脆弱性対策情報の利活用基盤 MyJVN の提案

寺田真敏　　杉山　賢　　相馬基邦
永安佑希允　　山岸　正　　小林偉昭

(独)情報処理推進機構 (IPA)
〒113-6591　東京都文京区本駒込 2-28-8

概要：国内においても，様々な層での脆弱性対策情報の提供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．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 を提案する．また，MyJVN の具体的な取り組みとして，プラットフォーム識別子，Web サービス API，XML フォーマットの整備と，JVN の脆弱性対策情報を用いたサービスとして，製品視点から対策情報を選別するフィルタリング型情報提供サービスの実装について報告する．
キーワード：セキュリティ，脆弱性，API

Proposal of MyJVN for vulnerability information service framework

Masato Terada　　Ken Sugiyama　　Motokuni Souma
Yukinobu Nagayasu　　Tadashi Yamagishi　　Hideaki Kobayashi

Information-technology Promotion Agency, Japan
2-28-8 Honkomagome, Bunkyo, Tokyo, 113-6591 Japan

Abstract: Currently ,most of security information is deployed as Web page information on HTML base. In order to re-construct the information and perform correlation between collected information, it is necessary to improve the security information service environment. In this paper, firstly we will describe the framework of MyJVN. Secondly, we will introduce our feasibility study on MyJVN for filtered vulnerability information service on JVN.
Key words: Security, Vulnerability, Web API

1　はじめに

2004 年の情報セキュリティ早期警戒パートーナーシップ開始以降，国内においても，JVN(Japan Vulnerability Notes)，製品開発ベンダ，コミュニティなど様々な層での脆弱性対策情報の提供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．

本稿では，脆弱性対策確認の容易化，脆弱性対策情報の流通基盤の整備を実現するため，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 を提案する．また，提案に基づき開発した MyJVN Web API と API を利用した脆弱性対策情報チェックツールについて報告する．

2　関連動向

本章では，脆弱性対策に関わる機械化処理の取組みを，施策と脆弱性対策情報データベースの 2 つの視点から整理する．

2.1　脆弱性対策の機械化処理に関連する施策

(1) SCAP(Security Content Automation Protocol)

SCAP は，米 NIST(National Institute of Standards and Technology：国立標準技術研究所)が中心となって推進している，米国政府を対象とした情報セキュリティ管理の技術面での自動化と標準化を規定した仕様群である(表 1)[1]．SCAP では，脆弱性管理，コンプライアンス管理の一部を自動化することにより，情報システムに対するセキュリティ対策の負荷軽減と情報セキュリティ施策の推進の両立を目的としている．

表 1：SCAP を構成する仕様群の概要

仕様名	概要
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
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
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
XCCDF (EXtensible Checklist Configuration Description Format)	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
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
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

(2) WARP(Warning, Advice and Reporting Point)

WARP は，英 CPNI(Centre for the Protection of National Infrastructure：国家インフラ防護センター)が推進している中小規模組織向けのセキュリティ情報共有サービスである．WARP では，セキュリティ関連情報をメンバに配信する Filtered Warning Service，メンバ間で情報共有する Advice Brokering Service，インシデントを匿名化して報告する Trusted Sharing Service を用意している．このうち Filtered Warning Service では，FWA(Filtered Warnings Application) [2]と呼ばれるツールを用いて，セキュリティ情報の選別やカテゴリ毎のフィルタリングを実現している．

(3) EISAS(European Information Sharing and Alert System) [3]

EISAS は，EU の情報セキュリティ専門機関である ENISA(European Network and Information Security Agency：欧州ネットワーク情報セキュリティ庁)が推進するセキュリティ関連情報の共有システムを実現するためのプロジェクトである．欧州全域で情報収集と分析を行い，国別に情報配信するモデルが検討されている．

2.2 脆弱性対策情報データベースでの機械化処理

(1) OSVDB(The Open Source Vulnerability Database)

OSVDB は，中立的で，かつ詳細な脆弱性情報の提供を目的とするセキュリティコミュニティが開設した脆弱性対策情報データベースである．OSVDB では，脆弱性対策情報の活用策として，XML，MySQL 形式を用いたエクスポートや表 2に示す 3 種，計 29 件の Web サービス API を提供している[4][5].

