CYBEXで標準化された サイバーセキュリティ情報の活用法

奈良先端科学技術大学院大学　情報科学研究科　准教授　門林（かどばやし）　雄基（ゆうき）

1．はじめに

近年、電子メールやウェブ、スマートフォンなどを介した問題はますます多様化しており、なくなる気配はない。これは膨大なヒトと情報通信機器の相互作用によって生じるサイバースペースにおいて、諸問題が様々な形で発現したもので、価値観の衝突、経済格差、制度的格差、教育格差、誤操作、誤認、バグ、誤動作、設定ミスなど、さまざまな次元の問題をはらんでいる。ITU-T参加国の中にもこれらを混同し、抜本的解決策を技術的な国際標準化に求めようとする意見も少なくない。その一方で、より正確にこの問題を取り扱うために、デジタルディバイド、サイバー法などの議論が進められるとともに、サイバースペースにおける安全性を取り扱うサイバーセキュリティが大きな課題として認識されることとなった。

このような状況の下、ITU-Tスタディグループ17課題4（以下、課題4）ではサイバーセキュリティに関する国際標準化の方向性として、技術的な防護装置の実現方式ではなく、サイバーセキュリティ向上のための情報交換技法に的を絞って標準化を進めてきた。その理由は以下のとおりである。今日のサイバースペースの屋台骨を構成するインターネットは生来的にグローバルな空間であり、問題の根源を完全に排除することは不可能である。またこのため、信頼できない第三者とのインタラクションを前提としてシステムを構成する必要がある。継続的な脅威を前提として、情報交換によってリスクの発生源と対峙していく必要がある。

課題4では手始めに、安全上問題となるバグ、すなわち脆弱性情報を取り扱うための国際標準化を推進した。脆弱性情報は、セキュリティ対策、高信頼ソフトウェア開発、及びシステムの安全性評価などで活用することができる。

以下ではまず、脆弱性情報のための標準について、その概要を説明する。次にシステム管理等における活用法、そしてソフトウェア開発、安全性評価における活用法について説明する。

なお本稿では、サイバーセキュリティの制度面での課題である制度的格差や教育格差、並びにヒューマンファクターの課題である誤操作、誤認については以降では言及しない。これらは重要な課題であるが、これまで課題4の標準化活動の範疇外であったためである。

2．脆弱性情報のための標準

CVE

今日、利用者が用いるソフトウェア数は膨大であり、また一つのソフトウェアに対して複数の脆弱性が発見されることがある。さらに一つのソフトウェアが複数のライブラリを用いており、そのライブラリに脆弱性が発見されることもある。近年、デバッグ技術及びバグ発見手法について飛躍的な進展があり、脆弱性発見のペースが格段に上がっている。

それぞれの脆弱性に一意な番号をつけることにより、異なるセキュリティ製品を用いている企業であっても共通の脆弱性について議論することができる。これが共通脆弱性識別子CVEである。

脆弱性が発見され報告された時、命名や分類をしている時間はないので、通し番号により採番する。脆弱性の番号としてはCVE-2012-nnnnのように年単位で採番する。近年では、年間数千件の脆弱性が報告されている。

脆弱性には大きく分けて2種類ある。まず、ネットワーク経由でその脆弱性が突かれてしまう脆弱性である。代表的なものとして、リモートコード実行の脆弱性が挙げられる。この種の脆弱性が発見されたソフトウェアをそのまま使い続けていると、ネットワーク経由でユーザ権限を奪取される危険性があるため、ただちに修正しなければならない。このほか、インジェクションの脆弱性にも情報漏洩や権限奪取につながる危険性があるため、早急な対処が必要である。

ネットワークを経由しない脆弱性というものもあり、その代表的なものとしては権限昇格の脆弱性が挙げられる。これは脆弱性が突かれることにより、一般利用者の権限しか持っていないユーザがシステム管理者の権限でしかできない操作を実行できてしまう、といった危険性をはらんでいる。

一般に、CVEデータベースでは脆弱性の深刻度を表記しているが、具体的にどのような操作をすれば問題が発生するか、といったことは不正アクセス抑止の観点から詳細に記していない。バグが発現するための条件は一般論としてかなり複雑であり、相当込み入った入力をしなければ脆弱性が発現

表１．近年問題となっている脆弱性の例

CVE番号	脆弱性の概要	対策手法の例
CVE-2011-0609	Flashにおけるリモートコード実行の脆弱性	OSにおけるデータ実行防止機能（DEP）の利用
CVE-2011-2462	Acrobatにおけるリモートコード実行の脆弱性	AcrobatにてJavaScriptの無効化
CVE-2013-0422	Java 7におけるリモートコード実行の脆弱性	ブラウザにてJavaプラグインの無効化

することはない。このため数年前までは、脆弱性は一部の専門家に委ねられていたトピックであったが、近年ツールによる自動化が進んでおり、脆弱性を放置するリスクが相対的に高まっている。

