情報処理学会論文誌 Vol. 52 No. 12 3775–3785 (Dec. 2011)

推薦論文

ブラウザのアドオンを利用したアドウェアのイベント注入による振舞い解析

糟谷正樹[†1] 河野健二[†1,†2]

アドウェアは効率的な広告表示を行うために必要な情報を収集するソフトウェアである．一般にアドウェアは無害であるいわれている一方，パスワード情報などを収集する悪質なアドウェアも存在する．このような悪質なアドウェアの多くは Browser Helper Object（BHO）を利用しており，その対策のためには BHO ベースのアドウェアの動作解析を行うことが必要である．しかし，BHO を利用した悪意あるアドウェアは，ブラウザの一部として動作するうえ，頻繁な動作を行わず，暗号化や難読化がなされているため，その解析は容易ではない．本論文では，BHO をブラウザから分離し，BHO を動作させる偽のイベントを生成・注入してアドウェアを実際に動作させて振舞いを解析するシステムである Blayzard を提案する．Blayzard によって，BHO のみの動作を抽出することができ，ステルス性の高い悪意あるアドウェアを意図的に動作させて解析を行うことができる．また，動的解析を行うために暗号化や難読化の影響は受けない．31 個の実在する悪意あるアドウェアを集めて解析を行った結果，めったに動作しない悪意あるアドウェアや Windows のプロダクト ID を暗号化して外部に送信しているといった振舞いを抽出することができた．また，10 個のよく利用される BHO を集めて実験を行ったところ，これらの一部はアドウェアと似た振舞いをすることが分かった．

Using Event Injection for Behavior Analysis of Adware Based on Browser Add-ons

Masaki Kasuya[†1] and Kenji Kono[†1,†2]

Adware collects various information for targeted advertisement. Although adware is usually harmless, there exists malicious adware that sends out sensitive information such as password. Our target in this paper is malicious BHO-based adware because most malicious adware samples are based on Browser Helper Object (BHO). To detect and defend against malicious adware, it is mandatory to analyze behaviors of the samples. Unfortunately, behavior analysis of malicious adware is not easy for the following reasons. First, malicious adware behavior is blend in with that of normal browser behaviors. Second, malicious adware is highly stealthy to hide its presence. Finally, adware is encrypted and obfuscated to hinder manual and static code analysis. This paper proposes Blayzard, an analysis tool for BHO-based adware. Blayzard provides a stand-alone execution environment for adware, injects carefully-crafted events to reveal the behavior of stealthy adware, and automatically decrypt and deobfuscate adware code by dynamic execution. Our experimental results demonstrate that Blayzard can extract the behaviors of real adware samples. We collected 31 malicious samples and Blayzard could extract their behaviors. Blayzard also suggests that some benign BHOs shows the behaviors similar to adware.

1. はじめに

アドウェア[⋆1]は効率的な広告表示を行うために必要な情報を収集するソフトウェアである．一般にアドウェアは，コンピュータシステムを破壊したり，重要な情報を外部に送信したりしないため，しばしば害がないといわれている．しかし，実際にはアドウェアの中にもパスワード情報など本来収集するべきでない情報を外部に送信するものも存在することが知られている[1]．カスペルスキーによる調査[2)–5)]では，アドウェアが継続的に上位 15 位以内にランクされている．ほかにも，パンダセキュリティによる調査[6),7)]では，アドウェアは金銭被害を与えるソフトウェアであるという報告もあり，その被害の規模は増大しつつある．Moshchuk らにおける調査報告[8)]では 18,000,000 個の URL のうち 13.4%がスパイウェアを含んでおり，そのうち 91%がアドウェアの機能を持っていたと報告されている．

多くの悪意あるアドウェアが Internet Explorer（IE）の Browser Helper Object[9)]（BHO）を利用している[8),10)]．実際に，様々なマルウェアを収集しているウェブサイトの 1 つである Offensive Computing[11)]から入手した悪意あるアドウェアの約 1/3 が BHO に基づくもの

†1 慶應義塾大学理工学部情報工学科
Department of Information and Computer Science, Keio University
†2 科学技術振興機構 CREST
CREST, Japan Science and Technology Agency

⋆1 本論文では，アドウェアという用語を広義のアドウェアを指すものとして用いる．そのため，スパイウェアなどと統合されたものもアドウェアとして扱っている．

本論文の内容は 2010 年 10 月のコンピュータセキュリティシンポジウム 2010 にて報告され，同プログラム委員長により情報処理学会論文誌ジャーナルへの掲載が推薦された論文である．

であった．BHO はイベント駆動型の IE のアドオンであり，IE は処理内容に応じて様々なイベントを BHO に対して送信するため，対応するイベントハンドラを実装することによって IE の動作を変更したり，新しい機能を追加したりすることができる．たとえば，BHO が IE のページ遷移を制御することが可能になる．

悪意あるアドウェアの検出や防御手法を確立するためには，その振舞い解析を行うことが必要不可欠である．本論文でいう振舞いとはアドウェアが受信するイベント列，アドウェアが発行するコールバックメソッドや Windows のシステムコール列のことを表している．このような振舞いを抽出することができれば，アドウェアによる情報漏洩の内容を特定することができる．また，BHO の振舞いを監視することでアドウェアの存在を検知することも可能となる．

しかし，悪意あるアドウェアの振舞い解析を行うことは容易ではない．悪質なアドウェアは BHO としてブラウザに組み込まれて動作をするため，悪意あるアドウェアの動作とブラウザ本体の動作を区別しにくい．また，悪質なアドウェアの中には自身の存在を隠蔽するために頻繁な動作を避け，ブラウザ本体の動作に自身の動作を紛れ込ませるなどステルス性が高い．最後に，暗号化や難読化を行った悪意あるアドウェアの場合，デバッガやリバースエンジニアリングを用いた解析は困難となる．

本論文では，BHO ベースの悪意あるアドウェアの振舞い解析を行う解析システムである Blayzard を提案する．Blayzard では，ブラウザと BHO を分離し，BHO を動作させる偽のイベント列を生成・注入する動的解析を行う．以下に Blayzard の特徴を説明する．

