インターネットの脅威に対抗しうる脆弱性データベースと検証システムの構築

大野 浩之　　武智 洋　　永島 秀己

総務省 通信総合研究所　横河電機株式会社　横河電機株式会社

あらまし インターネットは，実社会のさまざまな実用的なアプリケーションの基盤として利用されるようになり，重要な社会基盤として認知されるに至った．これに伴い「インターネットの危機管理」が重要であるという考え方が広まってきている．既存の重要社会基盤とは異なり，現状のインターネットでは，一般利用者や，一般利用者が所属する組織も危機管理を念頭においた対処が必要となる．著者らはこの認識に基づき研究を進めており，本論文では，その一環として，脆弱性データベースの構築と DDoS attack 検証システムを構築し，これらのシステムの有効性を確かめた．

1 まえがき

インターネットは，人々の生活に根付き，重要な社会基盤とみなされるようになった．重要な社会基盤であるならば，さまざまな危機を想定し，それに対応する方法を整えておかなければならない．重要社会基盤とみなされた分野では，従来から危機管理が検討されてきたが，従来型の重要社会基盤である電力供給や鉄道などは，危機管理は行政あるいは当該サービスを提供する事業者が検討するものであって，それらのサービスを享受するだけの一般の利用者が自ら危機管理を検討したりそれを実施することはなかった．これに対して，新たに重要社会基盤となったインターネットにおいては，インターネットを構成する技術情報の多くが開示されていて，誰もがサービス提供側にも享受側にもなれることもあり，サービスの提供側と享受側という単純なモデルでは危機管理を検討できない．少くとも現状では，ネットワークサービスプロバイダだけでなく，利用者ごとにあるいは利用者が所属する組織ごとに検討しておかなければならない事項が多数ある．著者らは，この考え方に基づき，現在のインターネットに迫る脅威を分析して対処方法を検討し，「インターネットの危機管理機構」構築に向けた研究を続けている．本論文では，一連の研究の一部をなす，脆弱性データベースの構築とネットワークに及ぶ脅威のひとつである DDoS attack (分散型サービス不能攻撃) を検証するシステムについて論じる．

* Vulnerability Data Base and DDoS Attack Simulator
Hiroyuki Ohno (Communications Research Laboratory, MPHPT), Hiroshi Takechi (Yokogawa Electric Corporation), Hideki Nagashima (Yokogawa Electric Corporation).

2 現状と問題点

インターネットが実社会のさまざまな実用的なアプリケーションの基盤として利用されるようになり，重要な社会基盤として認知されるに至ったが，上述したように他の社会基盤とは異なる状況もあって，インターネットのセキュリティに対する関心は，一般利用者の間でもここ数年の間に多いに大きく高まってきた．加えて，昨今の国内外における不正アクセス事例の多発，米国政府の「情報システム防御に関する国家計画」の発表，日本における不正アクセス禁止法の施行なども，インターネットのセキュリティに対する関心の一層の高まりの一助となっている．その結果，インターネット上のセキュリティに関する技術的な対策に関する情報や，さまざまなセキュリティ関連製品/サービスが数多く出てきている．

しかし，以下のような理由から，セキュリティ対策に関して，万全であるとは言えない．

- セキュリティ技術分野の複雑性・広がり・進歩性

  セキュリティを万全にするためには，システム全体にわたってある程度の水準をクリアする必要があり，そのための知識や技術は広範で多岐に及ぶ．また，新しい技術の出現に伴って新たなセキュリティも考慮する必要があり，新しい知識・技術を習得することは容易ではない．また，一人のセキュリティ担当の技術者のみで全てをカバーすることは不可能である．

- セキュリティ技術者不足

  セキュリティ技術者の数か少なく，全ての組織に十分なセキュリティ対策を行える人材がいるわけではない．

- 組織内でのセキュリティ対応体制の不整備

  セキュリティ対策を，組織全体で対応している組織は少なく，多くの場合が，その組織のシステム管理者個人の考え方に依存している場合が多い．これは，セキュリティ対策が企業などの組織が本来行うべき役割ではなく，いわば「余計な仕事」であることも影響している．

