多数のマルウェア検体を並列解析可能な動的解析システムの提案

鉄穎† 田辺瑠偉† 水戸慎† 神薗雅紀‡ 星澤裕二†‡ 吉岡克成† 松本勉†

†横浜国立大学
240-8501 神奈川県横浜市保土ヶ谷区常盤台 79-7
{tie-ying-fc ,tanabe-rui-nv, mito-makoto-xy, yoshioka, tsutomu }@ynu.ac.jp

‡セキュアブレイン
102-0083 東京都千代田区麹町 2-6-7
{masaki_kamizono, yuji_hoshizawa}@securebrain.co.jp

あらまし 我々は，ボットのように攻撃者に操作されて不正な活動を行うマルウェアを，ネットワーク接続された動的解析環境内で長期的に動作させ，その振る舞いを継続的に観測することで，攻撃者の動向を把握する技術の研究開発を実施している．このように長期的に動的解析を実施する場合，多数のマルウェア検体を解析するためには実行環境も多数用意する必要がある．そこで，我々は効率的なシステム仮想化と適切なネットワーク制御により，限られた計算機資源で多数の検体を同時に並列解析可能なシステムの実装を提案する．提案システムでは，QEMU/KVMを利用し，Kernel SamePage Merging (KSM)技術を用いて，1台のサーバマシン上に50個以上の独立したマルウェア実行環境を実現可能である．また，仮想ネットワーク上でタグVLANを導入することで，実行環境間の接続を容易に設定可能とする．さらに，解析中の複数のマルウェア検体がサーバ化した場合にも，外部からの接続要求を適切に各実行環境へ転送するためのリバースプロキシ機能や，これらのネットワーク接続やルーティング設定を統合的に管理する管理サーバを導入する．

キーワード マルウェア動的解析，並列解析，統合管理，リバースプロキシ

A Malware Sandbox System for Analyzing Multiple Samples in Parallel

Ying Tie† Rui Tanabe† Makoto Mito† Masaki Kamizono‡ Yuji Hoshizawa†‡
Katsunari Yoshioka† Tsutomu Matsumoto†

†Yokohama National University
79-7 Tokiwadai, Hodogaya, Yokohama, Kanagawa 240-8501, Japan
{tie-ying-fc ,tanabe-rui-nv, mito-makoto-xy, yoshioka, tsutomu }@ynu.ac.jp
‡Secure Brain
2-6-7 Koujimati, Tiyoda, Tokyo 102-0083, Japan
{masaki_kamizono ,yuji_hoshizawa}@securebrain.co.jp

Abstract Malware sandbox is a flexible testing environment for observing malicious behaviors of malware. By observing a long-term behavior of the malware samples executed in the sandbox, we aim to monitor and trace malicious online activities of attackers who

remotely control malware infected hosts. However, analyzing long-term behavior of many malware samples would require many sandbox environments. Therefore, this paper proposes a malware sandbox system which can analyze multiple malware samples in parallel with a limited computational resource by utilizing virtualization technology and appropriate control of network. Using QEMU/KVM and Kernel SamePage Merging (KSM) technology we realize more than 50 independent sandboxes on a physical server. In addition, by introducing a tagged VLAN on the virtual network, we can easily set network connection between the sandboxes. Moreover, we deploy a reverse proxy server that can handle multiple connection requests from the Internet and redirect the requests to the appropriate sandboxes according to the pre-defined priority. Also, we set up an integrated management server to manage network connections and routing configurations.

Key words malware sandbox system, parallel analysis, integrated management, reverse proxy

1 はじめに

重要ファイルやパスワードの流出，盗聴，スパムメール，サービス妨害攻撃など，インターネット上のセキュリティ脅威の多くは，攻撃者により効率的に操作されたマルウェアが原因と言われている．我々は，このようなマルウェアの不正活動を国際的な連携により広域的かつ詳細に把握し，攻撃の予測を目指す PRACTICE (Proactive Response Against Cyber-attacks Through International Collaborative Exchange)プロジェクトを推進している．PRACTICE では，マルウェアを操作する攻撃者の動向を把握するため，多数のマルウェア検体を動的解析環境内で長期的に動作させ，その振る舞いを継続的に観測することを検討している．このように長期的に動的解析を実施する場合，多数のマルウェア検体を解析するためにはマルウェア実行環境も多数用意する必要がある．そこで，我々は効率的なシステム仮想化と適切なネットワーク制御により，限られた計算機資源で多数の検体を同時に並列解析可能なシステムの実装を提案する．提案システムでは，QEMU/KVMを利用し，KSM技術を用いて，1台のサーバマシン上に最大で50個以上の独立したマルウェア実行環境を実現可能である．また，仮想ネットワーク上でタグ VLAN を導入することで，実行環境間の接続を容易に設定可能とする．さらに，解析中の複数のマルウェア検体がサーバ化した場合にも，外部からの接続要求を適切に各実行環境へ転送できるようにリバースプロキシサーバを導入する．加えて，ネットワーク接続やルーティング設定を統合的に管理する機能も実現する．