- **IE と BHO の分離：**BHO は Dynamic Link Library（DLL）として IE プロセスに読み込まれるため，BHO が発行するシステムコールは IE が発行するシステムコールと区別することができない．そこで，BHO と IE プロセスを分離するために，Blayzard は両者を別々のプロセスに配置する．プロセスの分離によって，IE の影響を排除することができるため，Blayzard は BHO が発行するシステムコールのみを抽出することができる．また，IE と BHO を別プロセスに配置しているため，Blayzard は悪意ある BHO が IE のメモリを直接操作することを防ぐことができる．
- **偽のイベントの生成・注入：**ステルス性を有するアドウェアを活性化させるために，Blayzard は偽のイベント列を生成・注入することで，意図的に解析対象のアドウェアを動作させて解析を行う．通常，イベントは IE の動作状況に応じて生成する．しかし，Blayzard は IE の動作をともなわずに偽のイベントを生成できるため，容易に大量のイベント列を生成することが可能となる．その結果，稀にしか動作しない BHO を強制的に動作させることができる．

  また，偽のイベント列を注入することによって，暗号化してある通信内容を推測することができる．通信を行うシステムコールの引数を記録すれば，解析対象のアドウェアが外部に送信する情報を取得することができる．しかし，通信内容が暗号化してある場合は，その内容の判別を行うことができない．そこで，注入するイベント列の内容を工夫することによって，Blayzard では暗号化された通信内容の推測を支援するようになっている．たとえば，正しい情報（例：Windows のプロダクト ID）を含むイベント列と，偽の情報（例：偽の Windows のプロダクト ID）とを含むイベント列を注入し，解析対象のアドウェアが送信する通信内容の比較を行う．両者の通信内容が異なれば，プロダクト ID を送信している可能性が高い．
- **動的解析：**Blayzard は BHO ベースの悪意あるアドウェアの振舞いを抽出するために動的解析を行う．Blayzard が生成したイベント列を解析対象のアドウェアに意図的に注入することで，実際にアドウェアを動作させて，解析対象のアドウェアが実行したコールバックメソッドやシステムコール列を抽出する．動的解析を用いる利点の 1 つは，暗号化や難読化に耐性を持つことである．そのため，暗号化や難読化が行われている場合であっても，暗号化や難読化の影響を受けずに解析を行うことができる．

Blayzard の有用性を示すために Blayzard を実装し，マルウェア収集サイト[11),12)]などから 31 個の BHO ベースの悪意あるアドウェアを収集し，Blayzard による振舞い解析を行った．その結果，収集したすべての悪意あるアドウェアの振舞いを抽出することができた．特に，ステルス性の高いアドウェアであっても，偽のイベント列を大量に注入することで活性化させ，その振舞いを解析することができた．また，虚偽の情報を含むイベント列を注入することによって，Windows のプロダクト ID を暗号化して送信する振舞いを解析することができた．Blayzard による解析結果が正しいことを確認するため，商用のディスアセンブラである IDA Pro[13)] を用いて検体の解析を行ったところ，Blayzard の解析結果は正しいことが確認できた．

本論文の構成は以下のとおりである．2 章では，BHO の概要とアドウェアによる利用方法を説明する．3 章では，Blayzard の設計と実装について説明する．4 章では，実験結果を示し，5 章では，関連研究について取りあげる．最後に，本論文を 6 章でまとめる．

2. BHO ベースのアドウェアの振舞い

BHO は IE のプロセスに読み込まれる DLL である．BHO はイベント駆動型のアドオン

であり，Windows の Component Object Model [14]（COM）を利用する．IE は動作状況に合わせて様々なイベントを生成し BHO に送信する．たとえば，ブラウザがあるリンクをたどって別のページに遷移する際，`BeforeNavigate2` というイベントを IE が生成し，遷移先の URL，HTTP ヘッダ，POST データなどをイベントの引数として BHO に渡す．

IE は `Navigate` や `LocationURL` などの様々なコールバックメソッドを提供している．たとえば，ページ遷移を行うときに生成される `BeforeNavigate2` イベントのハンドラ内で `Navigate` メソッドを呼び出すと，ページの遷移先を変更することができ，本来遷移すべきウェブページとは異なるウェブページに遷移させることができる．また，IE と BHO は同一プロセスとして動作するため，BHO は IE と同じ権限でシステムコールを実行することができる．そのため，IE がアクセスできる様々な情報を取得できる．たとえば，レジストリの値を取得する `NtQueryValueKey` というシステムコールを発行すれば，BHO もレジストリの値を取得することができる．

IE 以外のウェブブラウザでも BHO に相当するアドオンを利用することができる．Blayzard は IE と BHO に特化した実装となっているものの，他のブラウザ用のアドオンでも Blayzard と同等の機能を実現することが可能である．これは多くのブラウザが IE と同様にイベントやコールバックによるアドオンを提供しているからである．

3. Blayzard

Blayzard は BHO を動作させるイベントを入力し，BHO ベースの悪意あるアドウェアの振舞いを抽出する解析システムである．本論文でいう振舞いとは，BHO ベースのアドウェアが受信するイベント，発行するコールバックメソッドやシステムコールのことである．Blayzard はステルス性のあるアドウェアを解析するために，単純に解析対象のアドウェアを監視するのではなく，偽のイベントを注入してその振舞いを抽出する．Blayzard が生成する偽のイベントは，悪意あるアドウェアに偽のイベントであることを気付かれないように，ブラウザが送信する実際のイベント列と類似したものになるようにしている．これに加えて，偽のイベントは引数に与える URL を変更して，どのような情報が外部に流出したかを知る手がかりになるように生成している．以下に Blayzard の設計および実装の詳細について説明する．

3.1 IE と BHO の分離

BHO ベースのアドウェアは IE と同一プロセス上で動作するため，Blayzard は BHO ベースのアドウェアと IE の振舞いを区別することができない．たとえば，システムコールを発行する場合，Blayzard はそのシステムコールが IE と BHO のどちらが発行したのかを区別できない．BHO のみの振舞いを抽出するために，Blayzard は IE と BHO をそれぞれブラウザプロセスと BHO プロセスに配置する．プロセスの分離を行うことで，IE の影響を排除して BHO の振舞いを調べることができる．

