Vol.2010-DPS-142 No.10
Vol.2010-CSEC-48 No.10
2010/3/4

転送ファイルの構造を考慮したアノマリ型侵入検知システムの提案

鴨狩裕紀[†]　寺田真敏[†]　土居範久[†]

概要：ネットワーク経由による，マルウェアの侵入を検知する手段の 1 つとして，アノマリ型の侵入検知システム（IDS）がある．アノマリ型 IDS とは，正常時の手続きや動作を定義し，定義からものを異常として検知するシステムである．アノマリ型 IDS は主に，プロトコル異常に着目した検知手法であるため，マルウェアが正規の手続きによって侵入する場合，検知することができない．

本稿では，プロトコルに基づく異常検知を一歩進め，画像ファイルなどの拡張子を偽装することで利用者を騙し侵入する攻撃や，ファイル特有の機能を用いて攻撃する方法など，ファイルの構造に着目したアノマリ型 IDS を提案する．

A Proposal of File Anomaly-based Intrusion Detection System

Yuki Kamogari[†]　Masato Terada[†]　Norihisa Doi[†]

There is an anomaly-based intrusion detection system as one of the means to detect the invasion of the malware by way of the network. Anomaly-based intrusion detection system is a system that defines the procedure and the operation when it is normal, and detects the one from the definition as abnormality. When the malware invades because anomaly-based intrusion detection system is a detection chiefly technique for paying attention to an abnormal protocol by a regular procedure, it is not possible to detect it.

By this report, It proposes anomaly-based intrusion detection system that pays attention to the structure of the file like the method of attacking abnormal detection based on the protocol by cheating the user by going a step and camouflaging the extension of the picture file etc. and using a peculiar function to the invading attack and the file etc.

1. はじめに

インターネットの普及と共に，Web サイトを主体としたサービスを簡単に利用できるようになった．その反面，“いつも見ているホームページ”から知らない間に，マルウェア配信サイトに誘導され，マルウェア感染被害を発生させるなど脅威が増している． Web サービスを利用するにあたって，脅威の 1 つと考えられる事象が，悪意のあるファイル（マルウェア）を送り込むことを目的としている Web サイトの存在が挙げられる．Web サイトを利用したマルウェア感染は，利用者の不注意によるダウンロードによる侵入だけではなく，上述のように，改ざんされた Web サイトにアクセスした利用者を，マルウェアを頒布している Web サイトに誘導する攻撃方法など，多岐に渡ってきている．

マルウェアの侵入を検知する手段の 1 つとして侵入検知システム（Intrusion Detection system：IDS）が挙げられる．IDS における検知手法には，シグネチャ型とアノマリ型の 2 種類がある．シグネチャ型 IDS はマルウェアのパターンを特徴として保持し，検査対象ファイルと保持した特徴とを比較することで侵入を検知する方式である．シグネチャ型 IDS は既知のマルウェアに対しては検知が容易であるが，パターンのない未知のマルウェアを検知できない．また，新しいマルウェアが見つかるたびにパターンを作成しなければならないため非常に手間がかかる．一方，アノマリ型 IDS は，予め定義しておいた正常時の手続き動作と，マルウェアの侵入や，感染後の動作とを比較することで検知する方式である．アノマリ型 IDS はマルウェアのシグネチャに依存しないため，既知のマルウェアはだけでなく，未知のマルウェアに対しても有効である．

未知のマルウェアに対して有効であるアノマリ型 IDS だが，これまでのアノマリ型 IDS は，主に，プロトコル異常に着目した検知手法であった．このため，"いつも見ているホームページ"から知らない間に，マルウェア配信サイトに誘導され，マルウェア感染被害を発生させるような正規の手続きに基づく異常検知は苦手とする．

本稿では，プロトコルに基づく異常検知を一歩進め，画像ファイルなどの拡張子を偽装することで利用者を騙し侵入する攻撃や，ファイル特有の機能を用いて攻撃する方法など，ファイルの構造に着目したアノマリ型 IDS を提案する．また，提案方式を実装したシステムを用いた評価実験を通して，提案方式の有効性を示す．

2. ファイル構造に関する調査

本章ではファイル構造に関する調査結果について述べる．

[†]中央大学大学院 理工学研究科 中央大学大学院
Graduate School of Science Engineering, Chuo University

