Vol.2011-CSEC-54 No.33
Vol.2011-SPT-1 No.33
2011/7/13

MyJVN バージョンチェッカ ～MyJVN を用いた脆弱性対策自動化の実現～

寺田真敏　斉藤良彰　大森雅司
相馬基邦　小林偉昭

国内においても，脆弱性対策情報の提供環境が充実してきている．しかし，対策情報の多くは主に文書として構成されているために，脆弱性の有無をチェックして対策を促すなどの脆弱性対策に関わる処理の機械化については発展途上にある．特に，2009 年に流布したウェブ誘導型マルウェアである Gumblar 以降，アプリケーションの脆弱性を悪用したマルウェア感染への対策は急務であり，アプリケーションを常に最新バージョンに維持することが必要となってきている．本稿では，このような問題解決に向けて，JVN で整備を進めている脆弱性対策に関わる処理の機械化を目指すフレームワーク MyJVN について述べる．また，情報セキュリティ問題をチェックする手続き仕様 OVAL (Open Vulnerability Assessment Language)を用いた MyJVN バージョンチェッカの実装について述べる．

The concept and implementation of MyJVN Version Checker

Masato Terada, Yoshiaki Saito, Masashi Ohmori,
Motokuni Souma, Hideaki Kobayashi

The spread of "Attacks via a Legitimate Website" in 2009, in which a legitimate website is defaced and users accessing it suffer from certain damages. For these threats, we can apply traditional measures such as keeping up-to-date operating systems, applications, plug-ins, and virus definition files of antivirus software. In order to improve the keeping up-to-date environment, it is necessary to improve the security information service environment for automation of vulnerability countermeasure. In this paper, firstly we will describe the framework of MyJVN, which is JVN Security Content Automation Framework to establish SCAP collaboration. Secondly, we will introduce our implementation of OVAL (Open Vulnerability Assessment Language) for MyJVN Version Checker.

1. はじめに

2008 年以降，普段の業務や日常的に使用しているアプリケーションの脆弱性を狙った攻撃が，標的型攻撃や Web 誘導型マルウェアの流布において利用される傾向にある．この背景には，受動型攻撃を主流とする侵害活動に変わってきたこと[1]，良く利用されているアプリケーション，いわゆる定番ソフトウェアに内在する脆弱性の増加が挙げられる．国内の脆弱性対策データベース JVN iPedia によれば，定番ソフトウェアとして良く利用されているアプリケーションの脆弱性登録件数は，2008 年から 2010 年にかけて 2 倍近く増えている[2].

国内においても，JVN や JVN iPedia など脆弱性対策情報の提供環境は充実してきており，脆弱性の傾向などを把握しやすくなってきている．しかし，対策情報の多くは主に文書として構成されており，脆弱性の有無をチェックして対策を促すなど脆弱性対策に関わる処理の機械化については発展途上にある．特に，2009 年に流布した Web 誘導型マルウェアである Gumblar 以降，アプリケーションの脆弱性を悪用したマルウェア感染への対策は急務であり，アプリケーションを常に最新バージョンに維持することが必要となってきている状況にある．

本稿では，このような問題解決に向けて，JVN で整備を進めている脆弱性対策に関わる処理の機械化を目指すフレームワーク MyJVN(JVN 脆弱性対策機械処理基盤)について述べる．次に，このフレームワークの下で開発した MyJVN バージョンチェッカの概要と，情報セキュリティ問題をチェックする手続き仕様 OVAL(Open Vulnerability Assessment Language)を用いた実装について述べる．

2. 関連動向

本章では，脆弱性と脆弱性対策に関わる機械化処理の取り組みの動向について述べる．

2.1 脆弱性の動向

Web 誘導型マルウェアの侵害活動に関連する動向として，良く利用されているアプリケーション，いわゆる定番ソフトウェアと Web アプリケーションの脆弱性件数を取り上げる．

