サイエンティフィック・システム研究会　　2010年1月20日
2009年度 システム技術分科会　第２回会合

# 情報セキュリティの課題と対応策

高橋　正和
マイクロソフト株式会社

[アブストラクト]
情報セキュリティは，実際に起きた事件や攻撃手法の変化に伴って発展しており，時代と共に変化してきた．本稿では，情報セキュリティに関する事件と対策の推移から，情報セキュリティに対する認識の変化や，攻撃の発展の過程を分析し，情報セキュリティが，今取り組むべき課題を解説する．また，これらの課題について，最新のＯＳが，昨今の攻撃手法に対して，どのような取り組みが行われているのかを紹介する．

[キーワード]
不正アクセス，マルウエア，ＯＳ

## 1. はじめに

2004 年 4 月の *Sasser* ワームを最後に，大規模なウイルス感染は発生していないが，フィッシングやスパイウェアによる金銭的な被害の報道が増えている．

例えばフィッシングは国際的な犯罪組織と関係し[1]，年間 24 億ドル[2] (約 2 兆 9 千億円)もの被害を与えており，このような犯罪行為を支えるための基盤として，ボットネットが利用されているといわれている[3].

Telecom-ISAC Japan および JPCERT/CC の調査結果「フィールド調査によるボットネットの挙動解析」[4]によれば，ボットネットは，DDoS 攻撃a)，情報の収集，スパムメールの送信などの機能を効果的に実施出来るとされており，ボットネットが犯罪行為を支える基盤として利用される理由と考えられる．

一方で，このような機能を備えたウイルスは過去にも存在していたが，大規模な犯罪行為の基盤としては考えられていなかった．従来は，多数の感染ノードが単独で存在するモデルであり，特定の感染ノードに対する操作が可能であっても，多数の感染ノードを一括して制御する機構が存在しなかった．これに対し，ボットネットは，感染ノード群を有機的に結合し，分散システムとして一括して制御する機構を実装している．この点に，ボットネットの本質的な脅威がある．

本稿では，まず過去のウイルスやワームがどのように基本要素を実装していったかを調査し，各基本要素がウイルスに組み込まれた経緯と目的を考察する．また，基本要素を持ったウイルスの例として，DDoS ツールである *TFN* と，メール型ウイルスの *Sobig* を取り上げ，技術的な問題点について分析する．

次に，ボットネットが，*Sobig* 等の技術的な問題点をどのように解決しているのかについて考察を行い，ボットネットがもたらした，本質的な脅威の変化について分析する．

最後に，これらの分析の結果，これまでのセキュリティ対策のアプローチの問題点を取り上げ，最新のオペレーティングシステムが，これらの問題に対して，行っている対策について解説する．

## 2. マルウエア（広義のウイルス）の推移

ウイルスやワームに代表される，悪意を持ったプログラム（以下，マルウエア）の歴史についてまとめた資料はあまりない．ここでは，数少ない資料のひとつとして著者らが執筆した「有害プログラム-その分類」[5]を中心に，海外の文献の調査結果を加えて，マルウエアの歴史を振り返る．

### 2.1 ウイルス・ワームという名称について

コンピュータウイルスという呼び名は，1984 年に Feed Cohen が発表した論文が最初といわれており，他のプログラムに寄生して広がっていくことを基本的な概念として定義している．

現在，一般にウイルスと呼ばれているプログラムは，電子メールを媒体として利用することが多い．電子メールや文章などのドキュメントファイルに寄生するものも，ウイルスと呼ぶことが出来る．

一方，ワームという名前の由来は，1975 年に公表された John Brunner の SF 小説 “The Shockwave Rider” に登場する *Tapeworm* に由来する．ワームは，独立したプログラムであり，コンピュータウイルスのように他のプログラムに寄生しない点が，ウイルスとは異なっている．

### 2.2 ワームの原型

1982 年に公表された，John Scoch 等の文献”The “Worm” Programs – Early Experience with a Distributed Computation”[6]に，ネットワークで有用な機能を果たす自動化された分散型プログラムという概念で，ワームが述べられており，試作したワームプログラムについての分析が行われている．

試作したプログラムのひとつである *Blob* は，夜間にリソースに余裕のあるコンピュータを探し出して利用し，朝になると処理を終了するというものであった．*Blob* は，夜間に活動することから *Vampire*(バンパイア)とも呼ばれた．

*Blob* を使った実験では，誤動作（バグ）により，すべてのコンピュータがクラッシュするという事故が発生した．この事故では，システムをリブートしても，すぐに *Blob* がシステムリソースを奪取してしまい，なかなか復旧作業を行う事が出来なかったという．Scoch 等は，これらの経験から，ワーム作成上の重要な問題は，ワームの制御である（”Key problem: Controlling a Worm”）と述べている．

文献[6]では，あわせて，インターネットの前身である Arpanet におけるワームプログラムの歴史が取り上げられており，自らコンピュータ間を移動するプログラムとして，Arpanet のルーティングプログラムである *IMP* や，1971 年に B.Tomas によって開発された *Creeper*(クリーパー)が紹介されている．*Creeper* は，自己複製機能がなかったが，R.Tomlison が自己複製機能を追加し，ワームとして動作させた．なお，R.Tomlison は，*Creeper* を探し出して削除する *Reaper* という，現在のアンチウイルスソフトに相当するツールも作成している．

