Vol.2009-CSEC-46 No.43
2009/7/3

Nimrod Project: フィルタドライバを用いたネットワークアプリケーションのOS内部挙動観測とデータセットの公開

安藤　類央[†1]　　門林雄基[†1]　　篠田陽一[†1]

本論文では、フィルタドライバを用いてネットワークアプリケーションがインターネットや P2P ネットワーク上でセキュリティインシデントに遭遇する際の OS 上の内部挙動を観測する手法と、データセットの公開を提案する。提案する手法では、Windows OS の NATIVE API フックとフィルタマネージャの仕組みを用いて、API 呼び出しをインターセプトし、ネットワークアプリケーションのメモリ、ファイル、ソケットバッファアクセスなどの OS の内部挙動を観測する。

Nimrod Project: An introspection method for network application using filter driver and generating dataset

Ruo Ando ,[†1] Youki Kadobayashi [†1]
and Yoichi Shinoda [†1]

In this paper we propose an introspection method for network application using filter driver and generating dataset. In proposed system of Microsoft Windows, we apply filter driver, filter manager and Dll injection to intercept Library, API and file IO. We discuss the way to use filter driver to obtain the sequence of application such as P2P and Web browser connected to Internet. Detailed observation changed according to the destination and state of netwok is possible by our system. This makes it possible to monitor network application of memory, file socket buffer acccess and its fine-grained logging.

1. はじめに

ここ数年のデバッグ技術の発達は目覚しいものがあり、仮想化技術の発展とあわせて、より詳細な観測と、高粒度なロギングが可能になってきている。特に、Microsoft Windows が提供するデバッギングＡＰＩ、ネットワークアプリケーション、そしてカーネルモジュールの開発のための環境も急速に整備されつつある。特に、サードパーティの開発を簡素化し、既存のカーネルモジュール、ファンクションドライバの機能を追加修正するためのフィルタドライバの開発環境が提供されたことで、自動するシステムの機能修正やチューニングが可能になった。フィルタドライバは、デバイスドライバより先にＩＲＰを参照することが可能であり、従来のＤＬＬインジェクションなどの技術とあわせて、アプリケーションの挙動を参照する際には有効なモジュールである。本論文では、ネットワークアプリケーションの挙動の観測するためのフィルタドライバの開発と、それによってデータセットを作成し、公開するための手法を提案する。

2. 提案システム

図１は、本論文で提案するフィルタドライバを用いたネットワークアプリケーションの OS 内部挙動観測システムの概要を示したものである。インターネットまたは、仮想化ネットワークに接続したネットワークアプリケーションを、クローラーまたは自動化のソフトウェアが制御し、これによって生じるネットワークアプリケーションの内部挙動をフィルタドライバとライブラリが補足し、ロギングをとるシステムである。提案システムにより、ネットワークの接続、流通状況によって変化するネットワークアプリケーションの内部挙動の詳細なログをとることが可能になり、精密な解析のためのデータセットを作成することが可能になると思われる。

3. 適用技術

3.1 フィルタドライバによるカーネルＡＰＩフック

Microsoft Windows のフィルタドライバは、ＸＰから積極的に導入、活用が始まったソフトウェアモジュールである。フィルタドライバは、ＩＯマネージャとカーネルドライバの間