表1に、近年問題となっているサイバー攻撃において用いられた脆弱性を例示する。

CPE

ソフトウェア資産については現在、米国NISTが共通プラットフォーム一覧CPEを国内標準とし、体系的に命名を行っている。CPEではURI表記でソフトウェア資産を表記する。例えばMicrosoft Windows 2003をcpe:/o:microsoft:windows_2003と表記する。このように、市販のソフトウェア製品に対するCPE表記があらかじめ定められているので、ソフトウェア資産を一意に識別することができる。

このようなソフトウェア資産に対する標準的な表記法は膨大な脆弱性情報を膨大なソフトウェア資産に対して照合していくときに必要不可欠である。仮にこのような表記がなかった場合、Windows 2003、Win 2003、Win2k3といった表記のゆれが生じてしまい、機械的な照合が難しくなる。

このため米国の国家脆弱性データベースNVDでは脆弱性を有するソフトウェアをCPE表記で記述している。企業や政府機関におけるソフトウェア資産がCPE表記で管理されていれば、資産の台帳と脆弱性情報を照合し対策導出につなげることが容易となる。

一般的にはエンドポイントプロテクション製品を導入することで、各端末に導入されたソフトウェア資産の洗い出しと脆弱性データベースとの照合を自動化することができる。エンドポイントプロテクション製品は複数のセキュリティ対策製品ベンダから市販されており、CVE互換製品の一覧（http://cve.mitre.org/compatible/compatible.html）から対応製品・サービスを知ることができる。

CVSS

各企業が保有、又は利用するソフトウェア資産を把握しており、それらに対する脆弱性情報を入手した場合、個々の脆弱性の影響度・深刻度を計算することができる。脆弱性情報は、当該脆弱性への攻撃方法の有無など、日々変わる状況により対策の優先度を変えなければならない。また該当するソフトウェア資産の有無によっても対策の優先度を変える必要がある。CVSSによって、このような影響度や深刻度をスコアとして計算することができ、個々の脆弱性のスコアに基づいて対策の優先度を決めることができる。

まずCVEデータベースには計算の出発点となる基本スコア、影響度サブスコア、深刻度サブスコアが記載されている。これは各社の事情や攻撃の有無などを一切考慮しないものである。影響度サブスコアは、当該脆弱性の秘匿性への影響、及び完全性、可用性への影響を評価し算出される。深刻度サブスコアは、脆弱性発現の容易さ（例えばネットワーク経由で発現するか否か等）を評価し算出される。これらをまとめたものが基本スコアとなる。

次にセキュリティ製品ベンダやサービス事業者、JPCERT/CCなどの調整機関からの情報を基に現状値を計算する。これらの事業者や調整機関から、当該脆弱性について解決策・回避策の有無、攻撃手法の完成度、脆弱性の信憑性などの情報を入手し、現状値が計算される。

最後に、当該脆弱性が企業に与える損害、脆弱性を有する資産の割合、秘匿性要件、完全性要件、可用性要件を考慮し、各企業に合わせた環境値が計算される。

表2に、表1で示した脆弱性に対するCVSSの基本スコアを示す。

表２．近年問題となっている脆弱性のCVSS基本スコア

CVE番号	基本スコア	影響度サブスコア	深刻度サブスコア
CVE-2011-0609	9.3	10.0	8.6
CVE-2011-2462	10.0	10.0	10.0
CVE-2013-0422	10.0	10.0	10.0

CWE

CVEが脆弱性に通し番号をつけてデータベース化したものであったのに対し、CWEは脆弱性の分類辞書である。脆弱性は年間数千件というペースで報告されるが、脆弱性の発生原因、つまり安全上問題となるバグは類型化できることから、それほど件数が増加しない。CWEでは脆弱性のインパクト、脆弱性の発見方法、脆弱性につながったプログラムの例、CVEデータベース中に見られる発現例、回避方法、参考情報へのリンクなどがデータベース化されている。CWEはCigital、Coverity、Fortify、Microsoft、Whitehat Securityなどのセキュリティ検査企業とCERT/CC、MITRE、NIST、OWASP、WASCなどの非営利機関や団体が、それぞれの知見を持ち寄って出来上がったものであり、ソフトウェア開発者向けの分類、脆弱性研究者向けの分類など複数の基準で分類がなされている。

CVEは市販ソフトウェア製品向けの脆弱性データベースであるのに対し、CWEは独自開発ソフトウェア向けの脆弱性データベースであるととらえることもできる。例えば電子商取引のためにショッピングカートを自社開発した場合、仮に脆弱性があったとしても市販ソフトウェアではないためCVEデータベースには掲載されない。このような場合CWE番号に基づいて問題を管理し、安全性向上につとめることができる。

表3に、ソフトウェア開発において問題となるCWEの上位10件を示す。

表３．ソフトウェア開発において問題となるCWEトップ10（2011年版）

CWE ID	名称
CWE-89	SQLインジェクション
CWE-78	OSコマンド・インジェクション
CWE-120	バッファ・オーバーフロー
CWE-79	クロスサイト・スクリプティング
CWE-306	重要機能における認証の欠如
CWE-862	権限管理の欠如
CWE-798	認証子の埋め込み
CWE-311	重要情報における暗号化の欠如
CWE-434	危険な型のファイルのアップロード
CWE-807	信頼できない入力に依存したセキュリティ判定

