



INFORMATION-TECHNOLOGY PROMOTION AGENCY, JAPAN

脆弱性やセキュリティ設定をチェックする 「OVAL (Open Vulnerability and Assessment Language) 」 ～MyJVNバージョンチェッカの裏側～

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
情報セキュリティ技術ラボラトリー

2010年8月6日 公開

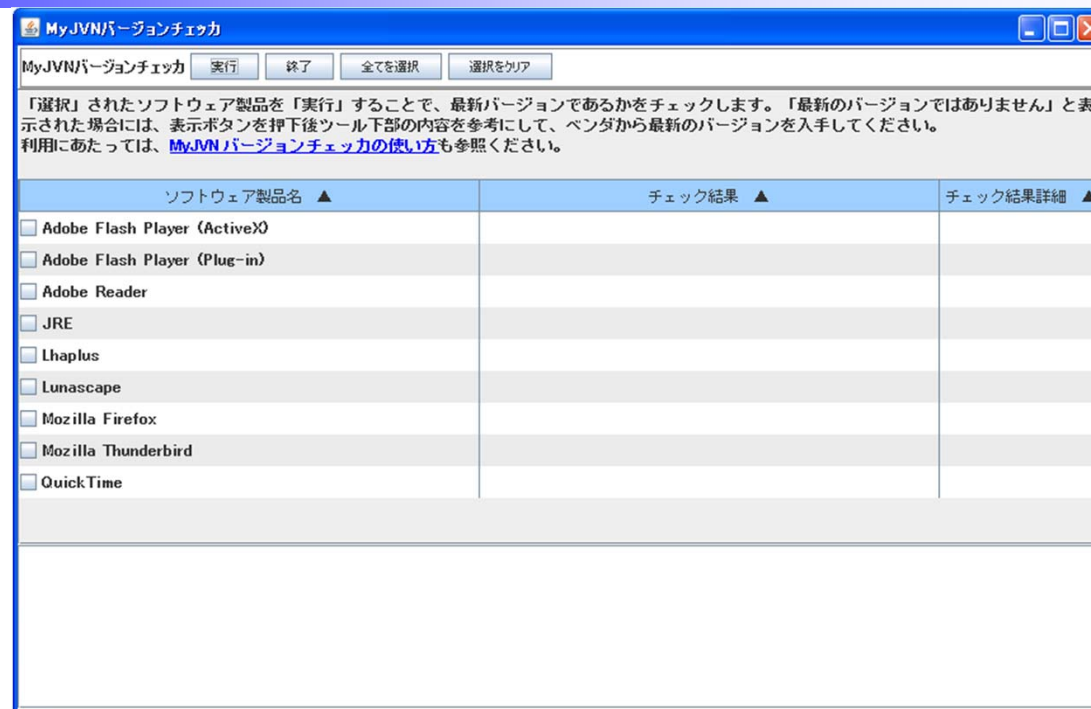
アジェンダ

1. MyJVNバージョンチェッカとは
2. OVALとは
3. MyJVNバージョンチェッカの裏側
4. OVALを使ったバージョンチェッカの作り方
5. まとめ



MyJVNバージョンチェツカとは

MyJVNバージョンチェッカとは



- 利用者のPCにインストールされているソフトウェア製品のバージョンが最新であるかを、簡単な操作で確認するツール
- MyJVNバージョンチェッカではOVALを使用

<http://jvndb.jvn.jp/apis/myjvn/#VCCHECK>

MyJVNバージョンチェッカへの声

- 「**ガンブラー対策**」に使える！という声
 - 「**ガンブラー**」感染防ぐ**無料**チェックソフトとは
 - <http://www.yomiuri.co.jp/net/qanda/20100115-OYT8T00821.htm>
 - **ガンブラー対策大丈夫？ まずは無料のMyJVNバージョンチェッカで3分チェック**
 - <http://web-tan.forum.impressrd.jp/e/2010/02/02/7293>
 - **IPA、入れてあるソフトが最新か否かを確認「MyJVNバージョンチェッカ」公開**
 - <http://journal.mycom.co.jp/news/2009/11/30/008/index.html>
- **他にも多数ご紹介をいただき、またIPAへご要望をいただいております**

