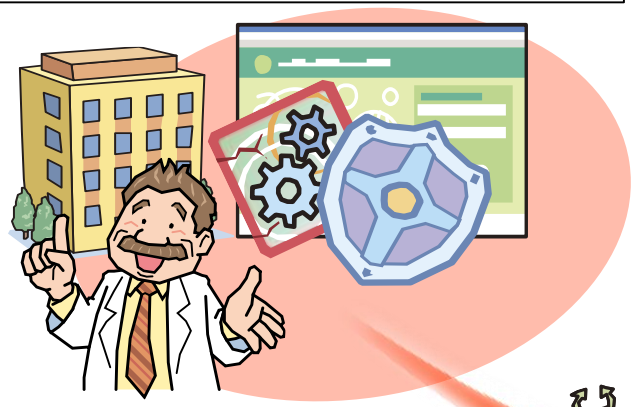
- 最終的な評価値まで判断した上で、脆弱性対応を検討するのが望ましい。

基本値だけを見ると深刻度が高くない脆弱性も、
場合によっては優先して対応が必要になることも…

• SQLインジェクション	基本評価値：7.3
• 悪用事例なし	現状評価値：6.7
• 機密情報のない内部向け環境	環境評価値：4.4
• サービス運用妨害（DoS）	基本評価値：5.3
• 悪用事例あり	現状評価値：5.3
• 重要な対外向け環境	環境評価値：6.1

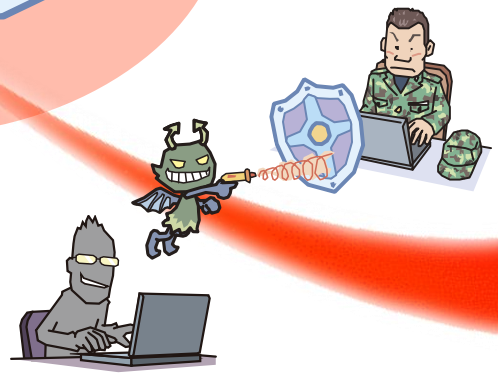
2.1 CVSSとは
～CVSSの活用イメージ～

ベンダ・セキュリティ機関 等

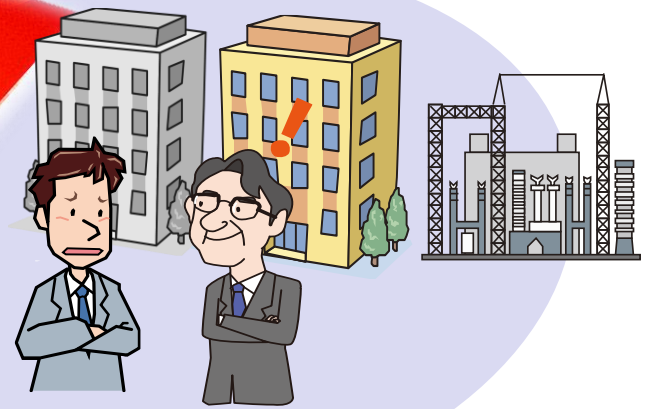


I. 基本評価値を評価
脆弱性そのものの深刻度を評価

II. 現状評価値を評価
脆弱性の現状の深刻度
（悪用状況や修正状況）を評価



組織・利用者



III. 環境評価値をユーザが評価
個別の環境下での脆弱性の深刻度を評価
→ **脆弱性対応を検討**

2.2 届出にCVSS値を利用する意図

- IPAが、脆弱性情報届出受付フォーム上で、CVSSv3 の入力欄を設けている意図
  - CVSSv3 基本値に沿った評価をいただくことで、悪用条件や影響などを正確に把握したい
  - 不明点などの確認事項を減らし、よりスムーズな脆弱性のハンドリングを行いたい

しかしながら…

評価が難解な項目もあるため
以降で評価方法を説明します

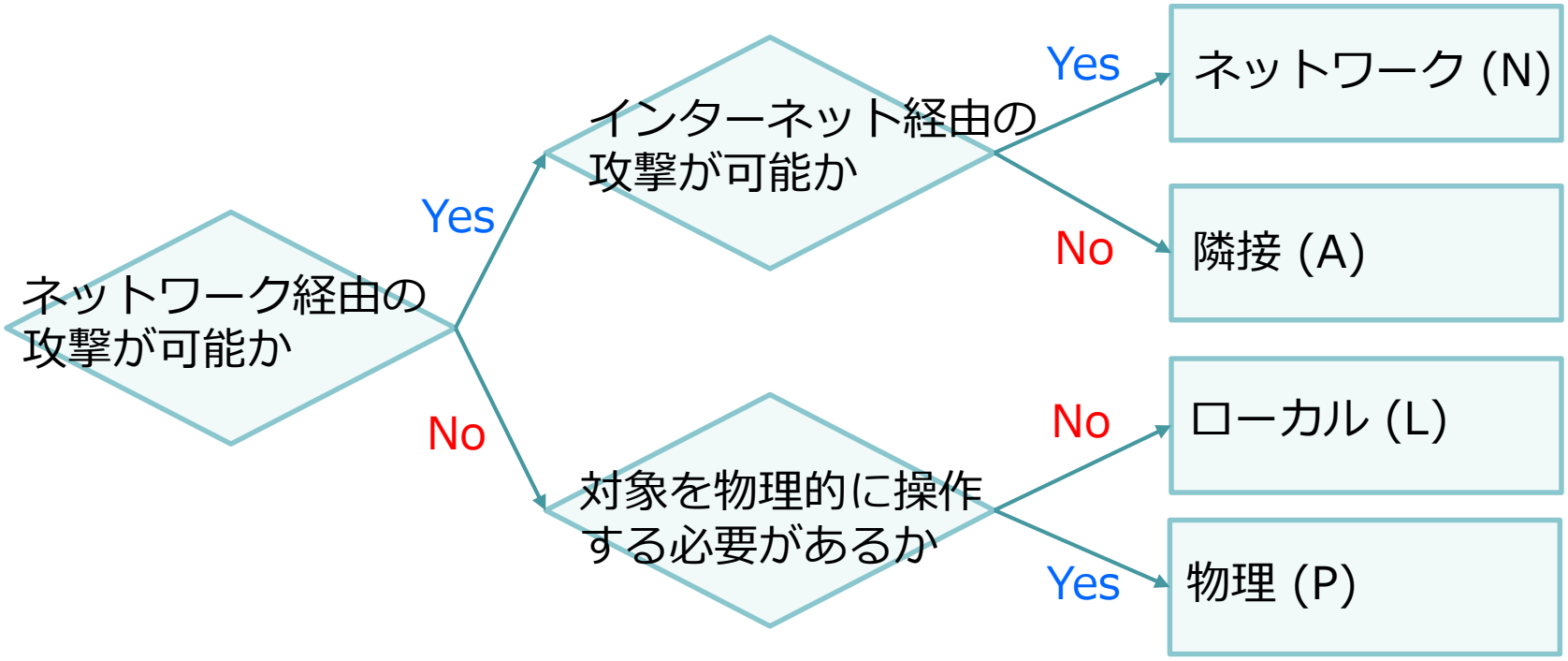
2.3 基本評価基準(Base Metrics)