CAPEC

CAPECは攻撃パターンのデータベースである。CWEが脆弱性の類型であったのに対し、CAPECは脆弱性が発現するパターンを類型化したものであると言える。CAPECにより、様々な攻撃パターンを想定してシステムの設計や開発、導入を行うことができ、セキュリティ対策の網羅性を向上させることができる。

CAPECでは攻撃の流れ、攻撃の発見方法、攻撃の帰結、予防方法、深刻度、影響度、CVEの例、関連するCWEなどがデータベース化されている。なおCAPECは本稿執筆時点ではITU-T勧告案X.1544であり、問題なく承認されれば今年4月に勧告となる見込みである。

表4に脆弱性情報のための勧告を示す。CVE、CPE、CWE、CAPECは元はと言えばMITREが米連邦政府向けに作っていた標準であり、ITU-T勧告は互換性要件である。例えばX.1520はCVEへの互換性要件であり、日本の脆弱性データベースJVNはCVE互換であるのでX.1520に準拠していることになる。CVSSはインシデント対応チームの国際的団体FIRSTが策定し、後にITU-T勧告X.1521として認めたものである。

3．システム管理における活用

システム管理においてセキュリティを考えるとき、機器管理、ソフトウェア資産管理、ユーザ管理、鍵管理、アクセス制御と並んで脆弱性管理は重要なタスクである。いくら強固な暗号を採用していても、高度な認証システムを採用していても、システムに深刻な脆弱性があれば情報漏洩は免れない。

脆弱性情報を基に定期的に脆弱性管理を行い、リスクの顕在化を防ぐ視点が重要である。また針の穴を通すような小さな脆弱性であっても、いったん悪用されると連鎖によって拡大する可能性がある。このため、ネットワーク経由で発現する深刻な脆弱性でなかったとしても、脆弱性管理の対象とすることが求められる。

脆弱性検査製品やエンドポイントプロテクション製品を用いている場合、間接的にCVE、CPE等を活用していることになる（本原稿を依頼された趣旨は「標準の活用」であったが、本来良い標準とは意識しないうちに使っているものだと思う）。またCVEを定期的にチェックし、最新の脆弱性情報に注意を払いつつCVSSを参考に対策の優先度を決定するという活用方法もある。

表４．脆弱性情報のための勧告（括弧書きは勧告案）

ITU-T勧告	標題	互換性
X.1520	Common vulnerabilities and exposures	CVE
X.1521	Common vulnerability scoring system	CVSS
X.1524	Common weakness enumeration	CWE
X.1528	Common platform enumeration	CPE
（X.1544）	Common attack pattern enumeration and classification	CAPEC

４．ソフトウェア開発における活用

ソフトウェア開発の現場ではCWEを活用し、セキュアソフトウェア開発につなげることができる。近年では数百種あるCWEエントリの中から、CWEトップ25が選ばれ公表されている。これはソフトウェア安全性検査に携わる20以上の専門機関からのフィードバックに基づくもので、ソフトウェア開発においてセキュリティを確保する上で注意すべき重要25項目であると言える。その詳細についてはhttp://cwe.mitre.org/top25/を参照されたい。

このほか、開発中のソフトウェアと類似の機能を持つ市販ソフトウェア名で検索するなどして、失敗データベースとして脆弱性データベースを活用することもできる。また開発予定のソフトウェアで使用するライブラリやフレームワークを選定する際のデータベースとしても脆弱性データベースを活用できる。過去に数多くの脆弱性が発見されたソフトウェアは、一般的な傾向として、今後も脆弱性が発見される可能性が高い。

５．安全性評価における活用

システムの安全性を評価するとき、多岐にわたる攻撃シナリオを想定する必要が生じる。このときCAPECによって想定する攻撃パターンの網羅性を評価できる。またCAPECの個々の攻撃パターンに与えられた名称及び攻撃パターン番号を用いることで、攻撃の発見方法、予防方法、想定するバグなどに関するデータベースの記載を再利用することができる。

６．おわりに

人間のやることにはミスがつきものである。設計ミス、プログラムのミス、設定ミス、操作ミス、伝達ミスなどあらゆる間違いがサイバースペースにおいても日々起きているし、これらのミスがなくなることはないであろう。

秀才の発想では、セキュリティ事故が起きないシステムを作ればよいという理屈になる。しかしながらコンピュータが人間の言うことに従う限り、膨大なヒトと情報通信機器の相互作用によって生じるサイバースペースにおいて問題が根絶されることはないであろう。

筆者はリスクから目を背けるのではなく、リスクを正しく理解し、細かな問題をつぶさに観察することがサイバーセキュリティ向上への近道であると考える。CYBEXで標準化されたサイバーセキュリティ情報がその一助になれば幸いである。

参考文献

1. Robert A. Martin, Being Explicit About Security Weaknesses, In CrossTalk, March 2007.
2. Robert A. Martin, Managing Vulnerabilities in Networked Systems, IEEE Computer, November 2001.