Vol.2010-DPS-142 No.10
Vol.2010-CSEC-48 No.10
2010/3/4

2.1 ファイル構造に着目することで検知可能な攻撃

（1）GIFAR[1][2]

GIFAR とは, 画像ファイルである GIF ファイルの末尾に, Java アーカイブである JAR ファイルを付加したファイルである．GIF ファイルはファイルの先頭から読み込まれるのに対し，JAR ファイルはファイルの末尾からファイルを読み込むので，同一ファイル内で混在が可能となっている．GIFAR の拡張子は“.gif”になっており，Web ブラウザでも正しく表示することができるため，攻撃を発見することが難しい．このため，危険なコードを仕込んだ JAR ファイルを，容易に送り込むことが可能となる．

（2）拡張子偽装[3]

画像ファイルに拡張子に偽装したトロイの木馬が，2007 年 12 月末頃から話題となった．拡張子は“jpg”などの画像ファイルではあるが，中身は JavaScript を埋め込んだファイルで，Internet Explorer（IE）などでアクセスすると攻撃を受けてしまう．IE は，デフォルトの設定で，拡張子ではなく，内容でファイルタイプを判断し実行しているため，拡張子の偽装が容易である．これにより利用者を騙してアクセスさせることができ，攻撃コードを含んだ JavaScript を実行させることが可能となる．

（3）悪意を持った JavaScript を埋め込んだ PDF[4]

Acrobat Reader の脆弱性を悪用するため, PDF ファイルに JavaScript コードを埋め込む攻撃が流行っている．悪意を持った JavaScript を埋め込んだ PDF ファイルは，開くだけで JavaScript が動作し，悪質なプログラムを実行する．PDF ファイルは，Web ブラウザから直接実行できるため, 攻撃のトリガとして非常に有効な手法となっている．

2.2 ファイルの構造

2.2.1 保存形式によるファイル構造の定義

ファイルの構造は，ファイルの保存形式から大きく 3 つに分けることができる．本稿では，便宜上次の 3 つの形式を定義する．

- サイズ参照型
- 終端子型
- 不定型

（1）サイズ参照型

サイズ参照型は，ブロックのサイズを取得し，順番に読み取れるようにした，ファイルの保存形式である．サイズ参照型の例として，GIF や JPEG，PNG などが挙げられ，次のような特徴を持っている．

- ブロックの先頭付近には，ブロックを識別するための文字列が存在する
- ブロックのサイズは固定なものと，可変なものが存在する
- ブロックの種類が特定できれば，ブロックのサイズが取得可能である

サイズ参照型は，ブロックの先頭に，自分がどの種類のブロックか示すための文字列を用意している．この文字列によって，ブロックの種類を特定し，サイズの取得をおこなう．ブロックには，サイズが固定であるブロックと，サイズが可変であるブロックが存在する．

サイズが固定であるブロックは，ブロックの種類が特定できればサイズの取得が可能である．サイズが固定であるブロックの読み取りを，図 1 に示す．

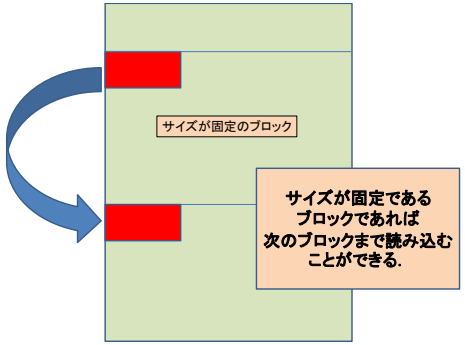

図 1　サイズが固定であるブロックの読み込み

サイズが可変であるブロックは，ブロックの種類を特定したあと，ブロック内部に存在するサイズを管理する場所からサイズを取得する．サイズが可変であるブロックの読み取りを，図 2 に示す．