図 1 に Blayzard の構成を示す．BHO プロセスは BHO が独立したプロセスとして動作できるようにした実行環境である．BHO プロセスには Event Dispatcher モジュール，Callback Method Invoker モジュールが組み込まれている．IE プロセスには BHO として Event Generator モジュール，Callback Dispatcher モジュール，Behavior Logger モジュールが組み込まれている．BHO プロセス内の Event Dispatcher と Callback Method Invoker は，IE が BHO に対して提供するのとまったく同じ API を提供しており，BHO プロセスに組み込まれたアドウェアからは IE と同等に見える．また，Event Generator，Callback Dispatcher，Behavior Logger は IE の BHO として組み込んであるため，IE からはこれらのモジュールが BHO として見える．Event Generator は偽のイベントを生成し，IPC（inter-process communication）を用いて BHO プロセスにイベントを送信する．BHO プロセス内の Event Dispatcher がそのイベントを受信し，解析対象のアドウェアに配信する．イベントを受信したアドウェアがコールバックメソッドを起動すると Callback Method Invoker が起動され，IPC により IE プロセス内の Callback Dispatcher に通知される．Callback Dispatcher は IE の提供するコールバックを実際に呼び出す．監視を行っている BHO の振舞いを抽出するために，受信したイベント列，コールバックメソッドやシ

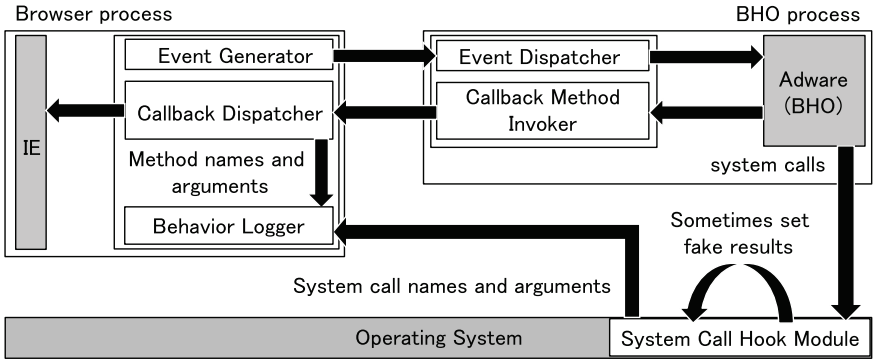

図 1　Blayzard の構成
Fig. 1　Architecture of Blayzard.

ステムコール列は Behavior Logger に送られ，その引数とともに記録されるようになっている．悪意あるアドウェアによるログの改竄が起きないように Behavior Logger は BHO プロセスではなく IE プロセス内に配置している．

なお，解析対象のアドウェアが IE プロセス内で動作していると思わせるため，BHO プロセスの実行イメージ名は IE と同じ `iexplore.exe` としている．実際，自身が実行するプロセス名が `iexplore.exe` であるときのみ，悪意ある動作を行うアドウェアが存在している．

3.2　偽のイベントの生成と挿入

BHO ベースの悪意あるアドウェアの振舞いを抽出するために，Blayzard は偽のイベントを生成し，解析対象のアドウェアに注入する．偽のイベントは解析対象のアドウェアが通常どおり動作するように生成する必要がある．これは，適切でないイベントをアドウェアに送信することで，悪意あるアドウェアが Blayzard による監視に気付いてしまい，悪意ある振舞いを行わない可能性があるためである．

Blayzard はブラウザが実際に発行するイベントの順番を真似て，偽のイベントをあたかも IE が生成したように作成している．実際，ブラウザはその実行状態に応じて様々なイベントを生成し BHO に送信する．Blayzard ではブラウザの実行状態として，初期化，ページ遷移，タブおよびウィンドウの生成，ブラウザの終了という 5 つの実行状態を用意し，それぞれの状態に応じて実際のブラウザが送信するのと同等のイベント列を生成するようにしている．

表 1 に Blayzard で用意されているイベントの一覧を示す．Blayzard では，IE が BHO に送信する 45 種類のイベントのうち 40 種類のイベントを送信することができる．実装していない 5 種類のイベントは，そもそも IE で実装されていないイベントや，仕様の詳細が公開されていないイベントである[15),16)]．上述の 5 つの実行状態を用いると，実装された 40 種類のイベントのうち 31 種類が自動的に注入されるようになっている．自動的に注入されることのない 9（= 40 − 31）種類のイベントは，通常のブラウジング動作で発行されることは稀であるため，必要に応じて Blayzard の利用者が手動で注入する必要がある．

たとえばブラウザのページ遷移を模するために，ページ遷移を行う前に生成される `BeforeNavigate2` イベント，ページ遷移が完了したときに生成される `NavigateComplete2` イベント，ドキュメントの表示が完了したときに生成される `DocumentComplete` イベントなどを順に生成する．ただし，実際にブラウザがページ遷移を行うときには，タイトルの変化を通知する `TitleChange` イベントや，より詳細な状態変化を通知する `StatusExchange` イベントなどが通知される．こうしたイベントも適切に再現することによって，BHO ベースのアドウェアから見れば，実際のウェブブラウザが動作しているように見えるようになっている．

表 1　Blayzard が用意しているイベントと用意していないイベント
Table 1　Prepared and non prepared events by Blayzard.

自動注入を行うイベント		
BeforeNavigate	NavigateComplete	SetSecureLockIcon
StatusTextChange	Quit	DownloadComplete
DownloadBegin	NewWindow	CommandStateChange
ProgressChange	PropertyChange	TitleChange
FrameBeforeNavigate	FrameNavigateComplete	TitleIconChange
BeforeNavigate2	NewWindow2	DocumentComplete
NavigateComplete2	OnQuit	OnAddressbar
OnVisible	OnToolbar	OnMenubar
FileDownload	NavigateError	OnStatusbar
PrivacyImpactedStateChange	NewWindow3	SetPhishingFilterStatus
WindowStateChanged		
手動注入を行うイベント		
OnFullScreen	OnTheaterMode	WindowSetResizable
WindowClosing	WindowSetLeft	WindowSetTop
WindowSetWidth	WindowSetHeight	ClientToHostWindow
未実装のイベント		
WindowMove	WindowResize	WindowActivate
FrameNewWindow	ViewUpdate	

すでに 2 章で述べたように IE が BHO に通知するイベントには引数をともなうものが多い．これらの引数の多くは毎回同じ値が設定されていることが多いため，Blayzard もその値を設定するようにしている．しかし，たとえば遷移先の URL などを引数としてともなうイベントの場合，Blayzard のほうで適切な値を設定する必要がある．現在の Blayzard の実装ではあらかじめ定めておいたいくつかの値から選択した値を設定するようになっている．いくつかの値から選択することで，引数が特定の値の場合にのみ動作するようなアドウェアでも活性化できる可能性が高まる．実際，4 章の実験では，Google など特定のサイトを訪問したときにのみ活性化するアドウェアがあった．より現実的なイベント列を生成するためには，IE の操作履歴などを多数集め，その履歴に沿った形でイベント列を生成することが望ましい．

