グローバル PSIRT を支える 脆弱性管理システムの構築

（株）デンソー

■ 執筆者Ｐｒｏｆｉｌｅ ■

長井　健一

2016 年　（株）デンソー入社
安全環境・品質本部
情報セキュリティ推進室
2018 年　現在 コーポレート基盤本部
情報セキュリティ推進室

■ 論文要旨 ■

IoT（Internet of Things）時代が到来しつつある昨今、自動車業界においてもコネクテッドカーへのシフトが加速している。一方、ネットワークに接続することによって、サイバー攻撃などのセキュリティリスクに晒される。そのため、製品のセキュリティ対策強化が必要であり、当社においても PSIRT（Product Security Incident Response Team）を設置し、活動を開始した。

PSIRT の運用実態を踏まえ、製品に内在する脆弱性を管理するためのシステムが必要であるという仮説を立てた。そして、構想に基づくシステムを開発した。開発工程では、Web ユーザーインターフェースや文字列検索処理を工夫し、操作性の向上と処理の高速化を図った。その結果、業務効率を約 30 倍に高めることができた。本稿では、仮説立案から評価までを論じるとともに、今後の展望についても考察する。

■ 論文目次 ■

■ 図表一覧 ■

１．はじめに

１．１　当社の概要

株式会社デンソー（以下、当社という。）は、愛知県刈谷市に本社をおく自動車部品メーカーであり、欧州・米国をはじめとする世界中の自動車メーカーとビジネスを行っている。主要分野は、熱機器（カーエアコン、ラジエータなど）、パワトレイン機器（スタータ、コモンレールシステムなど）、情報安全（ミリ波レーダー、ヘッドアップディスプレイなど）、電子機器（エンジンコントロールユニット、半導体センサなど）及びモータ（ワイパ、パワーステアリングなど）に大別される。製品開発拠点は日本国内のほか、欧州や北米などにも拠点を有している。生産拠点や販売会社を含めた全世界の連結会社子会社数は、2018 年３月末時点で 221 社となっている。

１．２　製品セキュリティの強化

近年、自動車のコネクテッド、自動運転、シェアリング、電動化が加速しており、自動車業界は 100 年に一度の変革期といわれている。当社においても、情報安全分野は今後更に注力していく分野と位置づけている。一方、自動車がネットワークに接続されることによって、セキュリティリスクが増大する。例えば、2015 年、Wi-Fi 経由で自動車を乗っ取り、遠隔操作できることが米国のセキュリティカンファレンスで発表された[1]。この発表を受け、米国の自動車メーカーが対策のため約 140 万台をリコールしたと報道されている[2]。

こうした情勢を鑑み、当社では従来から取り組んできたコーポレートのセキュリティ対策に加え、製品や工場などの「つながる社会」へのセキュリティ対策を強化するため、2016 年６月に全社 CISO を設置した。同時に、CISO の配下に「情報セキュリティ推進室」を設置し、製品や工場のセキュリティ対策を強化するための全社的な取組を開始した。

１．３　PSIRT の概要

PSIRT（Product Security Incident Response Team）は、製品の脆弱性に起因するリスクに対応するための組織である[3]。「IoT セキュリティガイドライン[4]」や「IoT セキュリティ総合対策[5]」などに出荷後の脆弱性対応や相談窓口の設置が謳われており、国内企業にお

---

[1] Charlie Miller & Chris Valasek, “Remote Exploitation of an Unaltered Passenger Vehicle”, Black Hat USA 2015
https://www.blackhat.com/us-15/briefings.html#remote-exploitation-of-an-unaltered-passenger-vehicle

[2] Chrysler Recalls 1.4 Million Vehicles After Jeep Hacking Demo
http://www.darkreading.com/vulnerabilities---threats/chrysler-recalls-14-million-vehicles-after-jeep-hacking-demo-/d/d-id/1321463

[3] JPCERT コーディネーションセンター, “FIRST PSIRT Services Framework 1.0（日本語抄訳版）”
https://www.jpcert.or.jp/research/psirtSF.html

[4] IoT 推進コンソーシアム, “IoT セキュリティガイドライン ver 1.0”
http://www.soumu.go.jp/main_content/000428393.pdf