危険小 ⟷ 危険大

	評価項目	選択肢・ポイント			
攻撃の難易度	どこから攻撃可能であるか 攻撃元区分　(AV：Attack Vector)	物理(P)	ローカル(L)	隣接(A)	ネットワーク(N)
	攻撃する際に必要な条件の複雑さ 攻撃条件の複雑さ(AC：Attack Complexity)	高（H)		低(L)	
	攻撃する際に必要な特権のレベル 必要な特権レベル(PR：Privileges Required)	高(H)	低(L)		不要(N)
	攻撃にユーザの関与が必要か否か ユーザ関与レベル(UI：User Interaction)	要(R)		不要(N)	
	別のコンポーネントへの影響 スコープ(S：Scope)	変更なし(U)		変更あり(C)	
攻撃による影響	機密情報の漏えい被害 機密性への影響　(C：Confidentiality Impact)	なし(N)	低(L)		高(H)
	情報の改ざん被害 完全性への影響　(I：Integrity Impact)	なし(N)	低(L)		高(H)
	業務が遅延・停止する被害 可用性への影響 (A：Availability Impact)	なし(N)	低(L)		高(H)

攻撃元区分（Attack Vector）

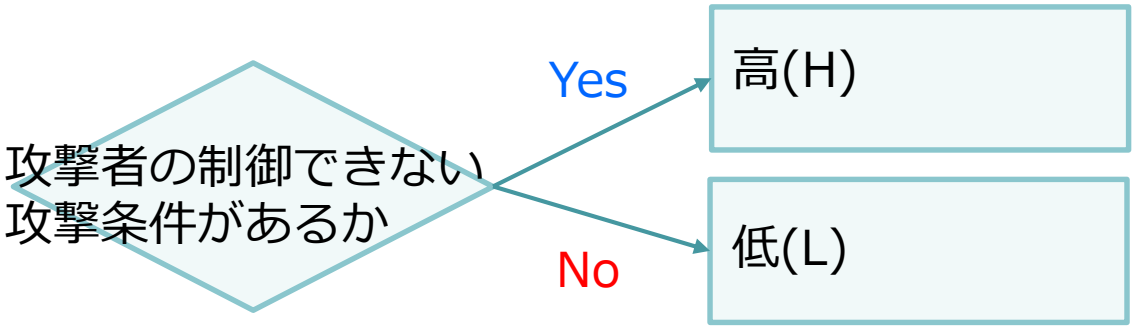
- どこから攻撃可能かを評価



※対象コンポーネントへの攻撃元がどこか
※ファイルを開かせること等が必要な場合は L となる

攻撃条件の複雑さ (Attack Complexity)

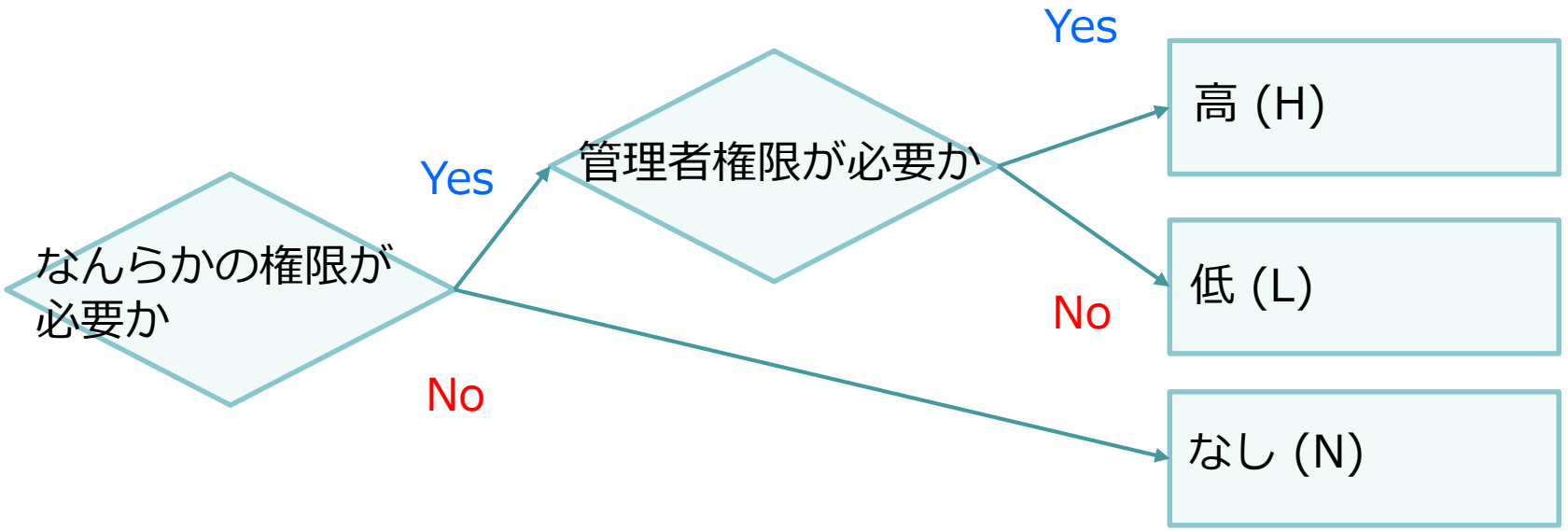
- 攻撃者が制御できない要因を評価



※総当たりや、事前の情報収集、中間者攻撃などは
　高（H）となる
※攻撃者のスキルについては考慮しない
※ユーザー操作の要因は除く（後述するUIで評価）

必要な特権レベル(Privileges Required)