本論文の構成は以下のとおりである．2 章で動的解析システムに求められる要件を説明し，3章では2章の要件を満たす動的解析システムを限られた計算機資源で実現する実装方法について述べる．4 章では提案した実装方法が 2 章の要件を満たす構成になっていることを検証・確認し，5章では先行研究を紹介する．最後に，6 章でまとめと今後の課題を述べる．

2 要件

本章では，PRACTICE プロジェクトにおいて動的解析システムに求められる要件を説明する．

・並列解析機能

前述のとおり，マルウェアを操作する攻撃者の動向を把握するためには，インターネット接続された動的解析環境内でマルウェア検体を長期的に動作させ続ける必要がある．また，提案システムの並列解析機能においては，複数

のマルウェアを同時に実行することによる解析結果への影響を考慮し，1 つの実行環境で動作させる検体を 1 つとしている．そのため，多数の検体を同時に解析し，攻撃者の動向を把握するためには，多数の実行環境が必要となる．具体的には，1 台のサーバマシンにより，数十個の仮想的な Windows マシンが動作し，その上でマルウェアを実行し，解析が行えることが望ましい．

・ネットワーク構成の柔軟性

各マルウェア実行環境を接続するネットワーク設定を柔軟に変更できる必要がある．例えば，動作環境が互いに干渉しないように独立したサブネットを適用したり，特定の動作環境群を同一サブネットに接続することが容易に可能であることが望まれる．また，インターネット上の外部ホストと通信する際に利用可能な IP アドレス数が限定的である場合は，NAT 等により IP アドレスの共有が可能であることが望まれる．

・サーバ機能をもつマルウェアへの適切な通信転送

マルウェア検体の中には攻撃の踏み台となるためにプロキシサーバとして動作するものや，感染ホスト間での通信のためポート待ち受け状態となるものが存在する．さらに一般家庭向けブロードバンドルータ等による NAT 接続状態においても各種サーバとして動作できるように，UPnP によりルータの通信転送設定を変更するものが存在する[5]．動的解析システムは，このような挙動も観測できることが望ましい．加えて，複数の検体がサーバとして動作している場合にも，事前に管理者が設定した優先順位に従い，優先度の高い環境に外部からの接続要求を転送する機能が必要である．

・統合管理

多数のマルウェア実行環境を効率的に利用するため，ネットワーク接続やルーティング設定から，通信トラフィックおよび各サーバ・ネットワーク機器の運行状態の監視までを統合的に管理できる必要がある．

・攻撃流出の防止

実行環境にて動作するマルウェア検体そのものやマルウェアからの攻撃が解析環境外に流出しないように適切なアクセス制御を行うとともにリモートエクスプロイト攻撃などの危険な通信についてはシステム内のダミーサーバ等に転送したり，必要に応じて管理者に警告を発行する機能をもつことが望まれる．

・ネットワーク機器・サーバ機器の安全性

マルウェアによりルータのルーティング設定が変更される可能性があるため，ネットワーク機器の設定状況の変更について把握し，必要に応じて管理者に警告を発行する機能をもつことが望まれる．

3 提案システム

2 章で説明した要件を満たす動的解析システムを，サーバマシン 3 台，ルータ 1 台，L2 スイッチ 1 台というシンプルな機器構成で実現する実装方法を提案する．

3.1 全体構成

本節では，提案システムの全体構成について述べる(図 1)．構成要素は並列解析環境サーバ，リバースプロキシサーバ，管理サーバ，L2 スイッチ，ルータの 5 つである．なお，以下では，30 台の仮想マシンを用意し，うち 29 台をマルウェア実行環境 VEMU1～VEMU29 として用い，残り 1 台をダミーサーバが動作するための環境として用いる場合について説明する．