†1 情報通信研究機構〒 184-8795 東京都小金井市貫井北町４－２－１

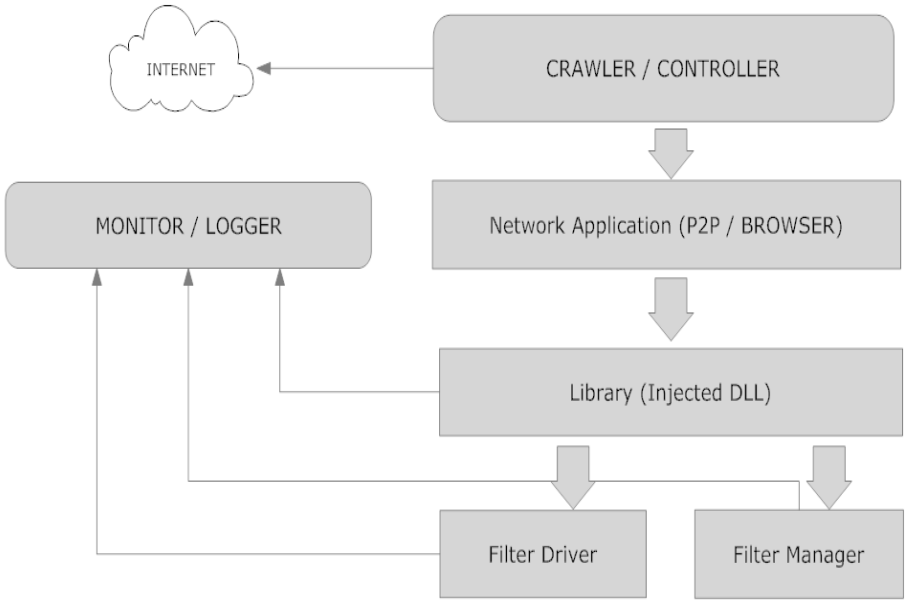

図 1 フィルタドライバを用いたネットワークアプリケーションの OS 内部挙動観測システム

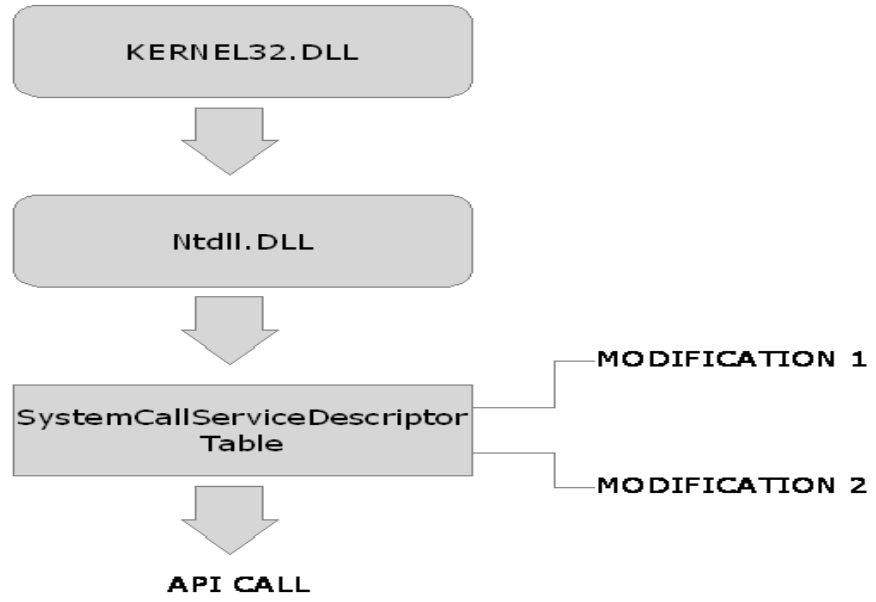

図 2 Service Descriptor Table (SDT) の修正によるカーネルモードＡＰＩフック。

に位置して、ファンクションドライバ（既存のデバイスドライバ）の前後に既存の Windows が提供する機能を利用して呼び出され、新しい機能を追加したり、機能修正、デバッグなどを行うソフトウェアである。フィルタドライバを用いることで、Native API フックを行うことができる。

Native API Hook は、カーネルモードでのイベント検出に用いられる。ユーザモードでの API 発行は、ntdll.dll 経由でカーネル空間に伝達される。Windows では、カーネルモードで動作する API を、native API といい、これらは、SystemServiceDescriptorTable という構造体によって、制御されている。そのため、Native API Hook を行うためには、SystemServiceDescriptor に修正を加える必要がある。

Modification 1 は、InterlockExchange を用いて、システムコールテーブルを書き換える方法である。Modification 2 は、ドライバのロード時にシステムコールテーブルを書き換え、フック関数を通過する方法である。

3.2 フィルタマネージャによるＡＰＩフック

開発環境や稼動する状況にも依存するが、通常のフィルタドライバを用いたファイルＩＯのフックは、安定動作しない場合がある。フィルタドライバは、Microsoft が新たに提供をはじめたファイルシステムフィルタドライバである。フィルタドライバは、大量に呼び出される割り込みやＡＰＩに関して、順序などを適切に制御し、サードパーティの開発を簡素化、支援することを主眼に設計されている。図３は、フィルタマネージャの機能概要を示したものである。フィルタマネージャは、カーネルドライバ（ファンクションドライバ）とフィルタドライバの間を仲介し、フィルタドライバの実装を簡素化し、機能を安定にする。

4. ネットワークアプリケーションの制御

4.1 DLL Injection

DLL Injection とは、任意の関数をライブラリ化して特定の API が発行されたときに、実行させるもので、既存のアプリケーションに新しい機能を加えたいとき、ソフトウェアの