MyJVNバージョンチェッカはガンブラー対策**専用**のツール？

MyJVNバージョンチェツカは 脆弱性対策の自動化推進の一環

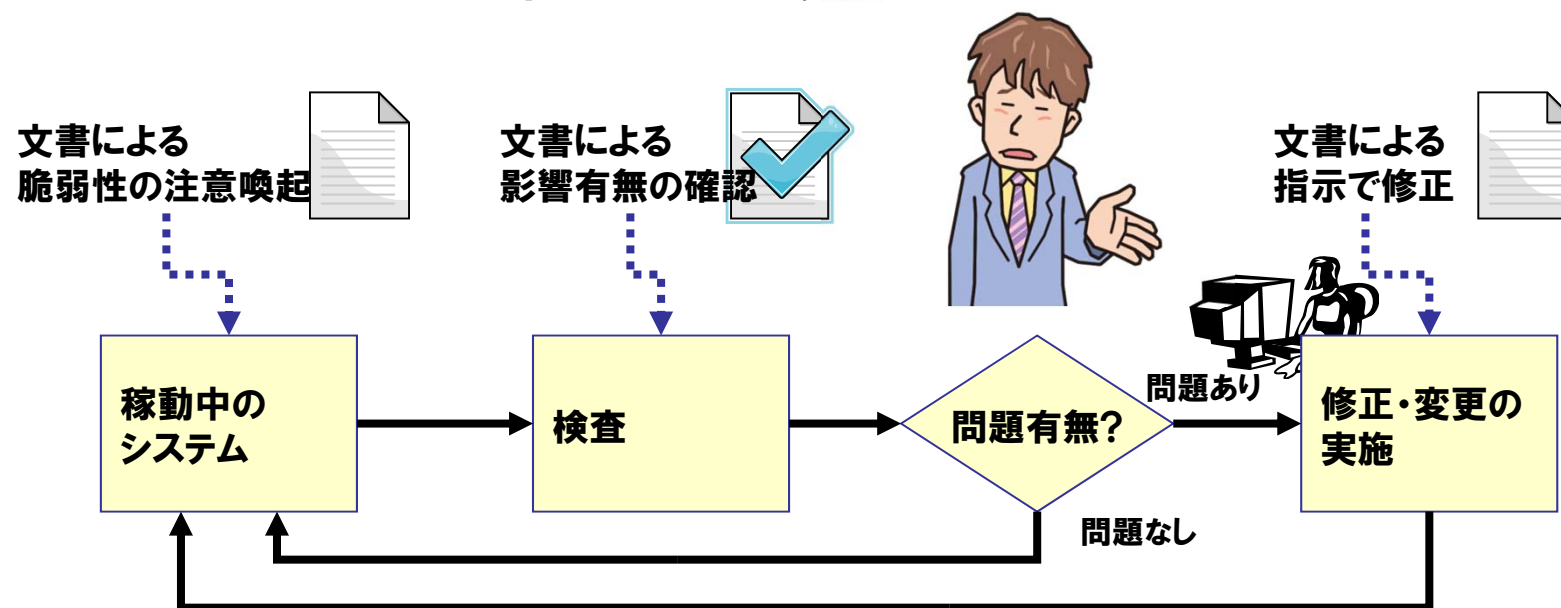


- MyJVNバージョンチェツカは
 - 脆弱性対策の自動化（機械処理）推進の一環
 - SCAPの根幹であるOVAL (Open Vulnerability and Assessment Language:セキュリティ検査言語) を使ってできることを示したもの
 - MyJVNバージョンチェツカの開発からリリース
 - 2008年 : MyJVNバージョンチェツカを企画
 - 2009年3月頃～ : ガンブラーの騒ぎが起き始める
 - 2009年11月30日 : MyJVNバージョンチェツカのリリース