また，セキュリティに関しては，常に最新の情報に基づいて対応/対策を考える必要があり，情報収集およびその整理が重要になってくる．そのために，各方面でセキュリティに関する情報を収集し，アドバイザリといった形式で整理しセキュリティ対策を行う人の便宜を図る活動が行われている．しかし，これらについてもいくつかの問題点がある．

- 全情報を網羅することは不可能

  CERT のアドバイザリ [1] などセキュリティに対する情報ソースは数々あるが，一つで全てのセキュリティ対策をカバーできるわけではない．

- 組織間連携

  CVE(Common Vulnerabilities and Exposures)[2] により脆弱性データベースのデータ相互関連性を維持管理する活動は徐々に広まりつつあるが，同じような活動を行っている組織間で相互互換にするところまでは至っていない．セキュリィに関連する情報は膨大であり，組織間で協力しなければ，今後対応が難しくなっていくと予想される．

- 情報公開性の問題

  セキュリティの性格上，非公開とされている情報がかなり存在し，対策手段を考える上では情報が不十分である．

- 情報のローカライズ

  日本語で記述されたものは少なく，日本人が緊急な対応をする場合などには障害となることがある．

3 問題点の改善方法

以上のような現状と問題点をふまえ，今後のセキュリティ技術を向上させ，インターネットをより安全なものとしていくために，まず，各組織での対策を講じるために必要な情報を提供するデータベースの構築と，セキュリティの問題が起きた場合にできるだけ早くその問題に対抗する手段を見い出すための活動が必要である．

そのために，以下の活動を整備していかなければならない．

- さまざまなセキュリティ問題を専門的かつ専任で調査できる体制
- セキュリティ問題を再現し，その詳細を明らかにするためのシステム
- セキュリティに関するさまざまな情報を，さまざまな点から比較分析するための情報収集を行う
- 集めた情報を分類，整理し，データベースを作成

本研究では，セキュリティ対抗手段研究システムについての研究にまず着手した．対抗手段を見いだすためには，以下のことができなければならない．

- 脅威についての情報を集めて分析すること
- 脅威を実際に再現して，その動作/影響などの詳細を知ること
- 効果的な対抗手段を検討し，実際にその効果を計測すること

そのためのシステムとして，以下が必要である．

- セキュリティテスト環境
- セキュリティデータベース

3.1 セキュリティテスト環境に求められる要件

セキュリティテスト環境は，各種セキュリティの問題を再現し，対抗手段の有効性を実証する環境であり，以下の機能が求められる．

- 現実のネットワークトポロジやシステムの環境を容易に再現できる機能

  さまざまな形態の現実のシステムを再現するためには，ネットワークトポロジの再構築容易性と柔軟性が必要で，さらに，各種システムソフトウェアの変更容易性も必要である．

- 柔軟な計測機能

プローブ周期といったパラメータや測定項目などの変更が観測中にでき，計測対象のシステム系に悪影響を与えずに変更できなければならない．

- テスト結果のアーカイブと閲覧機能

テストの結果が保存されなければならず，過去のテスト記録の検索・比較などが行いやすいよう，テスト仕様書，テスト報告書などの書式などが定められていなければならない．

3.2 セキュリティデータベースに求められる要件

セキュリティデータベースは，他のセキュリティ関連情報へのポインタも含め，セキュリティに関する情報が網羅的に収納されていなければならず，必要な情報を絞り込んだり，関連付けもできる，セキュリティ情報ポータルサイト的な機能が必要がある．また，セキュリティ情報のうち，重要な情報の1つである脆弱性情報については，網羅的であるとともに，情報の詳細さも，有効な対抗手段を考える上では必要となる．