[5] 総務省, “IoT セキュリティ総合対策”
http://www.soumu.go.jp/main_content/000510701.pdf

いても徐々に PSIRT の設置が進んでいる。当社も出荷後の脆弱性対応や、万一、製品に関わるセキュリティインシデントが発生した場合に備え、一部の部門を対象に、2016 年度から PSIRT の活動を開始した。本稿では、PSIRT が製品に内在する脆弱性を日々モニタリングし、管理するための仕組みについて論じていく。

２．課題と仮説

２．１　従来の運用方法と課題

当社 PSIRT は、脆弱性データベースである National Vulnerability Database[6]（以下、NVD という。）や JVN iPedia[7]から脆弱性情報を収集し、当社製品に脆弱なコンポーネントが含まれていないかどうかを検査することを基本的なアプローチとして活動を開始した。具体的には、製品に含まれるソフトウェアコンポーネント（以下、構成情報という。）を予め抽出した上、NVD や JVN iPedia の脆弱性情報に記載されているソフトウェア名やバージョン情報で照合し、当該コンポーネントが脆弱性の影響を受けうるかどうかを検査する。そして、脆弱性の評価基準である CVSS[8]値などを参考に、もし製品が一定の影響を受ける可能性があると判断された場合は、製品開発部門が詳細な影響調査を行うというものである。NVD だけでなく JVN iPedia も併用する理由は、NVD あるいは JVN iPedia にしか登録されていない脆弱性情報が存在し、かつ、当社 PSIRT は世界各国で報告された脆弱性に対処する必要があるためである。なお、将来的には、NVD や JVN iPedia で公表されている公知の脆弱性以外にも、顧客である自動車メーカーや、セキュリティ情報を共有する機関である ISAC（Information Sharing and Analysis Center）、セキュリティ企業や研究者といった外部組織から提供される脆弱性に対しても、迅速に対処する必要があると考えている。

2016 年度から開始した初期段階の運用では、Excel マクロを活用して、NVD 及び JVN iPedia からの脆弱性情報を収集、構成情報との月次照合を行っていた。しかしながら、運用上の課題をいくつも抱えていた。例えば、マクロ処理の実行に時間を要し、しばしばフリーズが発生した。また、マクロの単純なロジックでは精度の高い照合が難しいため、処理結果に対して人間系で実施する判別作業（以下、目検仕分けという。）の負荷が非常に大きかった。そこで、マクロの実行や目検仕分けは外部委託も活用したものの、前月に公開された NVD 及び JVN iPedia の脆弱性情報と構成情報との照合を完了するまでに、２週間程度のリードタイムが発生した。そのほか、脆弱性情報の軽微な更新に伴い、前月以前に照合された脆弱性が再出力されるという管理上の問題も抱えていた。

中長期的な課題として、初期段階の運用では２製品を対象としていたけれども、今後、製品のコネクテッドカー対応などに伴い、対象製品の数は右肩上がりに増加していくと予測されていた。加えて、公表される脆弱性の数も年々増加傾向であることを考慮すれば、恒久的にマクロで運用していくことは難しかった。

---

[6] NIST, “National Vulnerability Database” https://nvd.nist.gov/

[7] 独立行政法人情報処理推進機構, “JVN iPedia” https://jvndb.jvn.jp/

[8] 独立行政法人情報処理推進機構, “共通脆弱性評価システム CVSS 概説” https://www.ipa.go.jp/security/vuln/CVSS.html

こうした状況を踏まえ、脆弱性情報の収集と照合を自動化し、製品の脆弱性を日々モニタリングするためのシステムが必要であるという仮説を立てた。なぜなら、このようなシステムが構築できれば、目検仕分けの負担を大きく軽減させ、対応状況の管理に人的リソースを集中することができる。加えて、今後業界全体のセキュリティ意識が高まれば、脆弱性が公表されてから影響調査に着手するまでのリードタイム短縮は必須であると考えたからである。

２．２　システム構想

前節の運用内容を踏まえ、システムに要求される機能を具現化した。いずれの機能も、脆弱性対応の基本的なアプローチは変えずに、省力化やリードタイムの短縮、照合精度や管理精度、作業性の向上といった観点から必要であると判断したものである。