(1) 良く利用されているアプリケーションの脆弱性

国内の脆弱性対策データベース JVN iPedia に登録されている，良く利用されているアプリケーションの脆弱性件数に関する年別推移を図 1 に示す. 特に, Adobe Acrobat, Adobe Reader，Adobe Flash Player は年々登録件数が増加しており，2008 年から 2010

(独)情報処理推進機構 (IPA)
Information-technology Promotion Agency, Japan

Vol.2011-CSEC-54 No.33
Vol.2011-SPT-1 No.33
2011/7/13

年では 3 倍(Adobe Flash Player は 20 件から 57 件に増加)から 4 倍(Adobe Acrobat と Adobe Reader はそれぞれ 17 件から 68 件に増加)となっている．

(2) Web アプリケーションの脆弱性

米国の脆弱性対策データベース NVD(National Vulnerability Database)に登録されている脆弱性総件数ならびに，Web アプリケーションの脆弱性件数に関する年別推移を図 2 に示す．2008 年から Web アプリケーションの脆弱性件数の割合が増加している[3].

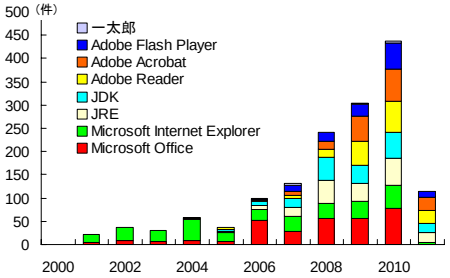


図 1：良く利用されているアプリケーションの脆弱性推移(JVN iPedia)

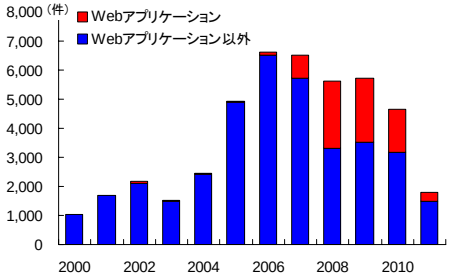


図 2：Web アプリケーションの脆弱性推移(NVD)

2.2 脆弱性対策に関わる機械化処理

脆弱性対策に関わる機械化処理の取り組みについては，図 3 に示すサイバーセキュリティ情報交換に関する仕様の包含関係を中心に述べる．

(1) SCAP [4]

米国では，2002 年の FISMA(Federal Information Security Management Act：連邦情報セキュリティマネジメント法)の施行以降，情報セキュリティ規格やガイドラインに従い，情報システムにセキュリティ要件を反映する活動が進められている．2007 年には，米 OMB(Office of Management and Budget：行政予算管理局)主導により，Windows ソフトウェアを共通セキュリティ設定に準拠させる FDCC(Federal Desktop Core Configuration：連邦政府共通デスクトップ基準)が導入された．2010 年には，連邦政府 CIO 評議会(Federal CIO Council)主導で，広く使われているプラットフォームを対象に共通セキュリティ設定に準拠させる USGCB(The United States Government Configuration Baseline：米国政府共通設定基準)が導入された．

SCAP(Security Content Automation Protocol：セキュリティ設定共通化手順)は，米 NIST(National Institute of Standards and Technology：米国国立標準技術研究所)が中心となって，連邦政府で使われているプラットフォームが FDCC，USGCB に沿っているかを自動チェックするための仕様群として普及展開されている(表 1).