MyJVNバージョンチェツカは、**ガンブラー対策に有用**ですが、**ガンブラー対策を視野に入れて作ったツールではない**のです。

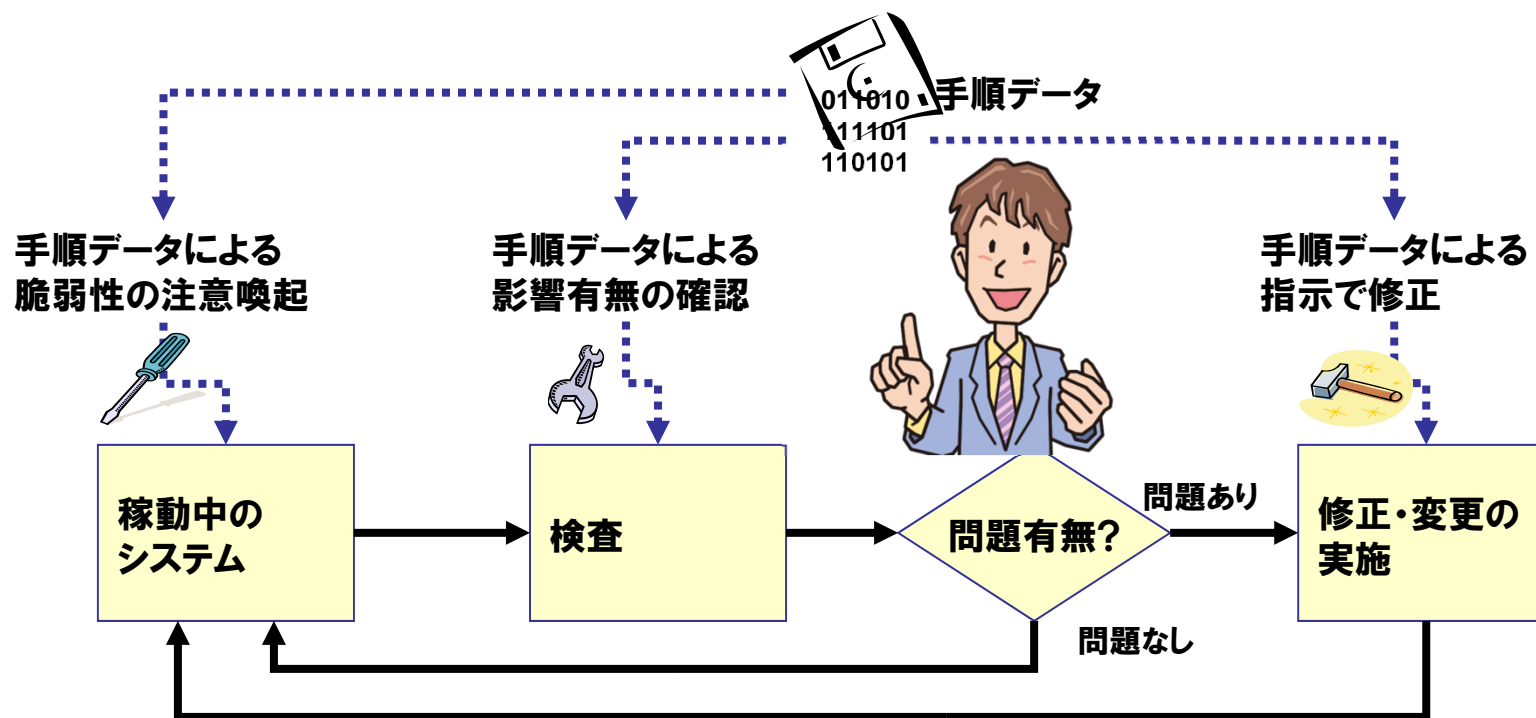
なぜMyJVNバージョンチェツカか？

- 脆弱性の対策を文書で組織内に適用させるのは大変
 - 毎日脆弱性対策情報を追い、影響が深刻な脆弱性があれば稼働中のシステムに対して対策を文書で通知する
 - 文書による脆弱性対策情報だけで影響有無を判定する手法では抜け漏れが発生してしまう可能性がある
 - ※これらの作業は基本的に手作業で実施



なぜMyJVNバージョンチェツカか？

- 手作業より**機械処理**の方が**確実**だし**早い**！何より**楽**！
- **漏れもなく、確実にできる**！



なぜMyJVNバージョンチェツカか？

- **機械処理を進めるためにOVALを利用してMyJVNバージョンチェツカを作ろう！**（2008年、IPAにて）
- **OVALを使う理由**
 - **オープンな仕様**
 - IPAだけが作れる仕様では意味がない
 - 多くの人が利用できる仕様である
 - **多くのOS・ソフトウェアを考慮した仕様**
 - OSやソフトウェアはたくさんの種類がある
 - OS: Windows, Linux, Mac・・・
 - ソフトウェア: Adobe Reader, Adobe Flash, Java, Apache, 一太郎・・・
 - これらの環境に対して、統一した仕様で機械処理ができる



そのOVALとは一体何？

OVALとは

OVALとは



IPA®

- OVAL: Open Vulnerability and Assessment Language
:セキュリティ検査言語
- コンピュータの**セキュリティ設定状況を検査するため**
の仕様
 - OS・ソフトウェアの脆弱性対策の確認に
 - OS・ソフトウェアのセキュリティ設定の確認に
- OVALの利用方法
 1. **OVAL定義データ** (OVALのフォーマットに則ったXML*1
ベースの定義ファイル) を作成する
 2. **OVALインタプリタ** (OVAL定義データを解釈するプログラ
ム) で処理をさせる