Blayzard ではステルス性の高いアドウェアを活性化させるため，ブラウザの初期化から終了までの一連のイベントを数回から十数回にわたって注入するようにしている．このよう

にすることで，何回かに 1 度だけ活性化するアドウェアであっても活性化させることができる．また，ブラウザの初期化から終了までのイベント注入を繰り返す際，イベントにともなう引数は毎回違った値を選択するようにしている．これは引数が特定の値のときのみ活性化するアドウェアでも活性化させるためである．なお，このようなアドウェアの場合，Blayzard を用いても活性化できないことがある．それについては，3.4 節で議論する．

2 章で述べたように BHO ベースのアドウェアは IE と同じ権限でシステムコールを実行することができる．悪意あるアドウェアの振舞い解析を行うため，Blayzard では特定のシステムコールの返値を改竄するようにしている．たとえば，レジストリの値を取得するシステムコールを実行した場合，数回に 1 度，虚偽の値を返すようにしている．これはすでに述べたように，システムコールから取得した値を暗号化して送信するアドウェアの振舞い解析を支援するためである．

3.3 BHO の振舞いの抽出

Blayzard はアドウェアの振舞いとして，解析対象のアドウェアが受信したイベント列，起動したコールバックおよびシステムコールとその引数を記録する．図 1 に示したように，IE が送信するイベント列，起動されたコールバックとその引数は Event Generator，Callback Dispatcher で取得する．現在，700 以上のコールバック関数に対応しており，BHO が利用するコールバックメソッドを抽出するのに十分である．BHO プロセスが実行したシステムコール列は，オペレーティングシステムに組み込んだシステムコールのフック・モジュールによってフックされるようになっている．このフック・モジュールは `sysenter` 命令の飛び先を変更することで，システムコールのフックを行うようになっている．このフック・モジュールはシステムコールとその引数を IE プロセス内の Behavior Logger に通知するだけではなく，特定のシステムコールについては適宜，虚偽の返値を返すようになっている．

BHO ベースのアドウェアが別のプロセスを立ち上げ，アドウェアが取得した情報をそのプロセスを使って外部に送信する場合がある．Blayzard では新しいプロセスを作成する `NtOpenProcess` をフックし，BHO プロセスが起動したプロセスについてもシステムコールのフックを行うようになっている．このようにすることで，BHO プロセスの振舞いを追跡することが可能になっている．

3.4 Blayzard の制限

Blayzard では適切なイベントを解析対象のアドウェアに注入することによってアドウェアの振舞い解析を行っている．そのため，解析対象のアドウェアに与えるイベントが不適切であると，正しくその振舞いを解析することができない．たとえば，イベントの引数が特定の値のときのみ活性化するアドウェアの場合，その特定の値を注入することができなければそのアドウェアを活性化させることができず，振舞い解析を行うことはできない．たとえば，特定の URL を訪問したときだけ活性化するアドウェアの場合，その URL をイベントの引数として注入する必要がある．

Blayzard の現在の実装では，イベントの引数としてあらかじめ定めたいくつかの値のみを利用している．たとえば，イベントの引数として使われる URL には `google.com`，`yahoo.com` などを用意している．4 章でも示すように，多くのユーザが訪問する URL を用意しておくだけでも十分に効果的である．しかし，あらゆる URL を事前に準備しておくことは現実的ではなく，稀にしか訪問されることのない URL に対してのみ活性化するアドウェアについては Blayzard での解析には適さない．このような場合，Fuzzing や静的解析を援用することが効果的であると考えられる．

また，ブラウザが表示しているウェブページの内容によって動作を変えるアドウェアが存在する場合，Blayzard では適切な解析結果を得ることができない．BHO からウェブページの内容を取得するためには，`get_innerHTML` というコールバックメソッドを呼び出す．このようなアドウェアの解析を行うためには `get_innerHTML` の返値として様々な内容を返さなければならない．しかし，Blayzard を用いて `get_innerHTML` が呼び出されていることは解析できるため，それを手がかりに静的解析などを援用することにより，より詳細な解析結果を得ることができる．

静的解析と動的解析の網羅性と簡易性はトレードオフの関係にある．Blayzard は動的解析を行っているため，すべての実行経路を網羅することは難しい．一方，静的解析を行えば理論上はすべての実行経路を網羅できる．しかし，アドウェアのバイナリが暗号化・難読化を行っている場合，IDA Pro などを用いた静的解析によって暗号化・難読化を行ったアドウェアの動作を理解するためには相応の技量と時間を必要とする．一方，動的解析を行うとバイナリの暗号化・難読化によらず，実行時の振舞いを抽出することができる．

4. 実　　験

実在する悪意あるアドウェアとよく利用される BHO を対象に，Blayzard の解析結果を示す．評価に用いるアドウェアは悪質なものを対象としているため，31 個の悪意あるアドウェアを Offensive Computing [11] や VX Heavens [12] などのマルウェア収集サイトから収集した．また，Google Toolbar などのよく利用される BHO を 10 個収集した．実験は Windows XP SP3 上の IE 7（ver. 7.0.5730.13）と IE 8（ver. 8.0.6001.18702）を用いた．また，悪

意あるアドウェアによる被害を防ぐため仮想化環境である VMware Fusion 3.1.1 上で実験を行った．

ここで用いたアドウェアの検体の中には，Browser Accelerator のように文献によってアドウェア（たとえば文献 17)）またはスパイウェア（たとえば文献 18)）に分類されているような検体も含まれている．Blayzard は悪質な振舞いを行うアドウェアをその解析対象としているため，そのような検体も含めて実験を行っている．

4.1　悪意あるアドウェアの振舞い

表 2 に実験に用いた悪意あるアドウェアとその振舞いの一覧を示す．表 2 に示したように，Blayzard を用いてすべてのアドウェアの振舞い解析を行うことができた．この解析結果から分かるようにアドウェアの振舞いは，URL を流出しつつページ遷移を行う，広告表示のためのスクリプトやテキストの挿入，Windows のプロダクト ID や Cookie の流出など，多岐にわたっている．なお，IE 7 と IE 8 のブラウザのバージョンの違いによる解析結果の違いは確認できなかった．以下，特に興味深い振舞いを行った 3 つのアドウェアについて詳細な解析結果を示す．