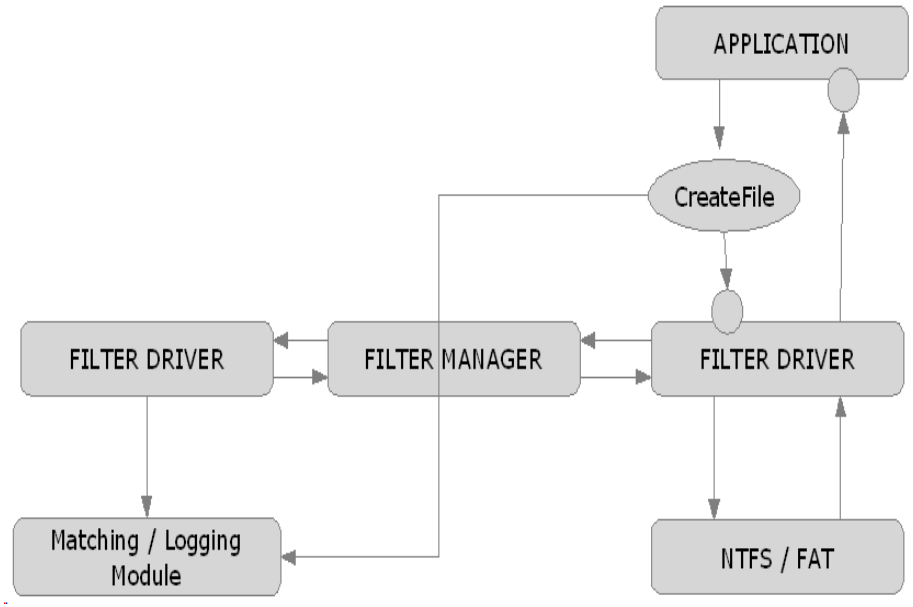

図 3 フィルタマネージャを用いたファイルフィルタドライバの作成と動作概要。

デバッグの際に用いられる。開発の要請から、DLL（独自 API) を他のプロセスに任意のタイミングで実行させる DLL Injection という技術が用いられることがある。DLL Injection の方法には、Mi-crosoft Windows が提供している機能を使うもの、デバッグ機構の機能を使うもの、リモートスレッドを使うもの、そしてモジュールのインポートセクションの変更によるものなどがある。本論文では、モジュールのインポートセクションの変更による方法により、対象ソフトウェアのデータ送受信関数をフックし、ここに適切な操作を入れることにより、ソフトウェアの挙動を追跡し、問題となるトラフィックが発生または到着する前に防止を行うことを可能にした。図１は、インポートテーブルの変更よる DLL 注入を示したものである。インポートテーブル（インポートセクション）とは、セクションテーブルの中にあるデータ構造体（ヘッダ）で、プログラムが実行される前に必要な DLL のアドレスと、利用する DLL 内のシンボルのアドレスのリストが格納されている。ここで、フックしたい DLL のアドレスを、任意の DLL へのアドレスに書き換えることで、ソフトウェアの動作の修正を行うことができる。この方法は、特定の CPU の仕様に依存しなく、またスレッドの同期の問題もないため、非常に柔軟な方法として用いられることが多い。処理としては、１）注入したい DLL を用意する。２）対象となるソフトウェアのインポートテーブルアドレスを取得する。３）フックしたい関数のアドレスを探す。４）発見したアドレスを、注入したい DLL（関数）へのアドレスに書き換える。関数形は

```
void ReplaceIATTableInP2Psoftware
("kernel32.dll",
funcORG,
funcINSERT",
moduleHandler);
```

対象とするプロセスの構造などにより、実装方法はいくつか存在するが、大枠は以上で示した通りである。

4.2 Windows 上の自動化

Windows 上のアプリケーションの自動化には、前節で述べたライブラリの修正や、その他の Code Injection 系の手法の他に、タスクマネージャの利用や、WSH、VBScript などのスクリプト言語の利用がある。

5. ネットワークアプリケーションの観測制御

5.1 P2P アプリケーションの挙動の観測と制御

本節では、Ｐ２Ｐアプリケーションの挙動の観測と制御について述べる。近年、Ｐ２Ｐアプリケーションで問題になっているのは情報漏洩などによるデータセキュリティに関するセキュリティインシデントであり、これらの観測を、ソケット関係の DLL Injection によって行うことができる。また、銃砲漏洩によって発生することが考えられるＰ２Ｐアプリーションによる異常なファイルアクセスに関しては、フィルタマネージャなどによって検出が可能である。