表 1：SCAP を構成する仕様群の概要

共通基準	概要
脆弱性の共通識別子 CVE (Common Vulnerabilities and Exposures)	プログラム自身に内在するプログラム上のセキュリティ問題に一意の番号(脆弱性識別子)を付与する仕様
共通セキュリティ設定一覧 CCE (Common Configuration Enumeration)	プログラムが稼働するための設定上のセキュリティ問題に一意の番号を付与する仕様
共通プラットフォーム一覧 CPE (Common Platform Enumeration)	情報システム，プラットフォーム，ソフトウェアパッケージに一意の名称を付与する仕様
共通脆弱性評価システム CVSS (Common Vulnerability Scoring System)	脆弱性自体の特性，パッチの提供状況，ユーザ環境での影響度などを考慮し脆弱性の影響度を評価する仕様
セキュリティ設定チェックリスト記述形式 XCCDF (EXtensible Checklist Configuration Description Format)	情報セキュリティチェックリストやベンチマークなどの文書を記述するための仕様
セキュリティ検査言語 OVAL (Open Vulnerability Assessment Language)	プログラム上のセキュリティ問題や設定上のセキュリティ問題をチェックするための手続き仕様

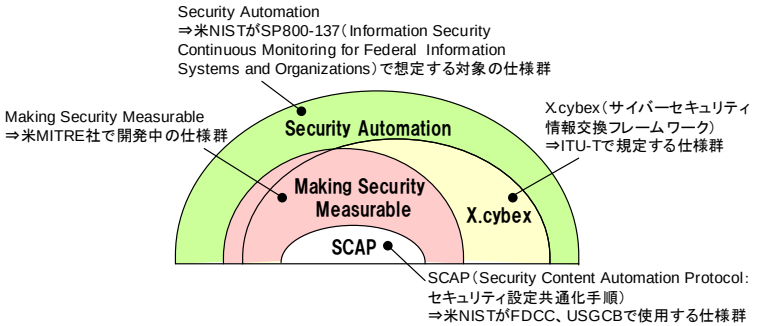


図 3：サイバーセキュリティ情報交換に関する仕様の包含関係

(2) Making Security Measurable [5]

情報セキュリティに関連する共通識別子，共通仕様を整備することにより，情報セキュリティ対策全般に関わる処理の機械化も促進されることになる．SCAP の仕様開発を担当している米 MITRE 社では，2007 年頃から共通識別子，共通仕様，リポジトリの視点から”Making Security Measurable”と呼ぶ活動を推進している．表 1 に示す SCAP を構成する仕様群の他に，攻撃パターンの識別を共通化する CAPEC(Common Attack Pattern Enumeration and Classification)，イベントの識別を共通化する

Vol.2011-CSEC-54 No.33
Vol.2011-SPT-1 No.33
2011/7/13

CEE(Common Event Expression)，機械化(自動化)できないセキュリティ問題をチェックするための質問型の手続き仕様 OCIL(Open Checklist Interactive Language)などがある．

(3) X.cybex

ITU-T において，脆弱性対策情報のフォーマット，番号体系などの技術仕様を規格化するサイバーセキュリティ情報交換フレームワーク X.cybex(X.1500)(Cybersecurity Information Exchange Framework)の標準化が進められている[6]．X.cybex は，Making Security Measurable の仕様群を中心に構成されている．また，ISO では，脆弱性対策(脆弱性情報の開示，脆弱性対応手順)に関する規格の標準化活動を行っており，X.cybex は，ISO の推進する脆弱性対策を補完する技術文書として位置付けられている(表 2)．

(4) Security Automation

米 NIST では，2010 年 12 月，定常的なシステム監視の概念規格 SP 800-137 のドラフトをリリースした[7]．この規格の監視対象範囲は，脆弱性管理，パッチ管理，マルウェア検知，資産管理，設定管理，ソフトウェア保守，イベント管理，情報管理，ライセンス管理，インシデント管理，ネットワーク管理と多岐を想定しており，技術仕様は明示されていないが，広範囲の自動化を視野にいれている．