4.1.1　`xml2u32h.dll` の振舞い

このアドウェアは Google などの検索エンジンに入力した検索語と Windows のプロダクト ID を外部に送信する．これらの情報を外部に送信するために，`xml2u32h.dll` は HTTP の GET メソッドを用いている．そのメッセージフォーマットは**表 3** のとおりである．

ページ遷移を行うことを示す `BeforeNavigate2` イベントを受け取ると，その後，`xml2u32h.dll` は引数として渡される URL に応じて送信するメッセージ内容を変更する．Blayzard では `BeforeNavigate2` イベントの引数として Google，Yahoo などのよく知られた URL を与えるようになっているため，このような振舞いを解析することができた．実際，Google，Yahoo，MSN を訪問したとき，送信メッセージ内の検索エンジンを表す数字（表 3）がそれぞれ 1，2，3 となる．また，検索語は `BeforeNavigate2` イベントの引数として渡される URL に含まれるものから取得していると予想できる．また，表 3 の暗号化してある文字列の部分は Windows のプロダクト ID が含まれている．これは，`xml2u32h.dll` がレジストリから Windows のプロダクト ID を入手していること，偽のプロダクト ID を返すと暗号化された部分の文字列が変化することから分かる．

Blayzard による振舞い解析の結果を検証するために，商用のディスアセンブラである IDA Pro [13] を利用して確認を行った．`BeforeNavigate2` メソッドの引数から URL，検索語を得ていること，検索エンジンによって 1，2，3 の数値を選んでいること，暗号化された文

表 2　評価を行った悪意あるアドウェアとその振舞い
Table 2　Evaluated malicious adware samples and the behavior.

アドウェア名	抽出できた振舞い
xsfer.dll	`Navigate` を用いて URL を流出させる
autoex.dll	`Navigate` を用いて URL を流出させる
C Java Object	アダルトサイトにアクセスする
ini.dll	`write` を用いてスクリプトを挿入する
nada64.dll	`write` を用いてスクリプトを挿入する
navfilter.dll	`write` を用いてスクリプトを挿入する
iefltr.dll	`write` を用いてスクリプトを挿入する
nvgf.dll	`write` を用いてスクリプトを挿入する
tasdgim.dll	`write` を用いてスクリプトを挿入する
dhofozr.dll	`write` を用いてテキストを挿入する
sdsheol.dll	`write` を用いてテキストを挿入する
gofpa.dll	`write` を用いてテキストを挿入する
uslpazhfbcbafrssa.dll	`showBrowserBar` を用いて検索語を外部に送信する
skype Helper Object	定期的にブラウザの起動情報を送信する
BhoApp Class	Cookie と URL を流出する
IeHelperEx.dll	レジストリに値を追加する
msfacat32.dll	警告メッセージを読み込む
ylsvevnbhjdaj.dll	HTML ファイルをダウンロードする
jwixfukqsgxwy.dll	テキストファイルをダウンロードする
acrobat.dll	コンピュータ名を流出させる
UCMTSAIE.dll	URL を収集する
BrowserAccelerator.dll	URL 流出する
Platrium.dll	URL を流出する
cpush.dll	広告を表示する
marwin32.dll	広告を表示する
baiduc.dll	広告を表示する
mqodcuwucedvr.dll	`get_innerHTML` と `put_innerHTML` を呼び出す
QQMenu	イベントを受信後，表示しているウィンドウを廃棄する
LsVpd	`get_cookie` を用いて Cookie を収集する
xml2u32h.dll	検索語とプロダクト ID を外部に送信する
mws31209.dll	検索語とプロダクト ID を外部に送信する

表 3　`xml2u32h.dll` が情報を外部に送信するために利用する GET メソッド
Table 3　GET method used by `xml2u32h.dll` to leak sensitive information.

GET メソッドの内容
GET /qinfr2?| 検索語	暗号化文字列	検索エンジンを表す数字　HTTP/1.1

字列は Windows のプロダクト ID を含んでいることが確認できた．

4.1.2 BhoApp Class の振舞い

BhoApp Class はブラウザが所有する Cookie と URL を収集して，暗号化を行い外部に送信する．Cookie と URL を入手していることは，NavigateComplete2 イベントを受信したときに get_LocationURL メソッドと get_cookie メソッドのコールバックを行っていることから推察できる．また，取得したシステムコール列から，system32 ディレクトリに perfz9368.dat というファイルを作成し何らかの情報を保存していることが分かった．Cookie や URL の値を変更すると，perfz9368.dat ファイルに保存される内容が変わることから，このファイルには Cookie や URL を暗号化して保存されていることが推察される．

BhoApp Class では perfz9368.dat ファイルが一定のサイズを超えるとその内容を外部に送信するようになっている．これは Blayzard が多くのイベントを注入するようになっているため，外部に情報を送信する動作を発動させることができた．Blayzard を用いて大量のイベントを注入すると，BhoApp Class が perfz9368.dat の内容を POST メソッドを用いて送信していることが確認できた．

IDA Pro[13] を用いて解析を行ったところ，xor 0x96 による暗号化を行って URL と Cookie を含む文字列を外部に送信していることが確認できた．このことから，Blayzard による解析が正しいことが確認できる．

4.1.3 baiduc.dll の振舞い

baiduc.dll は広告表示を行い，MAC アドレス，URL およびこのアドウェアが生成した ID を外部に送信している．baiduc.dll を BHO プロセスに読み込むと，system32 ディレクトリに mprmsgse.axz というファイルを作成し，そのファイルに何らかの値を書き込んでいることが確認できる．その後，ドキュメントの表示完了を知らせる DocumentComplete イベントを受信すると，get_LocationURL メソッドをコールバックし，ドキュメントの URL を取得していることが確認できる．その後，MAC アドレスおよび mprmsgse.axz に書き込んだ値を外部に送信していることが解析できる．

なお，baiduc.dll を含め，広告を表示させるタイプのアドウェアには共通する特徴的な動作がある．それは，IE のウィンドウがフォーカスされているときのみ広告が表示されるという点である．Blayzard では偽のイベントを送り込む際には，IE のウィンドウをフォーカスするようにしている．

4.2 アドウェアではない BHO の振舞い

アドウェアとは見なされていない BHO の振舞いを調べるために，Google Toolbar などの 10 個の BHO の振舞い解析を行った．**表 4** に解析結果を示す．8 個の BHO は広告を表示したり情報を流出させたりすることはなかったものの，Google Toolbar と Earthlink は情報漏洩を行うアドウェアに似た振舞いをすることが分かった．以下，Google Toolbar と Earthlink について詳細を述べる．

表 4　評価を行ったよく利用される BHO とその振舞い
Table 4　Evaluated benign BHOs and the behavior.