5.2 Web ブラウザの挙動の観測と制御

最近 Web ブラウザの機能を悪用したセキュリティインシデントが多発している。例えば、ＰＤＦのオープンをトリガーにして、Internet Explorer にコードがインジェクトされるケースなどである。マルウェアの多くがＷＥＢを経由して混入してくる場合が多く、このようなセキュリティインシデントを解析する際には、Web ブラウザの挙動の観測と制御が重要になる。提案システムでは、DLL Injection を用いて、ユーザモードのＡＰＩの呼び出しを観測制御し、同時にフィルタドライバを用いてレジストリアクセスやファイルアクセスをモニ

Vol.2009-CSEC-46 No.43
2009/7/3

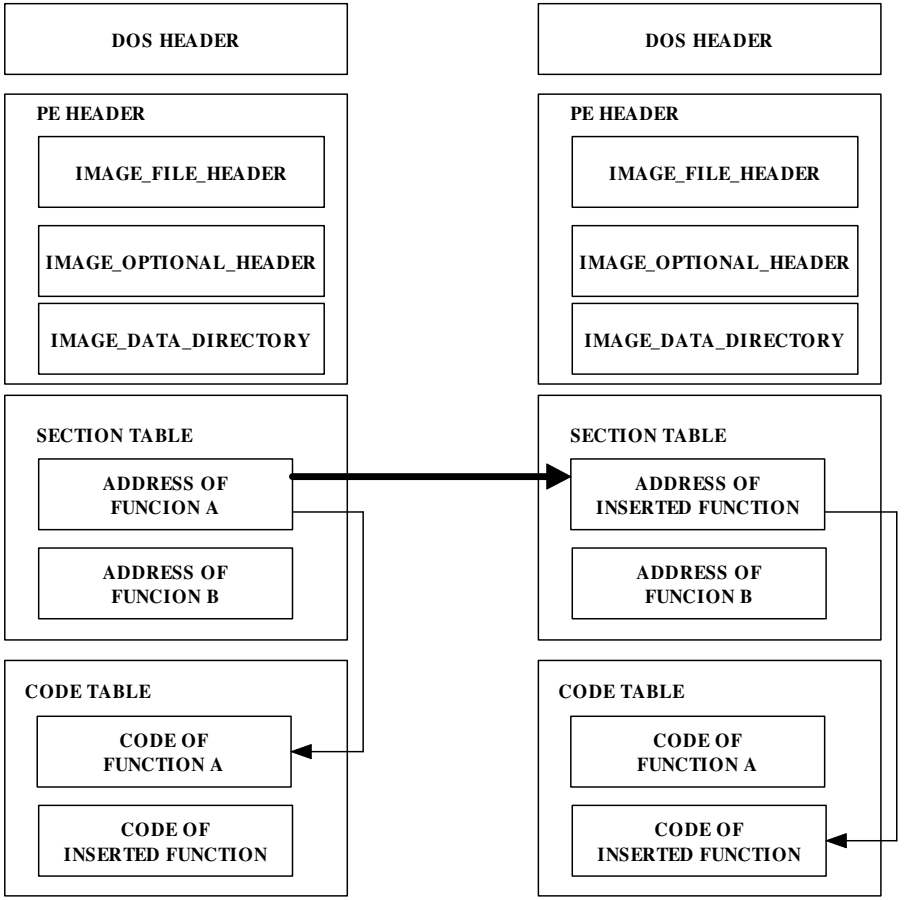

図 4 DLL Injection。インポートセクションテーブルの変更により、特定のＡＰＩの機能を追加修正する。

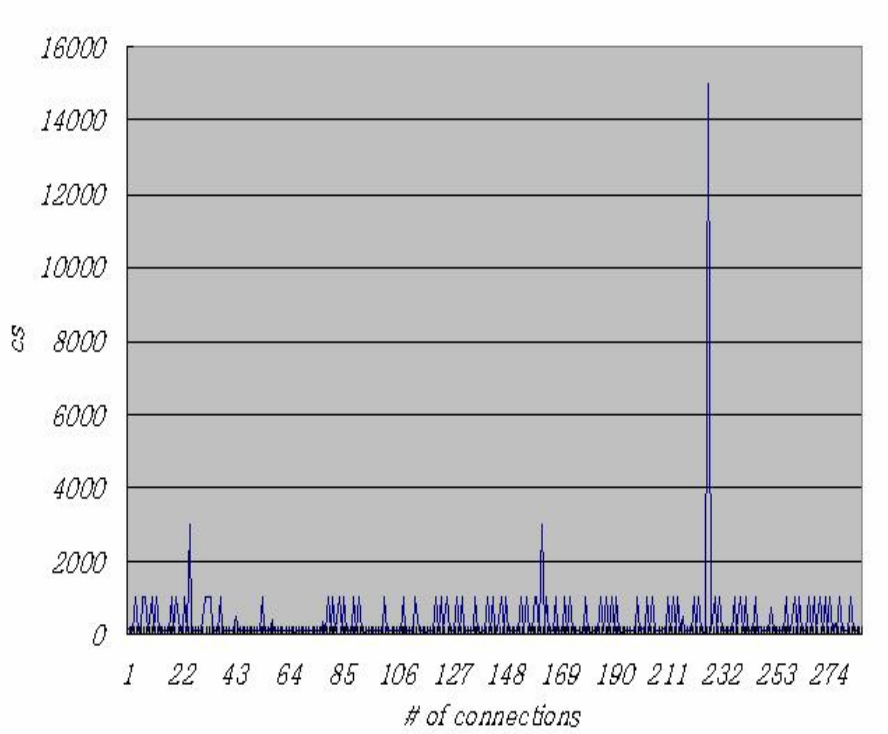

図 5 上図では、提案システムによるＰ２Ｐアプリケーションで使われたソケットバッファの解析により、ネットワークアプリケーションに接続されたノードの通信速度をプロットした。

タすることで、Web ブラウザの異常な挙動を検出し、データセットの作成を行う。図５は、Ｐ２Ｐアプリケーションのソケットバッファを解析し、接続されてくるノードの通信速度を図示したものである。これにより、通信速度の速いスーパーノードがどの時間帯に接続し、どのような挙動を示すのかについてロギングを行うことが可能であると想定される。

6. まとめと今後の課題

本論文では、フィルタドライバを用いてネットワークアプリケーションがインターネットや P2P ネットワーク上でセキュリティインシデントに遭遇する際の OS 上の内部挙動を観測する手法と、データセットの公開を提案した。提案する手法では、Windows OS の NATIVE API フックとフィルタマネージャの仕組みを用いて、API 呼び出しをインターセプトし、ネットワークアプリケーションのメモリ、ファイル、ソケットバッファアクセスなどの OS の内部挙動を観測する手法を提案した。

参 考 文 献

1) Kernel Based Virtual Machine, http://kvm.qumranet.com/kvmwiki
2) XEN virtual machine monitor, http://www.cl.cam.ac.uk/Research/