表 2：標準化活動の関係

活動主体	件名	概要	関係性
ITU-T/SG17 Q4	X.cybex サイバーセキュリティ情報交換フレームワーク	脆弱性対策情報のフォーマット，番号体系などの技術仕様について標準化を進めている．	脆弱性情報その他の情報共有の客体を特定するための付番や形式化のための方法を整理，メニュー化する．
ISO/SC27 WG3	29147 "Vulnerability Disclosure" 脆弱性情報の開示	ベンダが脆弱性情報を受信する際の心得，ベンダが脆弱性情報に関する情報を開示する際の心得など，運用上，考慮すべき点を標準化している．	運用を規定する．上記の付番や形式の中から必要なものを選択し使用することになる．
	30111 "Vulnerability Handling Processes" 脆弱性対応手順	ベンダが脆弱性情報を受信してから，対策を完了させるまでの手順を中心に，考慮すべき点を標準化している．	

3. JVN 脆弱性対策機械処理基盤

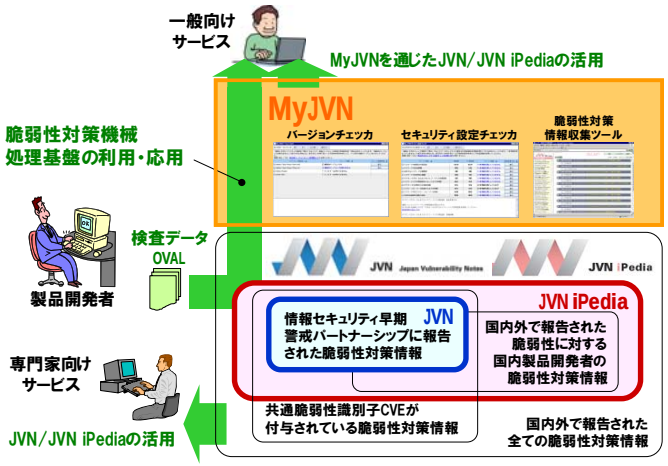
本章では，国内での脆弱性対策推進を支援するために，脆弱性対策に関わる処理の機械化を目指すフレームワーク MyJVN について述べた後，解決したい課題について提示する．

3.1 (JVN+JVN iPedia)×MyJVN

JVN 脆弱性対策機械処理基盤は，図 4 に示す通り，JVN，JVN iPedia，MyJVN の 3 つのコンポーネントから構成されている．

JVN(Japan Vulnerability Notes)は，情報セキュリティに関わるシステム管理者ならびにシステムエンジニア向けに脆弱性対策情報を広く告知することを目的とした情報公開サイトである．2003 年 2 月に JPCERT/CC の試行サイトとして運用を開始した[8]．2004 年 7 月には経済産業省告示「ソフトウェア等脆弱性関連情報取扱基準」[9]を受け，日本国内の製品開発者の脆弱性対応状況を公開するサイトとして情報発信を行なっている[10]．2007 年 4 月からは，即時性と網羅性とを備えた情報提供を実現するため，JVN と JVN iPedia[11]の 2 つのコンポーネント構成に拡張している．MyJVN は，処理の機械化や自動化を考慮した流通基盤であり，JVN と JVN iPedia に登録された脆弱性対策情報を用いたサービスを構築するフレームワークである．2008 年 10 月からフィルタリング型情報提供サービスを提供している[12]．

JVN と JVN iPedia の構成と外部連携を図 5 に示す．JVN iPedia では，国内の脆弱性対策の基点データベースとしての役割を果たすために，JVN ならびに NVD から国内で利用されている製品を対象にした脆弱性対策情報を取り込んでいる．また，JVN と JVN iPedia に登録されている脆弱性対策情報は，インターネット向け脆弱性対策情報の情報源として NVD などの主要な脆弱性対策データベースから参照されている．



【MyJVN】JVN と JVN iPedia に登録されている脆弱性対策情報を対策実施に直結したサービスに繋げるための仕組みを提供する．

【JVN iPedia】国内の製品開発者から公開された対策情報，および海外の脆弱性対策データベースに登録された情報に基づき，国内で利用されている製品を対象にした脆弱性対策情報を網羅し蓄積する．

【JVN】国内の情報セキュリティ早期警戒パートナーシップ，および海外の調整機関に届けられた脆弱性関連情報に関して製品開発者と調整した脆弱性対策情報をタイムリーに公開する．