また，以下のような情報を持つ必要がある．

- 脆弱性についての情報
- 各種研究機関の情報
- さまざまなセキュリティに関連した URL の情報
- クラッカたちが取り交わしている情報

3.3 今回のアプローチ

今回は，以下の二つのアプローチを実施した．

まず，インターネットにおけるさまざまな脅威の中で，最近話題に上ることの多い分散型サービス不能(DDoS)攻撃 [3] のシミュレーション環境を構築し，各種 DDoS 攻撃ツール [4,7] を動作させ，その挙動と攻撃された場合のシステムの状況について計測し，詳細を調査した．

次にセキュリティデータベースの大きな部分を占める脆弱性情報について，どのような形にまとめれば良いかを検討し，実際にインターネット上で流通している各種データベース [5] から脆弱性情報を収集し，脆弱性データベースとしてまとめた．

4 DDoS シミュレータおよび脆弱性データベースの設計と実装

DDoS シミュレータ，脆弱性データベースとも OS に依存しない動作を共通の前提条件として設計を行ない，実装した．

4.1 DDoS シミュレータへの要求

DDoS ツールは 3 階層或いは 2 階層からなる Attacker 構成を形成する．また Attacker の規模の大小が攻撃の成否に関わるツールもあり，DDoS 攻撃を正確に模擬する為にはその階層と規模を任意に構成できる環境が必要となる．また，本システムでは DDoS ツールはインターネットなどを通じ入手することを基本としており，ツールを修正せずネイティブな環境で実行することも求められた．

一方，DDoS 攻撃によってサービス不能に陥る状況を正確に把握するために，攻撃条件を容易に変更でき，Victim の影響を正確に把握することが求められた．また，攻撃条件による実験成否（サービス不能の成否）がリアルタイムに把握でき，また比較検証するため実験毎のデータ保存が必要とされた．

DDoS 攻撃については，ベンダーから対抗手段がいくつか提供されているが，この有効性についても正確に評価できることも本システムでは求められた．

4.2 DDoS シミュレータの設計

DDoS ツールはツール毎にソフトウエア的見地での完成度が異なる．DDoS シミュレータではソフトウエア的に未熟な DDoS ツールも管理できるように，管理デーモンを開発し，このデーモン上で DDoS ツールを実行させる事とし，DDoS ツールにあわせ OS 非依存で実行できる事とした．

また，DDoS ツールをツール毎にネイティブな環境で実行させる為に，OS を含む実行環境を容易に変更可能な設計とした．

さらに，DDoS 攻撃を受けた際の Victim 側の状況をネットワークを経由し計測・収集するが，その際に攻撃トラフィックの影響を排除することも考慮した．

4.3 DDoS シミュレータの実装

- Attacker 部

Attacker 部を構成する AttckerPC は，DDoS の規模のシミュレーションを行うために相当数が必要となるので，省スペース化のためにラックマウント可能な 2UnitHeight の PC を選定した．またこの PC のディスクは OS を含む実行環境の迅速で容易な変更を可能とするために，ホットスワップ対応ディスクを実装した．OS 非依存で DDoS ツールの管理可能とするため，JAVA により DDoS ツールコントロールデーモンを作成し，コントロールデーモンが XML で記述された各

AttackerPC の設定条件を管理 PC から読み込むことにより Attcker 部全体の動作の集中管理を行っている．また，このコントロールデーモンを使い 1 台の PC 上で DDoS Agent を複数動作させることができ，実台数以上の構成での DDoS シミュレーションが可能となっている．

DDoS 攻撃化の Victim 部の状態はネットワーク経由で計測用の PC に収集する．データ収集時に DDoS 攻撃のトラフィックの影響を受けないために，DDoS 攻撃を受けるネットワークとデータ収集のためのネットワークを分離し，Victim 部を構成する全てのサーバにはそれぞれのネットワークにインターフェースを用意した．データ収集は SNMP を基本としたが，DDoS 攻撃の状況を把握する上で SNMP MIB で対応できない部分については専用の Agent を作成し，収集を行った．収集したデータは計測対象毎にリアルタイムで表示し，実験成否の判断が即座に行えるようにした．また同時に全ての DDoS 攻撃パケットをキャプチャし，MAC アドレス-IP アドレス対応表示を行い，IP スプーフィング状況を観測可能とした．収集したデータは実験終了時に，タイムスパンやスケール調整を行った後，CSV 形式のファイルとして保存し，表計算ソフトウエアなどによる解析を可能としている．