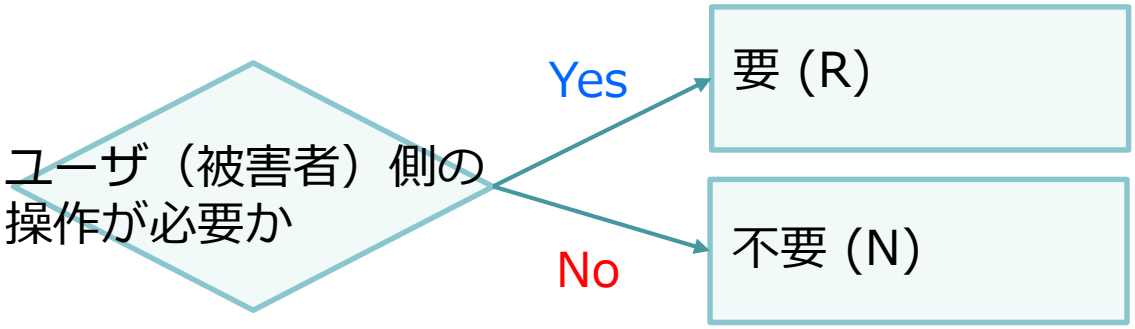
- 攻撃に必要となる特権レベルを評価



※認証の有無を、利用者が任意に設定できる場合などは不要（N）として評価する場合がある

ユーザ関与レベル(User Interaction)

- 攻撃における、ユーザの関与レベルを評価

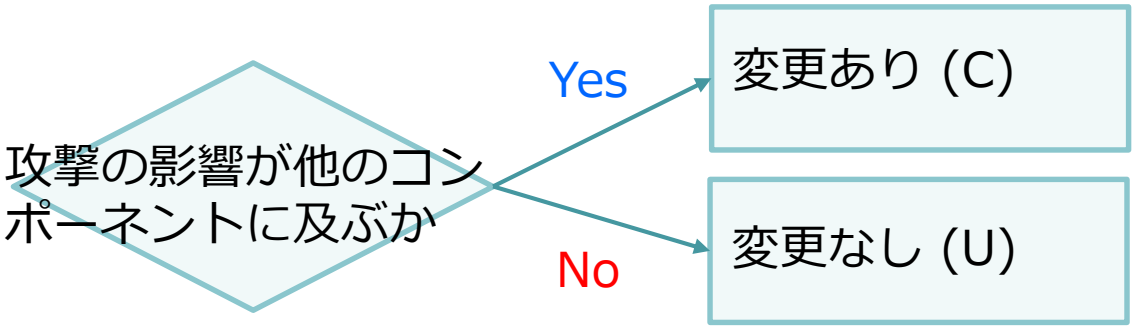


※ユーザ側の操作の例：
　リンクのクリック、ファイルの実行など

※あくまで操作を評価し、状態は評価しない
　ログインの操作をする　→　要（R）
　ログインしている前提　→　不要（N）

スコープ(Scope)

- 被害の影響範囲を評価



※XSSのように、対象コンポーネント側の脆弱性の影響が、ユーザのブラウザ上にまで及ぶもの

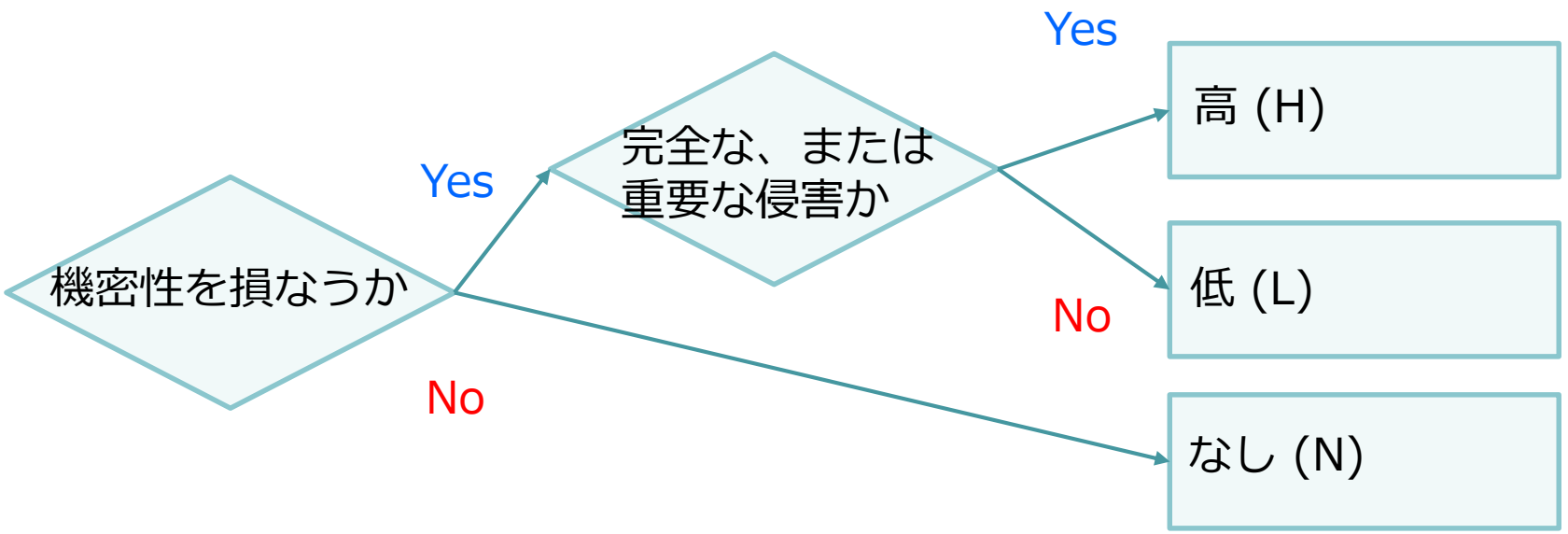
基本評価基準(Base Metrics)

危険小 ← → 危険大

	評価項目	選択肢・ポイント			
攻撃の難易度	どこから攻撃可能であるか 攻撃元区分　(AV：Attack Vector)	物理(P)	ローカル(L)	隣接(A)	ネットワーク(N)
	攻撃する際に必要な条件の複雑さ 攻撃条件の複雑さ(AC：Attack Complexity)	高（H)		低(L)	
	攻撃する際に必要な特権のレベル 必要な特権レベル(PR：Privileges Required)	高(H)	低(L)		不要(N)
	攻撃にユーザの関与が必要か否か ユーザ関与レベル(UI：User Interaction)	要(R)		不要(N)	
	別のコンポーネントへの影響 スコープ(S：Scope)	変更なし(U)		変更あり(C)	
攻撃による影響	機密情報の漏えい被害 機密性への影響　(C：Confidentiality Impact)	なし(N)	低(L)		高(H)
	情報の改ざん被害 完全性への影響　(I：Integrity Impact)	なし(N)	低(L)		高(H)
	業務が遅延・停止する被害 可用性への影響 (A：Availability Impact)	なし(N)	低(L)		高(H)