図 4：JVN 脆弱性対策機械処理基盤(JVN+JVN iPedia)×MyJVN

Vol.2011-CSEC-54 No.33
Vol.2011-SPT-1 No.33
2011/7/13

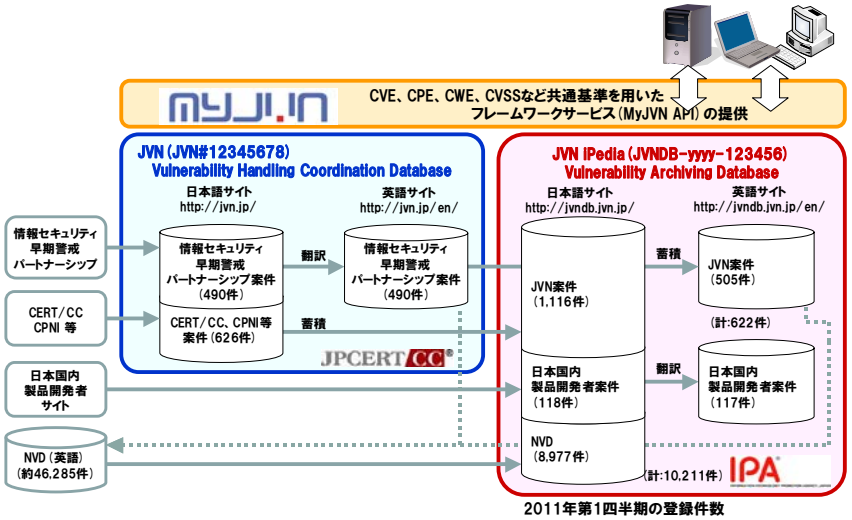


図 5：JVN と JVN iPedia の構成と外部連携

3.2 解決したい課題

本節では，脆弱性対策に関わる処理の機械化を目指すフレームワーク MyJVN において解決したい課題を示す．

- 国際性と地域性(現地化)とを両立させた脆弱性対策機械処理基盤の整備
  脆弱性対策情報などの国際的な流通に利用可能な機械処理基盤であり，かつ，言語，製品マーケットやインシデントなどの地域性(現地化)を考慮した機械処理基盤であることは必要不可欠である．
- 脆弱性対策確認を容易にする機械処理基盤の整備
  脆弱性の有無をチェックして対策を促すなどの脆弱性対策に関わる処理の機械化については発展途上にある．特に，多くの利用者が複数ベンダから提供される定番ソフトウェアを利用しており，日々公表される脆弱性対策情報をチェックし，脆弱性の影響有無を判断するには手間が掛かる．

4. 機械処理基盤としての検査データ提供サービスの実現

本章では，前述の課題を解決する，脆弱性の影響有無の判定を自動化するための検査データ提供サービスの実現方式について述べる．

4.1 システム構成

検査データ提供サービスは，次の 3 つのコンポーネントから構成されている(図 6).

(1) OVAL データベース

OVAL[13]は，プログラム上のセキュリティ問題や設定上のセキュリティ問題をチェックする手続き仕様である．2002 年 10 月に開催された SANS Network Security 2002 において，米 MITRE 社から，その仕様が開示された．XML を用いて汎用的に作られた仕様であり，セキュリティ修正プログラムの適用状況や，稼動しているプラットフォーム／ソフトウェアパッケージの判定に利用できる．

MyJVN においては，セキュリティ問題をチェックする手続き仕様を整備し，検査データ(OVAL 定義データ)として提供していくことで，脆弱性の有無をチェックして対策を促すなどの脆弱性対策に関わる処理の機械化が実現できる．将来的には，チェック項目が記載された OVAL 定義データを流通基盤上で交換することにより，脆弱性対策の支援や国際間での脆弱性対策情報の相互運用といった可能性が広がる．

