社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2006－DD－56（8）
2006／7／28

Web技術を悪用する攻撃に対するサーバ側セキュリティ要件

松永 豊† 大場 みち子‡

† 東京エレクトロン株式会社コンピュータ・ネットワーク事業部 〒107-8481 東京都港区赤坂 5-3-6
‡ 株式会社日立製作所ソフトウェア事業部 〒140-8573 東京都品川区南大井 6-26-2
E-mail: † matsu@kabuki.tel.co.jp, ‡ michiko.oba.cq@hitachi.com

あらまし Web 技術を悪用したサーバ攻撃が急増しており，サイト閉鎖や個人情報漏洩，金銭的被害が実際に発生している．Web フォームに対する SQL インジェクションや，PHP モジュールに対する XML を使った手法が多く報告されてきたが，最近は例えば Ajax における Java Script を利用した攻撃が現れている．Web 2.0 といった言葉に代表される開発手法やビジネスモデルの発展に伴い，今後さらに新しい技術を悪用した不正行為の増加が懸念されている．攻撃手法の進化により従来のセキュリティ技術では防御できなくなっているサーバへの攻撃手法を調査し，これらの攻撃を防御するための技術要件を検討した．

キーワード セキュリティ，Web 2.0，SQL インジェクション，クロス・サイト・スクリプティング，イベント相関，データベース・セキュリティ，Ajax，RSS，SOAP，REST，個人情報漏洩，ID 盗難

Server-side security requirements against the Web technology threats

Yutaka MATSUNAGA† Michiko OBA‡

† Computer Network Division, Tokyo Electron Ltd. 5-3-6 Akasaka, Minato-ku, Tokyo, 107-8481 Japan
‡ 6-26-2, Minamioi, Shinagawa-ku, Tokyo, 140-8573 Japan
E-mail: † matsu@kabuki.tel.co.jp, ‡ michiko.oba.cq@hitachi.com

Abstract The number of the attacks to the Web servers exploiting Web technologies is increasing. These attacks have caused forced site-closure, personal information leakage, and economic damage. These attacks were using SQL injection to the Web forms and unexpected commands in XML data to the PHP modules, but recently some incidents shows the exploit of JavaScript in Ajax applications. As the new development method and business models described by the words like Web 2.0 become ubiquitous, it is anticipated that the attacks exploiting such new technologies to increase. This paper will outline the latest attack methods to the servers that cannot be detected or mitigated by the traditional security technique, and discuss the technical requirements to protect against those attacks.

Keyword security, Web 2.0, SQL injection, cross site scripting, event correlation, database security, Ajax, RSS, SOAP, REST, personal information leakage, identity theft

1. はじめに

米 CSI/FBI の調査[1]によると，コンピュータ犯罪による被害は総件数が減少の傾向にあり，平均損失額も 2001 年をピークに減っているという．

しかし一方，大量の個人情報漏洩に代表される大規模なコンピュータ犯罪は続々と発生しており，最近でも 2006 年 5 月に米国政府機関から最大 2600 万件，米国ホテル予約サイトから 24 万件の個人情報漏洩，また日本国内でも 2006 年 6 月に通信会社で 400 万件の顧客情報漏洩などが発覚している．CSI/FBI の調査でも，全体的な損失が減少しているにもかかわらず，情報への不正アクセスと個人情報の盗難による損失は増加している事が報告されている．こうした大規模犯罪の増加は，情報システムを狙う攻撃の性格がいたずら的なものから実利を得るための計画的・組織的なものに変化している事を表している．

その対策面を見ると，攻撃の性格の変化に追随した効果的な対策は今までのところ行われていないのが実情である．前述の漏洩事件の中でも，それぞれの情報流出経路や原因はさまざまである．ただし大量の機密情報は通常データベース・サーバなどのサーバに保管されており，漏洩においてもまず何らかの形でサーバからのデータ取得が行われているわけで，こうしたデータ盗難犯罪の防止にはサーバへのデータアクセス制御が最も重要な対策となる．CSI/FBI の調査でも，情報盗難増加の傾向とあわせて，サーバが稼動する Web サイトでのセキュリティ犯罪が劇的に増加していると報告されている．

