情報処理学会論文誌 Vol. 52 No. 9 2613–2623 (Sep. 2011)

ネットワークシステムにおける脆弱性影響の測定手法とシステム実装

原田敏樹[†1] 金岡晃[†2] 加藤雅彦[†3]
勝野恭治[†4] 岡本栄司[†2]

複数機器・複数機能の相互接続により構成されるネットワークシステムは電子社会において現在必要不可欠の基盤システムである．しかし現状のネットワークシステムの安全性に関する設計は経験に大きく依存して行われていて定性的であり，方法論や理論を用いて再現性を持った定量的な評価がほとんど行われていない．安全性を論じる尺度の 1 つに脆弱性の影響があげられ，脆弱性に対する定量的な評価手法の 1 つに CVSS（共通脆弱性評価システム）がある．CVSS は基本・現状・環境の 3 つの評価基準を持つが，各環境での対策判断指標となるべき環境値においては，一意なパラメータ決定が困難である．よって，環境値の利用にあたっては，この困難性を解決する必要がある．本論文では，脆弱性影響範囲の一意な決定手法を提案した．また，機械的な影響範囲特定を可能にするために，提案手法を実装したシステムも開発した．実装したシステムの評価を通して，CVSS における影響範囲のパラメータ決定が容易になったことを示した．

A Method for Vulnerability Risk Assessment in Networked Systems and Its System Development

Toshiki Harada,[†1] Akira Kanaoka,[†2]
Masahiko Kato,[†3] Yasuharu Katsuno[†4]
and Eiji Okamoto[†2]

A networked system interconnected with several network equipments is becoming essential in current infrastructure. However, it is still needed expert knowledge to design secure networked system in the absence of quantitative evaluation method based on methodologies and theories, while the vulnerability impact is one of the most important security measurements. CVSS (Common Vulnerability Scoring System) is one of well-known frameworks for scoring IT vulnerabilities. CVSS environmental metrics in CVSS metrics groups is expected to be used for countermeasures to vulnerabilities in networked systems, but, it is difficult to determine the parameters of the environmental score. Thus the method solving the difficulty is necessary for using the environmental score. In this paper, we proposed the method determining uniquely extent of the vulnerability impact. Furthermore, we implemented the method for enabling determining extent of the vulnerability impact automatically. Through the evaluation of implemented system, the ease of determing the CVSS parameter of the vulnerability impact extent is shown.

1. はじめに

近年インターネットの利用者は増加の一途をたどり，高速回線の普及やそれにともない多様なサービスが展開されている．そうしたインターネットを介したサービスは一般に単一のサーバにより提供されるのではなく，複数の機器を接続しネットワーク化したシステム（ネットワークシステム）により提供される．たとえば近年急速に利用が広まっているクラウドコンピューティングなども，非常に大規模なネットワークシステムによって構成されている．

ネットワークシステムは実際にサービスを行う Web サーバ，アプリケーションサーバ，データベースサーバなどのほか，それらの機器を中継するルータやスイッチ，ファイアウォールなどで構成される．また冗長構成をとることで，ある機器に不具合が生じた際などにシステムを停止することなくサービスを継続することが可能となる．こうしたメリットがある一方で，様々なソフトウェアがそれぞれにアクセス制御を行うため，小規模なネットワークシステムでも構成が複雑化してしまうというデメリットも存在する．その複雑性は，ネットワークシステム内のソフトウェアに脆弱性が発見された際の影響範囲の特定を困難にし，セキュリティ面にも影響が及ぶ．

†1 筑波大学大学院
Graduate School, University of Tsukuba
（投稿当時．現在の所属は株式会社野村総合研究所であり，本研究は株式会社野村総合研究所の事業とは無関係である．）
†2 筑波大学大学院
Graduate School, University of Tsukuba
†3 株式会社インターネットイニシアティブ
Internet Initiative Japan Inc.
†4 日本アイ・ビー・エム株式会社東京基礎研究所
IBM Research, Tokyo Research Laboratory

現在 OS やソフトウェアには 45,000 件を超える多くの脆弱性が発見されている[1]．それらの脆弱性は共通脆弱性評価システム（CVSS: Common Vulnerability Scoring System）[2] により深刻度が定量的に算出され，情報が一般に公開されている．CVSS は基本，現状，環境の 3 つの評価基準によりそれぞれ値を算出する．CVSS 基本評価基準により算出される基本値は米国の国立標準技術研究所（NIST: National Institute of Standards and Technology）[3] をはじめとして多くの組織で採用されているほか，これを利用したリスク分析などの研究も行われている[4)–9)]．一方，複数の脆弱性がシステム内に存在した場合に，それらの影響の規模に応じた優先順位を付けて迅速に対応する必要があり，CVSS 環境値はそのための指標となるものである．しかし環境値は一意にパラメータを決定することが困難であるため，現時点での環境値の利用は難しい．

そこで本研究は，CVSS 環境評価基準におけるパラメータの一意な決定を可能にし，環境値を用いた脆弱性評価の利用性を向上させることを目的とする．そのために，CVSS 基本評価基準に基づく脆弱性の性質から，ネットワークシステム内における脆弱性の影響範囲を一意に特定する．特定された脆弱性影響範囲を CVSS 環境評価基準に適用することで，パラメータの決定を可能にする手法を提案した．また，提案手法を実装したシステムを用いて，評価ならびに，出力結果についての考察を行った．

2 章では CVSS について説明し，3 章で CVSS についての関連研究を紹介する．4 章では本研究で用いるマルチレイヤネットワークモデルを説明する．5 章で脆弱性影響測定手法を提案し，6 章で提案手法の自動化システムを実装する．7 章では実装したシステムを用いていくつかのネットワーク構成に対して影響範囲測定を行い，その結果について評価と考察を行う．最後に 8 章でまとめる．

2. 共通脆弱性評価システム

共通脆弱性評価システム（以下 CVSS）は脆弱性の深刻度を数値化する手法として最も広く利用されている．CVSS は基本，現状，環境の 3 つの評価基準により構成され，それぞれの基準において値を算出する．