並列解析環境サーバ:仮想マシンにより 29 台マルウェア実行環境を動作させる．図 1 の GuestOS#1～GuestOS#29 がマルウェア実行環境である．各仮想マシン間は仮想ネットワークで接続されているが，タグ VLAN を導入することで接続設定を容易に変更可能とする．また HostOS 上で，iptables[4]を利用し，危険性が高い通信トラフィックをダミーサーバ(図 1 の GuestOS#30)に転送する．ダミーサーバは

445/TCP などの脆弱なポートへの通信を受信するためのサーバ群であり，参考文献[4]と同様に低対話型ハニーポットプログラムや高対話型ハニーポットを用いる．

L2 スイッチ：並列解析環境サーバ，管理サーバと接続し，通信トラフィックを中継する．さらに，ポートミラーリングにより，並列解析環境サーバからの通信トラフィックを監視する．また，設定変更や機器故障などの場合，管理サーバへのイベント通知を行う．

ルータ：内部ネットワークとインターネット間，及び内部サブネット間のデータのやり取りを中継する．NAT により IP アドレスの共有を行ない，グローバル IP アドレスを節約する．実行環境を接続する VLAN 間のルーティングは行わないことで VLAN 間の独立性を保つ．また，UPnP 機能を有効にすることで，該当機能を悪用するマルウェア検体の挙動を観測できるようにする．

リバースプロキシサーバ：リバースプロキシサーバは，インターネット上の外部ホストからの同一ポートに対する接続要求を一旦受信し，事前に設定された優先度によって，適切に各実行環境に転送する．これにより，解析中の複数の検体がサーバとして動作している場合にも対応可能とする．

管理サーバ：管理サーバは，ネットワーク機器のルータ，L2 スイッチおよび並列解析環境サーバ，リバースプロキシサーバに対して，SNMP プロトコルを用いて運行状況を監視する．また，L2 スイッチのミラーポートを接続し，Trunk ポートを通過するトラフィックをすべて監視する．さらに，ネットワーク機器の設定変更やコンフィグのバックアップを行う．

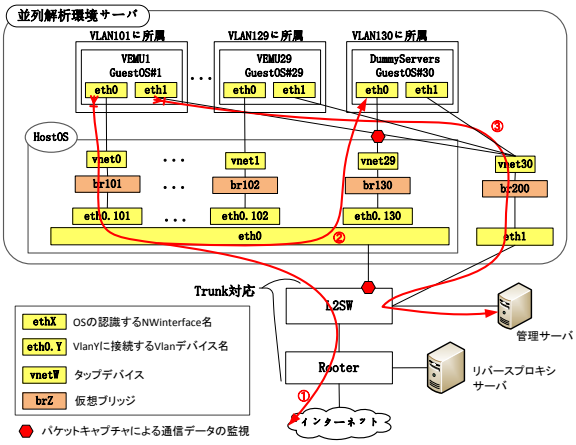

図 1. 多数のマルウェア検体を並列解析可能な動的解析システムの全体図

実行環境からのトラフィックの流れ：実行環境は並列解析環境サーバ上の仮想マシンにより実装する．実行環境 VEMU1～VEMU29 はそれぞれ仮想 NIC を 2 つ有し，eth0 を通常通信用，eth1 を運用管理用とする．eth0 からのトラフィックが各実行環境のタップデバイス vnet1～vnet29 に届き，次に，タップデバイスがセットされている仮想ブリッジ br101～br130 にそれぞれ届く．これで，トラフィックが仮想ブリッジを関連付けた VLAN デバイス eth0.101～eth0.130 に転送され，タグ付けされる[1, 2]．タグ付けされたトラヒックのうち，感染やサービス妨害攻撃などに繋がる危険性のある通信は図 1 の VLAN130 内の GuestOS#30 上で動作するダミーサーバ群に転送する（図 1 の赤線②）．一方，危険性が十分低いと判断される通信は HostOS の実 NIC eth0 から L2 スイッチ，ルータを経由してインターネットに出る（図 1 の赤線①）．トラヒックの危険性の判断については文献[4]の手法の利用を想定している．仮想 NIC eth1 からのトラフィックは，タップデバイス vnet30，仮想ブリッジ br200，HostOS の実 NIC eth1，さらに L2 スイッチを経由し，運用管理サーバに届く（図 1 の赤線③）．