表 2：OSVDB の提供する Web サービス API

分類	概要
Mapping API	他組織の脆弱性 ID と OSVDB の ID とを関連付ける
Core Queries	製品提供者情報(製品提供者名，製品提供者の URL，問い合わせ窓口など)や製品情報(製品名，製品内容，以前の製品名など)を取得する．
Supporting Queries	製品提供者 ID，製品 ID から製品提供者情報，製品情報を取得する

(2) NIST NVD(National Vulnerability Database)

米 NIST が提供する NVD では，NVD/CVE XML Schema[6]を用いて NVD に掲載されているコンテンツを XML 形式でエクスポートしている．また，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である CPE や，プログラム上のセキュリティ問題の分類を共通化する CWE(Common Weakness Enumeration)[7]など，脆弱性対策に関係する一意の識別子を積極的に導入している．

3 MyJVN

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 と開発したフィルタリング型情報提供サービスのシステム構成について述べる．

3.1 解決したい課題

脆弱性対策に関わる処理の機械化を通して解決したい課題は，次の通りである．

- 脆弱性対策確認の容易化

利用者にとって，日々公表される脆弱性対策情報をチェックし，どの脆弱性が自身に関係するものか判断するには手間が掛かる．

- 脆弱性対策情報の流通基盤の整備

利用者が必要とする脆弱性対策情報は，利用者の使用している製品や専門レベルにより異なるため，提供者側のサービスだけで網羅するには限界がある．

3.2 脆弱性対策に関わる利活用基盤の整備

上述の課題を解決するために，処理の機械化や自動化を考慮した利活用基盤上に，JVN の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

MyJVN を提案する．具体的には，処理の機械化や自動化を考慮した利活用基盤として，プラットフォーム識別子，Web サービス API，XML フォーマットの 3 つを整備する．次に3.3節で述べる，JVN の脆弱性対策情報を用いたサービスとして，「製品にどのような脆弱性が存在するのか」という視点から対策情報を選別するフィルタリング型情報提供サービスを実現する．

(1) プラットフォーム識別子

インストールされているソフトウェアパッケージの識別や，同じソフトウェアであってもバージョンによって脆弱性の影響は異なるため，使用しているソフトウェアパッケージを識別することは，脆弱性対策にとって重要である．そこで，MyJVN では，ソフトウェアパッケージ識別のために，米 NIST が推進している CPE(共通プラットフォーム一覧：Common Platform Enumeration)を採用する．

CPE では，ソフトウェア／ハードウェアなどのプラットフォームを一意に識別するための体系として，「パート(ハードウェア／OS／アプリケーション)」「ベンダ名」「製品名」「バージョン」「更新版／サービスパック」「エディション」「言語」を規定している．また，このようなプラットフォーム識別の仕組みを整備していくことで，将来的に脆弱性有無やパッチ適用有無の確認といった機械化処理の範囲拡大と，脆弱性対策情報の相互参照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．

cpe:/{種別}:{ベンダ名}:{製品名}:{バージョン}
:{アップデート}:{エディション}:{言語}

図 1：CPE 2.0 の識別子の基本構成

(2) Web サービス API

Web サービスの普及と共に，サービスを利用するための API を規定し，Web サービス同士の連携を図る環境整備が進められている．JVN の脆弱性対策情報を使用するための Web サービス API を規定することは，利用者自身ならびに開発者らが JVN を活用し，必要とされる新たなサービスを作り出す環境の整備につながる．MyJVN では，JVN が脆弱性対策情報を「製品にどのような脆弱性が存在するのか」という視点から対策情報を選別するフィルタリング型情報提供サービスを実現するために，製品提供者一覧，製品一覧，脆弱性対策概要情報一覧，脆弱性対策詳細情報を取得する Web サービス API と，図 2に示すフィルタリング条件を Web サービス API のパラメタとして規定する．

- 製品提供者名／CPE ベンダ名
- 製品名／CPE 製品名
- 脆弱性深刻度(危険／警告／注意)
- 脆弱性発見日
  - 日付指定
  - レンジ指定(直近 1 週間／直近 1 ヶ月)
- 脆弱性更新日
  - 日付指定
  - レンジ指定(直近 1 週間／直近 1 ヶ月)
- 言語(日本語／英語)

図 2：Web API で利用可能なフィルタリング条件

(3) XML フォーマット

OSVDB，NVD ではコンテンツ自身の活用策として，XML 形式を用いたコンテンツのエクスポートを実施している．脆弱性対策情報の活用策という点では，JVN に掲載されているコンテンツの再利用についても検討しなければならない．そこで，MyJVN では，情報の記述粒度による使い分けを考慮した概要記述向けと詳細記述向け XML フォーマット(図 3)を用意することで，コンテンツ自身の利用促進を図る．