基本評価基準は，情報システムに求められる 3 つのセキュリティ特性である機密性 C（Confidentiality），完全性 I（Integrity），可用性 A（Availability）に対する影響の大きさと，攻撃容易性から潜在的かつ不変的な脆弱性固有の深刻度（基本値）を評価する．3 つのセキュリティ特性（CIA）に対する影響程度の定義[10] は以下のようになっている．C は「機密情報あるいは重要なシステムファイルが漏洩する可能性」を，I は「情報やシステムファイルが改ざんされる可能性」を，A は「リソースの枯渇や業務の遅延・停止が引き起こされる可能性」を評価し，評価は「なし」,「部分的」,「全面的」の 3 段階で行われる．

現状評価基準は攻撃コード，対策情報の有無，脆弱性情報の信頼性から脆弱性の現在の深刻度（現状値）を評価する．

環境評価基準は組織での製品利用状況や，攻撃を受けた際の二次被害規模，CIA への要求などを基準とした，製品利用者ごとに変化する各環境での深刻度（環境値）を評価する．環境評価基準の 1 つである TD（Target Distribution）は，対象の脆弱性による影響を受けるシステムの割合を示し,「なし（0%）」,「小規模（1〜25%）」,「中規模（26〜75%）」,「大規模（76〜100%）」の 4 段階で評価される．

現状値は基本値を，環境値は現状値をそれぞれ入力とするが，現状・環境評価基準の適用は任意となっており，適用しない場合は入力となった値がそのまま出力の値となる．現在一般的に用いられている CVSS は，主に基本評価基準によって算出された基本値を指しており，現状値と環境値の利用は進んでいない．現状値は基本値と同様に利用者環境に依存しない値であるため，第三者による提供の可能性がある．しかし現状値は時間経過により変化する値であるため，正確に提供するためにはすべての脆弱性の現状について逐次監視を行い値を更新する必要がある．これには膨大なコストを要することが明らかであり，実際の運用は困難である．またネットワークシステム管理者の観点では，最も重要なことは脆弱性によって及ぼされる影響の程度，すなわち対応決定の指標である．これには CVSS における環境値に該当するが，環境評価基準の各パラメータは，入力方法やパラメータの選択に関する指針が定量的ではないため利用が困難である．

3. 関 連 研 究

CVSS 基本値を用いたリスク評価の研究はいくつか行われている．Clark らの研究[6] では，主に企業が掲げるミッションとミッションを実現するための IT 資産がかかえるリスクの評価手法を提案している．ここでのリスク評価はミッション分析，脅威分析，リスク分析の 3 段階で行われる．ミッション分析ではミッションと資産の依存関係を，脅威分析では重要な資産と脆弱性の依存関係を，それぞれ木構造で表現する．そしてリスク分析では 2 つの木を統合し，フォルトツリー分析による定量化を行う．

また，CVSS 環境評価基準に対する研究も行われている．Fruhwirth らの研究[11] では，基本値のみから組織内の対応優先度を決定することは適切でないとして，現状値・環境値を利用するための手法を提案している．彼らは現状評価基準の一部と環境評価基準の一部の値

を，統計やヒアリングを用いた研究成果に基づく，コンテキスト情報と呼ばれる付加情報を用いて決定することで，現状値・環境値の算出を可能にした．

Clark らの研究においてフォルトツリー分析を行う際には，機器の故障内容やその原因，部位などを詳細に行う必要がある．そのため作業コストが膨大になることに加え，故障内容や原因，部位の候補抽出は機械的な抽出でなく再現性を欠くといった問題がある．Fruhwirth らの研究では，環境評価基準における CIA に対するセキュリティ要求の項目に絞ったパラメータ決定の手法を提案しており，影響範囲に関する研究は行われていない．

4. ネットワークシステムモデル

本章では，本研究で用いるネットワークシステムモデルである Networked systems Security Quantification モデル（NSQ モデル）[12] について説明する．金岡らにより提案された NSQ モデルは，ネットワークシステムを構成する各機器の特徴を失わないマルチレイヤネットワークモデルであり，依存関係の明確化や，複数種類の機器を同一モデル内に表現可能とするものである．ここでネットワークシステムとは，複数の機器がローカルエリアネットワーク（LAN）技術によって接続されネットワークを構成し，ローカルネットワークで 1 つ以上のサービスをローカルネットワーク外部に提供しているシステムと定義する．

NSQ モデルはグラフ $G = (V, E)$ を拡張したものであり，ノードとリンクの集合で表現される．しかし既存モデルとは異なり 1 つの機器は 1 つのノードでは表されず，機能の要素としてノードが各レイヤに存在し，それらノードとリンクの集合により 1 つの機器（モジュール）を表現する（**図 1**）．

4.1 モデルの定義

定義 4.1（レイヤ） レイヤは 5 つからなる．レイヤ 1（L1）は物理的接続の層であり，レイヤ 2（L2）は Ethernet ネットワークの層，レイヤ 3（L3）は IP ネットワーク層，レイヤ 4（L4）は TCP/UDP ネットワーク層，そしてレイヤ 5（L5）は抽象化したサービスの層である．

定義 4.2（ノード） 通信の終端あるいは中継点となる要素をノードと呼ぶ．ノードは終端ノードと中継ノードの 2 種類が存在する．

終端ノードは通信の始点あるいは終点となるノードであり，中継ノードは通信の始点あるいは終点ではないが，通信を行うにあたりそれら始点のアイデンティティ情報と終点のアイデンティティ情報からデータ配送可否の判断や配送する通信路の決定を行うノードである．

ノードは 4 つの属性を持つ．レイヤ情報，終端ノードか中継ノードの種別情報，所属モジュール情報，そしてアイデンティティ情報である．アイデンティティ情報はシステム内で一意に識別されるための情報であり，IP アドレスや MAC アドレス，ポート番号などが適用される．

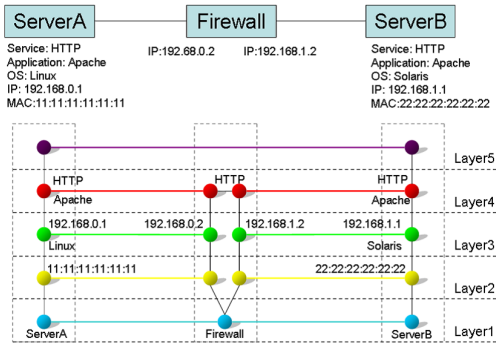

図 1　NSQ モデルによる表現例
Fig. 1　Networked system example of NSQ model.