Webサイトを攻撃するための手段としては，従来のWebアプリケーションへの攻撃に加えて，新たな開発技術を悪用する攻撃が目立ち始めている．新たな技術が使われ始めるたびに，それらが持つ脆弱性やセキュリティまで念頭に置いた開発手法の不在などのために，攻撃者の格好の標的となるからである．

本稿では，最近のWebサイト攻撃に見られる脅威の傾向や攻撃手段の情報について，実例を含めて調査した結果を報告し，それらを防御するためのセキュリティ対策を提案する．セキュリティ対策については，今まで手薄だったと思われるシステム上の対策におけるアプリケーション層の防御を中心に提案する．

2. 関連技術の動向

2.1. Webを狙う脅威の動向

かつてWebサーバを狙う攻撃と言えば，ホームページを書き換えるなどの他愛ないいたずらや，DoS (Denial of Service = サービス停止)攻撃による迷惑行為などであった．しかし最近は実利を求める攻撃が増加し，個人情報の不正入手やアプリケーションの不正操作を目的にインターネットからWebサーバを攻撃し，実害をもたらす例が増加している．

具体的な手段としては，SQLインジェクションやクロス・サイト・スクリプティング(XSS)といった従来から知られている手法が多く使われているが，これらの攻撃を可能にする脆弱性の発見や攻撃行為に使える自動化ツールが出回っており，攻撃の実施を容易にしている[2]．

これらの攻撃手法については防御の方法も良く知られており[3]，対策が進んでいるが，被害の報告も相次いでおり，特にSQLインジェクションによる攻撃の被害は重大化している．これは，攻撃手法が進化し，2.3項で説明する新たな脅威が出てきているためである．

2.2. Web開発技術の動向

Web 2.0と総称される新たなネットワーク利用形態の拡大に伴い，アプリケーション開発において新たな技術が使われている[4]．Perl, PHP, JavaScriptなどスクリプト言語の利用とAjaxに代表されるサーバ通信の柔軟化，RSSによる情報配信，APIによる部品としてのWebサービス公開，部品を組み合わせることで新たなサービスを作り上げるマッシュアップ，などである．

これらの開発技術が使われた結果，アプリケーションが新たな特徴を持つようになってきている．

- 柔軟な画面設計と更新
- サービスやその部品の境界の不明確化
- クライアント側での自動処理（スクリプトの高度な利用）
- 頻繁なサーバへのデータアクセス
- XMLによるデータ交換

これらの特徴の中で，攻撃の性質に影響を及ぼす部分としては，

- 画面設計，境界，自動処理などの変化によって，よりユーザに検知および予防されにくい攻撃が可能になってきていること
- データアクセスとデータ交換の変化によって，アプリケーションの表面（Web）だけでなくバックエンドのデータを狙いやすくなっていること

が挙げられる．

2.3. 新たな脅威の動向

単純な攻撃手段に対する対策が普及してきた結果，攻撃の方法は3つの方向に進化してきている．

1) 対策の裏をかく - 防御のためにフィルタされがちなキーワードを検知されない書式に偽装する，など．防御の手法が公知となるに従って，それをすり抜ける方法が編み出される．さらにそうした方法がサンプル・コードや自動化ツールに組み込まれ，短期間に普及する．
2) 新たな技術を使う - セキュリティ対策がされていない新たな技術や，新たに見つかった攻撃手法など．特に前項で紹介したようなアプリケーション開発上の新技術は利便性を最優先にしている場合が多く，セキュリティ面では脆弱であることが珍しくない．
3) 特定の対象を狙う - 以前は侵入行為にしろワームのような自動ソフトウェアにしろ，脆弱なサイトならどこでも狙うといった汎用的なものが多かったが，特定の対象を狙う傾向が強まっている．価値の高いデータが存在する対象，攻撃に都合のいい機能を提供している対象，などが選択され，その対象に特化した攻撃手法が使用される．