(1) 脆弱性情報の収集・管理

脆弱性データベースである NVD 及び JVN iPedia から、１日１回以上の頻度で脆弱性情報を自動取得する。その際、NVD と JVN iPedia の脆弱性情報を CVE[9]番号で照合し、CVE 番号が合致したものは同一脆弱性情報として扱う。既に取得した脆弱性情報に更新があった場合は、更新内容を反映し、最新化する。

また、PSIRT の担当者がソフトウェア名や CVE 番号などから脆弱性情報を検索することができ、必要に応じて補足情報を追記することや、NVD と JVN iPedia に登録されていない脆弱性情報を新規に登録することができる。この機能は、将来的に PSIRT が外部組織から脆弱性情報を受け付ける際に利用することを想定している。

(2) 構成情報の管理

製品で利用しているオープンソースソフトウェア（以下、OSS という。）や商用ソフトウェアの名称及びバージョン情報を構成情報として管理することができる。また、PSIRT の担当者や製品開発者がシステム上で構成情報を編集することができ、CSV などのファイル形式でインポート及びエクスポートができる。更に、製品のソースコードをスキャンし、自動的に構成情報を作成することもできれば望ましい。

(3) 脆弱性情報と構成情報の照合

脆弱性情報と構成情報をソフトウェア名やバージョン情報で照合し、合致した場合は対応状況をチケットとして管理することができる。照合処理は、脆弱性情報や構成情報の新規登録または更新をトリガーとして、自動的に実行される。また、チケットが新規に発行された際、メールなどで PSIRT の担当者や製品開発者に通知することができる。

(4) 対応状況の管理

前項で発行されたチケットを利用し、各製品が有する脆弱性の対応状況を管理することができる。チケットの入力項目としては、対応予定日、対応実施日、対応内容の詳細など

---

[9] 独立行政法人情報処理推進機構, “共通脆弱性識別子 CVE 概説” https://www.ipa.go.jp/security/vuln/CVE.html

が必須である。そのほか、対応が滞っている場合（現在日と対応予定日に一定の乖離がある場合）には、自動的にメールなどでフォローを促すことが望ましい。

(5) そのほかの機能

利便性や操作性の観点から、Web ユーザーインターフェースであり、ユーザーの所属やロールに応じて構成情報や対応状況に関するアクセス制御を行うことができる。また、主要な製品開発拠点は日本国内であるものの、海外拠点でシステムを利用することも想定されるため、日本語と英語の２ヵ国語に対応していることが望ましい。

２．３　市販品の調査

システム開発において、一般的には市販品を活用した方が開発期間を短縮でき、開発コストも抑えることができる。脆弱性情報を自動的に収集し、構成情報と照合する市販品もいくつか存在したため、市販品に対する評価を行った。評価は次の３製品を対象とし、前節の機能を満足できるものであるかどうかを机上評価した。一つ目の製品は、OSS 利活用のためのリスク管理ツール（以下、OSS リスク管理ツールという。）であり、代表的な製品として Black Duck Hub[10]や FlexNet Code Insight[11]（旧製品名は Palamida）などがある。今回は、知名度の高い Black Duck Hub を評価対象として選定した。二つ目の製品は、オープンソースの脆弱性スキャンツール Vuls[12]である。三つ目の製品は、NTT テクノクロス社の提供する、脆弱性対処支援サービス TrustShelter/VA[13]である。

表１　市販品の比較表

製品名		スクラッチ開発	Black Duck Hub	Vuls	TrustShelter/VA
機能	(1) 脆弱性情報の収集・管理	◎ 当社要件に基づき開発可能 （脆弱性の追加・編集も可能）	○ NVDより脆弱性情報を取得 オプションでVuln DBを利用可	△ NVD、JVN等から取得 （ただし、Linux OS等に限定）	△ NVD、JVN等から取得 （ただし、サーバOS等に限定）
	(2) 構成情報の取得・管理	◎ ※抽出機能は他の商用ツールとの併用を検討する必要あり	◎ ソースコードより自動抽出	◎ サーバ機器より自動収集	◎ サーバ機器より自動収集
	(3) 脆弱性情報と構成情報の照合	◎ 当社要件に基づき開発可能	◎ CPEによる照合と同等 （照合ロジックは不明）	◎ CPEによる照合	○ 名称、バージョンによる照合
	(4) 対応状況の管理	◎ 当社要件に基づき開発可能	○ ステータス管理可能 （メモの入力は不可）	△ ステータス管理は未実装 他のツールとの連携が必要	◎ ステータス管理可能 （メモの入力も可能）
	(5) その他	◎ 日本語/英語に対応	○ 英語のみ対応	◎ 日本語/英語に対応	○ 日本語のみ対応