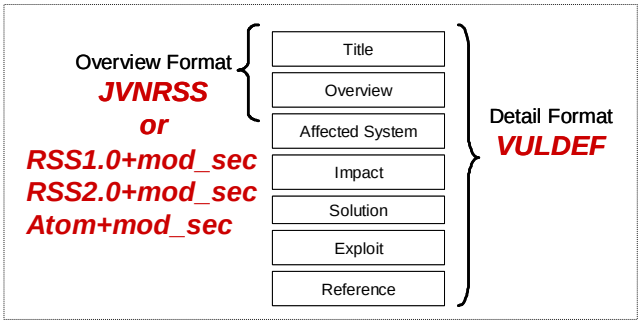


図 3：概要記述(JVN RSS)と詳細記述(VULDEF)

(a) JVNRSS(JVN RDF Site Summary)[8]

脆弱性対策情報の概要記述用 XML フォーマットで，サイトの概要をメタデータとして簡潔に記述する XML フォーマット RSS 1.0 をベースとした仕様である．脆弱性対策情報の概要を早く広く配布するためのフォーマットとして使用する．

(b) mod_sec(Qualified Security Advisory Reference)[9]

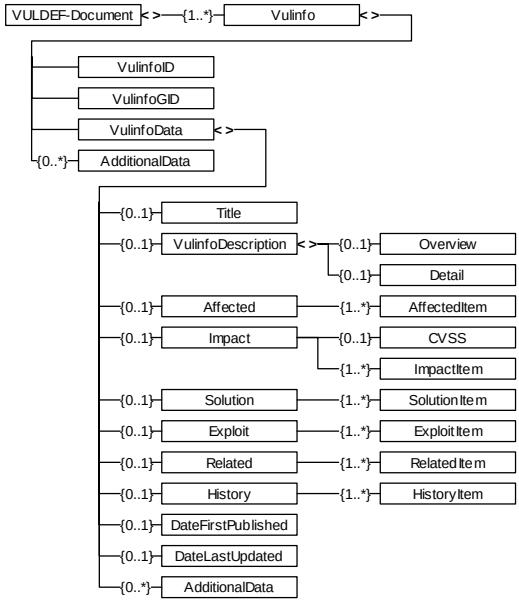
RSS 1.0，RSS 2.0，Atom 向けにセキュリティ情報の属性記述を拡張するための仕様である．脆弱性対策情報のグループ化やのために，セキュリティ情報識別子，セキュリティ関連情報の URL，共通脆弱性評価システム CVSS の評価値，CPE を用いたプラットフォーム識別子を格納する(図 4).

```
<sec:identifier>製品提供者固有のセキュリティ情報識別子</sec:identifier>
<sec:references>セキュリティ関連情報のURL</sec:references>
<sec:cvss score="全体評価値" severity="深刻度の指標"
          vector="各評価値" version="バージョン" />
<sec:cpe-item name="cpe名">
  <vname>製品提供者名</vname>
  <title>製品名</title>
</sec:cpe-item>
```

図 4：セキュリティ情報の属性記述を拡張するための仕様 mod_sec

(c) VULDEF(The VULnerability Data publication and Exchange Format data model)[10]

脆弱性対策情報を詳細に記述するための XML フォーマットであり，脆弱性対策情報のコンテンツ自身の利活用のためのフォーマットとして使用する(図 5).



クラス	説明
Title	脆弱性対策情報の題名
VulinfoDescription	脆弱性に関する情報(概要，詳細など)
Affected	脆弱性により影響を受けるバージョン，システムに関する情報
Impact	脆弱性により想定される影響
Solution	脆弱性を回避するための施策
Exploit	脆弱性の攻略に関する情報
Related	脆弱性ならびに対策に関連する情報
History	改訂履歴など
DateFirstPublished	対策情報の初版公開日
DateLastUpdated	対策情報の最新更新日
AdditionalData	備考用

図 5：VULDEF のデータモデル

3.3 フィルタリング型情報提供サービスの実現

本節では，JVN の脆弱性対策情報を「製品にどのような脆弱性が存在するのか」という視点から対策情報を選別するフィルタリング型情報提供サービスの実現方式について述べる.

3.3.1 システム構成

フィルタリング型情報提供サービスは，次の 3 つのコンポーネントから構成する(図 6).

(1) JVN データベース

JVN が保有する既存の脆弱性対策情報データベースであり，日本語／英語による，概要，CVSS による深刻度, 影響を受けるシステム, 想定される影響, 対策，ベンダ情報，参考情報が格納されている.

(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 名とをマッピングするデータベースである. CPE データベースを使用して，製品の CPE 名による識別を行った後，JVN データベースから該当する製品の脆弱性対策情報のみを取得する.

(3) Web API モジュール

JVN ならびに CPE データベースを利用して，「製品にどのような脆弱性が存在するのか」という視点から対策情報をフィルタリングする Web サービス API サービスを実現する.