### 2.3 実在のウイルス（Virus in the wild）

最初の実在のウイルス(Virus in the wild）は，1981 年に発見された，Apple II に感染する *Elk Cloner* といわれている．*Elk Cloner* が入っているフロッピーディスクをコンピュータに挿入すると，*Elk Clonner* はメモリ中で活動し，他のフロッピーディスクが挿入されると，そのフロッピーに感染した．*Elk Clonner* は，フロッピーが 50 回挿入されると，表示画面を消去し，メッセージ（詩）を表示した．

a) DoS 攻撃:Denial of Service Attack，サービス不能攻撃 とも呼ばれる

このプログラムは，Rich Skrenta が作成したといわれている．Rich Skrenta は，いたずらを目的としたプログラムを作成して友人に配っていたが，自分のプログラムを使ってくれる友人がいなくなったことから，フロッピーディスクに感染する自己増殖プログラム（クローン型プログラム）を書くことを思いつき，*Elk Clonner* を書いたと言われている[7].

ワームの源流が，ネットワークコンピューターの分散処理の研究であるのに対して，ウイルスの源流がいたずらにある点は興味深いものがある．

IBM-PC に感染するウイルスとしては，1986 年の*パキスタン・ブレインウイルス*が最初といわれている．こウイルスはフロッピーディスクのブートセクタに感染し，「ウイルスに注意（Beware of the VIRUS….”）」というメッセージと，連絡先として BRAIN COMPUTER SERVICES”社の社名，住所，連絡先等を表示するもので，自社ソフトの不正コピー対策であったと言われている．パキスタン・ブレインウイルスは，IBM-PC 上の最初のウイルスであると同時に，現在アドウェアの源流とも言えるプログラムである．

### 2.4 電子メールを感染媒体としたウイルス

初期のウイルスは，フロッピーディスクなどのメディアを経由して感染を広げたが，1987 年の *CHRISTMA exe ワーム*は，IBM の社内ネットワークおよび，BITNET 上の電子メールを媒体として感染を広げた．このワームは，クリスマスツリーを画面上に表示するプログラムを電子メールに添付して送信するもので，プログラムが実行されると，アドレス帳に登録されているすべての人に，このメールを転送するというものであった．

電子メールを使ったウイルスとしては，インターネットが普及し始めた 1999 年の *Melissa* が有名であり，IBM PC 上の最初のメール型ウイルスと考えられる事が多い．しかし，1997 年に一部のアンチウイルスベンダーから *ShareFun* という電子メールを媒体としたウイルスが報告されている[8].

いずれにしても，*Melissa* は，ウイルスがインターネット時代に移行したことを示し，ウイルス対策をはじめとした，ネットワークセキュリティに大きな影響を与えた．

### 2.5 インターネットワーム（モリスワーム）

インターネット上で活動するワームとしては，1988 年のインターネットワーム(モリスワーム)が最初のものと考えられる．このワームは単に感染を繰り返すものであったが，当時インターネットに接続されていたコンピュータの 10%(6000 台)に直接的な被害を与えた事に加え，被害を恐れてインターネットからコンピュータを切り離したサイトも多かった事から，結果として，極めて大規模な DoS 攻撃となり，インターネットを大きな混乱に陥れた．

この混乱を契機に，大規模なインシデントに対応することを目的として，カーネギーメロン大学に CERT が設立される契機となった[9].

### 2.6 DDoS ツール

モリスワームは，結果として DoS 攻撃を行ったが，1999 年に DoS(DDoS)を目的としたプログラム（以下 DDoS ツール）が話題になった．

Dave Dittrich の文献[10]によれば，最初の DDoS ツールは 1998 年の，*fapi* および *fuck_them* というツールあり，1999 年 8 月 17 日にミネソタ大学に対して DDoS ツールを使った最初の攻撃行われた．

DDoS ツールは，1999 年 11 月に CERT が開催した DSIT (the Distributed System Intruder Tools Workshop[11])によって，一般に知られるようになり，その直後の 1999 年 12 月末には FBI から，DDoS ツールを検出するプログラムがリリースされている[12].

DSIT が開催された背景には，Solaris RPC サービスの脆弱性を利用した侵入行為が多数見つかっていること（CERT Incident Note 99-04[13]/99-05[14]），この侵入行為が行われたサイトで，多数の *trinoo* および *TFN(the Tribe Flood Network)*と呼ばれる DDoS ツールが見つかったことにある(CERT Incident Note 99-07[15]). *trinoo* は，スクリプト（リスト 1）を使った侵入が行われており，短時間に大量の侵入が可能である事から，DDoS ツールの更なる拡散と，DDoS 攻撃による被害に対する大きな危機感があった．