L2 以上に属するノードは，必ず 1 階層下のノードと依存関係リンクにより接続されていなければならない．依存関係リンクに関しては後述する．

ネットワーク上の各ノードは v_i，その集合は V で表される．

定義 4.3（リンク） ノード間を結ぶ要素をリンクと呼ぶ．リンクは通信路リンクと依存関係リンク，中継リンクの 3 種類よりなる．

通信路リンクは同一レイヤでの異なるモジュールに属するノード間を結ぶリンクであり，当該レイヤでの通信路を示すものである．モジュールに関しては後述する．通信路リンクは向きを持ち，通信の方向を示す．L2 以上に属する通信路リンクは，当該リンクの始点・終点となるノードの各下位ノード間で到達可能になっていなければ存在できない．

依存関係リンクは依存関係があるノードを結ぶリンクである．レイヤ間を結ぶことも可能であるがノード間のレイヤ属性値の差は 1 に限る．また依存関係リンクは依存ノードから被依存ノードへの向きを持つ．

中継リンクはモジュール内で同一レイヤの中継ノード間を結ぶリンクである．中継リンクは向きを持ち中継の方向を示す．

リンクは 2 つの属性を持つ．1 つはレイヤ情報であり，もう 1 つはリンク種別を示す種別情報である．

定義 4.4（モジュール） ネットワーク機器の機能を提供するものをモジュールと呼ぶ．モジュールは依存関係リンクと中継リンク，ノードにより構成される部分グラフである．

モジュールはその機能によりサービスモジュール，インターネットモジュール，中継モジュール 3 つに大別される．中継モジュールはレイヤごとに細分化して表される．

サービスモジュール（S）はサービスを提供するモジュールである．また他のサービスモジュールに対するクライアントになることも可能である．各レイヤのノードはアイデンティティ情報を持つ．

インターネットモジュール（I）はシステムの外部全体（インターネット）を代表するモジュールであり，サービスモジュールに対するクライアントになる．各レイヤのノードは複数のアイデンティティ情報を持つことが可能である．

中継モジュール（R）はあるレイヤでの通信の中継を行うモジュールである．レイヤにより L1R，L2R，L3R，L4R，L5R の 5 つに細分化される．L2 以上のノードはアイデンティティ情報を持たないことがあり，またモジュール内の最上位レイヤに存在するノードはすべて中継ノードとなる．

定義 4.5（ネットワークとシステム） ネットワークはノードとリンクの集合であり，システムは各レイヤのネットワークを結合したものである．

4.2　モデルの応用

NSQ モデルはネットワークシステムの最適設計を行うことを目的として提案されたものであるが，ネットワーク設計の最適性を論じるには最適を示す尺度が必要である．金岡らは安全性の視点でのシステムの最適性の尺度として，アクセス制御の状態，各機器の障害がシステムに稼働状態に与える影響，各機器の脆弱性がシステムの安全性に与える影響などの検討を行っている．

脆弱性影響の検討においては，集合を用いてその影響範囲を求める式が提示されているが，脆弱性の分類や，脆弱性の影響の定義，到達可能性の求める手法などを具体的に求める手法が欠けており，実際に影響範囲を求めることは難しい．

5. 提 案 手 法

本章では NSQ モデルにおける安全性に関する最適設計の尺度となる，定量的な脆弱性影響の測定手法を提案する．

5.1　脆弱性の分類

脆弱性の影響範囲は，脆弱性影響の性質によって異なる．たとえば，ルータ内に経路情報を閲覧されてしまう脆弱性と，システムダウンが引き起こされてしまう脆弱性が存在するとした場合，前者の影響がルータ以外に及ぶことはないが，後者ではルータを介して行われる通信がすべて遮断されてしまう．こうした脆弱性影響の性質を一意に判断するために，CVSS 基本評価基準における「CIA への影響」を利用する．CIA それぞれに対する影響の性質に応じて影響範囲を決定する．

C に影響を与える事例は情報漏洩であり，脆弱性により任意のファイルの閲覧やデータベースファイルのダウンロードなどが可能になった場合が該当する．これらの影響は対象機器内にとどまるものである．

I に影響を与える事例は情報改ざんであり，任意ファイルの作成や上書きなどが可能になった場合が該当する．ルーティングテーブルを改ざんすることで対象機器が周囲の機器と通信不可能な状態に陥る可能性があることから A への影響があるものと評価される．

A に影響を与える事例は，リソースの枯渇やシステムの停止であり，対象機器が周囲の機器と通信不可能な状態に陥ることなどが可能になった場合が該当する．

以上の分類ごとの性質を見ると，影響範囲が対象機器の周囲に拡大しうる脆弱性は，A への影響があるものに限られる．そこで，影響範囲の特定にあたり脆弱性の性質を，対象機器周辺への影響拡大が考えられる脆弱性 W_A と，対象機器のみに影響範囲がとどまる脆弱性 W_{nA} の 2 種に大別する．

5.2　脆弱性影響の定義

影響範囲を定義するにあたり，まずは NSQ モデル上での脆弱性影響の定義を明確にする必要がある．また脆弱性対象のノードにおける影響と，そこから周囲に波及する影響とではその性質が異なるため，本論文では脆弱性保有のノードにおける影響を「直接影響」，周囲に波及する影響を「間接影響」，それらを総称して「影響」とする．またそれらの影響を受けているノードを「直接影響ノード」,「間接影響ノード」,「影響ノード」と呼称する．本節ではこれらの性質と前述の脆弱性分類に沿って，NSQ モデル上の各レイヤにおける脆弱性影響を定義する．

定義 5.1（直接影響） あるノードが脆弱性を持っている場合に，その脆弱性に対する攻撃によって当該ノードまたは同一モジュール内の他のノードが直接受ける影響である．

W_A による直接影響は，ノードが利用するリソースの不足や，それによってサービスが停止してしまうことを意味する．

W_{nA} による直接影響は，ノードが持つ情報の漏えいや改ざんが行われることを意味する．

ここで，リソース不足による電気信号の停止は考え難いため，L1 における W_A の影響は

ないものとした．また脆弱性の影響による不当なハードウェア情報やイーサネット情報の漏洩・改ざんも考え難いため，L1 および L2 における W_{nA} の影響もないものとした．直接影響の被害対象となるノードの集合を「直接影響ノード集合」とし，脆弱性の分類に応じてそれぞれ $V_{A_{dir}}$，$V_{nA_{dir}}$ と表される．