名前	抽出できた振舞い
Google Toolbar	URL やバージョン情報などを送信する
Earth link	OS の種類を流出する
Yahoo! Toolbar	バージョン情報などは送信しない
Bing Toolbar	バージョン情報などは送信しない
MicroGuarden	バージョン情報などは送信しない
Spybot S&D	バージョン情報などは送信しない
Super Ad Blocker	バージョン情報などは送信しない
Alexa Toolbar	バージョン情報などは送信しない
Spyware Guard	バージョン情報などは送信しない
MoreGoogle	バージョン情報などは送信しない

表 5　Google Toolbar が送信したシステム情報
Table 5　Sent sensitive information by Google Toolbar.

ホスト名	外部に送信された通信内容
clients4.google.com	POST /tbproxy/...&v=6.5.708.1000&*browser=7.0.5730.13* &rlz=1T4GGLL_jaJP375JP376&... HTTP/1.1
www.google.com	GET /tools/...&*osver=5.1*&*ossp=3.0*&*osarch=32* &*browser=7.0.5730.13*&*browserarch=32*&... HTTP/1.1
toolbarqueries.google.co.jp	GET /tbr?client=navclient-auto&... &q=info:*http://www.a-blog.jp/*&... HTTP/1.1

4.2.1 Google Toolbar

Google Toolbar は訪問した URL や OS やブラウザのバージョンを外部に送信するなど，情報漏洩を行うアドウェアと似た振舞いをすることが分かった．**表 5** に Google Toolbar が送信した情報の一部を示す．これから分かるように，clients4.google.com に対してブラウザのバージョンを送信し，www.google.com に対して OS のバージョンやブラウザのバージョン

などを送信している．また，ページ遷移が完了したときに送信される NavigateComplete2 イベントを受信したとき，引数として受け渡される toolbarqueries.google.co.jp の URL を送信している．

4.2.2 Earthlink

Earthlink は使用している OS の種類を HTTP の GET メソッドを用いて外部に送信している．また，ドキュメントの表示が完了したことを通知する DocumentComplete イベントを受信したとき，広告表示に似た振舞いを行うことが分かった．DocumentComplete イベントを受信すると，数回に 1 度，insertAdjacentHTML メソッドをコールバックしポップアップを表示しようとしている．ただし，その URL はすでに無効であった．

5. 関 連 研 究

5.1 様々なアドウェア対策ツール

現在，数多くのアドウェア検出ツールや除去ツールが存在しており，実際に，Ad-Aware [19]，Spybot S&D [20]，Snort [21] や Bro [22] などのツールがある．これらのツールを利用するためには，あらかじめ該当するアドウェアを解析して得られる定義ファイルを用意する必要がある．定義ファイルは，主に人手による解析を行って作成している．しかし，マルウェアの数は年々増加し続けており，アドウェアにおいてもこのことは例外ではない．そのため解析を行う作業は煩雑となるため，効率良く解析を行う必要がある．Blayzard は BHO ベースのアドウェアの解析を行うために，ブラウザと BHO を分離し，偽のイベントを解析対象のアドウェアに注入し，その振舞いを抽出する動的解析を行うことで，アドウェアが発行したコールバックメソッド，システムコール列の呼び出しや，どのような情報を取得し，外部に送信したかを容易に知ることができる．Blayzard を利用することで，BHO ベースのアドウェアの解析を容易に行うことができる．そのため，定義ファイルの作成が容易になったり，より詳細な解析を行うための手がかりを得ることができる．

5.2 解析を行う環境を識別するマルウェアへの対策手法

近年，マルウェア解析を行うために VMware [23] などの仮想化環境や Qemu [24] などのエミュレータを用いて，解析を行う手法を様々な研究者が提案している．これらを用いた解析手法はマルウェアだけでなく，OS を含めたシステム全体を監視することができる．そのため，マルウェアは様々な方法[25)–28)] を用いて仮想化環境やエミュレータ環境を識別し，そのような環境でマルウェアを動作することを避ける．解析環境を識別するマルウェアに対抗するために，Cobra [29] や Ether [30] は解析環境を識別するマルウェアをそのような環境上で動作していないように騙し，解析を行うシステムである．現在のマルウェアは Cobra [29] や Ether [30] に対抗できないといわれているものの，解析環境の識別を回避する時間のオーバヘッドが大きいため，数多くのマルウェアを解析することには向いていない．Blayzard は仮想化環境やエミュレータに依存したシステムではないため，解析環境を識別する BHO ベースのアドウェアによる影響は受けない．

5.3 スパイウェアの対策手法

NetSpy [18] や Siren [31] はスパイウェアに感染した環境と感染していない環境を用意しておき，両者に同一の入力を与えたときに，出力が同じになるかどうかを調べてスパイウェアの検出を行う手法である．しかし，Blayzard はスパイウェアに感染した環境と感染していない環境を用意する必要はない．NetSpy [18] や Siren [31] では，仮想化環境や侵入検知システムを利用している．しかし，Blayzard はそのような環境に依存するシステムではない．

Kirda ら[32] は動的解析と静的解析を組み合わせ，BHO のイベントハンドラ内で Win32 API の send などの情報を外部に送信する可能性のある API が存在するかどうかを調べ，スパイウェアの検出を行う手法である．しかし，Kirda らの手法は静的解析を含んでいるため，暗号化や難読化が行われていると解析が困難になる．しかしながら，Kirda らの手法と Blayzard はお互いを補完することができる．静的解析を回避する場合は Blayzard を用いることで，どのような情報が外部に送信するかを調べることができる．反対に Blayzard が不得意とするすべての実行経路を調べたい場合は Kirda らの手法を用いることで，より詳細な結果が得ることができる．

SpyShield [17] は IE と BHO を分離して，BHO による不正なページ遷移や Cookie の取得などを防ぐためにセキュリティポリシを記述し，BHO の動作を制限する手法である．そのため，SpyShield はユーザがセキュリティポリシを設定する必要がある．しかし，適切なセキュリティポリシをユーザが設定することは難しい．いずれの手法[17),18),31),32)] も BHO を活性化させるために偽のイベントを利用しない．