こうした新しい脅威に対抗するためのセキュリティ対策としては,現れてくる新しい脅威に対する防御策を継続的に適用するだけでなく，未知の攻撃に対する抵抗力も持てるような，そしてそのサイトに特有な機能を保護できるような，総合的な対策を施すことが必要になってくる．

3. 新たな脅威の例

3.1. XML-RPC for PHPの脆弱性

PHPベースのさまざまなアプリケーションで使われているソフトウェア・コンポーネントXML-RPC for

PHP[5]に脆弱性が見つかっており[6]，これを悪用したワームが複数報告されている．

2005年11月に報告されたLupperは，Webサーバを無差別にスキャンし，脆弱なスクリプトがインストールされている場合にこれを攻撃して感染する[7]．感染後はバックドアのプログラムをインストールし，外部からサーバを操る事が可能になる．

この例で読み取れる新たな脅威の特徴は，

1) ワームのデータとしてXMLが使われ，XMLデータの中にコマンド呼び出しなどの命令が埋め込まれていることと，
2) ソフトウェア部品の脆弱性を利用し，多数のアプリケーションに影響を与えたこと，

が挙げられる．Lupperが使用したXMLデータは次のようなものである[8]．

```
<?xml version="1.0"?>
<methodCall>
<methodName>test.method</methodName>
<params><param><value>
<name>',''));echo  '_begin_';echo  `cd
  /tmp;wget xx.xx.193.244/lupii;chmod +x
  lupii;./lupii  xx.xx.193.244  `;echo
  '_end_';exit;/*</name>
</value></param></params>
</methodCall>
```

(見易さのために改行は著者が編集した．)

この例からも分かるように，最新の技術を使う不正ソフトウェアを遮断するためには，XMLデータ中に含まれるコマンド文字列を検知する機能が必要になる．またLupperの場合は，とくにXML-RPCがLinuxのパッケージやWiki，コンテンツ管理，グループウェアなど多数のアプリケーションで使われていたことから影響範囲が大きくなり，修正ソフトウェアの提供も各アプリケーションの開発体制に依存してまちまちになってしまったことが特徴的だった．今後，オープンソースを活用したソフトウェアの部品化が進むにつれ，同様の危険性が高まることが予想される．

3.2. ユーザの個人情報を盗むワームMW.Orc

2006年6月，セキュリティ製品を開発するFaceTime Security社が報告した[9]ワーム．ソーシャル・ネットワーキング・サービス(SNS)の掲示板を利用して，当該SNSユーザの間で拡散した．このワームはJPEG画像ファイルに偽装した実行モジュールの形で配布され，ユーザのコンピュータに2つのファイルを追加で自動インストールする．そしてそのユーザがMy Computerにアクセスしたタイミングで，ユーザが利用している銀行サイトの情報やパスワード情報を外部に電子メールで送信する．

このワームは，会員制サイトなら安全，というユーザの想定を裏切り，その会員制の機能を悪用する新たな形態をとった脅威の例である．

3.3. Samyによる Ajax ワーム

2005年10月にソーシャル・ネットワーキング・サービス(SNS)のユーザ「Samy」が，どうサービス内で他のユーザのデータを改竄し増殖していくプログラムを開発・使用した[10]．

SNSサイトのユーザプロファイルへコードを挿入し，それを見た他ユーザの環境でXMLHttpRequestを含むスクリプトを実行し，自分へのリンクを埋め込む．さらに被害にあったユーザのプロファイルにもコードが挿入され，幾何級数的に拡散，一日以内に百万件以上のリンクを獲得した（図1）．このコードでは，「javascript」の文字列を隠蔽するために文字列を細工したり，スクリプトのタグ除去を回避するためにスタイルシートの中に命令を混入するなど，対策の裏をかく手法が使われている．さらに，Ajaxの中心部ともいえるXMLHttpRequestを利用してサーバへの通信を行ってデータの不正取得と改竄を行っている．