定義 5.2（間接影響） 当該ノードは正常に動作しているが，当該ノードと異なるモジュールに属するノードへの脆弱性影響によって当該ノードの機能が果たされていない．

W_A による間接影響は，ノード自体は脆弱性を持っていないが，そのノードを宛先とする通信経路が W_A の直接影響または間接影響を受けており，そのノードへの通信可能性が失われていることを意味する．

W_{nA} による間接影響は，前述のとおり周囲へ影響が拡大しないため，ないものとする．

間接影響の被害対象となるノードの集合を「間接影響ノード集合」とし，$V_{A_{indir}}$ と表す．

次節以降で W_A と W_{nA} の影響範囲の特定を行う．

5.3　W_A の影響範囲

5.3.1　W_A の直接影響範囲

W_A の影響には計算リソースの圧迫やマシンのクラッシュなどがあげられ，これらの影響は L2 以上の通信階層におけるデータ通信を阻害すると考えられる．すなわち脆弱性が存在しうるノードは L3 または L4 に存在するが，その影響は計算リソースを利用するすべてのノードに及ぶ．このことを NSQ モデル上で表現すると，脆弱性を持つノードと依存関係リンクによって接続されたモジュール全体が直接影響範囲となる．しかし，たとえば複数のサーバマシンで 1 つの Web サービスを冗長化，すなわち 1 つの L5 ノードが異なる L1 ノードを持つモジュールに対して依存関係リンクで接続している場合は，必ずしも W_A の影響が L5 に到達しない．なぜなら複数あるサーバマシンのいずれか 1 台に脆弱性があり可用性が損なわれたとしても，それ以外のサーバマシンによるサービス提供の継続が可能であるためである．本来，同一サービスを提供するサーバを冗長化するのであれば，すべて同一のマシンや同一のバージョンのソフトウェアで構成するのが望ましい．しかし，冗長性の拡張の際に，導入時期が異なり，以前の構成とまったく同一のものを用いない可能性も考えられる．

5.3.2　W_A の間接影響範囲

W_A の影響はさらに周囲のモジュールへも及びうる．たとえば電子商取引サイトを考えた場合，ユーザはフロントの Web サーバにアクセスすることでバックに存在する DB サーバの情報を利用することが可能だが，Web サーバの可用性が侵害された場合，連鎖的に DB サーバも利用不可能になってしまう．Internet から Web サーバや DB サーバへの通信の流れがある場合の，Internet 側を「上流」，Web サーバや DB サーバ側を「下流」と呼ぶとき，影響範囲を特定するには下流ノードと上流ノードとの関係性から，下流ノード群の可用性を考える必要がある．

各レイヤで上流・下流を考えるとき，L4 および L5 においては通信路リンクの向きから上流と下流の判別が可能である．しかし L2 と L3 における通信は双方向に行われるため，リンクの向きのみから上流と下流の判別を行うことは不可能である．そこで L4 のリンクの向きに基づいて，L2 と L3 の上流と下流の判定を行う．

5.3.2.1　上流・下流の判定

L3 では，L4 において通信路リンクの送信元となっている L4 ノードの直下の L3 ノードを「最上流ノード」と定義し，各最上流ノードを $u_{3,i}$，その集合を $U_3 \subset V$ で表す．最上流ノードから通信路リンクと中継リンクを交互にたどった際に経由するノードの順列を「L3 パス」と定義し，各 L3 パスを $p_{3,i}$，その集合を P_3 で表す．L3 パスを抽出する場合は，はじめに L4 における通信路リンクの情報からすべての最上流ノードを特定し，その後にパス抽出を行う．なお冗長構成によっては 1 組の最上流ノードと最下流ノードに対して複数のパスが存在する可能性があり，またある 1 つのノードが複数のパスの要素になりうる．L3 における 1 つの最上流ノードに対する 1 つのパスを抽出するアルゴリズムを Algorithm 1 に示す．ここでアルゴリズム内の E_R はシステムの中継リンクの集合，E_C は通信路リンクの集合である．

L2 でも L3 と同様にパスを形成するが，L3 パス内で経由する L3R の中継ノードの直下ノードが L2 パス内の最上流もしくは最下流ノードとなりうるため，そうしたノードを検出するために 2 つの手順を踏む．L3 パスの最上流ノード直下の L2 ノードを，L2 パスの最上流ノード $u_{2,i}$ とし，その集合を $U_2 \subset V$ で表す．L3 パスの最下流ノード直下の L2 ノードを，L2 パスの最下流ノード $d_{2,i}$ とし，その集合を $D_2 \subset V$ で表す．次に最下流ノードとなっている L3R の L2 ノードを L2 における最上流ノードの候補から除外し，残った L3R の L2 ノードを L2 における最上流ノードに加える．以上で決まった各最上流ノードから通信路リンクをたどり，経由したノードの順列で「L2 パス」を形成する．各 L2 パスを $p_{2,i}$，その集合を P_2 で表す．なお最上流ノードに確定した L2 ノードは最下流ノードにはなりえない．

Algorithm 2 に L2 における最上流ノードを抽出するアルゴリズムを示し，Algorithm 3 に L2 の最上流ノード群 U_2 と 1 つの最上流ノードに対する 1 つのパスを抽出するアルゴリズムを示す．ここで m_x はノード x の所属モジュール種別，l_x はノード x のレイヤ情報を

Algorithm 1 L3 パス抽出アルゴリズム：$findL3path()$

Require: L3 node x
Ensure: L3 node array $p_3 \in P_3$

```
1: p3 ← φ
2: p3 ← p3 ∪ {x}
3: for all e_i = (x, a) ∈ E_C do
4:   if a ∉ p3 ∧ a ∉ U3 then
5:     p3 ← p3 ∪ {a}
6:     if a is a Terminal Node then
7:       return p3
8:     else
9:       for all e_j = (a, b) ∈ E_R do
10:        if b ∉ p3 then
11:          p3 ← p3 ∪ {b}
12:          p3 ← p3 ∪ findL3path(b)
13:        end if
14:      end for
15:    end if
16:  end if
17: end for
```