4.4 脆弱性データベースへの要求

脆弱性データベースは情報の急速な増加に対処できるかが重要である．CVE は，インターネット上に存在する各種の脆弱性データベースのレコードに共通の命名を 1999 年から行っており，総件数は約 1000 件である，そのうちの 350 件が最近 3ヶ月のものである．また新規の攻撃が発生した場合，既存の脆弱性データに類似の攻撃が無いかといった一刻を争う検索が行われるため，これに耐えられる検索性能が必要となる．さらにこの時，具体的な脆弱性のソースコードや攻撃方法のソースコードなどが無ければ対処方法の早急な確立はできないが，インターネットに公開されている情報そのままでは，ソースコードが無かったり，検索性が悪かったりする．そして実際の攻撃の際，あるいはそれに備える際に，これらの情報がどこにでも持ち運べることが組織的，組織間にまたがる防御に役立つ．

4.5 脆弱性データベースの設計

脆弱性データベースは，作成／更新／利用／破棄といった脆弱性データのライフサイクルを管理するワークフローを考慮した構造とインタフェースを設計し，これにより組織内，組織間でのコラボレーションが可能になった．コンテンツを日本語化し，検索手段も複数提供する上，どこにでも持ち運べるようにコンテンツを XML で出力できるので，他のアプリケーションとの連携や独自の検索方法を使用することも可能になり，攻撃が現在進行中でも検索，レポーティングが可能になった．また脆弱性データベースには脆弱性のソースコードと攻撃のソースコードを含めたので，攻撃の兆候・足跡を調査することで，対策の確立が可能になった．

4.6 脆弱性データベースの実装

- データベース部

データベース部は WindowsNT 4.0 Server 上で Oracle Workgroup Server R8.0.5，すなわち RDB を利用して実装しているため，SQL を使って直接検索する事も可能になっており，また，インターネットに流通する情報を日本語化して格納できるようになっている．日本語化に際してはアプリケーションなどから出力されるメッセージや操作を日本語版の物にして格納するようになっている．また脆弱性データとしては具体的な攻撃ソースコードまで格納できるようになっている．

- アプリケーションサーバ部

データベース部とデータベース部への情報の作成／更新／利用／破棄を行うためのインタフェースとしてのアプリケーションサーバ部は分離し，障害隔離とスケーラビリティを実現している．WindowsNT 4.0 Server 上で Internet Information Server 4.0，すなわち WWW サーバを利用して実装している．キーワード検索（項目毎）部分と更新部分は VisualBasic 6.0 で CGI として実装し，全文検索（曖昧検索）部分は Namazu を利用して実装した．検索結果は XML で出力され，更新部分では，XML のファイルをアップロードできるように，情報の送受信手段を XML over HTTP で構築した．

- クライアント部

データベース部からアプリケーションサーバ部を経由して XML で出力した脆弱性データは，クライアント部に保存しておく事により，オフラインで情報を扱うこともできる．独自のアプリケーションや検索方法でこれを利用する事も可能であるが，今回は，VisualBasic 6.0 で Windows 上にオフラインビューアを実装した．また VisualBasic 6.0 で Windows 上に脆弱性データを編集し，XML で編集結果を出力できる，オフラインエ

ディタを実装した．これを利用して，編集した脆弱性データをデータベースに反映する事もできる．

5 実験と評価

図 1 に示すようなシミュレーションを構築し，実際に数種類の DDoS ツールを実行し，Victim として構成した Web システムがどのようにサービス不能状態に陥るかを検証した．また，DDoS 攻撃に対する IDS の挙動やベンダーが提供している DDoS 攻撃対応手段についても，その評価を行った．