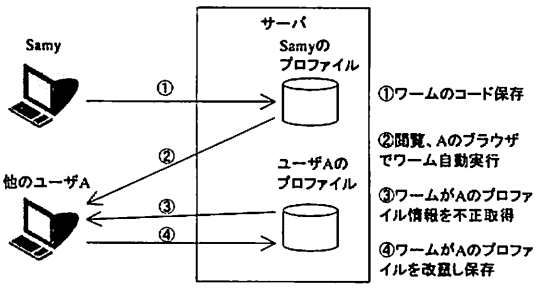

図 1 Samyによる Ajax ワームの動き

Fig.1 Ajax worm by Samy

この事件の被害としては，SNSサイトの「お友達」リンクを不正に多数取得した，ということに留まったが，本質的に重大な犯罪に発展する危険性を孕んでいる．このワームは，図1で示したように，サーバのデータベース中で本来アクセス権限のないデータに不正アクセスし，改竄している．しかも一連の不正なデータ操作は犯人とは別のユーザのデスクトップから行われ，そのユーザに対してデータ操作実行の事実を隠蔽することも容易である．従って同様の手法を利用して，大規模な個人情報の不正取得や金銭的な不正行為を秘密裏に行うことが可能である．

3.4. 公開サービスの脆弱性を突いたワーム Yamanner

2006 年 6 月中旬に発生したワーム Yamanner は，商用ポータルサイトのメールサービス上でスクリプト（JavaScript）を埋め込んだ HTML メールを表示することによって実行され，ユーザのアドレス帳にあるアドレスへワーム付きのメールを送付，取得したアドレスのリストを外部サーバへ送信する[11]．スクリプトを実行させる際に，ポータルサイトが標準提供している画像アップロード・サービスのスクリプトに存在した脆弱性を利用しており，やはり第三者ユーザの権限でそのユーザのデータを不正取得している．

この事件では，世界でも最大クラスのユーザ数を持つポータルサイトをピンポイントで狙っており，特定の対象を狙う新しい脅威の特徴が表れている．

3.5. RSS の危険性

ブログやニュースサイトなどの情報配信に利用される RSS に関しては，今まで特に攻撃例は報告されていない．しかしその柔軟性と普及度において，重大な攻撃につながる危険性が指摘されている．Trend Micro 社の David Sancho は，近い将来，RSS フィード・ハイジャックという手法を使ったワームが現れることを予見している[12]．この手法では，RSS フィードを購読するクライアント・ソフトウェアの参照先を変更することにより，悪意のあるサイトに接続させる．

さらに RSS には enclosure というフィールドが用意されており，バイナリ・データを格納しファイルタイプによりアプリケーションと関連付けできる．ハイジャックと合わせて悪用することにより，任意のプログラムをクライアント上で実行することが可能になる．

今後サーバ間における情報交換を RSS で行うようになると，同様の仕組みでサーバ内のデータを狙う攻撃に利用される可能性がある．

3.6. 新しいプロトコルに対する脅威

Web サービスで使われる SOAP, REST などの通信プロトコルも次第にセキュリティ面の考慮が必要になってきている．これらのプロトコルは主に http 上で使われ，ファイアウォールを通過して Web サーバに到達することを前提にしているため，インターネットを介した複雑な通信を実装するのに好都合である．ただしそのことはそのまま，悪意のある行為やプログラムに利用されやすいことにつながる．セキュリティ専門家の Bruce Schneier は，個人で発行するニュースレターの中で SOAP の危険性を指摘している．

「アプリケーション間でコマンドをやり取りしようとしても，うるさいファイアウォールが邪魔をしてしまう．しかし SOAP を使えば HTTP の中にコマンドを隠し，ファイアウォールに見つからないようにしてしまえるわけだ．」[13]