Algorithm 2 L2 パス最上流ノード群抽出アルゴリズム：$findL2upperNode()$

Require: L3 path set P_3
Ensure: L2 node set U_2

```
1: U2, D2, X2 ← φ
2: for all p3 ∈ P3 do
3:   for all e = (x', x) ∈ E_D s.t. x' is a First Node of p3 do
4:     U2 ← U2 ∪ {x}
5:   end for
6:   for all e = (y', y) ∈ E_D s.t. y' is a Last Node of p3 do
7:     D2 ← D2 ∪ {y}
8:   end for
9: end for
10: for all d2 ∈ D2 do
11:   X2 ← d2 s.t. m(d2) = L3R
12: end for
13: for all v ∈ V s.t. l_V(v) = 2 ∧ v ∉ X do
14:   U2 ← U2 ∪ {v}
15: end for
16: return U2
```

表し，$m : V \to M$ はノードの属するモジュール種別を示す写像，$l_V : V \to L_V$ はノードのレイヤ情報を示す写像である．

5.3.2.2　間接影響範囲の特定

L3 と L2 における上流・下流が決定することで，各レイヤにおけるノードの冗長性を判定し影響範囲の特定が可能になる．影響範囲特定は，はじめに直接影響ノードを決定し，続いて下流およびレイヤ間の間接影響について検討する．

まず下流への間接影響の特定を行う．L5 と L4 では，ある W_A 直接影響ノードの下流で隣接するノードにおいて，上流で隣接するすべてのノードが W_A の影響を受けている場合に，当該ノードは間接影響ノードとなる．L3 と L2 では，影響ノードが属するパス上の各ノードにおいて，当該ノードを要素を経由するすべてのパスの上流側に W_A 被影響ノードが存在するとき，当該ノードも間接影響ノードとなる．

続いて，特定された間接影響ノードのレイヤ間影響について検討する．まず，あるノードの直下レイヤのノードがすべて影響ノードに含まれている場合，そのノードの依存先の機能が果たされていないことを意味し，間接影響ノードとなる．さらに，新たに間接影響ノードとなったすべてのノードについて，直下レイヤのノードの影響有無に基づく影響判定を行う．すなわち，あるノード C の直下ノード D の直上ノード Y がすべて影響ノードだった場合，ノード D はその上位レイヤにおける機能を果たしていないと見なされ間接影響ノードとなる．レイヤ間影響特定の操作は新たな間接影響ノードが特定されなくなるまで行われ，終了時点までに 1 つ以上の新たな間接影響ノードがあった場合は，そのノードに対して再

Algorithm 3 L2 パス抽出アルゴリズム：$findL2path()$

Require: L2 node $u_2 \in U_2$
Ensure: L2 node array $p_2 \in P_2$

```
p2 ← φ
p2 ← p2 ∪ {u2}
for all ei = (u, x) ∈ EC do
    if u ∉ p2 ∧ u ∉ U2 then
        p2 ← p2 ∪ {x}
        if x is a Terminal Node then
            return p2
        else
            p2 ← p2 ∪ findL2path(x)
        end if
    end if
end for
```

び下流への間接影響特定を行う．

上記の下流影響特定とレイヤ間影響特定を互いに新たなノードが検出されなくなるまで交互に繰り返し，終了時点での直接影響ノードと間接影響ノードの和を影響ノードが影響範囲とする．

W_A の影響ノード V_A は以下の式で表される．

$$V_A = V_{A_{dir}} \cup V_{A_{indir}} \tag{1}$$

5.4 W_{nA} の影響範囲

W_{nA} の影響には脆弱性対象のソフトウェアにおける情報漏洩や改ざんがあげられる．これらの影響によって通信が阻害されることは考え難いが，同一モジュール内で脆弱性対象を利用しているもの（依存関係リンクの元）は何らかの影響を受けることが考えられる．たとえばある DB サービスを複数の DB サーバで冗長化しており，かつそのサービスが異なるアプリケーションや同一のアプリケーションであっても異なるバージョンで構成されている場合，それらのサーバのいずれか 1 つの情報が漏洩もしくは改ざんされてしまえば，上位のサービスは同様の影響を受けてしまうと考えられる．一方で上位層アプリケーションの影響が下位層の OS に波及することは考え難い．よって対象ノードとその上位層のノードが冗長化の有無にかかわらず直接影響ノードとなり，下流への影響はない．

5.5 CVSS 環境評価基準への適用

前節の影響範囲の定義を CVSS 環境評価基準の TD に適用する．これによって再現性のある影響範囲（TD）の決定が可能になった．TD は脆弱性の影響を受ける対象がシステムに占める割合を表しており，その程度に応じて 4 段階の評価を行う．TD は NSQ モデルにおける脆弱性影響範囲の割合と同一と見なすことができ，全ノード数に占める影響ノード数の割合がそのまま TD と考えられる．ただし Internet モジュールに属するノードは影響ノードに含まれることがないため，全ノード数に含めない．

TD は以下の式で表される．ここで，N_G はネットワーク全体のノード数，N_I はインターネットモジュールのノード数を示す．

$$TD = \frac{|V_A \cup V_{nA}|}{N_G - N_I} \tag{2}$$

6. システム実装

提案手法を用いて，CVSS 環境評価基準における TD を一意に測定するツールを開発する．開発にあたって，ツールには各脆弱性の情報を取得する機能が必要になる．その方法として，ツールがローカルで全脆弱性の情報を持つ方式が考えられるが，処理効率や情報更新の面で，オンラインで利用可能な脆弱性情報サービスの利用が望ましい．しかし，機械処理が可能な形で CVSS 基本評価基準を含んだ情報であり，かつ，国内ソフトウェアだけでなく全世界のソフトウェアに関する情報を網羅的に提供する脆弱性情報サービスは現状で存在しない．そこで Web API として利用可能なシステムを実装した．

本システムは，NSQ モデルで表されたネットワークシステム情報と脆弱性情報の組合せが同一であれば，同一の TD を算出する．すなわち，一意性が保たれた測定システムを実現した．