※ 本表は2017年3月末時点の内容であり、当社が独自に評価したもの。

表１は３製品を２．２節の機能に照らして評価したものである。結論として、各機能を

[10] Black Duck Software, Inc., “Black Duck Hub” https://www.blackducksoftware.com/ja/products/hub

[11] Flexera Software Inc., “FlexNet Code Insight” https://www.flexera.jp/producer/products/software-composition/flexnet-code-insight/

[12] フューチャーアーキテクト株式会社, “vuls” https://github.com/future-architect/vuls

[13] NTT テクノクロス株式会社, “TrustShelter/VA” https://www.trustshelter.jp/va/

有する製品は存在するものの、すべての機能を満足する製品は存在しなかった。特に、「脆弱性情報の収集・管理」機能については、NVD と JVN iPedia 両方の脆弱性データベースから脆弱性情報を取得し、かつサーバ OS 以外の組込ソフトウェアなどの脆弱性も収集対象とする製品は存在しなかった。

PSIRT では、外部組織から報告された脆弱性や、自社の脆弱性診断によって検出された脆弱性について、各製品開発部門に水平展開し、ほかの製品にも影響しないかどうかを調査する必要がある。そのため、脆弱性情報を柔軟に追加・編集できる機能は極めて重要である。しかしながら、このような機能を有する市販品は存在しなかった。加えて、対応状況や対策内容の管理機能も十分ではないため、既存の市販品では当社要求を満足できないと考えた。

市販品を活用せず、スクラッチ開発とした場合、ソースコードから構成情報を自動的に取得する機能を実装するためには多大な開発工数が必要となる。ところが、自動車業界の裾野は非常に広く、すべてのサプライヤからソースコードを提示させ、当社がすべてのソースコードを管理することは事実上困難である。加えて、当社では OSS のライセンス管理を目的として、従来から OSS リスク管理ツールを活用していた。このツールを活用すれば一定の精度で構成情報をエクスポートすることができる。そのため、ソースコードから構成情報を自動抽出する機能は必須ではなく、OSS リスク管理ツールからエクスポートされたファイルをインポートするかたちでも問題ないと考えた。

上記の理由から、OSS リスク管理ツールと疎結合に連携させることによって、開発工数を抑制しつつ、当社要求に即したシステムをスクラッチ開発することとした。

３．開発したシステム

３．１　開発手法及び IT インフラ

開発した脆弱性管理システム（以下、本システムという。）は、機能要件が明確であったことや、開発工程の節目で関係者へのレビューが必要であったことから、開発手法には伝統的なウォーターフォールモデルを採用した。ただし、バッチ処理の実行結果や Web ユーザーインターフェースのレイアウトについては綿密な摺り合わせが必要であったため、プロトタイプによる評価を行った。更に、Web ユーザーインターフェースについては、ほかの社内システムと項目ラベルなどの整合性を確保した上、ユーザーが直感的に操作できるよう３．３ (1)で後述する「インタラクションデザイン」も採用した。

IT インフラには、クラウドサービス（IaaS）を採用している。その理由は、情報セキュリティ推進室は戦略の立案・推進及び統括・監査が主たる役割であり、システム運用の実務部門ではないこと。社内計画として、2018 年度からの運用開始を目指しており、与えられた開発期間が約１年間と短かったこと。更に、将来的に国内外のグループ会社にも展開していくことが予定されていたためである。

なお、クラウドサービスを利用した場合、オンプレミスの社内システムと比較して、外部からパスワードリスト攻撃などの被害に遭う可能性がある。そのため、本システムでは、Web アプリケーションファイアウォールなどのクラウド標準のセキュリティ対策に加え、当社開発拠点以外からのアクセスを拒否するよう、クラウド側でのアクセス制御も施している。