(2) CPE データベース

JVN に登録されている製品ならびに製品提供者情報と CPE 名とを対応付けるデータベースである．OVAL 定義データが関連する CPE 名を識別した後，CPE データベースから製品名などの製品情報を取得する．

(3) MyJVN API モジュール

OVAL ならびに CPE データベースを利用して，「システムにはどのような脆弱性が存在するのか」「システムにはどのような製品がインストールされているのか」という視点からチェック項目が記載された OVAL 定義データを提供する．

4.2 MyJVN API

MyJVN API には，JVN データベースに登録されている脆弱性対策情報から，製品提供者一覧，製品一覧，脆弱性概要情報一覧，脆弱性詳細情報を XML 形式で出力するフィルタリング型情報提供用 API と，OVAL データベースから，OVAL 定義一覧，OVAL 定義データを XML で出力する検査データ提供用 API がある．

(1) リクエスト

URL の基本構成は次の通りである．

http://jvndb.jvn.jp/myjvn?method=メソッド&パラメタ

MyJVN API で指定された URL に対して，リクエストパラメタを指定し GET/POST にて HTTP 要求を発行すると，API の呼び出し結果であるレスポンスを XML 形式で取得できる．また，リクエストパラメタとして，表 3 に示す取得情報を指定する”method=メソッド”と CPE 名などのパラメタが使用可能である．

Vol.2011-CSEC-54 No.33
Vol.2011-SPT-1 No.33
2011/7/13

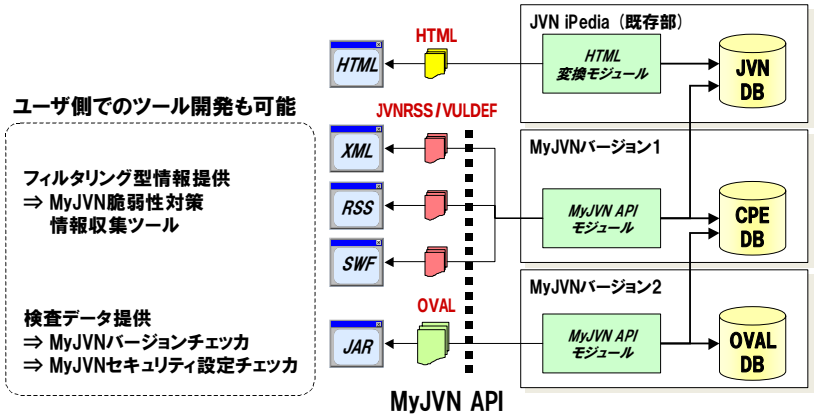


図 6：検査データ提供サービスのシステム構成

表 3：MyJVN API のメソッド一覧

分類	名称	概要
フィルタリング型情報提供用	製品提供者一覧取得 getVendorList	フィルタリング条件に該当する製品提供者一覧を XML 形式で取得する.
	製品一覧取得 getProductList	フィルタリング条件に該当する製品一覧を XML 形式で取得する.
	脆弱性対策概要情報一覧取得 getVulnOverviewList	フィルタリング条件に該当する脆弱性対策情報の概要一覧を RSS 1.0 + mod_sec 形式で取得する.
	脆弱性対策詳細情報取得 getVulnDetailInfo	フィルタリング条件に該当する脆弱性対策詳細情報を VULDEF 形式で取得する.
検査データ提供用	OVAL 定義一覧取得 getOvalList	フィルタリング条件に該当する OVAL 定義一覧を XML 形式で取得する.
	OVAL 定義データ取得 getOvalData	該当する OVAL 定義データを XML 形式で取得する.

4.3 MyJVN バージョンチェッカ

MyJVN バージョンチェッカ(以降，MyJVN VC)は，MyJVN API と連動して動作する脆弱性影響有無チェックツールであり，Web ブラウザ上で稼動する Java ベースの GUI ツールとなっている.