6.1 システム概要

本システムは，大きく 2 つの要素で構成されている．1 つは提案手法を実装した “脆弱性影響測定（VA: Vulnerability Assessment）ツール”，もう 1 つは “脆弱性情報提供 WebAPI” である．

脆弱性影響の測定は以下の手順で行うことである．はじめに，ネットワークシステムを NSQ モデルで表現した XML データを用意する．VA ツールはこの XML データからシス

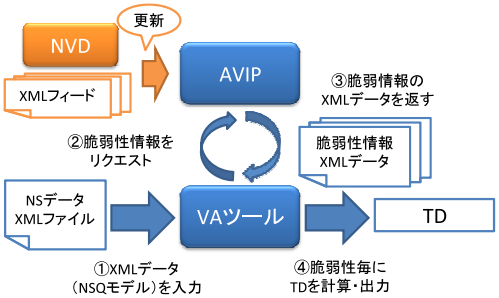

図 2　システム概要
Fig. 2　Outline of system.

テム構成情報を取得し，その情報に基づき脆弱性情報提供 WebAPI に対してリクエストを送信する．その後 VA ツールは WebAPI として利用可能なシステムから取得した脆弱性情報の XML に含まれる CVSS 基本評価基準の値を読み取り，その脆弱性による影響度を提案手法により算出し，最後に CVSS 環境値を出力する．上記の一連の処理を**図 2** に示す．

6.2　VA ツール

提案手法を実装し，自動的かつ仮想的にネットワークシステムにおける脆弱性影響範囲の測定を行うツールである．実装は Ruby 1.8.6-p111 で行った．本ツールは NSQ モデルの XML データ（NS データ）を入力として，OS やアプリケーションの脆弱性の影響範囲の割合を出力する．

本ツールに NS データを入力として与えると，L3 ノードおよび L4 ノードのノード ID と対応するソフトウェア情報が表示される．ユーザはその中から測定対象としたいソフトウェアをノード ID で選択する．ツールはそのソフトウェア情報に基づいて後述する WebAPI にリクエストを送信し，脆弱性情報を XML データとして取得する．取得した脆弱性情報の CVSS 基本評価基準の値から脆弱性の種類を識別し，提案手法に基づいて影響範囲を測定，すなわち TD を算出し，標準出力する（図 2）．対象のソフトウェアに対して複数の脆弱性情報が得られた場合は，各脆弱性すべてに対して TD 算出を行う．

6.3　脆弱性情報提供 WebAPI

VA ツールで用いる脆弱性情報を提供する WebAPI である．実装は Ruby on Rails 1.2.6 で行った．脆弱性情報のデータベースとしては，NVD，OSVDB，JVNiPedia などが一般的であるが，これらは主にブラウザベースでの利用を前提としたインタフェースとなっており，機械的な処理が困難である．また NVD などは XML 形式でのデータ配信も行っているが，一部の最新情報のみ，もしくは年ごとにまとめられた情報であるため非常に膨大なデータサイズとなっており，そのままのデータを利用することも困難である．

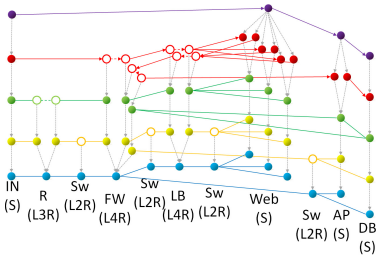

図 3　DMZ 構造
Fig. 3　DMZ architecture.

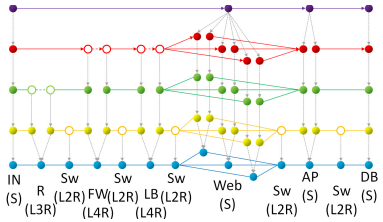

図 4　三層構造
Fig. 4　Three-tier architecture.

そこで任意のソフトウェアに限定した脆弱性情報を XML 形式で取得するための WebAPI を開発した．本 WebAPI を “AVIP: Automatic Vulnerability Information Provider” と呼ぶ．まず AVIP はあらかじめ NVD から脆弱性情報の XML データをダウンロードし，内部の情報を抽出して自らのデータベースに格納しておく．そして外部からのリクエストがあったときに格納しているデータを XML 形式で再構成し，リクエスト元に送信する．なおデータベースの更新は毎日行われ，2011 年 3 月の時点では 45,000 超の脆弱性情報を保有している．

7.　システム影響度計算の評価

提案手法をいくつかの一般的ネットワーク構成に適用し，再現性のある影響範囲の算出と，その処理時間が実用上問題がないかを評価する．

7.1　評 価 項 目

ソフトウェア工学では，システム構成に多層アーキテクチャが採用されることがある[13]．また，ローカルエリアネットワークの構成アーキテクチャとして内部ネットワークと外部アクセス可能なネットワークを分離した DeMilitarized Zone（DMZ）構造がある．本論文では，マルチレイヤアーキテクチャ（二層構造，三層構造）と DMZ を対象として，TD の測定を行った．

これらのネットワークは主にスイッチ（Sw），ルータ（R），ファイアウォール（FW），

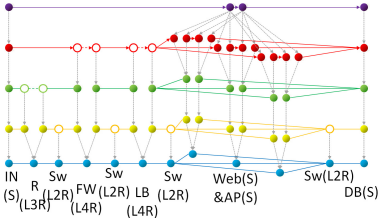

図 5　二層構造 A（Web と AP を統合）
Fig. 5　Two-tier architecture A (Web & AP in one).

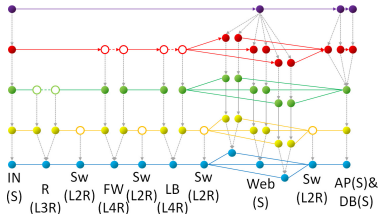

図 6　二層構造 B（AP と DB を統合）
Fig. 6　Two-tier architecture B (AP & DB in one).