機密性への影響(Confidentiality Impact)

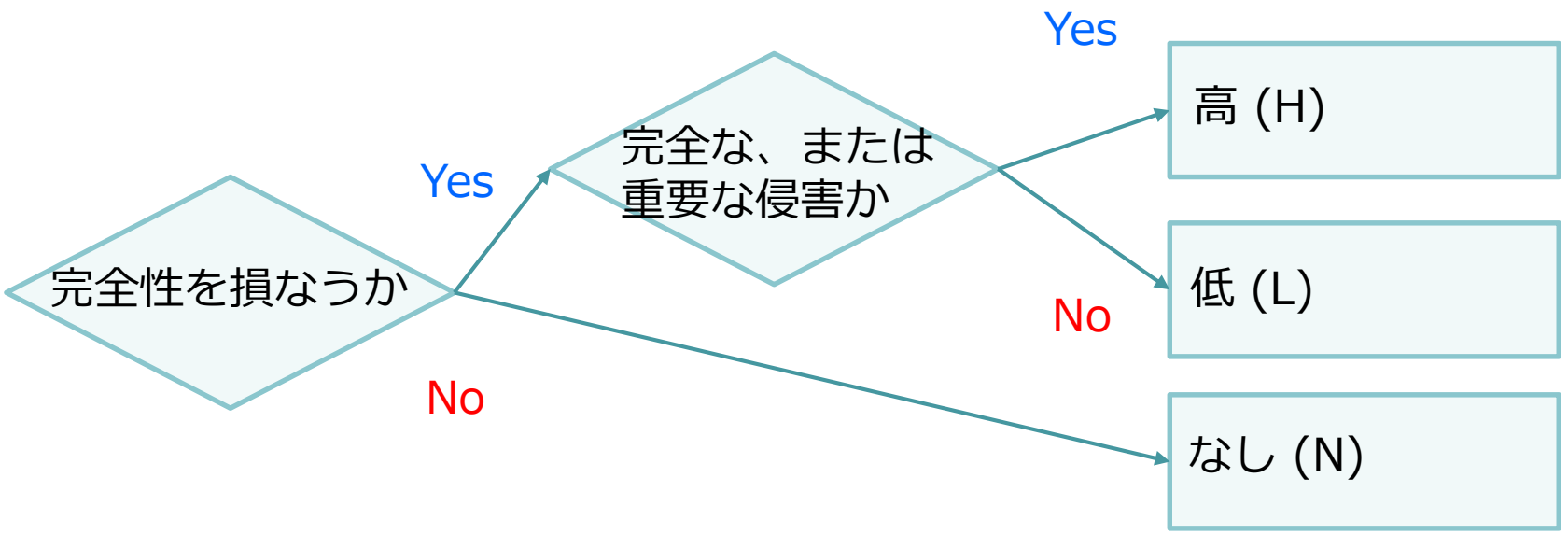
- 機密性に与える影響を評価



※全ての情報、または重大な情報が漏洩する場合、
高（H）と評価する

完全性への影響 (Integrity Impact)

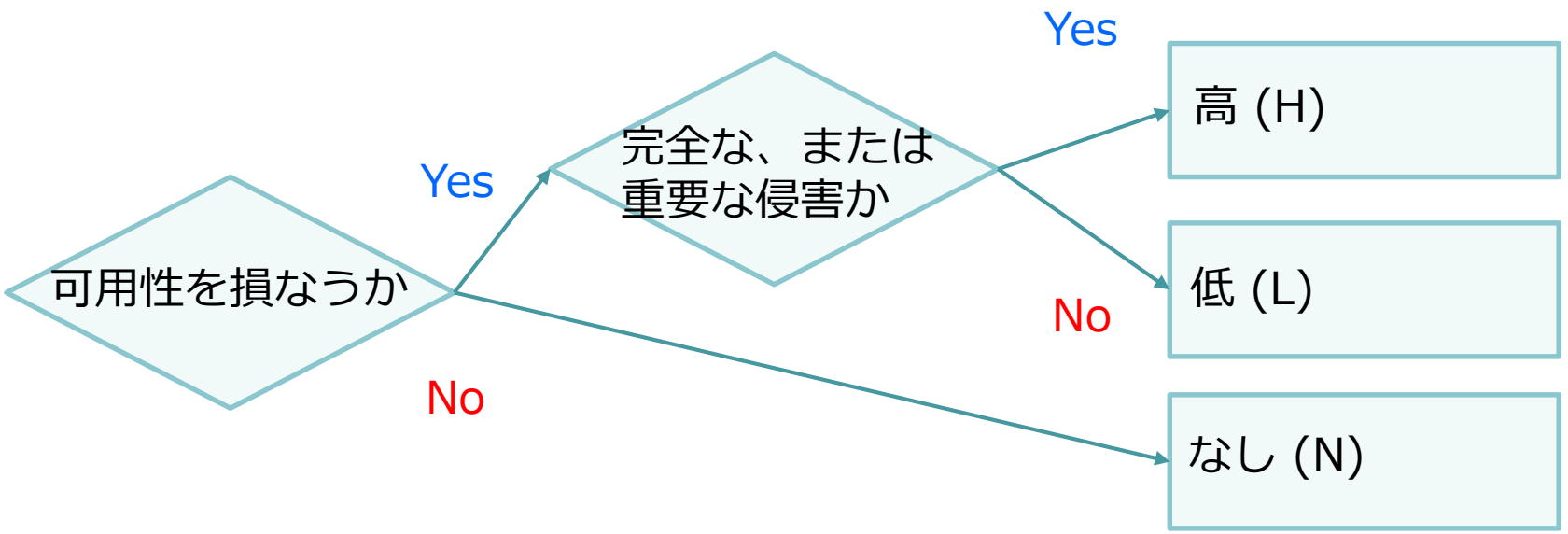
- 完全性に与える影響を評価



※全ての情報、または重大な情報が改ざん可能な場合、高（H）と評価する

可用性への影響 (Availability Impact)