(1) チェックリスト型 GUI

米国では，2002 年から進められてきた NCP(National Checklist Program)を通して，情報セキュリティのためのチェックリストという考え方が普及しており，その概念は，基準としての FDCC，USGCB，技術仕様としての SCAP に継承されている．また，国内でも情報セキュリティ対策項目を一覧とした紙ベースのチェックリストは認知されていることから，MyJVN VC では，チェックリストをベースとした GUI を採用した．さらに，脆弱性による影響有無の判定を，アプリケーションの最新バージョンがインストールされているかどうかに置き換えて判定することで簡易化すると共に，その判定結果を，「○：最新のバージョンである」，「×：最新のバージョンではない」，「－：インストールされていない」の 3 つとすることで利用者へのわかりやすさを実現している(図 7).

図 7：チェックリスト型 GUI (MyJVN バージョンチェッカ)

(2) MyJVN API との連動

MyJVN VC と MyJVN API との連動フローを図 8 に示す．getOvalList でチェックの対象となる製品の OVAL 定義一覧を取得した後，GUI 上にチェックリストを表示する．次に，OVAL 定義データの ID(例：oid=oval:jp.jvn:def:11)に基づき，getOvalData で該当する OVAL 定義データ(図 9)を取得する．MyJVN VC では，getOvalList，getOvalData の 2 つの API を組み合わせて使うことで，複数アプリケーションの最新バージョンのインストール有無に必要な OVAL 定義データを取得している.

(3) OVAL 定義データの解釈処理

MyJVN VC は，OVAL 定義データに記載された Windows システムのレジストリ情報，ファイル属性情報などの設定値を利用して，脆弱性影響有無をチェックする．例えば，Windows システムにおいて，MyJVN VC が最新の場合には，レジストリキー HKLM¥SOFTWARE¥IPA¥MyJVN¥CurrentVersion の値に 1.0 が設定されているとする．OVAL 定義データでは，これらの情報を<registry_object>と<registry_state>に格納し，