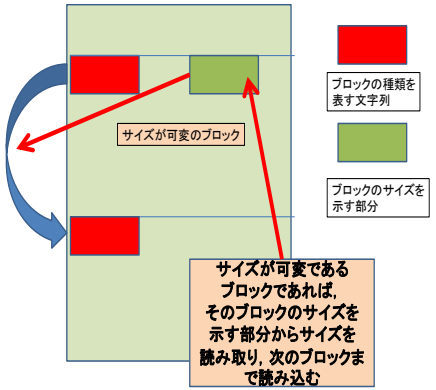

図 2　サイズが可変であるブロックの読み込み

Vol.2010-DPS-142 No.10
Vol.2010-CSEC-48 No.10
2010/3/4

（2）終端子型

終端子型は，ブロックの先頭と末尾に決められた文字列が存在し，決められた文字列によってブロックを判別することで読み込みをするファイルの保存形式である．終端子型の例として，PDF が挙げられ，次のような特徴を持っている．

- ブロックの先頭と末尾に決められた文字列が存在し，これによりブロックの判別が可能
- ブロックの長さは不定

終端子型は，ブロックの始まりを示す文字列と，ブロックの終わりを示す文字列の対によってブロックの特定をおこなう．終端子型のブロックの読み取りを，図 3.3 に示す．

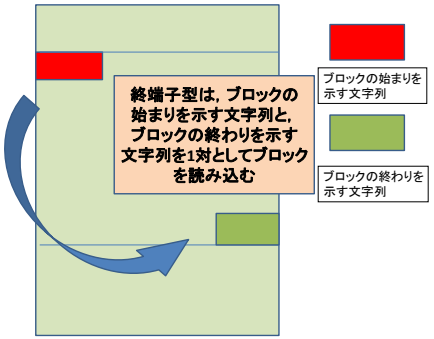

図 3　終端子型のファイルの読み取り

（3）不定型

不定型とは，ある特定の保存形式を持たないファイル保存形式である．例として，テキスト文やプログラムのコードなどが挙げられる．この型は，ファイルによってファイル構造が異なるため，ファイルの種類を特定することができない．

2.2.2 マジックナンバー

マジックナンバーとは，ファイルを読み取るときに，読み込むファイルがどの種類の保存形式か特定可能にするため，ファイルの先頭部分に付加される独特な文字列のことである．ファイルは通常，テキスト形式かバイナリ形式で保存されているので，その内容からファイルの種類を特定するのは困難である．そこで，ファイルの先頭に保存形式独自の文字列を付加することで，ファイルの保存形式を識別，読み込みが可能となる．ファイル保存形式とマジックナンバーの対応例を，表 1 に示す．

表 1　ファイル保存形式とマジックナンバーの対応例

ファイル保存形式	マジックナンバー
GIF	GIF89a もしくは GIF87a
JPEG(JIFI)	“FFD8”(バイナリ)
AVI(RIFF)	RIFFAVI
PDF	%PDF-1.2