- 可用性に与える影響を評価



※対象コンポーネントが完全に停止する、または継続的にパフォーマンスが低下する場合、高（H）と評価する

基本値評価のポイント

- 攻撃シナリオベースで評価する
  - 複数の攻撃シナリオが想定される場合は、よりスコアが高くなる方の攻撃シナリオを評価します。
  - 想定した攻撃シナリオも記入いただけると幸いです。
- 一次被害とその悪用条件のみを評価する
  - 例えば、管理者の機微情報が漏洩する脆弱性では、機密性のみの影響となります。
- 不明点等により、正確な判断が不能な場合、評価値がより高くなるように評価しておく
  - 評価をした項目を明記いただけると幸いです。
  - ベンダとの調整等を経て正しく再評価し公表します。

2.4 最後に

- CVSSについてもっと知りたい方は

共通脆弱性評価システムCVSS v3概説
https://www.ipa.go.jp/security/vuln/CVSSv3.html

Common Vulnerability Scoring System v3.0: Specification Document
https://www.first.org/cvss/specification-document

3. 脆弱性情報の取扱いについて

独立行政法人情報処理推進機構（IPA）
セキュリティセンター

3. 脆弱性情報の取扱いについて
目次

- 3.1 本プログラムの目的
- 3.2 脆弱性情報とは
- 3.3 脆弱性情報の取扱いの流れについて
  - 協調的に脆弱性情報を取り扱う
- 3.4 発見者の担う役割と注意点
  - 「発見」
    - 発見にかかわる注意点
  - 「報告」
    - 報告の内容について
    - 報告先について
      - 直接届出
      - 「情報セキュリティ早期警戒パートナーシップ」
- 3.5 まとめ

3.1 本プログラムの目的

脆弱性情報の性質について

- 脆弱性情報の2つの側面

脆弱性情報の取扱いの流れについて

- 協調的な脆弱性情報の取扱いと発見者の役割

脆弱性の発見と報告の際の注意事項について

- 「発見」の際の注意事項
- 「報告」の際の注意事項

適切な脆弱性の発見・報告を実施できるようにしましょう

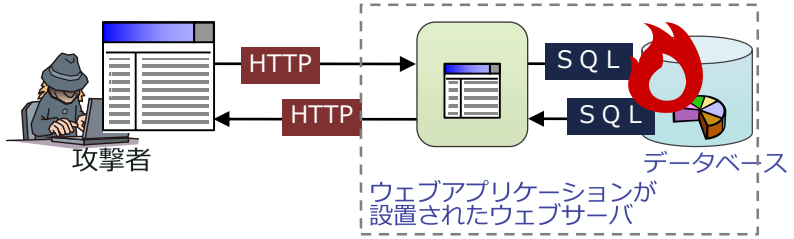
3.2 脆弱性情報とは

- 脆弱性の例

① 攻撃者が、 細工したリクエストを送信する
例：専用のソフトで通信データを直接細工する

② 動作が停止してしまう

① 攻撃者が、 細工したリクエストを送信する
例：通常の入力　･･･パスワード
細工した入力･･･「1 = 1」等



② 使用しているデータベースを不正に操作されてしまう

- アプリケーションが利用できなくなる
- データベースに保存している個人情報が漏洩する

セキュリティ上の弱点として「脆弱性」と呼ばれる

3.2 脆弱性情報とは
脆弱性情報の内容と性質

- 脆弱性情報とは
  - 「脆弱性情報」とは何か
    - どのソフトウェア/ウェブサイトにどのような脆弱性が存在するのかを示す情報

      例：
      - » 脆弱性のある箇所の情報
      - » 脆弱性を確認するための手順
      - » PoC
  - 脆弱性情報は誰に、どのように利用されるか
    - 脆弱性情報は相対する2つの性質をもつ
      - 脆弱性の**対策のために必要**な情報
      - 脆弱性を**攻撃するために悪用**される情報

脆弱性情報は適切に取り扱うことが重要

3.3 脆弱性情報の取扱いの流れについて
脆弱性情報の取扱方法

IPA

- 推奨できない脆弱性情報の取扱い

脆弱性情報を報告しない

- 製品開発者/ウェブサイト運営者にも他の機関にも脆弱性情報を報告しない
  - 製品開発者/ウェブサイト運営者によって**脆弱性が対策されない**
  - **時間が経てば攻撃者も同じ脆弱性を発見できるため悪用される可能性がある**

脆弱性情報を公開する

- 脆弱性情報を公開する
  - 製品開発者/ウェブサイト運営者は公表された情報を必ずしも把握できない
  - **脆弱性に対策されるまでの間、攻撃者は脆弱性を悪用できる**

利用者を危険にさらしてしまうことになります

3.3 脆弱性情報の取扱いの流れについて
脆弱性情報の取扱方法

IPA

♦ 推奨される脆弱性情報の取扱い

協調的に脆弱性情報を取り扱う

- 発見者が製品開発者/ウェブサイト運営者と**協調して**
  脆弱性開示・対策を実施する
- 脆弱性情報の性質を考慮した取扱いを行う
  - 脆弱性対策をするための時間的猶予が与えられる
  - 脆弱性情報は当事者以外に流出しないよう管理する