リスト 1 *Trinoo* のインストールシェル

```
./r -6 -k $1 "echo 'ingreslock stream tcp nowait root /bin/sh sh -i' ¥
      >>/tmp/bob ; /usr/sbin/inetd -s /tmp/bob"
./r -6   $1 "echo 'ingreslock stream tcp nowait root /bin/sh sh -i' ¥
      >>/tmp/bob; /usr/sbin/inetd -s /tmp/bob"
echo Sleeping 2 seconds...
sleep 2
telnet $1 1524
```

### 2.7 バックドア

ワームやウイルスと並んで，バックドアという呼称が使われることがある．バックドアは，主に次のような種類がある．

- 盗聴用のバックドア（スニファ,キーロガー等）
- リモートコントロールのためのバックドア
- バージョンアップのためのバックドア
- リダイレクタ（Proxy）

文献[5]では， 1998 年に 8 月に Black Hat カンファレンスで発表された *Back Orifice* を最初のバックドアとしているが，同年 3 月には *Netbus* と呼ばれる同様のツールがリリースされ，広く利用されていた．

また，1996 年にリリースされた *Netcat*(*nc*)というツールもバックドアの一種と考えられる．*Netcat* は,”TCP/IP swiss army knife”をコンセプトとしており，様々な機能が実装されている．この考え方は，現在のボットにも受け継がれており，*Phatbot* と呼ばれるボットのドキュメントでは，スイスアーミーナイフの画像が多用されている（図 1）.


図 1 *PhatBot* FAQ のスイスアーミーナイフ

### 2.8 スパム中継型ウイルス

インターネットを使った商用活動のひとつとして，電子メールによる広告活動を挙げることが出来る．この活動において，特に，不特定多数の電子メールアカウントに対して，一方的にメールを送りつける行為は，スパムメールと呼ばれている．スパムメールは，単に不要なメールを受け取ってしまうばかりでなく，様々な社会的な問題につながることから，これを防止するための対策が行われている．

初期のスパム送信は，ISP などの正規のメールアカウントが利用されていたが，サーバ管理者によりスパム送信者のアカウントを停止するなどの対策が行われた．

スパム送信者は，この対策に対して，送信者のアカウントを隠すために，関係のない第三者のメールサーバを利用するようになった（以下，第三者メール中継）．しかし，第三者メール中継を禁止する設定が普及したことや，ORDB（Open Relay DataBase ）[16]と呼ばれブラックリストを使った対策によって，スパムメール送信の実効性が著しく落ちていった．

これを解決し，効率的にスパムを送信する方法として登場したのが，2003 年の *Sobig* である．*Sobig* は，ウイルスにメール中継機能を持っており，これを利用してスパムを送信する．このため，多数の IP アドレスから独立してメールが送信されているように見せかけ，ORDB などのメール送信元の IP アドレスに基づいたスパムメール対策を回避することが可能となった．

## 3. マルウエアの基本機能についての考察

ここまでに紹介したウイルスに対して様々な機能が実装されてきた．これらは個別の事象として平行して起きているものである．ここでは，ウイルスが様々な機能を実装していった目的と経緯について考察する．

### 3.1 DDoS ツール(*TFN*)のアプローチ

DDoS ツールは，現在のボットネットとよく似た構造を持っている．ここでは，*TFN(Tribe Flood Network)*を例として，DDoS ツールの機能と，機構的な問題点について考察する．

*TFN* は，1999 年に公表された DDoS ツールで，Client(命令者)と Daemon(Zombie)の 2 階層で構成される．攻撃者は，自らが Client となることも，侵入したシステムを Client として利用することもできる．*TFN* の基本的な構成を図 2に記載する．

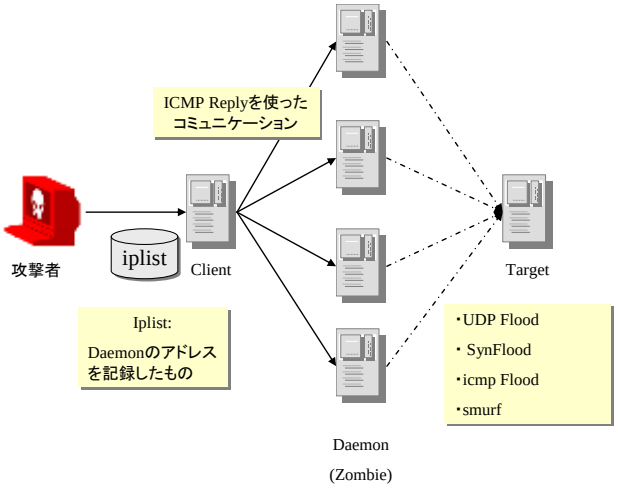


図 2　*TFN* の構成

*TFN* の Client・Daemon 間は，一般的にネットワークの接続性を確認するために利用される，ICMP ECHO REPLY を使って通信が行われる．このため，ファイアウォールを通過できる可能性が高く，また，通信を発見することが難しい面がある．ICMP を使った通信手段は，1996 年に *loki* [17]という名称で実証コード(POC：Proof of Concept)が公表されている．