ロードバランサ（LB），Web サーバ（Web），アプリケーションサーバ（AP），DB サーバ（DB）で構成され，Web サービスは 10 台の Web サーバで冗長化された場合を考える．対象ネットワークを NSQ モデルで表したものを**図 3**，**図 4**，**図 5**，**図 6** に示す．

測定方法は以下のとおりである．ネットワーク内の L3 ノードおよび L4 ノードにおいて，脆弱性 W_A，W_{nA} が存在した場合の影響範囲をモジュールごとに測定する．その際 Web サーバモジュールに関しては 10 台中の直接影響台数で場合分けを行い，各影響範囲の平均をとったものを影響範囲とする．

上記の測定を 4 種の各ネットワークに対して行い，測定結果の比較を行った．

7.2　結　　果

7.2.1　VA ツール計算時間

各ネットワークに対する脆弱性影響範囲の測定を VA ツールを用いて行った．各ネットワークに対して，それぞれのノードを脆弱ノードと仮定し脆弱性影響範囲測定を行うことを 10 回ずつ繰り返した．その結果 VA ツールによる影響範囲測定にかかる計算時間は平均で 71.40 秒であった．そのうち，AVIP への脆弱性情報リクエストとその回答までの時間が平均で 67.42 秒と全体の 94.4%を占めていた．実際に影響度測定に費やしているのは 3.98 秒であった．

7.2.2　影響範囲測定結果

各ネットワークに対する脆弱性影響範囲の測定結果を**表 1**，**表 2** に示す．

ここで，$P(W_{nA})$，$P(W_A)$ は，Internet モジュールのノードを除く全ノードを母数とした W_{nA}，W_A の数によりそれぞれの影響ノードの割合を表している．また $P_5(W_{nA})$，$P_5(W_A)$ は，測定結果に含める範囲を L5 に限定し，サービスへの影響にのみ焦点を当てた影響割合である．すなわち影響ノードの割合の母数を Internet モジュールのノードを除く全 L5 ノードとしている．

表 1　影響範囲測定結果（アーキテクチャ別）
Table 1　Vulnerability assessment result (each architecture).

	$P(W_{nA})$	$P(W_A)$	$P_5(W_{nA})$	$P_5(W_A)$
DMZ	11.9%	35.4%	27.7%	31.0%
二層 A	14.35%	47.5%	39.1%	28.3%
二層 B	18.5%	45.1%	28.4%	30.8%
三層	11.0%	40.8%	27.7%	31.0%

表 2　影響範囲測定結果（モジュール別）
Table 2　Vulnerability assessment result (each module).

	P(W_{nA})	$P(W_A)$	$P_5(W_{nA})$	$P_5(W_A)$
ルータ	1.8%	100.0%	0.0%	100.0%
FW	3.4%	93.6%	0.0%	100.0%
LB	3.3%	81.4%	0.0%	100.0%
Web	18.45%	35.4%	37.3%	10.0%
DB	4.2%	5.3%	41.7%	41.7%

表 1 は各モジュールの脆弱性影響範囲の割合について，ネットワーク構成ごとに平均したものである．表 2 は全測定パターンの結果に対してモジュールごとに平均をとったものである．なお表 2 において，二層構造では AP サーバが Web サーバまたは DB サーバに統合されてしまっているため，AP サーバモジュール個別の測定については省略した．

7.3　考　　察

本節では，得られた結果に対する考察をそれぞれシステムの動作パフォーマンス，結果からのネットワーク分析，提案手法自身の 3 つの視点から行う．

7.3.1　システムの動作性能について

VA ツールによる脆弱性影響範囲の計算時間は 71.40 秒であったが，そのうち AVIP のリクエストから回答までの時間が 67.42 秒と全体の 94.4%を占めていた．AVIP が格納する脆弱性数は 45,000 件あるが検索システムとして高速化が十分に図られていないため，検索に費やす時間が多くなっている．それをのぞいた計算時間は 3.98 秒であり，実用上では問題ない値であると考えられる．

7.3.2　ネットワーク構成が脆弱性影響範囲に与える要因について

まず表 1 について考察する．$P(W_{nA})$，$P(W_A)$，$P_5(W_{nA})$ を見ると，DMZ と三層構造が二層構造と比較して影響範囲は小さくなっており，より安全性が高いことが分かる．

$P_5(W_A)$ に関しては二層構造の方が影響範囲が小さくなっているが，これは AP サーバが Web サーバや DB サーバと統合されたことによって，1 つのアーキテクチャに対する測定回数が減少したためであると考えられる．

次に表 2 では，ルータや FW などの中継機器においては W_A の影響が非常に大きく，W_{nA} の影響が小さい．一方 Web や DB などのサーバマシンにおいては W_A の影響は小さく，W_{nA} の影響が比較的大きくなり，特にそれは L5 において顕著である．これは中継機器の役割がローカルネットワーク外とローカルネットワークの間や，多数のサーバマシン間の中継など，ネットワークの根幹を担うものであるためであり，それらの機能が失われた場合の影響が大きいことを示している．

なお $P_5(W_A)$ において，間接影響が拡大すると思われる Web サーバでの影響割合が DB サーバよりも低くなっているが，これは Web サービスが 10 台の Web サーバで冗長化されており，かつすべての Web サーバが影響を受けていなければ Web サービスに影響がないためである．また，Web サーバにおいて，全ノードを対象とした場合と L5 ノードのみを対象とした場合に，W_{nA} と W_A の割合の差の傾向が大きく異なっている．これは冗長化の有無にかかわらず W_{nA} の影響が L5 ノードに及ぶためである．一方，DB サーバでは，全ノードを対象にした場合は影響が小さく，L5 ノードのみを対象とした場合では大きくなっているが，これは DB サーバが 1 台で 1 つのサービスを持っていることによる．

7.3.3　提案手法により保たれる一意性とその応用について

本提案手法において，同一の脆弱性情報と同一のネットワークシステム情報を与えた場合には，同一の影響範囲が算出された．このことから，一意性が保たれていることが分かる．

一方本提案手法では，影響範囲が拡大する脆弱性の影響において，Web サービスが複数の Web サーバにより冗長化されている場合には，いずれかの Web サーバが稼働していればサービスに影響がないとした．しかし，サーバマシンおよびソフトウェアの性能やネットワークの速度，同時アクセス可能数や実際のアクセス数などの相互影響によっては，1 台では処理が間に合わない可能性がある．これについては NSQ モデルの拡張や，各ネットワークシステムの非機能要件などを考慮に入れることが対策として考えられる．