3.2 サーバ機器

本節では，各サーバとネットワーク機器の具体的な設定，動作の仕組みを説明する．

・並列解析環境サーバ

並列解析環境サーバでは，2 章の並列解析機能にて述べた通り，限られた計算機資源内で多数の実行環境を実現する必要がある．これらの要件を実現するため，並列解析環境は QEMU[10] /KVM (Kernel-based Virtual Machine)[11]を利用して構築し，特に資源の効果的な使用を実現するため Linux Kernel の Kernel SamePage Merging (KSM)技術[9]を利用する．

KSM は，メモリ領域の同一内容のページを 1 つにまとめ，メモリの使用量を削減することで限りある資源を有効に使う技術である．

具体的には，カーネルスレッドである ksmd がプロセスの使用メモリ(ユーザメモリエリア)を定期的に監視し，重複したページを自動的にマージして共有ページとする．マージされたページは“Copy on Write(CoW)”状態に設定され，ページへの書き込みが発生するタイミングでマージしたページを分離する．

KVM 向けに Linux Kernel に組込まれた機能であり，Kernel 2.6.32 から標準機能として組込まれている．

1 つの実計算機上で同じ構成の GuestOS が複数稼働する並列解析環境において，同じ内容のメモリ領域が存在する可能性が非常に高く，KSM の恩恵を受けることかができる．

また，ハードディスクの使用容量を削減するため，並列解析環境サーバでは，同じ種類の OS となる GuestOS についてはマスターディスク 1 つを用意し，GuestOS 数分の差分ファイルを用いて並列環境を構築する．

・リバースプロキシサーバ

負荷分散型プロキシの一種である Pound[12]を利用して，リバースプロキシサーバを構築する．

図 2 を用いてリバースプロキシサーバの動作を説明する．例として，図 2 のように 3 つの実行環境においてサーバ機能をもつマルウェア検体が動作している場合を考える．このとき，各検体は UPnP プロトコルの脆弱性[5]を悪用し，ルータの設定を改ざんして，外部からのポート 8000/TCP への接続要求を各実行環境へ転送することを試みるものとする．

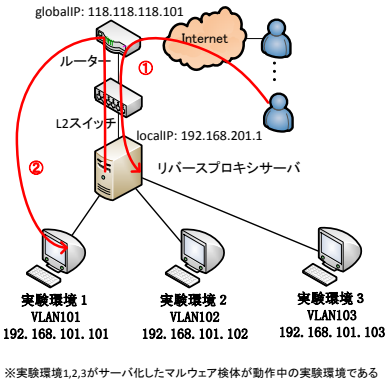

図 2. リバースプロキシサーバ動作説明図

外部からの接続要求はリバースプロキシサーバに届いた後(図 2 の赤線①)，内部の実行環境に転送される(図 2 の赤線②)．この際，接続要求をどの実行環境に転送するかをリバースプロキシサーバの設定により制御する．具体的には，pound.cfg と呼ばれる設定ファイル[3]の ListenHTTP ディレクティブにおいて，BackEnd として中継先サーバの IP アドレス，ポート番号,優先度を指定する．

・管理サーバ

管理サーバは L2 スイッチを流れるトラヒックのキャプチャと，ルータ，L2 スイッチ，各サーバの運用管理という 2 つの役割をもつ．この役割を果たすサーバの構築には様々な方法が考えられるが以下では，OS として Windows 7 Professional を想定した例を示す．まず NIC は 2 つ用意し，1 つを運用管理セグメントに接続し，もう1つを直接 L2 スイッチのミラーポットに接続し，Trunk ポートを通過するトラフィックをキャプチャできるようにする．キャプチャには，Wireshark 等のツールを利用する．

ルータ，L2 スイッチ，および，各サーバの運用・管理には teraterm など Windows 用ターミナルエミュレータを用いる．さらに，SNMP Service を有効にして，Windows 系監視ツール

TWSNMP マネージャ[6]を用いてネットワーク・サーバ機器の運行状況を常時監視する. 異常や故障が発生した場合は, 管理者にメールで通知を行う.

3.3 ネットワーク機器

・ルータ

L2 スイッチと接続するポートを Trunk ポートに指定し, VLAN ごとにサブインターフェイスを作成する. ルーティングテーブルにはインターネットと各マルウェア実行環境およびリバースプロキシサーバ関連のルーティングを設定する. 実行環境間の独立性を保障するため, VLAN 間ルーティングを ACL(Access Control List)によりブロックする.