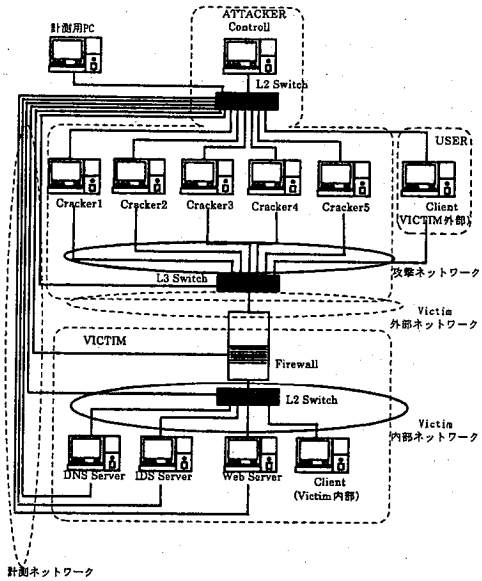

図 1: 実験構成図

5.1 mstream による TCP syn flood 攻撃

DDoS ツール「mstream」[6] による TCP ポート 80 への TCP syn flood 攻撃の結果を示す．図 2 はファイアウォールマシンの負荷状況を示し，DDoS 攻撃に対して特段影響が及んでいないことがわかる．図 3 は Attacker から Victim のファイアウォールに送信されたパケット量を示している．送信量は 38000 オクテット程度でありネットワーク帯域としては十分な余裕がある．図 4 は攻撃を受けている Vicitim 内 Web サーバに対して善意のユーザからのアクセス状況を示している．DDoS 攻撃が開始された直後から http に対する応答はまったくなくなっていることがわかる．一方で Web サーバに対する ping の応答，Victim 内 DNS サーバに対する応答は問題はなく，「mstream」による TCP syn flood 攻撃が特定のポートには致命的なダメージを与えてながらも他には影響を殆ど与えないことがわかる．

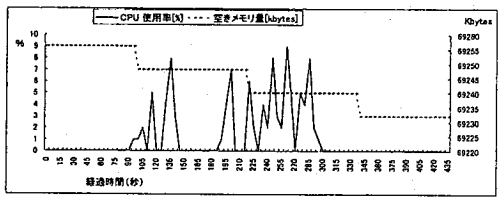

図 2: ファイアウォールマシンの負荷状況 (縦軸:負荷，横軸:経過時間)

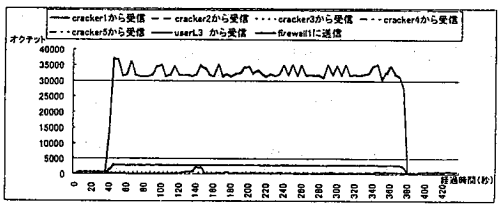

図 3: Attacker からファイアウォールに送信されたパケット量 (縦軸:パケット量，横軸:経過時間)

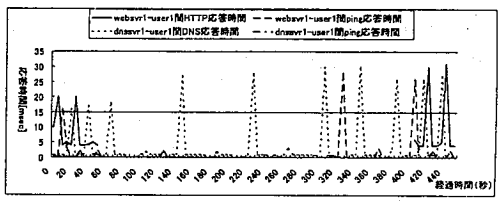

図 4: 善意のユーザーからのアクセス状況 (縦軸:応答時間，横軸:経過時間)

5.2 Syn Defender による TCP syn flood 対抗

次に DDoS ツール「mstream」による TCP ポート 80 への TCP syn flood 攻撃を行い，ファイアウォールソフトウエアである Firewall-1 が提供する TCP syn flood 対抗機能，Syn Defender を機能させた場合の結果を示す．図 5 はファイアウォールマシンの負荷状況が示されている．これによると Syn Defender を機能させた場合，メモリの消費量が大きくなることが観測される．図 6 は攻撃を受けている Vicitim 内 Web サーバに対し