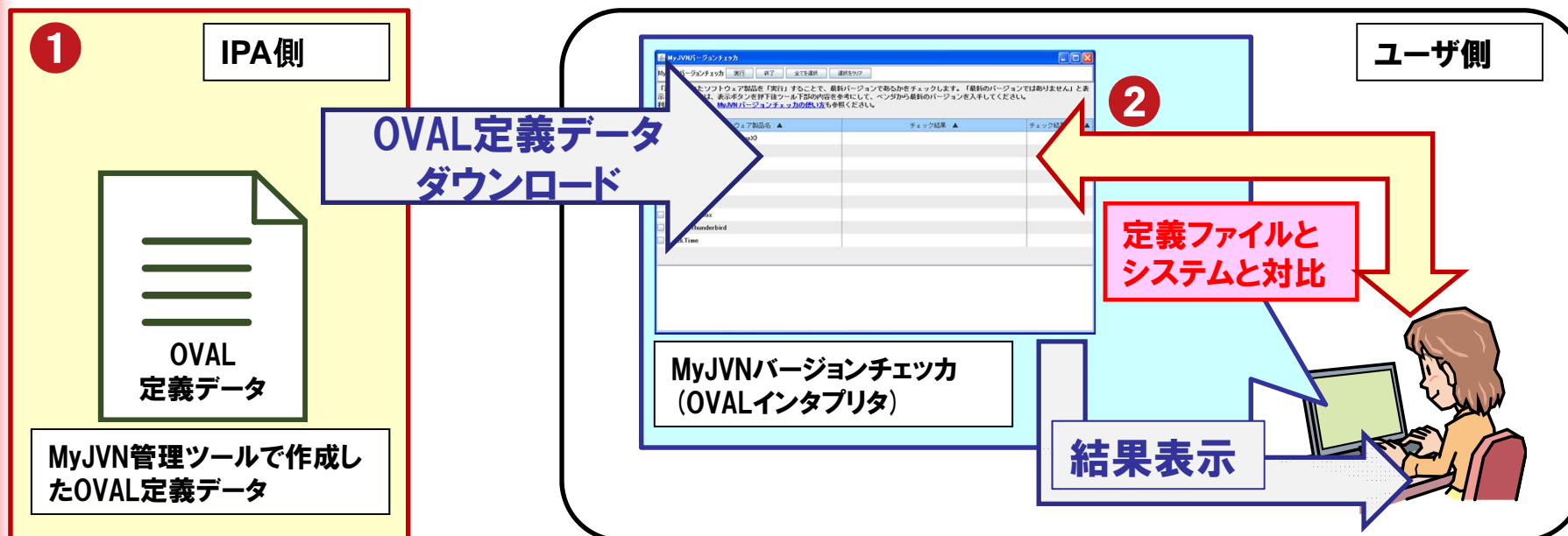
(*1) Extensible Markup Language

OVAlとは MyJVNバージョンチェッカでは



MyJVNバージョンチェッカでのOVAl利用方法

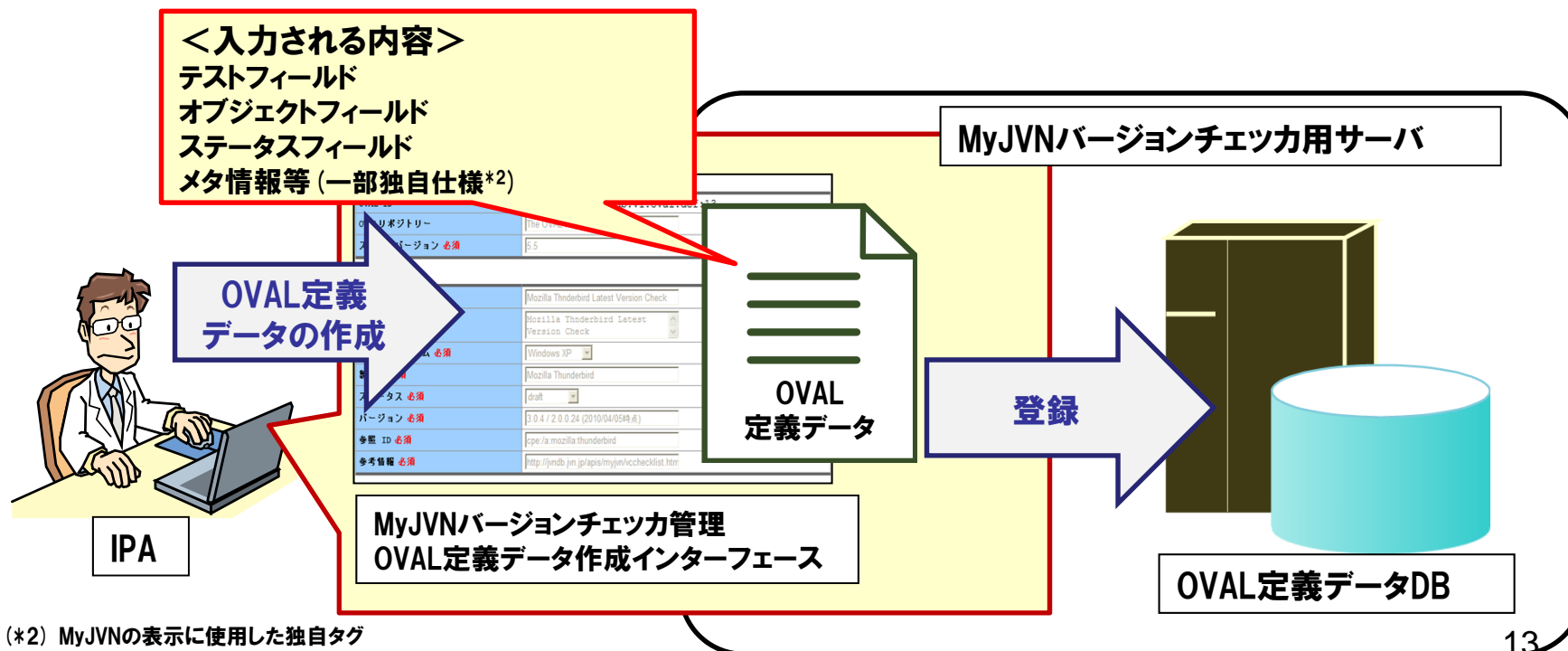
- 1 IPA内のMyJVN管理ツールを利用してOVAl定義データを作成する
- 2 OVAlインタプリタであるMyJVNバージョンチェッカでOVAl定義データとシステムの情報に対比させる