UPnP 機能を用いるマルウェアを観測する際は, ルータの UPnP 機能を有効にし, マルウェアがルータとの XML ファイルのやり取りができるようにする. ルータ設定が変更された際は, SNMP エージェントにより, 管理サーバへ通知することで, 管理者が必要に応じて適当な対処法を取れるようにする.

なお, セキュリティ面を考慮し, 管理サーバしかルータにログインできないように ACL を設定する. 加えて, SSH サーバも動作させ, 公開鍵暗号方式による認証を行う.

・L2 スイッチ

並列解析環境サーバおよびルータに接続するポートを Trunk ポートにして, 並列解析環境サーバ上に設定している各実行環境の VLAN のフレームを通過させる. そして, ポートミラーリングを用いて Trunk ポートに通過するトラフィックをすべて監視する.

4 考察

本章では, 3 章で提案したシステムが 2 章の各要件を満たす構成になっていることを確認する.

並列解析機能: 提案システムでは, 並列解析環境サーバ上で動作する仮想マシンによりマルウェア実行環境を多数用意し, 並列解析を可能としている. 以下では, 並列解析環境サーバにおいてパフォーマンス検証を行った結果を示す.

今回使用した HostOS のスペック概要を表 1 に示す. GuestOS は Windows XP SP3 の一種類とし, 一般的なアプリケーションでありメモリを比較的多く使用する Apache バージョン 2 をサービスとして起動した状態のイメージを, 検証用として順次起動する.

表 1 HostOS のハードウェアスペック

CPU	Xeon
Memory	8G
HD	300G

本検証では, メモリのスワップが発生すると処理速度が低下することを考慮し, GuestOS を順次起動させスワップが発生するまでの GuestOS 数を計測することとする. 計測した結果, 68 個目の GuestOS を起動した際にスワップが発生した. これより, 表 1 のハードウェア構成では 67 個の GuestOS はスワップすることなく起動できることが確認できた. また, 一つの GuestOS を起動する度に, 120Mbytes 程度しかメモリ使用量が増加していないことが確認できた. こちらも, 効率的に資源を利用でき, 1 つサーバ上に多数実行環境で解析可能と言える. ただし, 実際にマルウェアを解析する際は, WindowsAPI をフックするような解析プログラムを起動するため, 現状の結果より多少性能が変化する可能性がある.

ネットワーク構成の柔軟性: 提案システムの論理構成図が図 3 に示す. 各実行環境は VLAN により異なるサブネットに所属するため, お互いに影響せずに動作させることができる. また, 実行環境は, KVM の GuestOS 上に実装されているので, GuestOS の接続先のタップデバイスを変更すれば, 別のサブネットに容易に参加

できる．これにより，ネットワーク構成の柔軟性が保障される．

サーバ化するマルウェアへの適切な通信転送: リバースプロキシサーバの導入により，外部からの同一ポートへの接続要求を複数の実行環境に優先度順で適切に転送できる．

統合管理: 管理サーバは，ネットワーク機器やサーバ機器の運行状況を監視・管理する機能をもつ．加えて，ルータと L2 スイッチのコンフィグの設定変更や通信トラフィックの監視も行う．

攻撃流出の防止: iptables[4]の設定により，危険性の高い通信をダミーサーバ群に転送することによって，攻撃流出を防止する．危険性の判断は参考文献[4]の手法を前提としているが，長期的動的解析に特化した判断基準については今後の課題とする．

ネットワーク機器・サーバ機器の安全性: ネットワーク機器やサーバマシンの SNMP エージェント機能を有効にし，管理サーバが各機器の稼動状況を監視可能とした．また，異常状態の場合，管理者への通知によって，各機器の安全性を保障する．