３．２　各機能の実装と工夫した点

２．２節の構想に基づきシステム開発を行った。システムの利用イメージは図１のとおりであり、本節では各機能の実装結果や工夫した点などについて述べる。

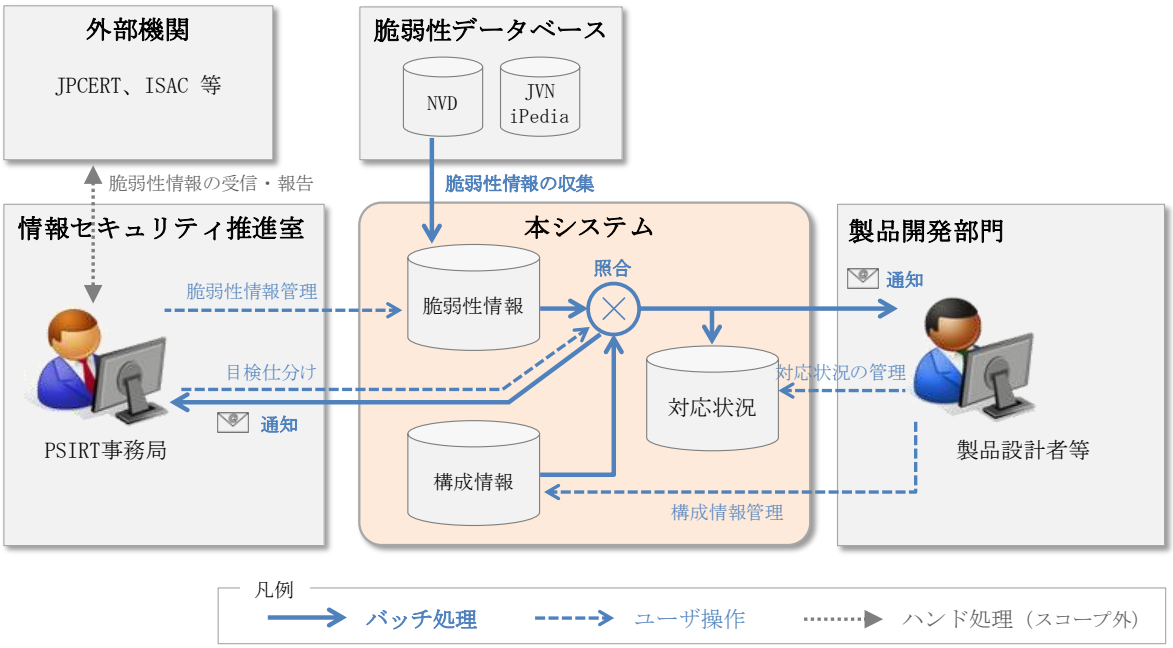

図１　システムの利用イメージ

(1) 脆弱性情報の収集・管理

脆弱性情報の収集は、１日２回、バッチ処理で実施する。日本時間（JST）及び北米時間（EST）の午前中に１回、午後に１回、自動収集される仕組みである。１日に何度もメールが通知されると煩わしさを感じるため、１日２回程度が妥当であると判断した。同様の理由から、製品ごとに照合結果を集計した上、PSIRT の事務局や製品設計者へサマリ情報を通知する仕組みとしている。

(2) 構成情報の取得・管理

構成情報は、ハンド入力のほか、OSS リスク管理ツールから出力したファイルをインポートできる機能も実装した。また、ハンド入力の場合は入力ミスが想定されるため、NVD 及び JVN iPedia に登録されている、最新の CPE[14]リストから選択できるよう工夫している（図２）。

そのほか、納品先によって製品バージョンが異なる場合も想定されることから、同一製品における世代管理機能も実装した。

[14] 独立行政法人情報処理推進機構，“共通プラットフォーム一覧 CPE 概説” https://www.ipa.go.jp/security/vuln/CPE.html