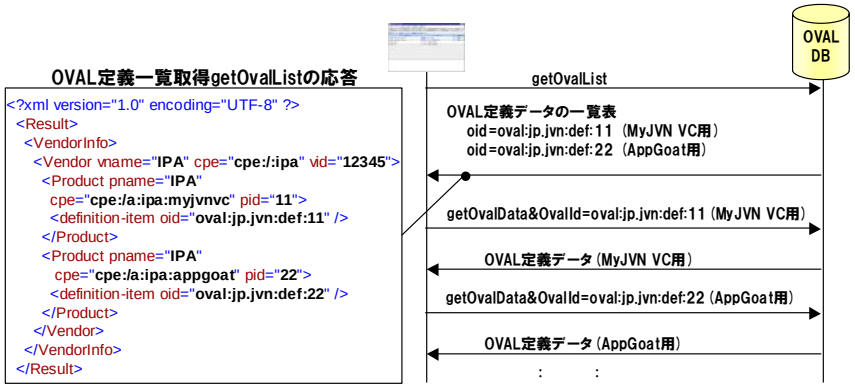


図 8：MyJVN VC と MyJVN API との連動動作

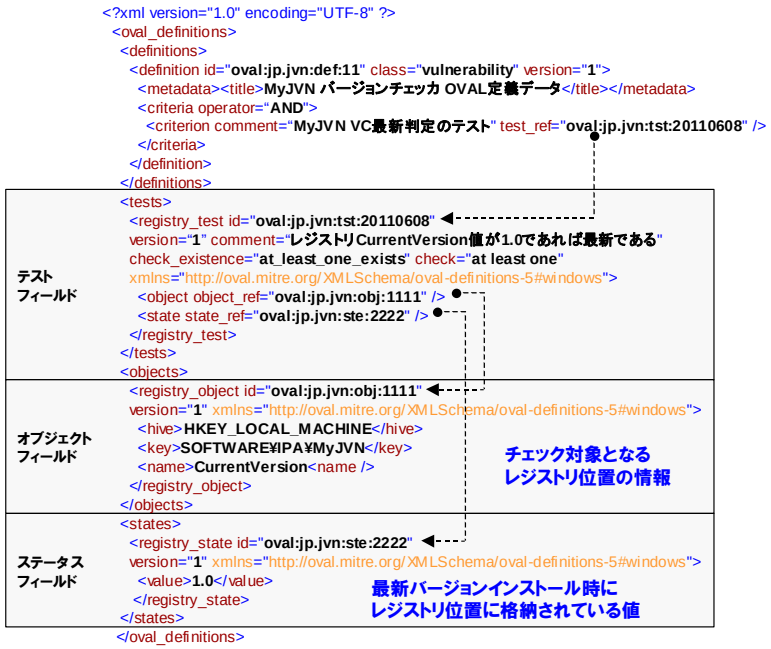


図 9：OVAL 定義データ取得 getOvalData の応答(OVAL 定義データ)

実際に格納されている設定値と比較することで判定を行なう．ただし，OVAL は仕様上，判定結果を ture か false の 2 値で出力する仕組みになっている．そこで，MyJVN VC では，HKLM¥SOFTWARE¥IPA¥MyJVN¥ CurrentVersion のキーが存在しない場合には「インストールされていない」というような独自の解釈処理を追加することで 3 値判定への拡張を実現している．

5. おわりに

本稿では，脆弱性対策に関わる処理の機械化を目指すフレームワーク MyJVN の課題として，脆弱性対策確認を容易にし，国際性と地域性(現地化)とを両立させた脆弱性対策機械処理基盤の整備の視点から示した．また，課題を解決するために，セキュリティ問題をチェックする手続き仕様 OVAL を用いた処理の機械化と，開発した MyJVN API サービスならびに，MyJVN バージョンチェッカについて報告した．

脆弱性対策に関わる処理の機械化を推進するにあたり，国際性(インターネットにおける脆弱性対策情報の情報源)と地域性(日本国内向けの脆弱性対策情報データベース)とを両立させた脆弱性対策機械処理基盤の整備は重要であると考えている．今後は，共通識別子，共通仕様の導入を進めながら，国際間での脆弱性対策情報の相互運用を視野に入れた MyJVN の普及推進を進めていく予定である．

参考文献

1) IPA: 2011 年版 10 大脅威 進化する攻撃, http://www.ipa.go.jp/security/vuln/10threats2011.html
2) IPA: 脆弱性対策情報データベース JVN iPedia の登録状況 [2011 年第 1 四半期(1 月～3 月)], http://www.ipa.go.jp/security/vuln/report/JVNiPedia2011q1.html
3) NVD CVE Statistics, http://web.nvd.nist.gov/view/vuln/statistics
4) The Security Content Automation Protocol (SCAP), http://scap.nist.gov/
5) MITRE: Making Security Measurable, http://measurablesecurity.mitre.org/
6) Cybersecurity Information Exchange techniques (CYBEX), http://www.itu.int/en/ITU-T/studygroups/com17/Pages/cybex.aspx
7) NIST: SP 800-137, Information Security Continuous Monitoring for Federal Information Systems and Organizations, http://csrc.nist.gov/publications/drafts/800-137/draft-SP-800-137-IPD.pdf
8) 寺田，高田，土居, “脆弱性対策情報データベース JVN の提案”, 情処論文誌 Vol.46 No.5 (2005)
9) 経済産業省，「情報セキュリティ早期警戒パートナーシップ」の運用開始について, http://www.meti.go.jp/policy/it_policy/press/0005399/
10) 脆弱性対策情報ポータルサイト, http://jvn.jp/
11) 脆弱性対策情報データベース, http://jvndb.jvn.jp/
12) 脆弱性対策情報共有フレームワーク-MyJVN, http://jvndb.jvn.jp/apis/myjvn/
13) OVAL: Open Vulnerability Assessment Language, http://oval.mitre.org/