3.　ファイル構造を考慮したアノマリ型侵入検知システム

提案するアノマリ型の侵入検知システムは，ファイルの保存形式を定義し，その定義した項目に関して異常と判定した場合には，攻撃が発生していると考える方式である．

3.1　概要

実装にあたっては，HTTP 通信用のプロキシサーバ上に提案手法の機能を実装した（図 4）．提案するシステムを構成する機能は，次の通りである．

- ファイル構造調査機能
- 異常ファイル処理機能
- アクセス解析機能
- URL フィルタリング機能

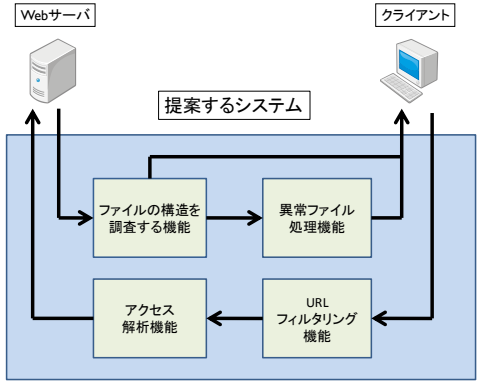

図 4 システム構成図

3.2　ファイル構造調査機能

この機能は，大きく分けて 2 つの項目について調査をおこない，マルウェアの侵入を検知する．調査項目は，次の通りである．

Vol.2010-DPS-142 No.10
Vol.2010-CSEC-48 No.10
2010/3/4

- ファイルの構造に異常がみられないか
- ファイルに提供されている特定の機能を使用していないか

異常なファイルを検知したら，異常ファイル処理機能により処理をする．異常ファイル処理機能については，3.3 節で詳細を述べる．

3.2.1 ファイルの保存形式に関する検査項目

提案方式の実装では，GIF ファイルと PDF ファイルを対象にファイル構造に関する検査項目と，異常に関する閾値を定義した．定義した検査項目を，表 2 に示す．

表 2 定義した検査項目

ファイルの保存形式	検査項目
GIF ファイル	拡張子
	ファイル構造
	ファイルサイズ
PDF ファイル	JavaScript の有無

今回，PDF ファイルのファイル構造に JavaScript の有無を検査項目として導入した理由は，次の通りである．

- PDF ファイルに JavaScript コードを埋め込む攻撃が流行っており、その脅威を低減する（2.3 節参照）
- PDF ファイルは，標準で JavaScript をサポートしているので，ファイル構造的には異常であるといえない．しかしインターネットに流通している PDF ファイルの多くに，JavaScript が使われていないとするならば，JavaScript を含む PDF をファイル構造上異常とみなすこともできる．筆者らの調査によれば，1373 個の Web サイトを対象に無作為に PDF をダウンロードして調査したところ，2665 ファイル中 1 つも JavaScript を使用した PDF ファイルを確認することはできなかった．

3.2.2 拡張子

ファイル拡張子の異常判定は，表 3 の通りである．送られてくるファイルの拡張子が”.gif”の事例で説明する．この場合 Content-Type は「image/gif」となり．このとき正規の GIF ファイルであるならば，GIF ファイルのマジックナンバーである「GIF89a」もしくは「GIF87a」がファイルの先頭に付加されている．一方，GIF ファイルのマジックナンバーがない場合，攻撃を意図したファイルである可能性を考えることができる．拡張子と送信データが一致する場合と，一致しない場合の処理概要を図 5，図 6 に示す．

表 3 ファイル拡張子の異常判定

項目	判定方法	異常時の処理
マジックナンバーが示すファイル保存形式と，HTTP ヘッダに含まれる Content-Type から得られたファイル保存形式が同じであるか.	拡張子の情報は，HTTP ヘッダ Content-Type から得ることができる．ここで得られた拡張子の情報と，送られてきたファイルのマジックナンバーと比較することによって，送られてきたファイル拡張子が異常であるか判定する.	拡張子に異常がみられた場合，送られてきたファイルは破棄する.

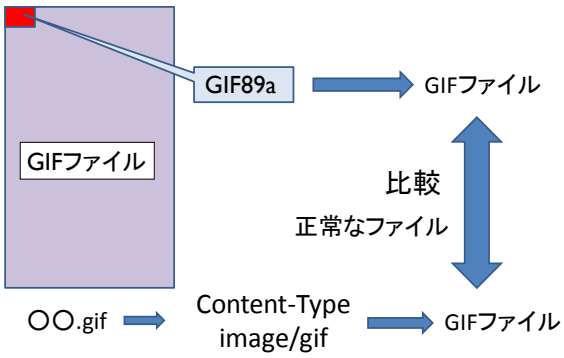

図 5　拡張子が正常な場合

Vol.2010-DPS-142 No.10
Vol.2010-CSEC-48 No.10
2010/3/4

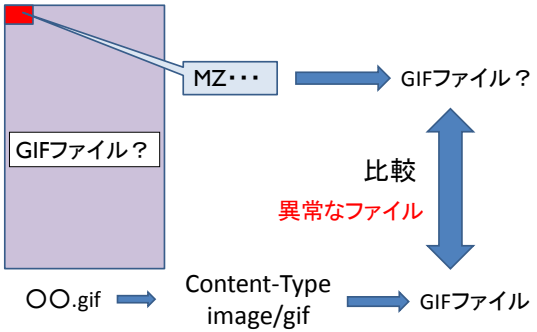

図 6　拡張子が異常な場合

3.2.3 ファイルの構造

ファイル構造の異常判定は，表 4 の通りである．GIF ファイルの例で説明する．GIF ファイルのファイル構造はサイズ参照型であり，順次ブロックの種類と，ブロックのサイズを取得することで，ブロックを順番に読み込むことができる．GIF ファイルにおける最後のブロックは”Trailer”ブロックであるので，このブロックまでたどり着くことができれば，正常なファイルであるといえる．一方，途中でブロックの種類が特定できなくなり，次のブロックを読むことができなければ，攻撃を意図したファイルの可能性がある．GIF ファイルにおける構造調査の流れを，図 7 に示す．また，GIF ファイルにおける構造調査のイメージを図 8 に示す．