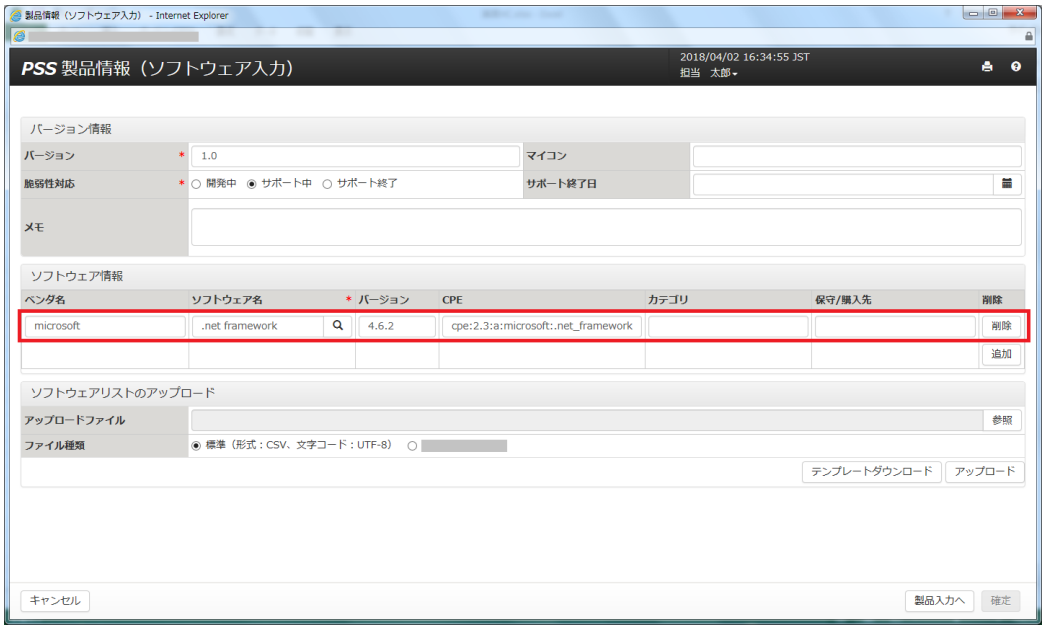

図２　構成情報の入力画面

(3) 脆弱性情報と構成情報の照合

脆弱性情報と構成情報の照合は、脆弱性情報に追加・更新があった場合、または構成情報に変更があった場合に照合が行われるよう実装した。このとき、既に対応状況が管理されている脆弱性については、深刻度（CVSS）や脆弱性の影響を受けうるソフトウェアに変更があった場合にのみ対応状況を更新し、不必要な通知メールが送信されないようにしている。

照合精度を高めるための工夫として、脆弱性情報と構成情報がともに CPE で記載されている場合は、CPE で照合するよう実装した。もし、脆弱性情報または構成情報のいずれかに CPE の記載がない場合は、ベンダー名、ソフトウェア名及びバージョン情報の組合せで照合する。このとき、３．３ (2)で後述する全文検索エンジンを用いて、意味のある単語に分解してから照合することによって、従来の Excel マクロによる照合から大幅に精度を向上させることができた。ただし、それでも文字列が部分的に一致した場合など、一部の照合結果については人間系での目検仕分けが必要となる。そこで、文字列全体に対する一致率を算出するようにした。この一致率を用いて、概ね全体が一致するものや一致率が極めて低いものはシステムで自動処理し、それ以外の曖昧なものだけを目検仕分けの対象とすることによって担当者の負担を軽減している。

(4) 対応状況の管理

脆弱性対応のステータス管理ができるほか、対応予定日を超過した場合、製品設計担当者に対してメールで１日１回自動通知する仕組みとした。そのほか、対策実施完了または影響なしと判断してクローズする際には、役職者の承認を得てからクローズする承認機能も実装した。

(5) そのほかの機能

ユーザーの管理やロール設定を行うユーザー管理機能、利用グループ会社や利用部署を管理するためのマスタメンテナンス機能も実装した。全社のクラウドサービス統合認証基盤が構築未完了であったことから、管理者によるパスワードリセット機能などもユーザー管理機能に実装した。

言語設定やタイムゾーンはメニューバーから切替が可能であり、出向者にも配慮し、独立して設定できるようにしている。

３．３　採用技術

(1) インタラクションデザイン

インタラクションデザインは、ユーザーとシステムとの対話（インタラクション）に注目し、利用目的に合致した画面遷移や、Web ユーザーインターフェースの自然な振る舞いをデザインするプロセスである。今回のシステム開発では、本システムの設計・開発者（以下、熟練者という。）が一連の流れを操作した時間と、同様の操作を一般利用者（以下、初心者という。）が操作した時間を測定し、その差を評価する NEM（Novice Expert ratio Method）法を利用した。具体的には、熟練者の操作時間を「熟練時間」、初心者に達成させたい時間を「目標時間」、操作方法が分からず続行不可能と判断する時間を「限界時間」として、各操作ごとに定義した。そして、目標時間や限界時間を超過した操作について、操作性を向上させるために改善の余地がないかどうかを検討した。熟練者は本システムの設計・開発者が担当し、初心者は本システムの利用者である PSIRT の事務局や当社製品の設計・開発者が担当した。