て善意のユーザからのアクセス状況を示している．これによるとSyn Defenderの機能が有効に働き，善意のユーザからのhttpを含むリクエストに対して応答が得られてことがわかる．ただし，httpの応答が750msecとなる状況も観測され，影響が皆無ではないこともわかる．

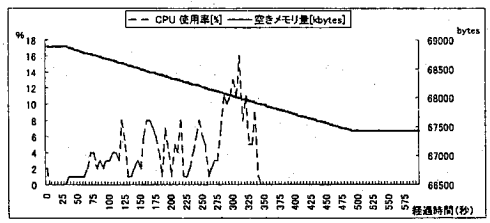

図 5: ファイアウォールマシンの負荷状況 (Syn Defender, 縦軸:負荷，横軸:経過時間)

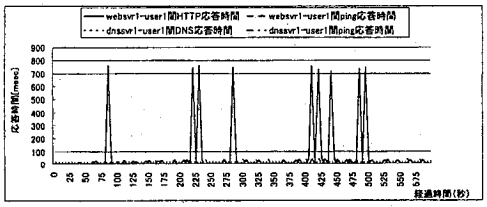

図 6: 善意のユーザのアクセス状況 (Syn Defender, 縦軸:応答時間，横軸:経過時間)

6 考察

脆弱性データベースは，今回のシステムにより情報の更新，検索，持ち運びが容易に行えることが確認された．今回の実装で，XML形式の脆弱性情報の持ち運び自体はプラットフォーム非依存となっていたが，ビューア，エディタ自体はプラットフォーム依存となっていた．プラットフォーム非依存なビューア，エディタの開発が当面の課題である．また次の課題として，DDoSシミュレータと脆弱性データベース連動，さらにはセキュリティテスト環境とセキュリティデータベースの連動によるセキュリティ情報収集と検証の更なる高速化が挙げられる．

DDoSシミュレータは，今回のシステムによりDDoSの動作を模擬し，影響を正確に観察することが可能であることが確認された．本システムではAttackerPC上にDDoS Handlerや複数のAgentをシミュレートする事により，限られたAttackerPC上でもスケーラブルにDDoS環境の構築が可能な事を一つの特徴としているが，この場合，Handler-Agent間に擬似のDDoS停止信号を送るタイプのDDoSカウンターツールの動作検証ができないことが判明している．スケーラブルな環境でかつDDoSツール間のパケットのやり取りやカウンターツールの検証可能な機構の検討は課題の一つである．

7 あとがき

本研究では，脆弱性データベースの整備とDDoSシミュレータを構築を行ない，これらの有効性の確認を行なった．この研究により，今後予想されるインターネットへの脅威に対して，ある程度の対抗力を備えることができたと言える．しかし，脆弱性データベースにもDDoSシミュレータにも考察で述べたような課題が残っているし，インターネットへの脅威に対して，なすべきことはまだ多数ある．今後も引続き研究を続け，インターネットの危機管理機構を整備し続ける必要がある．

8 参考文献

[1] CERT/CC Advisories, http://www.cert.org/advisories/

[2] CVE(Common Vulnerabilities and Exposures), http://cve.mitre.org/

[3] "DDoS: Is There Really a Threat?," USENIX Security Symposium, August 16, 2000. http://staff.washington.edu/dittrich/talks/sec2000/

[4] Distributed Denial of Service (DDoS) Attacks/tools, http://staff.washington.edu/dittrich/misc/ddos/

[5] CVE-Compatible Products, http://cve.mitre.org/compatible/

[6] The "mstream" distributed denial of service attack tool, by David Dittrich, George Weaver, Sven Dietrich, and Neil Long , http://staff.washington.edu/dittrich/misc/mstream.analysis.txt

[7] Analyzing Distributed Denial of Service Tools: The Shaft Case, by Sven Dietrich, Neil Long, and David Dittrich. In Proceedings of USENIX LISA 2000, December 2000.