発見者と製品開発者/ウェブサイト運営者の間での
当事者同士の協力が必要です

3.3 脆弱性情報の取扱いの流れについて
協調的な取扱いにおける流れ

- ソフトウェア製品の場合の協調的な脆弱性情報の取扱いの流れ

- 脆弱性の**発見**から**対策適用**まで
  - 脆弱性が解消されセキュリティ上の脅威がなくなる
- 各段階で各当事者が**役割を担う**

3.3 脆弱性情報の取扱いの流れについて
協調的な取扱いにおける流れ：発見

IPA

- 発見
- 報告
- 検証/対策
- 対策公開
- 対策適用

当事者

- 発見者

役割

- 脆弱性情報を入手すること
  - 脆弱性の調査して自ら発見する
  - 他人から脆弱性情報を受け取る

行動

- 脆弱性を特定する
- 再現手法・条件を確認する
- 証跡・ログファイルを作成する

3.3 脆弱性情報の取扱いの流れについて
協調的な取扱いにおける流れ：報告

発見

報告

検証/対策

対策公開

対策適用

当事者

- 発見者（報告者）
- 報告の受領者
  - 製品開発者　専門機関（IPA）

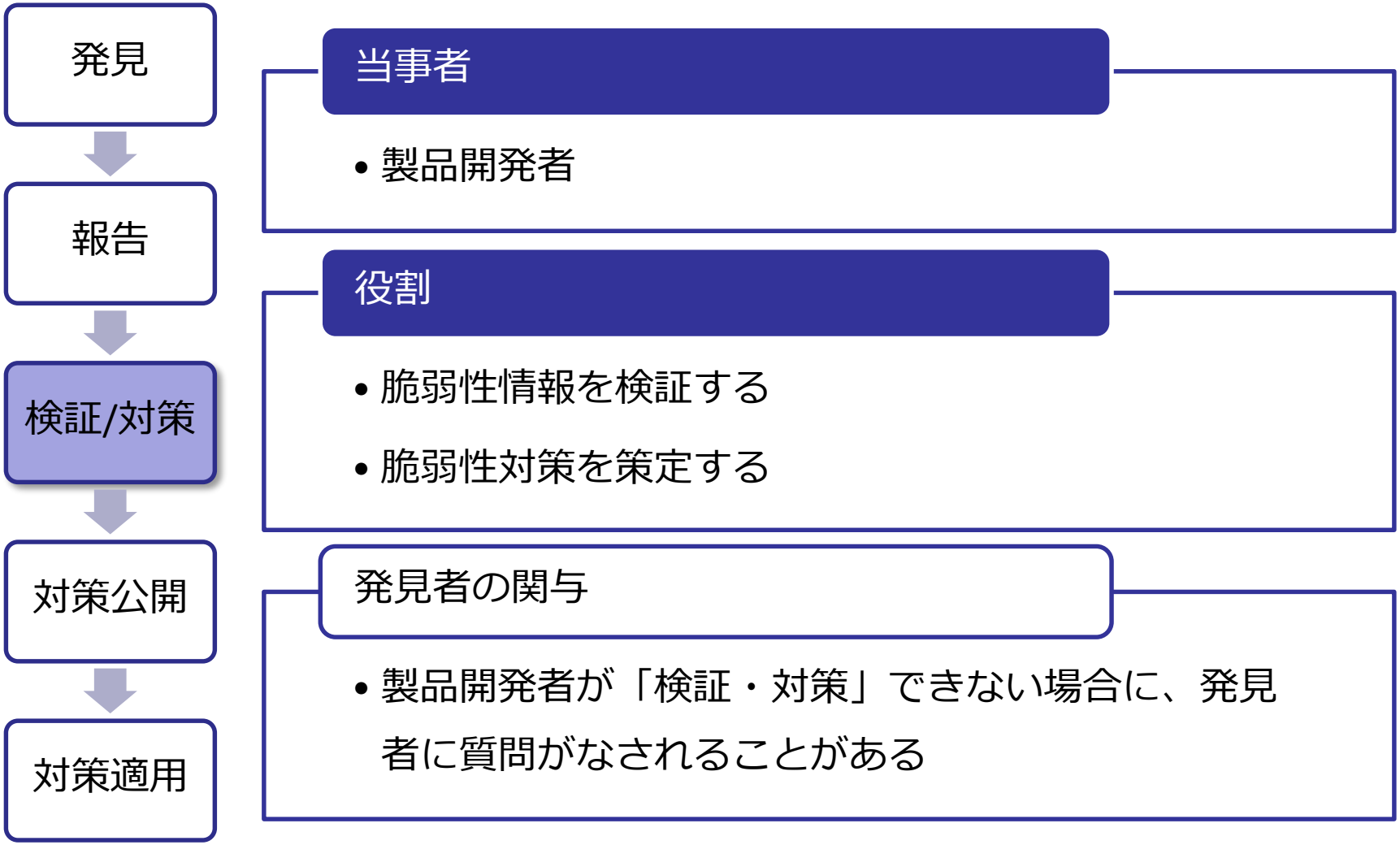
役割

- 発見者：脆弱性情報を当事者に報告すること
- 受領者：脆弱性情報を発見者から受領すること

行動

- 発見者：「検証・対策」のために必要な情報をまとめる
- 発見者：連絡先と連絡手法を調査・確認し、報告する

3.3 脆弱性情報の取扱いの流れについて
協調的な取扱いにおける流れ：検証・対策



3.3 脆弱性情報の取扱いの流れについて
協調的な取扱いにおける流れ：対策公開

当事者

- 製品開発者
- 製品の利用者

役割と行動

- 脆弱性の対策方法を利用者に伝達する

3.3 脆弱性情報の取扱いの流れについて
協調的な取扱いにおける流れ：対策適用

発見

報告

検証/対策

対策公開

対策適用

当事者

- 製品の利用者

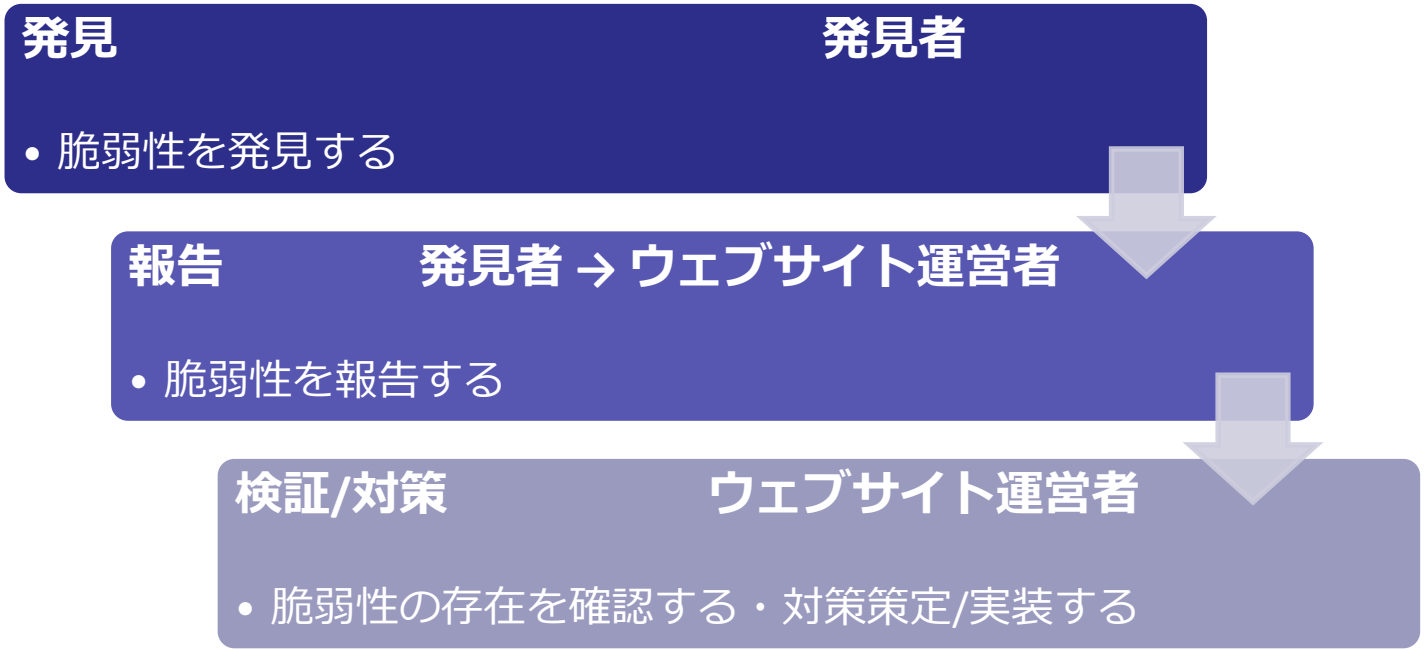
役割と行動

- 脆弱性の対策を実装する

脆弱性が対策され、脅威がなくなる

3.3 脆弱性情報の取扱いの流れについて ウェブサイトの脆弱性の場合の流れ

- ウェブサイトの脆弱性の場合
  - ウェブサイト運営者が対策を実施すれば脆弱性が解消するため、「対策公開」と利用者の「対策適用」は必須ではありません。



3.4 発見者の担う役割と注意点

- 協調的な取扱いにおける発見者の役割
  - 当事者と協調しながら脆弱性情報を取り扱うこと
    - 取扱いの流れの全体において、当事者と十分なコミュニケーションをとって相互に理解を図ることが重要です。
    - とくに**「発見」**と**「報告」**の段階で重要な役割を担います。

発見

- 他人に被害を生じさせないこと

報告

- 必要な情報をまとめること
- 適切な相手に報告すること

3.4 発見者の担う役割と注意点
発見時の注意点

- 他人に被害を生じさせないこと

ウェブサイト運営者の許可なく脆弱性検査を行うと、ウェブサイト運営者に危険を伝える目的であっても、トラブルになる可能性があります。

- 脆弱性の調査は、**ソフトウェアの機能不全**を引き起こしたり、**ウェブサイトの停止**をもたらしたりする可能性があります。
- 善意で調査したものであっても**当事者間でトラブルとなる可能性**があります。

3.4 発見者の担う役割と注意点
発見時の注意点

- 他人に被害を生じさせないこと
  - ウェブサイトの脆弱性
    - 稼働中のウェブサイトへの許可のない脆弱性調査は推奨できません。
    - ウェブサイトの脆弱性を、**偶然に**発見したとしても、必要以上に追加の調査行為はしないでください。