また SOAP を使った Web サービスは，入力フォームを使った攻撃と同様に(但しはるかに無警戒の状況で)SQL インジェクションやクロス・サイト・スクリプティングを仕掛けられるという報告もある[14]．この報告では，SOAP リクエストを使ってメッセージのサニタイジングの有無と程度，Web サービスの背後にデータベースが連結しているかどうか，SQL の使われ方，などを探る偵察行為の例を解説している．さらに得られた情報を利用して背後のデータベースからデータを引き出す過程をデモンストレーションしている．

このように SOAP はデータベースに対する新たな入り口を提供するだけでなく，XML の中に操作が隠れるため監視が難しく，そもそも SOAP が送られる POST リクエストが記録されないケースも多いため，事後の監査も難しい．

その点 REST の場合には各種インジェクションの手段として狙われる点は SOAP と同様であるが，HTTP リクエスト自体のメソッドで操作内容の使い分けもでき，各種パラメータも HTTP のパラメータとして渡されるため，入力の検査やメッセージの監視や記録は比較的容易である．但し，SOAP の場合には WS-Security などの標準規格で暗号化や認証などの手法が確立されているのに対して，REST にはセキュリティの標準規格は存在しない状態である．

SOAP, REST いずれの場合も，SQL インジェクションやクロス・サイト・スクリプティングといった従来の Web フォームに対する手法が応用されることを念頭に置き，SOAP メッセージ中の XML データや HTTP リクエストのパラメータを詳細にチェックすることが必要となる．

4. Web 2.0 時代のサーバ側セキュリティ要件

4.1. セキュリティ対策の種類

今まで見てきたように，セキュリティ対策を回避したい攻撃者の意図と，加速する開発技術の進歩があいまって，今までとは異なる脅威が現れている．こうした傾向に合わせてセキュリティ対策も見直していく必要がある．特に最近の脅威の傾向に効果的に対処するためには，次のような点に留意する必要がある．

- アプリケーション・レベルの攻撃に対する防御
- 会員制サイトなど環境に依存した攻撃に対する防御
- データを狙う攻撃に対応するデータベースの防御

これらの防御を実現していくためには，1) アプリケ

ーションの開発, 2) 開発後の検査, 3) 運用時のシステム, の 3 段階にわたってセキュリティ対策を施すことが不可欠となる. この 3 つの段階の中では, 開発段階と検査段階については, 今までもアプリケーション・レベルの攻撃を想定した対策が取られてきたので, そうした対策の増強を行うことになる. しかしシステム上の対策についてはこれまでネットワーク層の対策が主だったため, アプリケーション層の攻撃を検知・防御できる新しい手法も含めた検討が必要になる.

4.2. 開発上のセキュリティ対策

アプリケーション内でのセキュリティ対策としては, 入力される文字列の検査が重要である. 従来も Web アプリケーションのサニタイジングとして, 主に入力フォームに書き込まれる文字列の検査が行われてきた. しかし攻撃用の命令を挿入する経路が多様化してきた結果, 入力の検査もより幅広い対応が必要となってきている. たとえば, HTTP リクエストのパラメータ, SOAP メッセージの内容, クッキーの内容, スタイルシートの内容などである.

また, 禁止される文字列を複数に分割して挿入してくるような攻撃も確認されているため, 検査の方法も工夫する必要がある. たとえば Samy による Ajax ワームでは, xmlhttp.onreadystatechange という文字列が禁止事項として検査されていたため, 次のようなコーディングで回避していた[15].

```
eval('xmlhttp.onread' + 'ystatechange = callback');
```

このような回避策も想定した上での検査をする必要がある.

4.3. 脆弱性検査

アプリケーションの脆弱性を見つけるための検査も, より広範なテストが必要になる. セキュリティ・コンサルタントである Jaswinder S. Hayre と Jayasankar Kelath は, Ajax が脆弱性検査にもたらす課題として, 4 つの項目を挙げている[16].