また，本提案手法は CVSS 環境評価基準の中でも，影響範囲に特化したパラメータ決定手法であり，環境値の算出には他のパラメータを決定する必要がある．現時点においては，Fruhwirth らの研究と組み合わせることにより，現状評価基準の「脆弱性情報の信頼性」と，環境評価基準の「二次的被害の可能性」以外の項目を補うことができる．このことから，CVSS のユーザビリティが向上したといえる．

8. ま と め

CVSS は，脆弱性影響を定量評価するために広く利用されている手法である．これは基本・現状・環境の 3 つの評価基準を持つが，利用が進んでいるのは基本値のみであり，各環境での対策判断指標となるべき環境値はパラメータ決定が困難である．

そこで本論文では，CVSS 環境評価基準における TD（Target Distribution）の一意なパラメータ決定の困難性を解決するため，マルチレイヤネットワークモデルである NSQ モデルを利用し，現実に即した過程を考慮したうえで，再現性を持った脆弱性影響範囲の特定手法を提案した．また提案手法を実装したことで，自動的かつ一意なパラメータ決定が可能になり，CVSS のユーザビリティが向上したといえる．

提案手法を用いることで，機械的に脆弱性影響の環境評価を行えるようになったことに加え，マルチレイヤネットワークモデルにおける安全性の最適設計手法の検討も可能になる．今後は，CVSS における他の評価項目を一意に決定する手法の検討と，実際のサービスに求められる機器の性能など，非機能要件の調査も行う．また実環境からのモデル情報化による測定プロセスの完全自動化のための研究・実装も行う．

参 考 文 献

1) National Vulnerability Database, available from ⟨http://nvd.nist.gov/⟩.
2) CVSS: Common Vulnerability Scoring System, available from ⟨http://www.first.org/cvss/⟩.
3) NIST: National Institute of Standards and Technology, available from ⟨http://www.nist.gov/index.html⟩.
4) Wang, J.A., Zhang, F., Xia, M.: Temporal metrics for Software Vulnerabilities, *Proc. 4th Annual Workshop on Cyber Security and Information Intelligence Research*, pp.1–3 (2008).
5) Scarfone, K. and Mell, P.: Vulnerability Scoring for Security Configuration Settings, *Proc. 4th ACM Workshop on Quality of Protection*, pp.3–8 (2008).
6) Clark, K., Singleton, E., Tyree, S. and Hale, J.: Strata-Gem: Risk Assessment Through Mission Modeling, *Proc. 4th ACM Workshop on Quality of Protection*, pp.51–58 (2008).
7) Lai, Y.-P. and Hsia, P.-L.: Using the vulnerability information of computer systems to improve the network security, *Computer Communications*, Vol.30, No.9, pp.2032–2047 (2007).
8) Aime, M.D., Atzeni, A. and Pomi, P.C.: AMBRA–Automated Model-Based Risk

Analysis, *Proc. 4th ACM Workshop on Quality of Protection*, pp.43–48 (2008).
9) Frei, S., May, M., Fiedler, U. and Plattner, B.: Large-Scale Vulnerability Analysis, *Proc. 2006 SIGCOMM Workshop on Large-scale Attack defense*, pp.131–138 (2006).
10) 情報処理推進機構セキュリティセンター：共通脆弱性評価システム CVSS v2 概説 (2007)，入手先⟨http://www.ipa.go.jp/security/vuln/SeverityCVSS2.html⟩.
11) Fruhwirth, C. and Mannisto, T.: Improving CVSS-based vulnerability prioritization and response with context information, *Proc. 2009 3rd International Symposium on Empirical Software Engineering and Measurement*, pp.535–544 (2009).
12) 金岡　晃，原田敏樹，加藤雅彦，勝野恭治，岡本栄司：安全なネットワークシステム設計のためのマルチレイヤネットワークモデルの提案と応用，情報処理学会論文誌，Vol.51, No.9, pp.1726–1735 (2010)
13) Alonso, G., Casaty, F., Kuno, H. and Machiraju, V.: *Web Services*, p.354, Springer (2003).

(平成 22 年 11 月 30 日受付)
(平成 23 年 6 月 3 日採録)

原田　敏樹

2011 年筑波大学大学院システム情報工学研究科博士前期課程修了．修士（工学）．同年株式会社野村総合研究所入社．流通システム基盤に関する業務に従事．

金岡　　晃（正会員）

2004 年筑波大学大学院博士課程システム情報工学研究科修了．同年セコム株式会社入社．ネットワークセキュリティ，電子認証の研究開発に従事．2007 年より筑波大学大学院システム情報工学研究科研究員．2008 年より筑波大学大学院システム情報工学研究科助教．ネットワークシステムの安全設計方式，電子認証に関する研究に従事．博士（工学）．電子情報通信学会員．

加藤　雅彦（正会員）

1995 年豊橋科学技術大学大学院知識情報工学専攻修士課程修了．修士（工学）．1998 年株式会社インターネットイニシアティブ入社．セキュリティ情報統括室シニアエンジニア．日本ネットワークセキュリティ協会幹事，調査研究部会長，日本セキュリティオペレーション事業者協議会運営委員，日本クラウドセキュリティアライアンスボードメンバー．豊橋技術科学大学非常勤講師，電子情報通信学会会員．

勝野　恭治（正会員）

1998 年慶應義塾大学大学院理工学研究科計算機科学専攻修士課程修了．同年日本アイ・ビー・エム株式会社入社．東京基礎研究所主任研究員．2009 年筑波大学大学院システム情報工学研究科リスク工学専攻後期博士課程修了．2003 年ソフトウェア科学会高橋奨励賞受賞．情報セキュリティ，コンピューター・ネットワーク，エージェント技術に関する研究開発に従事．博士（工学）．日本ソフトウェア科学会会員．

岡本　栄司（正会員）

1973 年東京工業大学工学部電子工学科卒業．1978 年同大学院博士課程了．工学博士．同年日本電気中央研究所入社．その後，北陸先端科学技術大学院大学，東邦大学を経て 2002 年より筑波大学教授．情報セキュリティの教育・研究に従事．1990 年電子情報通信学会論文賞，1993 年本会ベストオーサ賞受賞．著書『暗号理論入門』（共立出版），『電子マネー』（岩波書店）等．