  - ソフトウェア製品の脆弱性
    - 外部との通信が発生しないような安全な環境のなかで検証してください。

3.4 発見者の担う役割と注意点
発見時の注意点

- バグバウンティについて
  - 脆弱性を発見し報告した方に報奨金を支払う取り組み
    - 運営者が指定する条件の範囲内で「稼働中の情報システムに対して調査する」ことができる場合があります。
    - なお、以下の条件に注意してください。
      - 対象となる脆弱性の種別
      - 対象となる情報システム
      - 発見手法の制限

3.4 発見者の担う役割と注意点
報告時の注意点

- 報告時の注意点

必要な情報をまとめること

- 「検証・対策」をするために必要な情報を記載する
- 「検証・対策」の実施者に情報が渡るようにすること

適切な相手に報告すること

- 製品開発者・ウェブサイト運営者に直接報告する
- IPA（情報セキュリティ早期警戒パートナーシップ）に報告する

3.4 発見者の担う役割と注意点
報告時の注意点

- 必要な情報をまとめること
  - 「検証・対策」をするために必要な情報を記載する
    - 脆弱性を特定するための情報
      - ソフトウェア製品の名称・バージョン情報
      - 脆弱性の種別
      - CVSSのスコア
    - 再現手法・条件
      - 環境情報
      - 設定の条件
    - 証跡・ログファイル
      - スクリーンショット
      - 脆弱性発見時のログ

3.4 発見者の担う役割と注意点
報告時の注意点

- 必要な情報をまとめること
  - 「検証・対策」の実施者に情報が渡るようにすること
    - 報告を受け取った方が、必ずしもセキュリティに詳しい技術者であるとは限りません。
      - サポート窓口や広報担当に連絡する場合
    - 「分かりやすい説明」をすることで、円滑に調整が進むことが期待できます。

3.4 発見者の担う役割と注意点
報告時の注意点

- 必要な情報をまとめること
  - 「検証・対策」の実施者に情報が渡るようにすること
    - 良く知られた用語に言い換える
      - 「脆弱性」→「セキュリティ上の問題」
    - 「脆弱性の種別」ではなく脅威の例を伝える
      - 「XSS」
        → 「フィッシング詐欺に悪用されたり、情報漏えいを生じさせるような弱点」

3.4 発見者の担う役割と注意点
報告時の注意点

- 適切な相手に報告すること

直接に製品開発者/ウェブサイト運営者に連絡する方法と第三者機関（IPA）に報告する方法があります。
それぞれに特色があるため、必要に応じて報告先を選択してください。

製品開発者/ウェブサイト運営者

- サポート窓口
- CSIRT/PSIRT
- バグバウンティプログラム

公的機関・専門機関

- IPA（情報セキュリティ早期警戒パートナーシップ）

3.4 発見者の担う役割と注意点
報告時の注意点

- 製品開発者/ウェブサイト運営者に直接脆弱性情報を報告する

  - 製品開発者ウェブサイト運営者に直接脆弱性を報告する方法です。
  - バグバウンティなど、報告することにより報償が得られる場合があります。

  - 報告先
    - 一般のサポート窓口
    - CSIRTやPSIRTなどのセキュリティ問題を受付する窓口
    - バグバウンティ(脆弱性報奨金制度)
      - 脆弱性を発見し、報告した方に報奨金を支払う

3.4 発見者の担う役割と注意点
報告時の注意点

- 下記のような問題・トラブルに直面する場合があります
  - 製品開発者/ウェブサイト運営者の連絡先がない
  - 連絡しても信用されない
  - 不正アクセスを疑われる