OVAL定義データ MyJVNバージョンチェッカでは



- MyJVN管理ツールでOVAL定義データを作成
 - MyJVN管理ツールに必要な情報を入力して作成
 - **OVAL定義データ** (OVALのフォーマットに則ったXMLベースの定義ファイル) を作成する (スライド11の1)



OVAL定義データ



- OVAL定義データは
 - XMLベースで記述する
 - 主に次の三つのフィールドを記述する

```
<?xml version="1.0" encoding="UTF-8" ?>
```

```
<oval_definitions>
```

```
<!-- 中略 -->
```

```
<tests>
```

テストフィールド

```
<registry_test id="oval:myjvn.oval:tst:1001" check_existence="at_least_one_exists" check="at least one">
```

```
<object object_ref="oval:myjvn.oval:obj:1001"/>
```

```
<state state_ref="oval:myjvn.oval:ste:1001"/>
```

```
</registry_test>
```

```
</tests>
```

どこのオブジェクトフィールド・ステータスフィールドを使って検査するかを記述するフィールド

```
<objects>
```

オブジェクトフィールド

```
<registry_object id="oval:myjvn.oval:obj:1001">
```

```
<hive>HKEY_LOCAL_MACHINE</hive>
```

```
<key>SOFTWARE\IPA\MyJVN</key>
```

```
<name>CurrentVersion</name>
```

```
</registry_object>
```

```
</objects>
```

確認対象を記述するフィールド

例:

バージョンが格納されているレジストリ位置
(レジストリだけではなく、ファイルの場合も)

```
<states>
```

ステータスフィールド

```
<registry_state id="oval:myjvn.oval:ste:1001">
```

```
<value>1.0</value>
```

```
</registry_state>
```

```
</states>
```

確認対象の状態を記述するフィールド

例:

比較対象となる最新バージョン値

```
</oval_definitions>
```

OVAL定義データ



IPA®

• OVAL定義データ

— 例:「Windows XP Firefox」の場合 at MyJVNバージョンチェッカ

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- 中略: 製品情報等々を記載 -->
<tests>
  <registry_test xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows" id="oval:jp.jvn.jvndb.v1.oval:tst:91" version="1" comment="Mozilla Firefox version greater than or equal 3.5.9 check" check="all">
    <object object_ref="oval:jp.jvn.jvndb.v1.oval:obj:91"/>
    <state state_ref="oval:jp.jvn.jvndb.v1.oval:ste:91"/>
  </registry_test>
  <registry_test xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows" id="oval:jp.jvn.jvndb.v1.oval:tst:92" version="1" comment="Mozilla Firefox version greater than or equal 3.6.3 check" check="all">
    <object object_ref="oval:jp.jvn.jvndb.v1.oval:obj:92"/>
    <state state_ref="oval:jp.jvn.jvndb.v1.oval:ste:92"/>
  </registry_test>
</tests>
<objects>
  <registry_object xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows" id="oval:jp.jvn.jvndb.v1.oval:obj:91" version="1">
    <hive>HKEY_LOCAL_MACHINE</hive>
    <key>SOFTWARE\Mozilla\Mozilla Firefox</key>
    <name>CurrentVersion</name>
  </registry_object>
  <registry_object xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows" id="oval:jp.jvn.jvndb.v1.oval:obj:92" version="1">
    <hive>HKEY_LOCAL_MACHINE</hive>
    <key>SOFTWARE\Mozilla\Mozilla Firefox</key>
    <name>CurrentVersion</name>
  </registry_object>
</objects>
<states>
  <registry_state xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows" id="oval:jp.jvn.jvndb.v1.oval:ste:91" version="1">