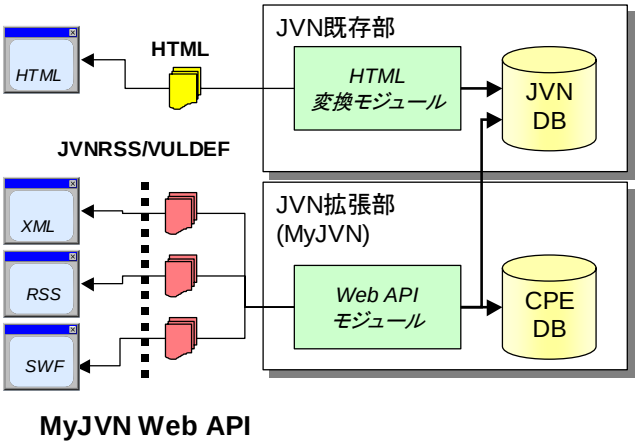


図 6：フィルタリング型情報提供サービスのシステム構成

3.3.2 MyJVN Web API

MyJVN API 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 Web サービス API である．本項では，Web API モジュールが提供する MyJVN Web API 仕様について概説する.

(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 Web 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，図 2に挙げた CPE 名によるフィルタリング条件などのパラメタが使用可能である．

表 3：MyJVN Web API のメソッド一覧

名称	概要
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． http://jvndb.jvn.jp/myjvn?method=getVendorList&cpeName=cpe:/*:j*&lang=en
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． http://jvndb.jvn.jp/myjvn?method=getProductList&cpeName=cpe:/*:jvn:*
脆弱性概要情報 一覧取得 getVulnOverviewList	フィルタリング条件に該当する脆弱性情報の概要一覧を RSS 1.0 + mod_sec 形式で取得する． http://jvndb.jvn.jp/myjvn?method=getVulnOverviewList&cpeName=cpe:/*:jvn:jvndb&rangeDatePublic=n&rangeDatePublished=w
脆弱性詳細情報 取得 getVulnDetailInfo	フィルタリング条件に該当する脆弱性詳細情報を VULDEF 形式で取得する． http://jvndb.jvn.jp/myjvn/MyJVN?method=getVulnDetailInfo&vulnId=JVNDB-2007-000001

(2) レスポンス

脆弱性概要情報一覧の出力には，図 7に示すように，RSS 1.0 に，RSS に不足しているセキュリティ情報の属性記述を mod_sec で補う形式を実装している．このように RSS をベースとすることにより，既存 RSS リーダを MyJVN Web API へのアクセスツールのひとつとして利用できる．

3.4 脆弱性対策情報チェックツール

本節では，フィルタリング型情報提供サービスの実現にあたり実装した脆弱性対策情報チェックツールについて述べる．脆弱性対策情報チェックツールは，MyJVN Web API を利用した JVN アクセスにより，定期的な脆弱性対策情報のチェック支援を目的とした Web ブラウザベースの GUI ツールである．また，脆弱性対策情報チェックツールは，フィルタリング条件設定，脆弱性対策概要情報表示，脆弱性対策詳細情報表，チェックリスト表示の 4 つの機能から構成している．

```
<?xml version="1.0" encoding="UTF-8" ?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns="http://purl.org/rss/1.0/"
        xmlns:dc="http://purl.org/dc/elements/1.1/"
        xmlns:dcterms="http://purl.org/dc/terms"
        xmlns:sec="http://jvn.jp/rss/mod_sec/">
 <channel rdf:about="http://jvndb.jvn.jp/myjvn">
  <title>JVN DB 脆弱性対策情報</title>
  <link>http://jvndb.jvn.jp/myjvn</link>
  <description>JVN DB 脆弱性対策情報</description>
  <items>
   <rdf:Seq>
   <rdf:li rdf:resource="http://jvndb.jvn.jp/apis/"/>
   </rdf:Seq>
  </items>
 </channel>
 <item rdf:about="http://jvndb.jvn.jp/apis/">
  <title>About JVN API</title>
  <link>http://jvndb.jvn.jp/apis/</link>
```

mod_sec

```
  <sec:identifier>JVNDB-2008-123456</sec:identifier>
  <sec:references>http://jvn.jp/</sec:references>
  <sec:cvss score="7.5" severity="High" vector="" version="2.0" />
  <sec:cpe-item name="cpe:/a:jvn:jvndb">
   <vname>JVN</vname>
   <title>Japan Vulnerability Notes</title>
  </sec:cpe-item>
```

```
  <dc:publisher>IPA</dc:publisher>
  <dcterms:issued>2008-10-10T10:10+00:00</dcterms:issued>
  <dcterms:modified>2008-10-10T10:10+00:00</dcterms:modified>
 </item>
</rdf:RDF>
```