インタラクションデザインを実施した結果、参照画面と更新画面の違いが分かり難いといったことや、何度か上下に画面をスクロールしなければならないなどの改善点が 24 件検出された。この結果を受けて、画面タイトルや操作メニューに「新規登録」や「参照」といった情報を付加することや、画面項目の配置を見直すなどの改善を設計工程で実施した。

(2) 全文検索エンジン

脆弱性データベースに登録されているソフトウェアは、要件定義着手時点においても約 150 万件が登録されていた。本システムに構成情報を新規に登録した場合、これら 150 万件の情報に対して１件ずつ検索・照合処理を実施していては処理の高速化が見込めない。また、独自開発したソフトウェアについては、脆弱性データベースのソフトウェア名と完全一致するものがほとんど存在せず、部分一致で照合することになる。しかしながら、データベース標準の部分一致検索機能はインデックスが有効でなく、150 万件のデータに対して処理を実行した場合、データベースの負荷が高まる。その結果、Web ユーザーインターフェースのレスポンスにも影響を及ぼす可能性があった。

そこで、特定の単語などを検索文字列としてインデックス化し、部分一致検索を行う全文検索エンジンを採用した。この全文検索エンジンを採用することによって、データベースに負荷をかけずに、標準機能よりも高速に検索することが可能となった。なお、インデックスは検索対象の文字列をルールにしたがって分解することによって得られる。この分解ルールには、意味のある文字列に分解する「形態素解析方式」を採用した。その結果、単語単位の検索が可能になると同時に、不要な照合が低減されるため、レスポンスへの影

響を更に低減されている。加えて、ヒットした単語数をカウントし、システムが照合精度を提示することによって、目検仕分けする際の判断基準として役立てることが可能となった。

４．評価

４．１　構想に対する評価

従来の運用では、マクロの性能や人間系の作業負担から、過去５年あるいは 10 年分の脆弱性情報と構成情報を一括照合するといったことは現実的に困難であった。しかしながら、本システム運用開始後は構成情報を新規登録した際、過去に公開された脆弱性情報を漏れなく、かつ機械的に照合することが可能になった。そして、初回照合以降は、新たに公開された脆弱性情報や、脆弱性の影響を受けうるソフトウェアなどの重要な要素に変更があった脆弱性情報のみが照合され、CVE 番号ごとに対応状況を管理できるようになった。更に、脆弱性情報が公開されてから照合完了までのリードタイムも 24 時間以内に短縮された。

このように、本システムは従来の運用で課題としてきたことを解消し、日々公開される脆弱性をモニタリングするという PSIRT の運用をあるべき姿に近づけることができた。したがって、２．２節のシステム構想が妥当であったこと、更には仮説が正しかったといえる。

４．２　業務効率化

従来の運用では、マクロ処理結果の目検仕分け作業のために、２製品の対応に 0.2 人月要していた。この目検仕分けの作業工数は、対象製品に含まれるソフトウェアの数に比例する。本システム運用開始後は、2018 年９月末現在、57 製品を 0.2 人月で処理することができており、大幅に業務を効率化させることができた。そのほか、工数には表れていないけれども、対応状況を一元管理し、フォローアップを自動化したことも PSIRT の業務効率化につながっている。PSIRT が取り扱う製品は今後も増加すると予想されるものの、本システムを活用することによって、必要最小限の人員で運用することができると考えている。

４．３　システム品質

システム障害については、運用開始後約３ヵ月の期間において、軽微なものが６件発生した。しかしながら、これらの障害は、脆弱性データベースのメンテナンスなどに起因するものであり、いずれも致命的な不具合ではなかった。

レスポンスについては、オンラインの参照処理で３秒以下、更新処理で５秒以下（いずれも遵守率 80%）という目標を達成した。バッチ処理についても通常１～２分程度で完了しており、目標の 10 分以下を達成した。