表 4　ファイル構造の異常判定

項目	判定方法	異常時の処理
ファイルのブロックを順次読み込み，ファイル末尾まで問題無く読み取ることができるか．	3.1 節で述べたファイル保存形式に基づいて，ファイルを順次読み込んでいき，ファイルを最後まで読み込むことができるかでどうかで，ファイル構造に異常があるか判定する．	拡張子に異常がみられた場合，送られてきたファイルは破棄する．

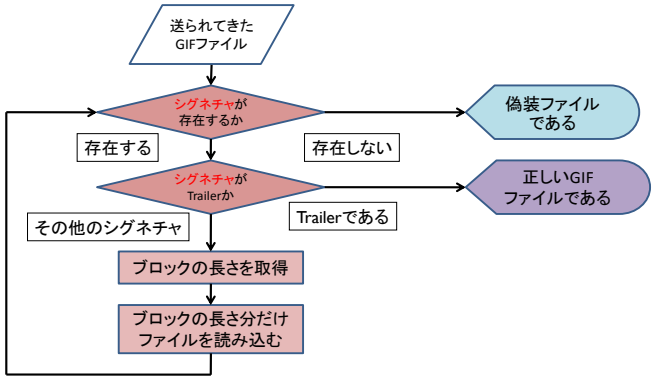

図 7　GIF ファイルにおける構造調査の流れ

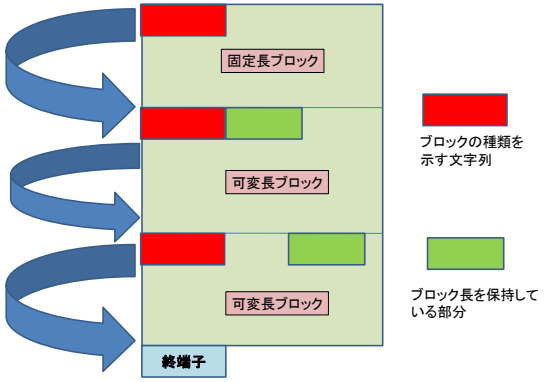

図 8　GIF ファイルにおける構造調査のイメージ

3.2.4 ファイルサイズ

ファイルサイズの異常判定は，表 5 の通りである．

Vol.2010-DPS-142 No.10
Vol.2010-CSEC-48 No.10
2010/3/4

表 5　ファイルサイズの異常判定

項目	判定方法	異常時の処理
3.2.3 節で読み込んだブロックサイズの合計と，HTTP ヘッダに含まれる Content-Length から得られる送信ファイルのサイズが同じかどうか.	送信ファイルのサイズ情報は，HTTP ヘッダ Content-Length から得ることができる．ここで得られた送信ファイルのサイズ情報と，3.2.3 節で読み込んだブロックサイズの合計と比較することによって，送られてきたファイルのサイズが異常であるか判定する.	サイズに異常がみられた場合，異常ファイル処理機能によって修正する.

3.2.5 PDF ファイル内に存在する JavaScript の検知

PDF は終端子型のファイル構造をしており，オブジェクトという単位で機能を分けている．オブジェクトは，自分がどの機能で動作するか，オブジェクトの先頭にタグをつけることで宣言している．PDF ファイル内に存在する JavaScript の存在判定は，表 6 の通りである.

表 6　JavaScript の存在判定

項目	判定方法	異常時の処理
PDF ファイル内に JavaScript オブジェクトを示すタグが存在するか.	PDF のオブジェクトの種類はオブジェクトに付けられたタグによって得ることができる．タグの中に，JavaScript オブジェクトを示すタグが存在するかどうかで判定する.	拡張子に異常がみられた場合，修正可能であれば，異常ファイル処理機能によって修正する．修正不可ならば破棄する.

3.3　異常ファイル処理機能

異常ファイルを検知したら，検知した検知項目により，2 通りの処理をおこなう．処理の方法は，次の通りである（表 7）.