- 製品開発者/ウェブサイト運営者も、予算や人員の問題など、脆弱性の対処のために様々な課題を抱えていることを考慮することも重要です。

- 製品開発者/ウェブサイト運営者との調整が難しい場合にはIPA（パートナーシップ）に届出を行ってください。

3.4 発見者の担う役割と注意点
報告時の注意点

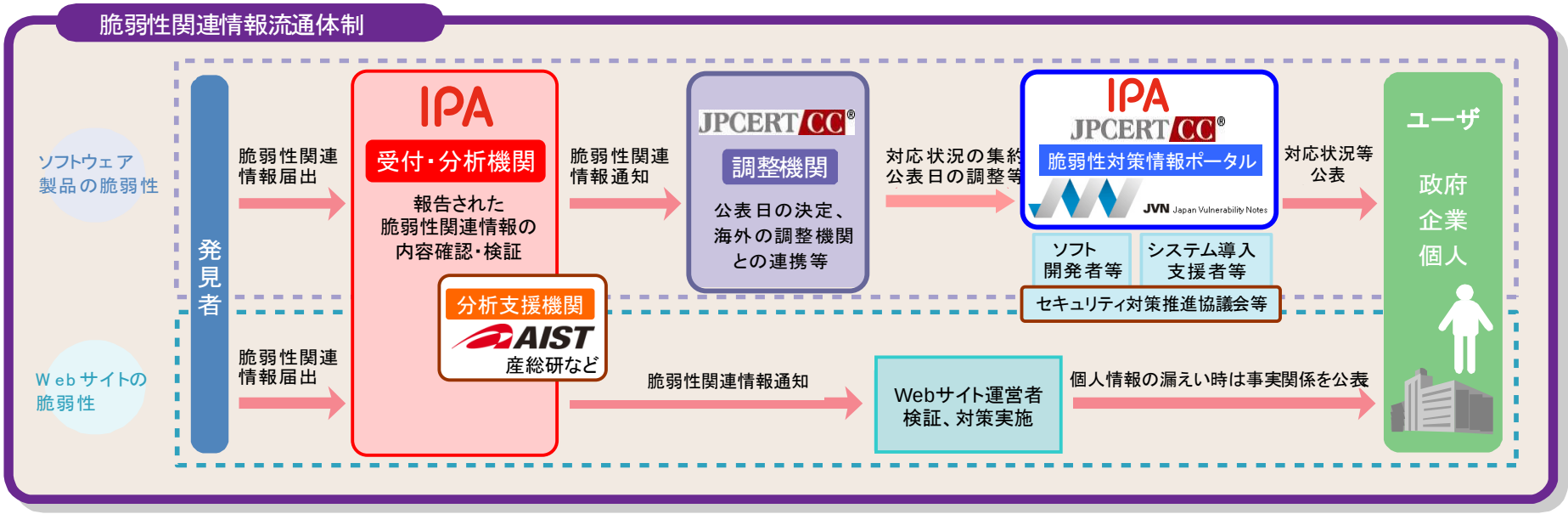
- IPAに報告する

脆弱性の報告を IPA が受け付け、製品開発者/ウェブサイト運営者に連絡し、対策を依頼します。
製品開発者/ウェブサイト運営者と直接やり取りすることなく、脆弱性情報を報告することができます。

  - 脆弱性情報を取り扱う**「情報セキュリティ早期警戒パートナーシップ」**を運営
    - 発見者からの届出を受け付け、製品開発者/ウェブサイト運営者に通知し、対策を促す制度
    - 各当事者の**報告/検証/公開** の役割をIPA・JPCERT/CCが部分的に担う

3.4 発見者の担う役割と注意点

報告時の注意点

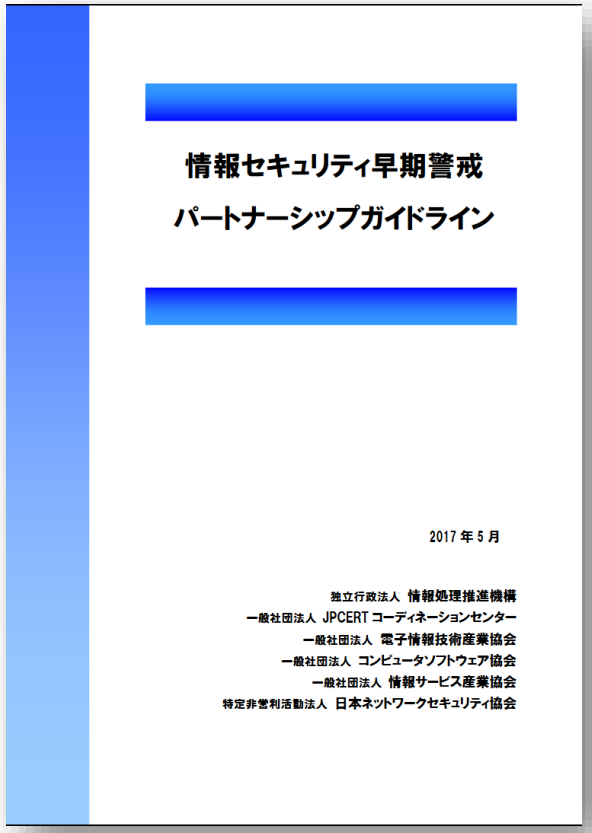


- ソフトウエア製品の脆弱性
  - 調整機関であるJPCERT/CCが製品開発者との連絡・調整を実施
  - JVNでの対策情報の公開
- ウェブサイトの場合
  - IPAがWebサイト運営者に連絡を実施
  - 脆弱性情報は公開されない
    - 個人情報漏えいの場合は公表を推奨

3.4 発見者の担う役割と注意点
報告時の注意点

- 制度に届出をする際の注意点

- 告示・ガイドラインに基づく運用をしているため、告示・ガイドラインの対象範囲に該当しない問題については取扱いできません。
- **「情報セキュリティ早期警戒パートナーシップガイドライン」**と**「脆弱性関連情報として取り扱えない場合の考え方の解説」**を届出する前に確認してください。



3.5 まとめ

脆弱性情報の性質について

- 対策のために必要な情報でもあり、攻撃に悪用される情報でもある
- 脆弱性情報にはその性質を考慮した取扱いが必要となる

脆弱性情報の取扱いの流れについて

- 発見者は「発見」と「報告」に密接にかかわる

脆弱性の発見と報告の際の注意事項について

- 他人に被害を与えないようにする
- 必要な情報を整理し、適切な相手に報告する

社会全体のセキュリティ向上のために
適切な脆弱性の発見・報告にご協力ください