1) 状態(state)が不明瞭 - ページの一部が動的に変更されることによって, 例えばリンクや入力フィールドが新たにできる可能性があり, そうした項目すべてについて脆弱性を検査する必要がある.
2) タイマーによるイベント - 例えば株価の定期更新のように, 定期的にサーバに要求を発信するアプリケーションもあり, 脆弱性検査はこうしたイベントも見つけてテストする必要がある.
3) 動的な更新 - 例えば入力欄に文字が入力されると同時に何らかの評価が行われ状態が変更されるようなきわめて動的なアプリケーションでは, 頻繁に発生するサーバとの通信をすべて把握し脆弱性を検査する必要がある.
4) XML データ - Ajax で使われる XMLHttpRequest は XML データで通信することがあり, 脆弱性検査では XML データへのコマンド挿入や XML の脆弱性も検査する必要がある.

このような課題が出てくる上に, 部分更新されるコントロールの多いアプリケーションでは, サーバ側のプログラムも多数用意されることになる. これらはすべて脆弱性の可能性があるサーバへの入り口となりうるので, 検査する必要がある.

4.4. 運用時のシステム対策

従来, Web システムにおけるセキュリティ対策としては, サーバの設定や, ネットワーク上でのファイアウォールや侵入検知システム (IPS) による検知・防御が一般的だった. しかし, 今までの対策の裏をかき, しかも新たに登場した技術を利用する攻撃が絶え間なく出てくる結果, システム上のセキュリティ対策への要求も, より高度化せざるを得ない. 主に考えなければいけない高度化の方法としては, 4 つの方向性がある.

1) ネガティブ・セキュリティだけではなく, ポジティブ・セキュリティの活用
2) 分散されてきたセキュリティ機能の一元化, あるいは一元管理化
3) データベースの対策
4) 多階層にわたる検査と統合検知

4.4.1. ポジティブ・セキュリティ

従来から行われているサニタイジング, フィルタリングなどの「悪い物を排除」するセキュリティ手法をネガティブ・セキュリティと呼ぶが, 悪用される技術が複雑かつ巧妙になるにつれ, ネガティブ・セキュリティでの対策は限界を迎えることが予想される. JavaScript のフィルタリングだけでも, 今まで実例で見てきたように, 悪意ある内容の検知は既にきわめて困難になっている.

この限界を克服するためには, 正しい通信を定義しそれ以外を排除するポジティブ・セキュリティの手法が必要になる. 原理的には, 対象のアプリケーションで受け取るべき正しい通信をモデル化し, そのモデルに合う通信のみ受け取るようにする. こうしたポジティブ・セキュリティによるモデルと, ネガティブ・セキュリティのルールを組み合わせることにより, 精度の高い検知が可能になる. また, ポジティブ・セキュリティの適用により, 未知の攻撃を遮断することも可

能になる．

課題としては，モデル構築の作業が個々のアプリケーションに依存すること，正しい通信の調査には大きな工数が必要なこと，アプリケーションの変更に追随する運用が必要なこと，モデルを完全に記述することは困難なため検知精度の向上が難しいこと，が挙げられる．これらの課題を克服するために，アプリケーション挙動の自動学習や，他の手法との連携による検知精度の向上などが試行されている．

4.5. 一元的なセキュリティ適用

古くから知られてきた SQL インジェクションといった攻撃による被害がなかなかなくならない背景としては，多数のアプリケーションや Web ページなどへの対策適用の徹底が難しいこと，アプリケーションが変更された場合のセキュリティ設定の追随が難しいこと，新たな攻撃バリエーションが見つかった場合の対策適用に時間がかかること，などがある．こうした課題を解決するためには，すべての Web ページやアプリケーションに対して，同一のセキュリティ対策を迅速に漏れなく適用することが必要になる．このためには，セキュリティ適用を一元的に行うことが効果的である．