表 7　検査項目ごとの処理

処理方法	検査項目
異常ファイルを破棄し，クライアントへの送信を停止する	拡張子 ファイル構造 PDF 内部に JavaScript
異常ファイルを修正し，クライアントへ送信する	ファイルサイズ PDF 内部に JavaScript

異常ファイルの種類により，異常ファイルが修正可能かを決定するため，検査項目により処理方法に違いが生じる.

- ファイルサイズが異常である場合，正常なファイルの末尾に別のファイルが付加されている状態であるので，後ろに付加されているファイルを取り除くことで，正常なファイルに修正する.
- PDF ファイルに施す処理は，JavaScript オブジェクトの形式によって変わる．JavaScript オブジェクトの形式は 2 通りある．JavaScript オブジェクトの形式別におこなう処理は，表 8 の通りである．実行文を別のオブジェクトに委託している形態では，委託先のオブジェクトの情報を収納する．この情報は非常にサイズが小さいので，コメント文など意味のない JavaScript の実行文による置き換えは難しいため、受信した PDF ファイルそのものを破棄する方法をとることとした.

表 8　PDF ファイルにおける JavaScript オブジェクトの形式と異常処理

JavaScript オブジェクトの形式	処理
オブジェクト内部に，JavaScript の実行文が存在する	オブジェクト内部に存在する JavaScript の実行文を，コメント文など意味のない JavaScript の実行文に置き換えることで修正する
オブジェクト内部に，JavaScript の実行文が存在せず，別のオブジェクトに委託している	JavaScript オブジェクト自体が，非常に小さく，意味のない JavaScript の実行文による置き換えはできない．修正不可であるので，PDF ファイルを破棄する.

3.4　アクセス解析機能

この機能は，異常ファイルをダウンロードしてきた URL を記録するための機能である．URL を記憶することで，再度悪意のあるファイルをダウンロードすることを防ぐ

Vol.2010-DPS-142 No.10
Vol.2010-CSEC-48 No.10
2010/3/4

ことを目的としている．

3.5 URL フィルタリング機能

この機能は，アクセス解析機能によって記録した URL を基に，クライアントからのアクセスを制御する機能である．

4. 評価

提案するシステムの有効性を検証するために, JavaScript を利用したウイルスが仕込まれた PDF ファイルを用いた検知評価と，ファイル構造チェックに伴う遅延評価を実施する．

4.1 ファイル構造を利用した異常検出の有効性

システムの有効性は, JavaScript を利用したウイルスが仕込まれた PDF ファイルを，提案するシステムによって無力化できるか実験をすることで評価をおこなう．

（1） 評価ファイル概要

評価に使用する PDF ファイルは，2008 年 6 月頃発生した，CSS2008 の CFP を騙ったウイルスメールにより使用されたもので，PDF に含まれる JavaScript を利用してウイルス（exe ファイル）を実行させる．

（2） 結果

実験において提案システムは，ウイルスが仕込まれた PDF ファイルを JavaScript の埋め込まれたファイルとして検知した．また，ウイルスが仕込まれた PDF ファイルに使用された JavaScript オブジェクトが，別オブジェクトに実行文を委託する形式だったことから, 異常ファイル処理機能によって該当ウイルス入り PDF ファイルを破棄することを確認した．

（3） 考察

実験結果より，提案システムは， JavaScript オブジェクトの有無によって，定義した異常を検知でき，適切な処理ができることを確認した．これにより，提案システムは，異常と定義したファイルの侵入に対して，有効に動作する可能性を示せたと考える．

4.2 提案システムを導入することで発生する遅延

提案システムを導入することで発生する遅延は，提案するシステムを経由した場合と，経由しなかった場合で，1 つのファイルを Web サイトから取得するのに要する時間を比較することで評価をする．

（1） 実験方法

実験方法は，次の通りである．

- Web サイトに存在する GIF ファイルと PDF ファイルを取得するのにかかる時間を計測する
- 提案するシステムを経由した場合と，経由しなかった場合の 2 通り計測し，得られた取得にかかる時間の差を比較する
- 計測は，Web サーバにリクエストを出してから，取得が終了するまでの時間である

実験に使用した GIF ファイルと PDF ファイルの詳細を表 9 に示す．

表 9 実験に使用したファイルの詳細

	GIF ファイル	PDF ファイル
ファイル数	3762	2199
平均サイズ（Byte）	2,943	2,816,614
最大サイズ（Byte）	135,047	37,606,205
最小サイズ（Byte）	101	228