そのほか、操作性に関しては、ユーザーからの目立った改善要望は確認されていない。CPU やメモリ使用量といったサーバリソースも 50%未満で推移しており、十分なリソースを確保できていることを確認している。

上記の理由から、本システムの品質は良好であり、今後の運用にも十分耐えられるものであると評価した。

５．今後の課題と展望

５．１　製品開発者の動機づけ

本システムの課題というより PSIRT の運営上の課題であるけれども、本システム上で製品設計者へ脆弱性の通知を行った後、期日を過ぎてもしばしば対応が滞っている。製品設計者は出荷後の製品よりも新製品の開発に追われており、全社的にも出荷製品への脆弱性対応の意識が高いとはいえない状況である。そのため、製品設計者への啓蒙活動を行うとともに、顧客との契約や社内ルールの整備などによる動機づけが必要であると考えている。

５．２　構成情報の入力揺らぎ

現時点では大きな問題となっていないものの、ハンド処理で構成情報を入力する際の”入力揺らぎ”がユーザーから指摘されている。具体的には、”WindowsVista”と”Windows Vista”といったのように、ユーザーによって入力内容が異なるというものである。このような入力揺らぎは、ソフトウェア名とバージョン情報を基に照合を行う際、単語分解の結果に影響を及ぼす。その結果、同一のソフトウェアを意図しているにも関わらず、両者の照合結果に差異が生じる可能性がある。また、全文検索エンジンのインデクッス生成にも影響することから、本システム全体のレスポンスが低下してしまうことなども懸念される。現在は PSIRT の事務局である情報セキュリティ推進室が、各製品の構成情報を横串でチェックしているけれども、本システムが自動的に検出あるいは補正する仕組みについて、今後検討していきたい。

５．３　脆弱性データベースの追加

本システムで利用している NVD や JVN iPedia といった脆弱性データベースは、脆弱性を漏れなく確認するという点では優れている。一方、脆弱性情報が初めて公開されてから、これらのデータベースに反映されるまでに数週間の期間を要する場合がある。PSIRT では、顧客影響が大きいほど、緊急対応が必要であることから、当社では JVN[15]の脆弱性レポートやベンダーのアドバイザリー情報なども参考としている。これらの情報は速報性が高い反面、NVD や JVN iPedia と比較すると１日あたりの更新頻度や更新件数は少ない。そのため、現在は人間系で収集し、必要に応じて本システムへ登録する運用としている。更なる業務効率化を目指し、NVD や JVN iPedia 以外の有償の脆弱性データベースや、ベンダーのアドバイザリー情報なども自動収集し、本システムへ反映する仕組みについても一考の余地がある。

６．おわりに

本稿では、自社製品の脆弱性を管理するためのシステムが必要であるという仮説を立て、スクラッチ開発したシステムの機能や工夫した点について論じた。そして、PSIRT の運用効率を大幅に向上させ、その運用をあるべき姿に近づけられたことから、仮説に基づくシ

[15] JPCERT コーディネーションセンター及び独立行政法人情報処理推進機構, “JVN” https://jvn.jp/

ステム構想が妥当であるということを示した。

一方、当社 PSIRT の歴史は浅く、製品開発者の動機づけや啓蒙活動がこれからも必要である。本システムについても改良余地があるため、PSIRT の業務効率を更に高められるシステムを目指して邁進していきたい。

最後に、インターネットに接続される機器の数は年々増加しており、世界の IoT デバイス数は 2020 年には約 400 億に到達するともいわれている[16]。今後も様々な業界で IoT 製品の開発が加速し、製品出荷後の継続的な脆弱性管理が必要となってくると予想される。本稿が皆様の取組の一助となれば幸いである。

参考文献

［１］独立行政法人情報処理推進機構：“IoT開発におけるセキュリティ設計の手引き”，（2018.4），PP22-24
https://www.ipa.go.jp/files/000052459.pdf

［２］鱗原 晴彦、龍淵 信、佐藤 大輔、古田 一義：“定量的ユーザビリティ評価手法：NEMによる操作性の評価事例およびツール開発の報告”，ヒューマンインターフェースシンポジウム2001，（2001.10）
https://www.ueyesdesign.co.jp/case/paper/his2001-nem.pdf

[16] 総務省，“情報通信白書（平成 30 年版）”
http://www.soumu.go.jp/johotsusintokei/whitepaper/h30.html