*TFN* Client は，Daemon(Zombie)に対して，以下のコマンドを指示することができる．

1. UDP　Flood
2. Syn Flood
3. ICMP(Ping) Flood
4. Smurf
5. Bind a root shell

*TFN* はソースとして入手が可能であったことから，これをもとに *TFN2K*，*Trin00*，*Stacheldraht* 等の次世代の DDoS ツールが開発された．これらの較的新しい世代の DDoS ツールと *TFN* の主な違いは次のような点である．

- 通信の暗号化
- 接続の際の認証方式
- Client と Daemon の間に Master と呼ばれるレイヤの追加
- 盗聴機能の追加（rcp コネクションのモニタなど）

DDoS ツールは強力なツールであり，効果的に DDoS 攻撃を行うことができる．しかし，ここに紹介した DDoS ツールを使った大規模な DDoS 攻撃の事例と考えられるのは，ミネソタ大学に対する UDP Flood などに限られる．Yahoo に対する DDoS は，*Smurf* と呼ばれる古典的な手法が使われている事から，DDoS ツールが利用された事を確認することは難しい．

DDoS ツールが，必ずしも広範囲に利用されなかった理由として，以下の点が原因となっている可能性がある．

- UNIX 系のシステムを主なターゲットとしていたため，Zombie 候補の絶対数が不足していた
- Zombie の台数を確保する作業に，時間と手間が必要だった
- Zombie の更新などのメンテナンス機能が実装されておらず，Zombie の維持が出来なかった
- Zombie の管理が非効率であり，実際に利用する事が困難であった

Zombie の台数を確保する事を考え場合，UNIX 系のシステムより Windows 系のシステムを使った方が有利な可能性が高い．さらに，Windows 系のシステムに対しては，ウイルスやワームという拡散技術が既に確立していたことから，DDoS トラフィックの発生源として，ウイルスが利用されるようになっていったものと考えられる．

文献[5]によれば，2001 年 7 月に発見された *CodeRed* は，ワームに DDoS 機能を実装した最初のウイルス/ワームであるとされている．*CodeRed* の他にも，2004 年の *Netsky* や *Mydoom* も DDoS を行うことを目的としており，*Netsky* や *Mydoom* はターゲットサイトをダウンさせることに成功しているが，*CodeRed*，*MSBlast* は， DDoS 攻撃を始める前に，回避策が実施されたため，攻撃に失敗している（文献[18]，文献[19]）．

### 3.2 バックドアについての考察

*Back Orifice*，*Netbus*，*Netcat* に代表されるバックドアは，一般的に侵入するための機能をもっておらず，いわゆるハッカーがシステムへの進入に成功した際に，設置されることが多かった．しかし，バックドアを設置する手法はある程度自動化されている場合が多く，多くの場合，次のような手法が利用されていた．

① Scanner と呼ばれるツールで，侵入が可能なターゲットを探し記録する．なお，この探査は，自動化されている場合が多い．
② ターゲットに対して攻撃を行い，実行権を取得する．
③ バックドアをターゲットにコピーし，起動した上で，システム起動時にも再実行されるように設定する．

この一連の流れを自動化したツール（Autorooter）も存在する．その一例が，「DDoS ツール」で紹介した *trinoo* のスクリプト（リスト 1）であり，典型的な例がネットワークワームである．多くのネットワークワームは次のような動作をする．

① 乱数などを使って，攻撃を行う IP アドレスを決定する
② 決定した IP アドレスに対して，攻撃コード(Exploit Code)を使って，ワーム本体をダウンロードするコマンドを実行する．
　例： CMD.exe TFTP xx.xx.xx.xxx wormbody.exe
　　wget http://xxx.xxx.xx.xx/bot.pl
③ 同様に，ダウンロードしたワームを実行する．
　例： CMD.exe wormbody.exe
　　/usr/bin/perl bot.pl

この一連の流れにおいて，バックドアのダウンロードを追加することは容易であり，また，ワーム本体にバックドア機能を持たせることも可能である． 具体的な例としては，2002 年の *CodeRed II*，*Badtrans*，*Bugbear*，2003 年の *Sobig* 等をあげることが

できる．

ウイルスが，バックドアをインストールする手法について，以下に *Sobig.A* を例として紹介する．

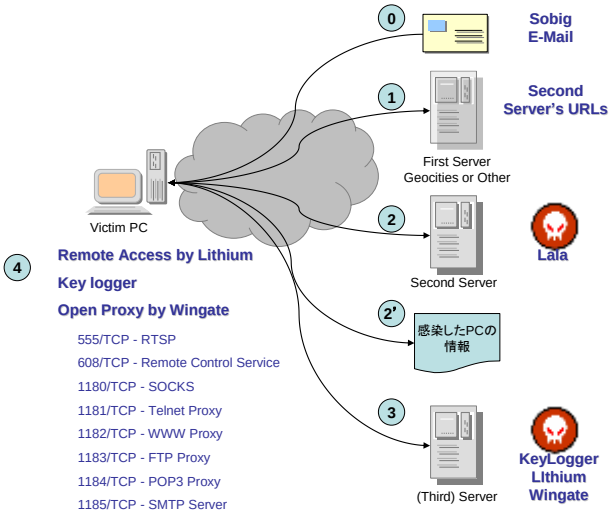


図 3　Sobig.A の挙動

*Sobig.A* は，主にメールを使って感染を行うウイルスだが，*Sobig.A* が実行されると，次の手順でバックドアと，リダイレクタをインストールする．

① ある Web サイトから，*Lala* というバックドアをダウンロードするためのアドレスを取得する
② 該当するアドレスから，*Lala* をダウンロードし，実行する．
③ *Lala* は，自分自身の場所をリモートサイトに通知し，キーロガーと，パスワードで保護されたリモートアクセスツール（*Lithium*）をインストールする．
④ 次に，*Lala* は *Wingate* プロキシサーバをインストールする．