ファイル数は，実験に使用した各ファイルの総数である．平均ファイルサイズは，1 ファイルあたりのサイズであり，”（総ファイルサイズ）÷（ファイル数）”によって算出した．最大サイズは使用したファイルの中で最大の，最小サイズは使用したファイルの中で最小のファイルサイズである．

（2） 結果

実験によって得られた, 各ファイルの取得時間と発生した遅延結果を表 10 に示す．平均遅延は，1 ファイルあたりの取得するときに発生する遅延で,”（各ファイルを取得する際に発生した遅延の総和）÷（総取得ファイル数)”によって算出する．また，各取得時間は，提案するシステムを経由しない場合のものである．遅延割合は，ファイル取得時間に対し，どの程度遅延したかを示す数値で,“(遅延）÷（取得時間)”で算出する．

Vol.2010-DPS-142 No.10
Vol.2010-CSEC-48 No.10
2010/3/4

表 10　実験結果

	GIF ファイル	PDF ファイル
平均ファイル取得時間（s）	0.0277	5.5062
平均遅延（s）	0.0027	0.2448
平均遅延割合	0.1	0.04
最大サイズのファイル取得時間（s）	0.0373	210.3125
最大サイズのファイル取得時に発生した遅延（s）	0.0284	2.3437
最大サイズのファイル取得時の遅延割合	0.75	0.01
最小サイズのファイル取得時間（s）	0.0193	0.2660
最小サイズのファイル取得時に発生した遅延（s）	0.0011	0.0935
最小サイズのファイル取得時の遅延割合	0.05	0.3

（3）　考察

平均遅延と，サイズによる遅延割合について考察する．

1)　平均遅延

表 5 より，Web サイトから GIF ファイルと PDF ファイルを取得する時に発生する遅延は，表 5 の平均ファイル取得時間平均から，取得時間の 1/10 程度であり，非常に小さい．これより，提案するシステムは Web サービスを通常利用することおいてに大きな障害が発生しないこと考える．

2)　サイズによる遅延割合

GIF ファイルの遅延割合をみると，ファイルサイズ最小のときは 0.05 であったのに対し，ファイルサイズ最大のときは 0.75 と大幅に大きくなっている．遅延割合が増加する原因は不明だが，提案システムのプログラムに問題があると考えられる．原因究明は今後の課題とする．

PDF ファイルの場合は，ファイルサイズの増加に伴って，遅延割合が減少する結果となった．これより，PDF ファイルは，サイズによる遅延の変動はあまり大きくないと考えられる．

5.　まとめ

本稿では，ファイルの構造に着目し，送られてきたファイルの構造を検査項目としたアノマリ型の侵入検知システムを提案し，実装した．また，提案したシステムは，JavaScript を利用したウイルス入りの PDF に対して，検知と無力化をすることができ，特定の機能を利用した攻撃に対して有効性を示した．

適用するファイル構造を増やし，提案システムの有効性を高め，より安心して Web サイトを利用できるようにするための今後の課題は，次の通りである．

- GIF ファイルに対する，遅延割合増加への対策をおこなう．
- ファイルの構造に着目した検査項目を充実させると共に、別の保存形式に対する検査項目を実装することで，提案システムの有効性を検証する．

参考文献

1)　ITpro，GIF を隠れ蓑に悪性 Java を勝手に実行
http://itpro.nikkeibp.co.jp/article/COLUMN/20090324/327100/

2)　ScanNetSecurity ，Black Hat Japan 2008 GIF＋JAR＝GIFAR ファイルでドメインベースの信頼は破壊される
https://www.netsecurity.ne.jp/3_12364.html

3)　ITpro，画像ファイルに偽装した，HDD をフォーマットしようとするトロイの木馬がネットで話題に
http://itpro.nikkeibp.co.jp/article/NEWS/20071231/290405/

4)　コンピュータセキュリティ研究会，CSS2008 の CFP を騙ったウイルスメールに関する情報
https://www.sdl.hitachi.co.jp/csec/css2008-cfp-secinfo.html

5)　しいしせネットワーク
http://siisise.net/

6)　アドビシステムズ著，ドキュメントシステム約：PDF リファレンス第 2 版，ピアソンエデュケーションズ