具体的な方法としては，

- セキュリティ適用が必要な機能をライブラリあるいはサービス化して複数アプリケーションから共有
- 個々のサーバやセキュリティ機器のセキュリティ設定を一元管理するソフトウェア
- 通信が集中するネットワーク上でセキュリティ適用を行うゲートウェイ

などがある．

4.6. データベースのセキュリティ対策

g アプリケーションへの攻撃が多岐にわたるのに伴って，そのすべてを検査・保護することは次第に困難になってきている．データを保護する観点では，データベースを直接保護することが考えられる(図 2)．今までデータベース内データの保護としてはユーザ管理によるアクセス制御やデータの暗号化などがあったが，Web アプリケーションの性格を考えるとそういった技術より，SQL インジェクションなど代表的な攻撃手法を前提とした，より具体的な防御が必要になる．

こうしてデータベースを保護する技術としては大きく分けて 2 種類ある．

1) データベース・サーバ上で監視・保護するソフトウェア．
2) ネットワーク上でデータベースの前段で SQL を分析して監視・保護するセキュリティ・ゲートウェイ．

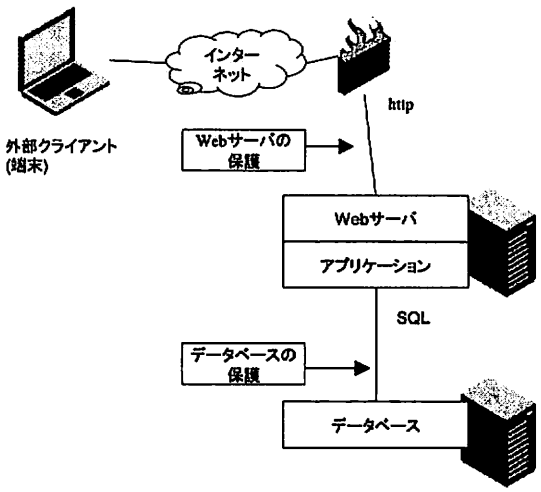

図 2 データベースの保護

Fig.2 Protecting database

前者はデータベースの稼動自体に悪影響を与える可能性があり，後者はサーバのコンソールからなど直接的なアクセスを検査できない．環境や要求に応じた技術の実装が必要になる．

4.7. 多階層にわたる検査と統合検知

対策の裏をかく攻撃手法が次々と生み出される環境においては，現れた脅威に対する対策を追加していくやり方では，常に後手に回ることになる．この点に対処するには，攻撃の現れる可能性がある階層すべてで検査を行うことで特定の階層における回避行動を無効にし，しかもそれらの検査結果を統合して分析することで，検知精度を上げることができる[17]．

Web サーバへの攻撃を想定した場合具体的には，http リクエストの挙動，HTML に埋め込まれる可能性があるスクリプト，XML データの内容，SQL 文の有無と内容，などの異なる階層で検査を行い，その結果を組み合わせて攻撃の判断を行うことが必要になる（図 2）．さらに，データベース保護機能の情報まで加えたり，ポジティブ・セキュリティとネガティブ・セキュリティの情報を組み合わせたりして相関処理することにより，特にデータベースへの不正アクセスについては極めて高い精度の保護機能を実現することができる．