5.4 Taint Analysis を用いた手法

近年，スパイウェアやアドウェアの対策手法として Taint Analysis が有用であるといわれている．Taint Analysis とはデータの流れに着目し，着目したデータが外部に送信される場合に，警告を出すシステムである．Egele ら[33] はスパイウェアの検出を行うために，Taint Analysis を用いて着目したデータを IE と BHO のどちらで利用しているかを調べ，BHO 側で着目したデータが外部に送信する場合はスパイウェアによるデータ送信と判断し，詳細な解析結果を提供する手法である．Panorama [34] はスパイウェアに限らずログインパス

ワードなどの情報にアクセスするマルウェアを検出および解析を行う手法である．ほかにも様々な Taint Analysis に関する研究[35)–39)] が存在する．しかしながら，Cavallaro ら[40)] は Egele ら[33)] の手法や Panorama[34)] に関する回避手法を示しており，Taint Analysis は万能な手法ではないことを示している．しかし，Blayzard は Taint Analysis を用いた手法ではない．

6. ま と め

近年，悪意あるアドウェアがセキュリティ上の問題となっており，アドウェアの対策を行うことは避けられない．悪意あるアドウェアの検出や防御手法を考案するために，アドウェアの振舞い解析を行うことは必要不可欠である．しかしながら，BHO ベースの悪意あるアドウェアがブラウザプロセスの一部として動作すること，ステルス性を有していること，暗号化や難読化を行う悪意あるアドウェアが存在することから，悪意あるアドウェアの振舞い解析を行うことは容易ではない．

本論文では，悪意あるアドウェア解析における問題点を解決するために Blayzard を提案した．Blayzard は BHO ベースのアドウェアを動作させるイベントを生成・注入し，その振舞いを抽出する解析システムである．Blayzard はブラウザプロセスによる影響を排除するために BHO を別プロセスに配置することで，BHO のみの振舞いの抽出を容易にしている．また，ステルス性のあるアドウェアを活性化させるために，偽のイベントを生成し，解析対象のアドウェアに注入することで，めったに動作しないステルス性のあるアドウェアを強制的に動作させることで振舞いの抽出を可能にしている．動的解析を行うことで，暗号化や難読化が行われたアドウェアであっても，その影響を受けることはない．

Blayzard の有用性を確かめるために，31 個の悪意あるアドウェアと 10 個のよく利用される BHO を収集して，振舞い解析を行った．実験結果から，Blayzard は偽のイベントを注入して，その振舞いを抽出することができた．特に，偽のイベントの注入の仕方を工夫することで，暗号化してある Windows のプロダクト ID や Cookie の流出を推測することができた．同様に，よく利用される BHO も解析を行ったところ，一部の BHO はアドウェアに似た振舞いをすることが分かった．

今後の課題は，Blayzard を利用した悪意あるアドウェア検出手法を考案することである．また，BHO を利用せずに実現されたアドウェアであっても，アドウェアとその外部とのインタフェースが明確に規定されていれば，本論文で示した手法を拡張して利用可能であると考えられる．たとえば，単独のプロセスとして動作するアドウェアであれば，オペレーティングシステムとのやりとりを監視し，適宜，システムコールの返値や非同期イベントを注入することで，Blayzard と同様に動作解析が可能となると期待できる．

参 考 文 献

1) Symantec Security Response: Techniques of Adware and Spyware, available from ⟨http://www.symantec.com/avcenter/reference/techniques.of.adware.and.spyware.pdf⟩.
2) Kaspersky Lab: Monthly Malware Statics, October 2010, available from ⟨http://www.securelist.com/en/analysis/204792146/Monthly_Malware_Statistics_October_2010⟩.
3) Kaspersky Lab: Monthly Malware Statics, September 2010, available from ⟨http://www.securelist.com/en/analysis/204792141/Monthly_Malware_Statistics_September_2010⟩.
4) Kaspersky Lab: Monthly Malware Statics, August 2010, available from ⟨http://www.securelist.com/en/analysis/204792135/Monthly_Malware_Statistics_August_2010⟩.
5) Kaspersky Lab: Monthly Malware Statics, July 2010, available from ⟨http://www.securelist.com/en/analysis/204792130/Monthly_Malware_Statistics_July_2010⟩.
6) Panda Labs: Quarterly Report PandaLabs (April-June 2010), available from ⟨http://www.pandasecurity.com/img/enc/Quarterly_Report_PandaLabs_Q2_2010.pdf⟩.
7) Panda Labs: Quarterly Report PandaLabs (January-March 2010), available from ⟨http://www.pandasecurity.com/img/enc/Quarterly_Report_Pandalabs_Q1_2010.pdf⟩.
8) Moshchuk, A., Bragin, T., Gribble, S.D. and Levy, H.M.: A Crawler-based Study of Spyware on the Web, *Proc. 14th Network and Distributed System Security Symposium* (*NDSS '06*) (2006).
9) Microsoft: Browser Helper Objects: The Browser the Way You Want It, available from ⟨http://msdn.microsoft.com/en-us/library/bb250436.aspx⟩.
10) Wang, Y.-M., Roussev, R., Verbowski, C., Johnson, A., Wu, M.-W., Huang, Y. and Kuo, S.-Y.: Gatekeeper: Monitoring Auto-Start Extensibility Points (ASEPs) for Spyware Management, *Proc. 18th Large Installation System Administration Conference* (*LISA '04*), pp.33–46 (2004).
11) Offensive Computing: Offensive Computing; Community Malicious code researh and analysis, available from ⟨http://offensivecomputing.net⟩.
12) VX Heavens: Welcome to VX Heavens! (VX Heavens), available from ⟨http://vx.netlux.org⟩.