図 7：脆弱性概要情報一覧のレスポンス例

(1) フィルタリング条件設定(図 8)

脆弱性対策情報のフィルタリング条件を設定する．

- 普及している製品等をフィルタリング条件とした既定設定と，ユーザが個別に設定するカスタム設定を選択可能
- カスタム設定においては，フィルタリング条件としてベンダ名，製品名，深刻度，脆弱性発見日，脆弱性更新日を設定／変更可能
- 起動時に保存したフィルタリング条件での脆弱性対策概要情報を表示

(2) 脆弱性対策概要情報表示(図 9)

ユーザの指定したフィルタリング条件で脆弱性対策概要情報を取得し，RSS+mod_sec で記載された XML 形式の結果を，ベンダ＞製品＞脆弱性番号ごとに階層化し，一覧表示する．

(3) 脆弱性対策詳細情報表示(図 10)

脆弱性対策概要情報の一覧から特定の脆弱性を選択すると，VULDEF で記載された XML 形式を当該脆弱性の脆弱性対策詳細情報として表示する．

(4) チェックリスト表示(図 11)

ユーザの指定したフィルタリング条件で脆弱性対策概要情報を取得し，RSS+mod_sec で記載された XML 形式の結果を，脆弱性番号，深刻度，脆弱性発見日，脆弱性更新日で並び替え可能なチェックリストに整形表示する．印刷して利用することも可能である．

図 8：getProductList で取得した
製品情報によるフィルタリング条件の設定

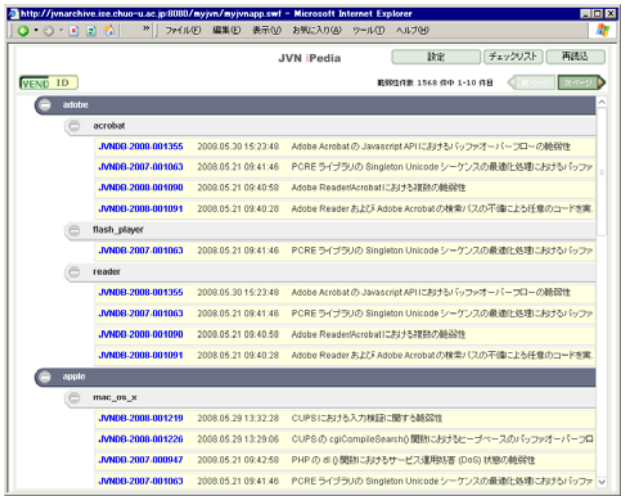


図 9：getVulnOverviewList で取得した
脆弱性概要情報の表示

図 10：getVulnDetailInfo で取得した
脆弱性詳細情報の表示

図 11：getVulnOverviewList で取得した
脆弱性概要情報のチェックリスト表示

4 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 を提案した．また，プラットフォーム識別子，Web サービス API，XML フォーマットの 3 つの整備と，JVN の脆弱性対策情報を用いたサービスとして，製品視点から対策情報を選別するフィルタリング型情報提供サービスの実装について報告した．今後も，他の脆弱性対策や機械化処理の取り組みを踏まえ，脆弱性有無を機械的にチェックするような更なる機械化処理に向けた検討を進めていく予定である．

参考文献

1) NIST: The Information Security Automation Program and The Security Content Automation Protocol, http://nvd.nist.gov/scap.cfm
2) WARP: Filtered Warnings Service, http://www.warp.gov.uk/FilteredWarnings.htm
3) ENISA: Final Report on the Feasibility Study on EISAS, http://www.enisa.europa.eu/doc/pdf/studies/EISAS_finalreport.pdf
4) OSVDB: Database Exports, http://osvdb.org/database_info
5) OSVDB: About the OSVDB API, http://osvdb.org/api/about
6) NVD/CVE XML Schema, http://nvd.nist.gov/schema/nvdcve.xsd
7) Common Weakness Enumeration, http://cwe.mitre.org/
8) JVNRSS: JVN RDF Site Summary, http://jvnrss.ise.chuo-u.ac.jp/jtg/jvnrss/
9) mod_sec: Qualified Security Advisory Reference, http://jvnrss.ise.chuo-u.ac.jp/jtg/mod_sec/
10) VULDEF: The VULnerability Data publication and Exchange Format data model, http://jvnrss.ise.chuo-u.ac.jp/jtg/vuldef/