*Wingate* プロキシの設定は，以下の通り．

555/TCP - RTSP
608/TCP - Remote Control Service
1180/TCP - SOCKS
1181/TCP - Telnet Proxy
1182/TCP - WWW Proxy
1183/TCP - FTP Proxy
1184/TCP - POP3 Proxy
1185/TCP - SMTP Server

⑤ 最後に，*Sobig.A* を削除する．

これにより，*Sobig.A* は次の機能を実現する．

- 完全なリモートアクセス
- キーストロークの記録
- スパム送信のためのリダイレクタ
- Wingate プロキシの自由な変更

この結果，*Sobig* に感染した PC のアドレスを持っている人物は，このリストを使って，大量のスパムを送信することが可能となる．

*Sobig* は，A～F まで 6 亜種が活発な活動を行ったが，*Sobig.F* が更新を行う際に，世界的な ISP の連携によって，*Lala* をダウンロードするためのサイトを全て閉鎖することで鎮圧された．

*Sobig* は，特定のサイトにアクセスを行わなければならなかったため，ここが弱点となったわけだが，バックドアの利用についても，次のような問題がある．

① 多くの PC がダイナミック IP アドレスを使っており，ある時点で実際に利用できる IP アドレスのリストを作成することが難しい．
② バックドアが開いていても，ファイアウォールの内側にある *Sobig* には接続できない．
③ 個別の *Sobig* の操作を行う事は容易だが，多数の *Sobig* を制御することは難しい．
④ 同じ理由で，大量の *Sobig* のバージョンアップを一斉に行うことも難しい．

*Sobig* では，IP アドレスを第三者に知られると，その制御権を与えてしまう事になるため，このリストは秘密にする必要がある．つまり，スパム送信者に *Sobig* を貸し出すなどの手段で，利益を得ることができず，常に自らスパムの送信を行う必要があったものと思われる．

*Sobig* は，多彩な機能を身につけたわけだが，ひとつのシステムとしてネットワークを構成するに至っておらず，単に大量の利用可能なノードがネットワーク上に存在するという状態であったといえる．

## 4.　マルウエアの一括制御へのアプローチ

従来からマルウエアは，多様な機能を持っていたが「多数の感染 PC を効率的に管理・運用する仕組み（以下 C&C ：Command and Control System)」が欠けていた．

マルウエアが C&C を持つことによる脅威の変化を，以下に考察する．

### 4.1　C&C 構築の利点

多数の感染 PC を管理する適切な C&C が構築できた場合，次のような利点がある．

① 別々の方法で感染させた PC をひとつのネットワークとして利用・制御することができる
② 構築したネットワークを利用することで再感染が容易になる
③ 意図する攻撃や活動に応じたマルウエアに入れ替えが出来る
④ リードタイムなしに攻撃等の活動が行える
⑤ マルウエアの入れ替えによりアンチウイルスの検出と削除を回避する

近年注目されているボットネットは，多数の感染 PC を使って，非常に堅牢なシステムを実現している．以下に，ボットネットの概要と C&C としての機能を解説する．

### 4.2　ボットネットによる C&C の構築

ボットネットは，ボットと呼ばれるウイルスの一種が構成するネットワークのことで，IPA（独立行政法人　情報処理推進機：Information-Technology Promotion Agency, Japan）ではボットを次のように定義している．

> ボットとは，コンピュータウイルスの一種で，コンピュータに感染し，そのコンピュータを，ネットワーク (インターネット)を通じて外部から操ることを目的として作成されたプログラムです．
> 感染すると，外部からの指示を待ち，与えられた指示に従って内蔵された処理 (後述)を実行します．この動作が，ロボットに似ているところから，ボットと呼ばれています．

ボットネットは，図 4のように多数のボットが，IRC メカニズ

ムを利用してネットワークを構成しているもので，HERDER[b]または，Master と呼ばれるボットネットの管理者によってコントロールされる．HERDER は，IRC サーバに指令を与えることで，IRC サーバに接続している全てのボットに対して，ほぼ同時に命令を伝えることができる．この特性を利用し，DDoS 攻撃やプログラムの更新といった，同時性が望まれる行為を，容易に実現できる．なお，必ずしも IRC メカニズムを利用する必要はなく，試験的に P2P プロトコルの利用をするボットも確認されている．

報道では，数万～数百万規模という大規模なボットネットが取り上げられることが多いが，多くのボットネットは 3 千～8 千台という比較的小規模な構成にとどまる傾向にある．これは，小規模なボットネットの方が，発見が難しいためと考えられている（Honey Net Project[20]）．