このように，提案システムは 2 章の要件を満たすといえる．

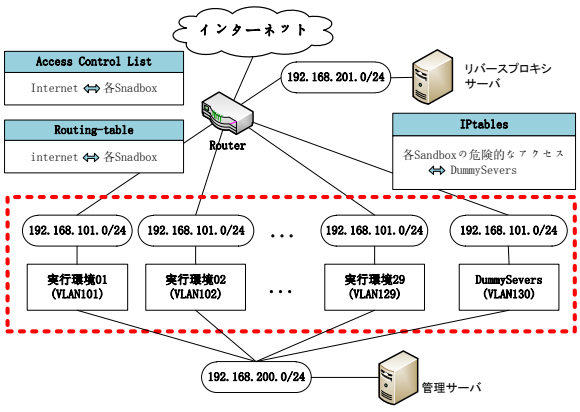

図 3. 多数のマルウェア検体を並列解析可能な動的解析システムの論理図

5 先行研究

nicter（Network Incident analysis Center for Tactical Emergency Response）は，NICT のネットワークセキュリティ研究所が開発した攻撃トラフィックの観測・分析システムである．論文[8]では nicter のマルウェア解析システムについて説明されている．このシステムでは，完全にインターネットから隔離の環境でマルウェアの挙動を分析している．また，仮想マシンを検知して動作を変更するマルウェアに対抗するため，仮想化技術を用いず実機を用いたマルウェア実行環境を適用していることから，並列解析を実現するには本稿で用いた仮想マシンによる並列化に比べて，より多くの計算機資源が必要になると思われる．

論文[4]では，ネットワーク接続型の解析システムを提案している．ほぼ隔離状態から解析を開始し，解析終了後に観測されたトラヒックを分析することで，危険性の低い通信を判別し，当該通信についてはインターネットへの通信を許可した上で再度，同様の検体の解析を行う解析手法を提案している．論文[4]では，マルウェア動的解析環境は 1 台の実機上で実現されているが，並列解析については検討されていない．

論文[7]では，インターネット接続型かつ多数の実行環境を 1 つのサーバマシン上で動かせるマルウェア動的解析システムを提案している点で本提案システムと関連が深い．しかし，論文[7]はマルウェア動的解析ではなく，高対話型ハニーポットの並列化を想定している．そのため，一般にハニーポットがマルウェアに攻撃を受けていないアイドル状態でいることが多い点に着目し，効率的な並列化を行っている．一方，マルウェア動的解析の場合，各実行環境では常にマルウェアが動作する状態になっており，このような高負荷状態での検証は論文[7]では行っていない．さらに，サーバ化したマルウェアへの適切な通信転送機能については触れられていない点が提案システムとは異なっている．

6　まとめと今後の課題

システム仮想化と適切なネットワーク制御により，限られた計算機資源で多数の検体を同時に並列解析可能なシステムを提案した．今後は，提案システムを実稼動させて，機能面を充実し，解析の自動化や安全性も詳細に検討する必要がある．また，実マルウェア検体を用いたパフォーマンス評価も実施予定である．

7　謝辞

本研究の一部は，総務省情報通信分野における研究開発委託／国際連携によるサイバー攻撃の予知技術の研究開発／サイバー攻撃情報とマルウェア実体の突合分析技術／類似判定に関する研究開発により行われた．

参考文献

[1] 中井悦司(2010)，「プロのための Linux システム構築・運用技術」，技術評論社．
[2] 中井悦司(2011)，「プロのための Linux システム・ネットワーク管理技術」，技術評論社．
[3] デージーネット(2005)，「Linux アドバンストネットワークサーバ構築ガイド」，秀和システム．
[4] K．Yoshioka and T．Matsumoto，“Multi-pass Malware Sandbox Analysis with Controlled Internet Connection，”IEICE Trans．Vol. E93-A，no.1，pp. 210-218, 2010.
[5] Playing with Universal Plug and Play,
http://www.symantec.com/connect/blogs/downadup-playing-universal-plug-and-play
[6] TWSNMP Manager,
http://www.twise.co.jp/twsnmp.html#5
[7] M. Vrable, J. Ma, J. Chen, D. Moore, E. Vandekieft, A.C. Snoeren, G.M. Voelker, and S. Savage,“Scalability, Fidelity, and Containment in the Potemkin Virtual Honeyfarm，” ACM Symposium on Operating System Principles (SOSP), vol. 39, no. 5, pp. 148 - 162, 2005.
http://www.read.seas.harvard.edu/~kohler/class/06f-aos/ref/vrable05scalability.pdf
[8] D. Inoue, K. Yoshioka, M. Eto, Y. Hoshizawa, K. Nakao, "Automated Malware Analysis System and Its Sandbox for Revealing Malware's Internal and External Activities," IEICE Trans. Vol. 92-D, pp. 945-954, 2009.
[9] Kernel SamePage Merging (KSM),
http://kernelnewbies.org/Linux_2_6_32.
[10] QEMU,
http://wiki.qemu.org/Main_Page.
[11] KVM,
http://www.linux-kvm.org/page/Main_Page.
[12]Pound，
http://www.apsis.ch/pound．