3) Tal Garfinkel and Mendel Rosenblum, ”A Virtual Machine Introspection Based Architecture for Intrusion Detection ”, In the Internet Society ’s 2003 Symposium on Network and Distributed System Security (NDSS), pages 191.206, February 2003.
4) BD Payne, M Carbone, M Sharif, and W Lee. Lares, ”An Architecture for Secure Active Monitoring Using Virtualization ”, In Proceedings of the IEEE Symposium on Security and Privacy (Oakland 2008), May 2008.
5) BAN Tao, ANDO Ruo, KADOBAYASHI Youki, ”Monitoring and Analysis of Network Traffic in P2P Environment”, Journal of the National Institute of Information and Communications Technology, Vol.55 Numbers 2/3 2008, ISSN 1349-3205, 2008.11.
6) 安藤類央、門林雄基、篠田陽一、「Code injection 検出のための VMM スナップショット機能の強化」、情報処理学会コンピュータセキュリティ研究会第４２回研究報告２００８年7月２５日
7) Ruo Ando, Youki Kadobayashi, Youichi Shinoda, ”Incident-driven memory snapshot for full-virtualized OS using interruptive debugging techniques”, International Journal of Security and Its Applications, vol.2, No,3,pp41-48, 2008 July, ISSN 1738-9976
8) Deepak Ravichandran, Patrick Pantel, Eduard H. Hovy: Randomized Algorithms and NLP: Using Locality Sensitive Hash Functions for High Speed Noun Clustering. ACL 2005
9) T.Kohonen, Self-Organizing Maps, Springer, 1996
10) V.Vapnik, ”The Nature of Statistical Learning Theory”, Springer, 1995