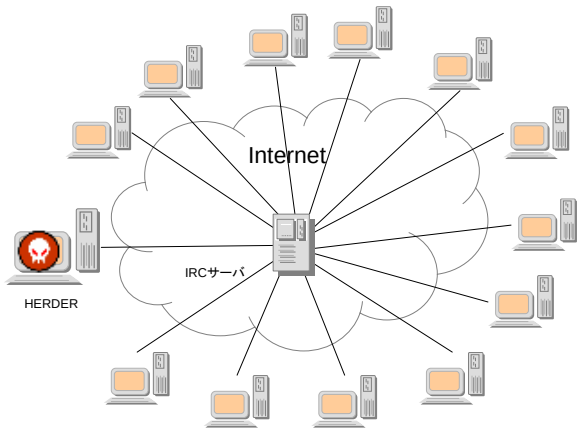


図 4　ボットネットの概要(IRC ボットネット)

なお，ボットは，ソースコードや解析やアンチウイルスによる検出を困難にするための難読化技術が，開発環境と共に流通していることが確認されており，機能の追加や変更が容易であり，様々な亜種が作成できるようになっている．

### 4.3　シーケンシャルマルウエアへの展開

IPA の調査[21]によれば，ボットネットで構築された機能は，一つの大きな実行ファイルから，用途に応じた小さな実行ファイルを必要に応じてダウンロードする形態が主流に変化しており，このような手法をシーケンシャルマルウエアと呼んでいる．なお，シーケンシャルマルウエアによっては，稼動しているプログラムに対するインジェクションを行う例や，あたかもオーバーレープログラムのように，機能モジュールをメモリ上にダウンロードすることで，ファイルを作成せずに新たな機能を取得する例もあるという．また，攻撃のトリガーが，ネットワークサービスから，メールや Web を使って様々なアプリケーションの脆弱性を利用する形態に変化している．

これらの事例は，個々のマルウエアが複雑な活動をしているように見えるが，実際はボットネットが変化したものであり，C&C または C&C に相当するものが，一括して感染したマルウエアをコントロールしている．また，ソースコード等で流通している様々なコードを“必要に応じて機能（モジュール）を組み込む”という形態から，“必要な機能（モジュール）だけを組み込む”という形態に変化したものと見ることができる（図 5）．

つまり，近年のマルウエアは，マルウエア基本機能のモジュール化による任意の機能の組み込みと，マルウエアの一括統制ができる点に特徴がある．

[b]HERDER：牧夫，群れを率いる人という意味で使われる．

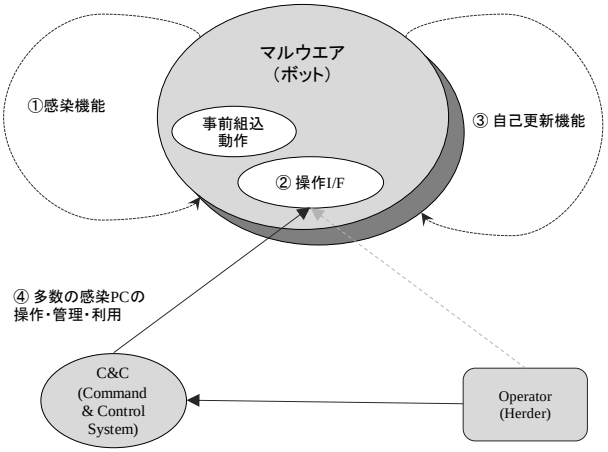


図 5　ボットネットの動作モデル

## 5.　最新システムにみるマルウエア対策

近年のマルウエア対策の難しさは次の点にある

- アンチウイルスで検知できないケースが増えている
- メールや Web を経由した感染が，イントラネットへの侵入を意味している
- 内部から外部に向けて通信を行うため，境界領域防御では，マルウエアと C&C の通信を防止できない
- 攻撃の対象となるアプリケーションが特定できない

このような近年のマルウエアの脅威に対して，最新のＯＳのひとつである，Windows 7 と，その基礎となった Windows Vista が実装している代表的な対策技術を紹介する．

### 5.1　OS のセキュリティレベル

Windows 7 については，まだデータが出ていないので，Windows 7 がベースとしている，Windows Vista のセキュリティレベルについて紹介する．

図 6は，主要なシステムの出荷後 1 年間に公表された脆弱性の比較である．Windows Vista は，Windows XP と比較して，約半分まで脆弱性を減少させている[22]．Microsoft では，脆弱性を減少させ，より安全なシステムを開発するために SDL(Security Development Lifecycle[23])という手法を適用しているが，この取り組みの成果を見ることができる．

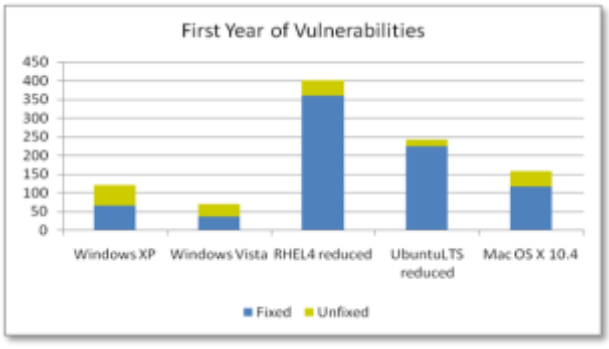


図 6　出荷 1 年間で公表された脆弱性の比較[24]