    <value datatype="string" operation="pattern match" var_ref="oval:jp.jvn.jvndb.v1.oval:var:91">^3.5.(9|[1-9][0-9]) *</value>
  </registry_state>
  <registry_state xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows" id="oval:jp.jvn.jvndb.v1.oval:ste:92" version="1">
    <value datatype="string" operation="pattern match" var_ref="oval:jp.jvn.jvndb.v1.oval:var:92">^3.6.([3-9]|[1-9][0-9]) *</value>
  </registry_state>
</states>
<!-- 後略 -->
```

テストフィールド

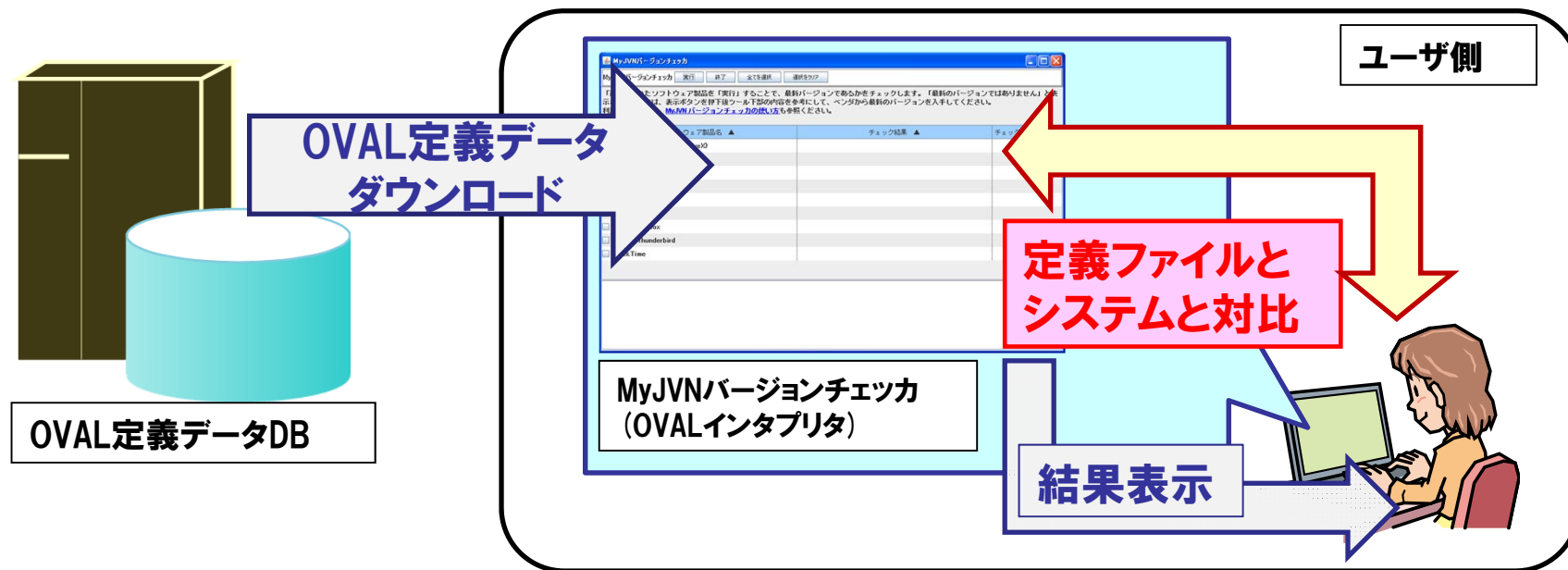
オブジェクトフィールド

ステータスフィールド

15



- MyJVNバージョンチェッカ (OVALインタプリタ) で
OVAL定義ファイルとシステム情報を対比
 - **OVALインタプリタ** (OVAL定義データを解釈するプログラム) で処理をさせる (スライド11の2)

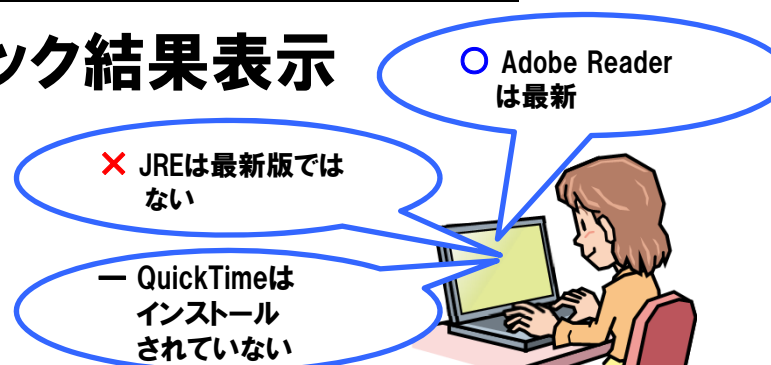


MyJVNバージョンチェツカの裏側



• MyJVNバージョンチェツカの動作の流れ

1. <ユーザ> MyJVNバージョンチェツカの起動
2. <MyJVN> システムのOS判定
3. <MyJVN> OVAL定義データのリストの取得
4. <ユーザ> 表示されているソフトウェアのチェック
5. <ユーザ> バージョンチェックスタート
6. <MyJVN> 該当のOVAL定義データのダウンロード
7. <MyJVN> OVAL定義データとシステムの比較
8. <MyJVN> バージョンチェック結果表示

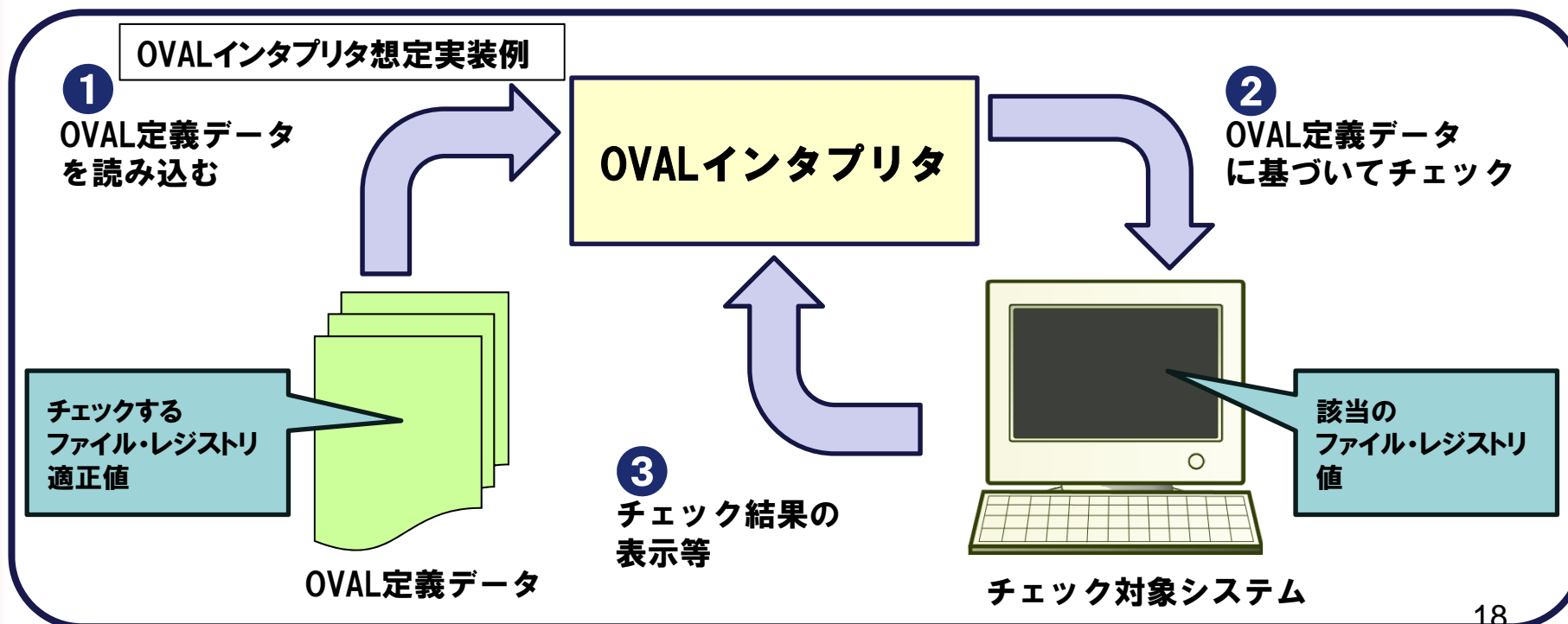


OVALインタプリタ



• OVALインタプリタ

- 「OVAL定義データ」に沿ってシステム情報を対比するプログラム
- それぞれの対比の結果を0か1(YesかNo)で返す



OVALのおさらい



- OVALのポイント
 - OVAL定義データを作成する
 - XMLベースで記述
 - チェックに必要な3つのフィールド(+メタ情報)を記述
 - テストフィールド
 - オブジェクトフィールド
 - ステータスフィールド
 - OVALインタプリタを作成する
 - OVAL定義データの記述どおりにチェックする
 - チェック結果を1か0で返す
- これでOVALを使ったオリジナルのバージョンチェッカを作れます！

OVALを使った オリジナル・バージョンチェツカの作り方

OVALを使った バージョンチェツカの作り方

- OVALを使うメリット = **皆で使える!**
 - OVAL定義データは必ずしも自分で用意しなくてもよい
 - 既に作成されているかもしれない
 - OVAL定義データを自組織用に作成すれば、既存のOVALインタプリタを利用して、自分専用のバージョンチェツカにカスタマイズできる
 - 最初から自分で作らなくてもよいかもしれない

OVALを使った バージョンチェツカの作り方



- バージョンチェツカの作り方
 - OVAL定義データの作成・取得
 - 方法1:自作する
 - 例えば、OVAL定義データ作成ソフトを作ると便利