13) Hex-Rays: IDA Pro Disassembler – multi-processor, windows hosted disassembler and debugger, available from ⟨http://www.hex-rays.com/idapro/⟩.
14) Microsoft: The Component Object Model, available from ⟨http://msdn.microsoft.com/en-us/library/ms694363(VS.85).aspx⟩.
15) MSDN Library: DWebBrowserEvents, available from ⟨http://msdn.microsoft.com/en-us/library/aa768309(v=vs.85).aspx⟩.
16) MSDN Library: DWebBrowserEvents2, available from ⟨http://msdn.microsoft.com/en-us/library/aa768283(v=vs.85).aspx⟩.
17) Li, Z., Wang, X. and Choi, J.Y.: SpyShield: Preserving Privacy from Spy Add-Ons, *Proc. 10th International Symposium on Recent Advances in Intrusion Detection* (*RAID '07*), pp.296–316 (2007).
18) Wang, H., Jha, S. and Ganapathy, V.: NetSpy: Automatic Generation of Spyware Signatures for NIDS, *Proc. 22nd Annual Computer Security Applications Conference* (*ACSAC '06*), pp.99–108 (2006).
19) Lavasoft: Ad-Aware by Lavasoft – Antivirus software, free spyware removal, firewall, available from ⟨http://www.lavasoft.com/⟩.
20) Spybot Search & Destroy, available from ⟨http://www.safer-networking.org/en/home/index.html⟩.
21) Snort, available from ⟨http://www.snort.org/⟩.
22) Paxson, V.: Bro: A System for Detecting Network Intruders in Real-Time, *Proc. 7th USENIX Security Symposium* (1998).
23) VMware: VMware Virtualization Software for Desktops, Servers & Virtual Machines for Public and Private Cloud Solutions, available from ⟨http://www.vmware.com⟩.
24) Bellard, F.: QEMU, a Fast and Portable Dynamic Translator, *Proc. USENIX Annual Technical Conference*, pp.41–46 (2005).
25) Klein, T.: ScoopyNG – The VMware detectioin tool, available from ⟨http://www.trapkit.de/research/vmm/scoopyng/index.html⟩.
26) Paleari, R., Martignoni, L., Roglia, G.F. and Bruschi, D.: A fistful of red-pills: How to automatically generate procedures to detect CPU emulators, *Proc. 3rd USENIX Conference on Offensive Technologies* (*WOOT '09*) (2009).
27) Raffetseder, T., Kruegel, C. and Kirda, E.: Detecting System Emulators, *Proc. 10th Information Security Conference* (*ISC '07*) (2007).
28) Rutkowska, J.: Red Pill... or how to detect VMM using (almost) one CPU instruction, available from ⟨http://www.invisiblethings.org/papers/redpill.html⟩.
29) Vasudevan, A. and Yerraballi, R.: Cobra: Fine-grained Malware Analysis using Stealth Localized-executions, *Proc. 27th IEEE Symposium on Security and Privacy* (*S&P '06*), pp.264–279 (2006).
30) Dinaburg, A., Royal, P., Sharif, M. and Lee, W.: Ether: Malware Analysis via Hardware Virtualization Extensions, *Proc. 15th ACM Conference on Computer and Communications Security* (*CCS '08*), pp.51–62 (2008).
31) Borders, K., Zhao, X. and Prakash, A.: Siren: Catching Evasive Malware (Short Paper), *Proc. 27th IEEE Symposium on Security and Privacy* (*S&P '06*), pp.78–85 (2006).
32) Kirda, E., Kruegel, C., Banks, G., Vigna, G. and Kemmerer, R.A.: Behavior-based Spyware Detection, *Proc. 15th USENIX Security Symposium*, pp.273–288 (2006).
33) Egele, M., Kruegel, C., Kirda, E., Yin, H. and Song, D.: Dynamic Spyware Analysis, *Proc. USENIX Annual Technical Conference*, pp.233–246 (2007).
34) Yin, H., Song, D., Egele, M., Kruegel, C. and Kirda, E.: Panorama: Capturing System-wide Information Flow for Malware Detection and Analysis, *Proc. 14th ACM Conference on Computer and Communications Security* (*CCS '07*) (2007).
35) Costa, M., Crowcroft, J., Castro, M., Rowstron, A., Zhou, L., Zhang, L. and Barham, P.: Vigilante: End-to-End Containment of Internet Worms, *Proc. 20th ACM Symposium on Operating Systems Principles* (*SOSP '05*), pp.133–147 (2005).
36) Crandall, J.R. and Chong, F.T.: Minos: Control Data Attack Prevention Orthogonal to Memory Model, *Proc. 37th International Symposium on Microarchitecture* (*MICRO '04*) (2004).
37) Newsome, J. and Song, D.: Dynamic Taint Analysis for Automatic Detection, Analysis, and Signature Generation of Exploits on Commodity Software, *Proc. 13th Network and Distributed System Security Symposium* (*NDSS '05*) (2005).
38) Portokalidis, G., Slowinska, A. and Bos, H.: Argos: an Emulator for Fingerprinting Zero-Day Attacks, *Proc. 1st ACM European Conference on Computer Systems* (*EuroSys '06*), pp.15–27 (2006).
39) Suh, G.E., Lee, J.W., Zhang, D. and Devadas, S.: Secure Program Execution via Dynamic Information Flow Tracking, *Proc. 11th International Conference on Architectual Support for Programming Langages and Operating Systems* (*ASPLOS '04*), pp.85–96 (2004).
40) Cavallaro, L., Saxena, P. and Sekar, R.: On the Limits of Information Flow Techniques for Malware Analysis and Containment, *Proc. 5th International Conference on Detection of Intrusions and Malware, and Vulnerability Assesment* (*DIMVA '08*), pp.143–163 (2008).

(平成 23 年 4 月 20 日受付)
(平成 23 年 9 月 12 日採録)

推　薦　文

広告表示や個人情報を収集するアドウェアがコンピュータ/ネットワークセキュリティ分野で生じる脅威を把握するためには，アドウェアの動作分析が必要である．本論文では，アドオンを監視プロセスに配置する手法によって分析に有用な情報を抽出し，世界の関連研究と比べても十分に新規性と実用性を主張できる動作分析を実現している．実験評価と考察も高い水準であり，学術的および実務的貢献が見込まれるので，推薦したい．

（コンピュータセキュリティシンポジウム 2010 プログラム委員長　松浦幹太）

糟谷　正樹（学生会員）

1986 年生．2009 年慶應義塾大学理工学部情報工学科卒業．2011 年慶應義塾大学大学院理工学研究科解放環境科学専攻修士課程修了．現在，同専攻博士課程在学中．IEEE，ACM 各学生会員．インターネットセキュリティ，オペレーティングシステム，システムソフトウェアに興味を持つ．

河野　健二（正会員）

1993 年東京大学理学部情報科学科卒業．1997 年東京大学大学院理学系研究科情報科学専攻博士課程中退．同専攻助手に就任．現在，慶應義塾大学理工学部情報工学科准教授．博士（理学）．平成 11 年度情報処理学会論文賞受賞．平成 12 年度山下記念研究賞受賞．オペレーティングシステム，システムソフトウェア，インターネットセキュリティに興味を持つ．IEEE/CS，ACM，USENIX 各会員．