こういった技術は具体的な実装としては，

- セキュリティ情報管理（SIM）と呼ばれるソフトウェアによる複数種類のセキュリティ機能

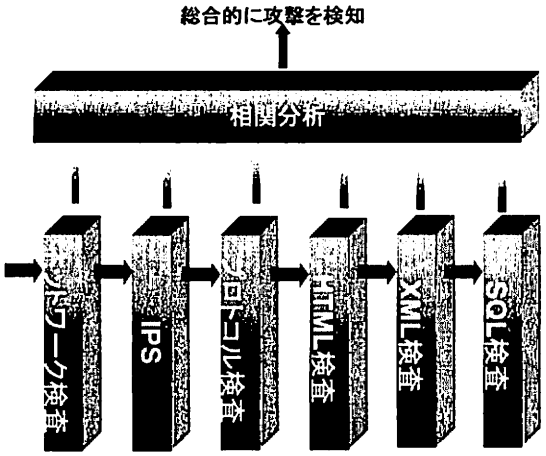

図 3 多階層での検査と相関分析

Fig.3 Multi-layer inspection and correlation

からの情報一元管理

- 複数のセキュリティ技術を集約したセキュリティ・ゲートウェイ

といった形で実現されている.

5. まとめ

高度化・悪質化するサーバへの攻撃について動向を報告し,対抗するためのセキュリティ対策を提案した. Web 2.0 という言葉に代表される利用技術の進歩に伴って, 攻撃手法も変化してきている. こうした中で Web システムに対するセキュリティ侵害に対処するためには, 従来のセキュリティ対策に加えて, ポジティブ・セキュリティの活用, セキュリティ機能の一元的な運用, データベースの保護, 多階層にわたる検査と相関分析が有効である. これらを用いたセキュリティ実装も現れてきており, 今後そうした実装の活用方法がベスト・プラクティスとして確立されることが期待される.

文 献

[1] Lawrence A. Gordon et al., 2005 CSI/FBI Computer Crime And Security Survey, Computer Security Institute, 2005

[2] JSOC Analysis Team, SQL インジェクションレポート 緊急レポート, 2006
http://www.lac.co.jp/business/sns/intelligence/jsoc_report.html

[3] 独立行政法人 情報処理推進機構 セキュリティセンター, 安全なウェブサイトの作り方, 2006
http://www.ipa.go.jp/security/vuln/20060131_websecurity.html

[4] 森山徹, "Web 2.0 の波, 開発現場へ, "日経 SYSTEMS, no.157, pp.57-62, May 2006

[5] XML-RPC for PHP Homepage
http://phpxmlrpc.sourceforge.net/

[6] Security Focus, XML-RPC for PHP Remote Code Injection Vulnerability, June 29 2005
http://www.securityfocus.com/bid/14088

[7] Joris Evers, New worm targets Linux systems, CNet News.com, November 7 2005
http://news.com.com/2100-7349_3-5938475.html

[8] The SANS Institute, XML-RPC for PHP Vulnerability Attack, Handler's Diary, November 5 2005
http://isc.sans.org/diary.php?storyid=823

[9] FaceTime Communications, FaceTime Security Labs Warns Against Data-Theft Worm Targeting Google's Orkut, Press Release, June 19 2006

[10] Computerworld, "Teen uses worm to boost ratings on MySpace.com, "Computerworld, October 17 2005
http://www.computerworld.com/securitytopics/security/holes/story/0,10801,105484,00.html

[11] Charles Babcock, Yahoo Mail Worm May Be First Of Many As Ajax Proliferates, InformationWeek, June 19 2006
http://nwc.networkingpipeline.com/shared/article/showArticle.jhtml?articleId=189500361

[12] David Sancho, The Future of Bot Worms, Trend Micro, 2005
http://www.trendmicro.com/en/security/white-papers/overview.htm#bot-wp

[13] Bruce Schneier, SOAP, Crypto-Gram Newsletter, June 15 2000
http://www.schneier.com/crypto-gram-0006.html#SOAP

[14] Sacha Faust, SOAP Web Services Attacks, SPI Dynamics Inc. Whitepaper, 2005

[15] samy, Technical explanation of the MySpace worm, November 2005
http://namb.la/popular/tech.html

[16] Jaswinder S. Hayre and Jayasankar Kelath, Ajax security basics, SecurityFocus, June 19 2006
http://www.securityfocus.com/infocus/1868

[17] Eugene Schultz, Intrusion Detection Event Correlation, CSI NetSec 2006 Conference, June 2006