脆弱性を減少させることは重要であるが，必ずしもマルウエアの感染を減少させるとは限らない．図 7は，マイクロソフトが半年毎に行っているセキュリティに関する調報告 Security Intelligence Report（以下 SIR)において OS 毎のマルウエアの感染率をグラフにしたものである[25]．この調査では，Windows

Vista は，Windows XP の約 15%の感染率まで減少しており，マルウエアに対しては，効果的な取り組みが行われていると考えることができる．

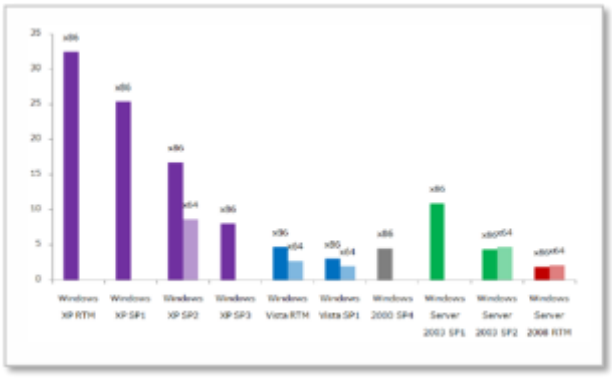

図 7　OS 毎のマルウエア感染率の比較[26]

SDL は，単に脆弱性の現象を目的としたものではなく，設計段階でのセキュリティの作り込みを目指したものである．マルウエアの感染率の低下も，SDL として実装された様々な機能が効果を挙げているものと考えられる．以下に、主要なセキュリティ機能について紹介する．

### 5.2　整合性レベル（Integrity Level）

整合性レベル(IL: Integrity Model)は，Windows Vista から導入されたもので，プロセスとオブジェクト間にレベルを設け，プロセスが持つレベルよりも高いレベルへの書き込みを禁止する仕組みである．

低整合性レベルのプロセスは，高整合性レベルに昇格をすることができず，また，システムエリアへの書込みができない（読み込みは ACL 次第）．このため，自動起動，システムファイルの置き換え，システム設定の変更や，他のプロセスにメッセージを送ることができない（UIPI）

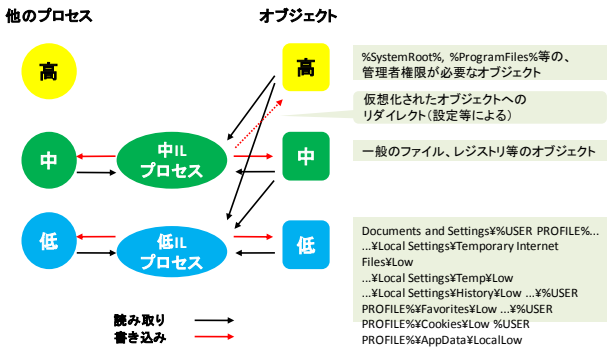


図 8 整合性レベルの概念

Internet Explorer 7/8 では，システムへのアクセスを制限する保護モードが用意されているが，これは IE と IE 上で稼動するプロセスを，低整合性レベルで動作させることで実現している．

### 5.3　DEP/NX

DEP/NX(Data Execution Prevention/No eXecution)は，ハードウェアを使ってデータ領域でのプログラムの実行を防止する．多くの攻撃に利用されるバッファオーバーフローを効果的に防止することができるため，マルウエアの感染の多くを，効果的に防止する．

DEP/NX は Windows XP SP2 から導入されているが，Windows Vista，Windows 7 と適用範囲を拡大している．具体的には，Windows 7 では，Internet Explorer 8 が出荷時設定として DEP/NX が有効に設定された．これにより，IE を経由して実行されるアプリケーションについても，DEP/NX が有効に機能することになった．

DEP/NX は，マルウエア対策として有効なものだが，アプリケーション側でも対応が必要であるが，徐々にアプリケーションの対応も進んでいる．

### 5.4　AppLocker

AppLocker は，プログラム，DLL，インストーラ，スクリプトに対して実行の許可・不許可を設定するツールで，パス，ハッシュ，証明書に基づいた制御を行うことができる．

従来から，同様の機能は実装されていたが，プログラムの更新のタイミングで設定を変更する必要があったことから，運用が煩雑で，あまり普及しなかった面がある．

AppLocker では，証明書に基づいた制御をサポートしており，プログラムのバージョン，プログラム，ファミリー，企業などの階層に応じた制御が行える．例えば，Microsoft Excel のバージョンに対する制御，Microsoft Excel に対する制御，Microsoft Office Suite に対する制御，Microsoft に対する制御を選択することができる（図 9）．

図 9　AppLocker の設定例

## 6.　まとめ

最近公表されたボットネット等の調査結果を分析すると，マルウエアは愉快犯ではなく，犯罪や疑わしい行為を目的として利用されるように変化している．この変化に伴い，マルウエアは複合的な機能を有するように変化したばかりではなく，マルウエアそのものを更新しながら活動するという，よりダイナミックなものへと変化している．