 - 方法2:公開されているOVAL定義データを使用する
 - OVALリポジトリ(後述)やMyJVN API*3を利用する等
 - OVALインタプリタの作成
 - 方法1:自作する
 - MyJVN APIを利用し、足りない分は自分で作成する等
 - 方法2:公開されているOVALインタプリタを使用する
 - 例:OVAL Interpreter
 - <http://sourceforge.net/projects/ovaldi/>

(*3) JVN iPedia の情報を、Web を通じて利用するためのソフトウェアインタフェース
OVAL定義データ用のAPIについても一般利用できるように整備中

OVALリポジトリ

- OVALリポジトリ

- OVAL関係者から構成されたコミュニティによって運用されているOVAL定義データのデータベース
- OVALリポジトリの例
 - MITRE^{*4}社: OVALリポジトリ管理サイト
 - <http://oval.mitre.org/repository/>
 - Red Hat社: 自社のセキュリティアドバイザリに対応したパッチ適用チェックのためのOVAL定義データ
 - <http://www.redhat.com/oval>
 - NIST^{*5}: ソフトウェア製品のセキュリティ設定に関するチェックと、脆弱性対策のためのチェックを目的としたOVAL定義データ
 - <http://nvd.nist.gov/scap/scap.cfm>

(*4) 米国政府向けの技術支援や研究開発を行う非営利組織

(*5) National Institute of Standards and Technology米国国立標準技術研究所

OVALの有効な利用方法

- **アップデート機能のないソフトウェアを最新バージョンに維持する**
 - アップデート機能がないソフトウェアでも、「OVAL定義データ」があれば、最新バージョンであることを機械処理で確認ができる
- **組織における一括のセキュリティ対策の機械処理**
 - 各端末に「OVALエージェント（仮称）」を入れておき、そこからシステムの情報を、管理端末へ送付する
 - OSやソフトウェアに依存することなく、一括の管理が可能
 - 「OVAL定義データ」を全て自分で作る必要はない

特にOVALを作ってほしい人

- **ソフトウェア製品開発者がOVAL定義データ**
 - 多くの開発者の方々がOVAL定義データを作成することでユーザはインストールしているソフトウェアの一括のバージョンチェックの一助に
- **SIベンダやセキュリティベンダ等がOVALインタプリタ**
 - 開発者のOVAL定義データだけではなく、そのOVAL定義データを処理できるOVALインタプリタが必須になる

**「製品開発者のOVAL定義データ」と
それを処理できる「OVALインタプリタ」によって、
セキュリティ対策の機械化処理の実現へ！**

まとめ

セキュリティ対策の機械処理

- OVALで脆弱性対策の機械処理が可能
- OVALでセキュリティ設定のチェックの機械処理も可能（例：MyJVNセキュリティ設定チェッカ）
- しかし、どこかで人が介在しなければなりません
 - 誰かが「OVAL定義データ」と「OVALインタプリタ」を作らなければなりません
- 皆がOVALを使うことで、これらの手間を軽減することが可能になっていきます

脆弱性対策の機械処理のためには

Q. どのようにOVALを利用すれば組織の脆弱性対策を自動化できるでしょうか

→すぐにOVALを採用することは難しいかもしれませんが、ちょっとした実験は可能です

ちょっとOVAL使ってみませんか？