これに対して，アンチウイルスソフトに代表されるマルウエア対策は，現在においても，大規模感染を最大の脅威とするモデルに基づいている．攻撃側は，このような対策側の特性を理解し，大規模な感染を意図的に回避し，また，アンチウイルスソフトによる発見を回避するための様々な手法を利用しており，明らかに攻撃側が有利な状況にある．

このような状況において，ＯＳやアプリケーション自体のセキュリティレベルを向上させる事がより重要となっており，最新のシステムにおいては，様々な対策が実装されるようになっている．より巧妙になっているマルウエアの対策を進める上では，これらの機能の特性を理解し，利用していく必要がある．

## 参考文献


1) 佐々木 俊尚：インターネット事件簿：恐るべきロシアマフィア vs 日本の幼稚なネット犯罪者，急増するフィッシング詐欺の実態
http://internet.watch.impress.co.jp/static/column/jiken/2004/08/18/
2) ガートナー社:米国においてフィッシング詐欺が増加と報告
http://www.cyberpolice.go.jp/international/north_america/20040617_190553.html
Gartner: Phishing on the rise in U.S.
http://news.com.com/Gartner:+Phishing+on+the+rise+in+U.S./2100-7349_3-5234155.html
3) 専門家グループ，フィッシングの自動化に警鐘
http://japan.cnet.com/news/media/story/0,2000047715,20076383,00.htm
Phishing Activity Trends Report November, 2004
http://www.antiphishing.org/APWG%20Phishing%20Activity%20Report%20-%20November%202004.pdf
4)高橋，村上，須藤，平原，佐々木，「フィールド調査によるボットネットの挙動解析」情報処理学会論文誌，第 47 巻第 8 号（2005）
5)内田勝也・高橋　正和「有害プログラム-その分類・メカニズム・対策」，共立出版　（2004）
6) John Shoch, Jon Hupp , “The "Worm" Programs - Early Experience with a Distributed Computation”,
Communications of the ACM, March 1982 Volume 25 Number 3, pp.172-180, ISSN 0001-0782, March 1982
http://vx.netlux.org/lib/ajm01.html
7) Decades after creation, viruses defy cure, Robert Lemos,
http://news.com.com/2009-7349_3-5111410.html
8) マカフィー社：WM/SHAREFUN.A
http://www.mcafee.com/japan/security/virS1999.asp?v=WM/SHAREFUN.A
9) CERT Coordination Center: Meet CERT
http://www.cert.org/meet_cert/meetcertcc.html
10) Dave Dittrich　University of Washington
DDoS attack tool timeline
http://staff.washington.edu/dittrich/talks/sec2000/timeline.html
11) Results of the Distributed-Systems Intruder Tools Workshop
http://www.cert.org/reports/dsit_workshop-final.html
12) Find Distributed Denial of Service (find_ddos)
by National Infrastructure Proctection Center
http://www.securityfocus.com/tools/822
13) CERT　Incident Note 99-04
Similar Attacks Using Various RPC Services
http://www.cert.org/incident_notes/IN-99-04.html
14) CERT　Incident Note IN-99-05
Systems Compromised Through a Vulnerability in am-utils
http://www.cert.org/incident_notes/IN-99-05.html
15) CERT Incident Note IN-99-07
Distributed Denial of Service Tools
http://www.cert.org/incident_notes/IN-99-07.html
16) Open Relay Database (ORDB)
http://www.ordb.org/
17) LOKI ICMP tunneling back door
http://xforce.iss.net/xforce/xfdb/1452
18) 残るはあと 1 サイト--NetSky の DDos 攻撃で 4 サイトがダウン
http://japan.cnet.com/news/sec/story/0,2000050480,20065376,00.htm
19) MyDoom ウイルス，一斉攻撃開始--SCO サイトがダウン
http://japan.cnet.com/news/ent/story/0,2000047623,20064064,00.htm
20) The Honynet Project & Research Alliance, “Know your Enemy: Tracking botnets”　http://www.honeynet.org/papers/bots/
21)　IPA: 近年の標的型攻撃に関する調査研究－調査報告書－
http://www.ipa.go.jp/security/fy19/reports/sequential/index.html
22) Vista one year vulnerability Report, Jeff Jones
http://blogs.technet.com/security/archive/2008/01/23/download-windows-vista-one-year-vulnerability-report.aspx
23) Microsoft Security Development Lifecycle
http://www.microsoft.com/security/sdl/default.aspx
24) Microsoft Security Development Lifecycle
http://www.microsoft.com/security/sdl/default.aspx
25) Microsoft Security Intelligence Report Ver.7
http://www.microsoft.com/downloads/details.aspx?displaylang=ja&FamilyID=037f3771-330e-4457-a52c-5b085dc0a4cd

26) Microsoft Security Intelligence Report Ver.7
http://www.microsoft.com/downloads/details.aspx?displaylang=ja&FamilyID=037f3771-330e-4457-a52c-5b085dc0a4cd