社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－DPS－130 (9)
2007－CSEC－ 36 (9)
2007／3／1

DLL injectionを用いたＰ２Ｐソフトウエアの情報漏洩の追跡と防止

安藤類央　外山英夫　門林雄基

独立行政法人　情報通信研究機構 情報通信セキュリティ研究センター
〒 184-8795　東京都小金井市貫井北町 4-2-1
（株）コムラッド
〒 168-0074 東京都杉並区上高井戸１－８－１７第３保谷ビル新館
奈良先端科学技術大学院大学
〒 630-0192 奈良県生駒市高山町 8916 番地の 5

あらまし P2P ソフトウェアは、フリーで公開／利用されるケースが多い反面、ソースコードが公開されず、新たな攻撃方法に対して、バージョンアップやサポートがされない場合も多い。また、開発側の対応が諸般の事情で止まってしまう場合もある。本論文では、このような現状の上で、ユーザが P2P ソフトウェアを安全に利用するためのソフトウェア修正技術を提案する。提案システムでは、対象 P2P ソフトウェアのインポートテーブルを変更し、フィルタ／プロテクト用の DLL（独自の関数）をプロセスに注入することで、ソフトウェアの挙動を追跡し、情報漏洩の防止を行う。これにより、開発やバージョンアップの行われない P2P ソフトウェアにおいても、新たな不正アクセスと情報漏洩に対して対策を行うことが可能になる。

Tracing behavior of P2P software using DLL injection

Ruo Ando, Hideo Toyama, Youki Kadobayashi

National Institute of Information and Communication Technology, Tracable Network Group
4-2-1 Nukui-Kitamachi, Koganei,
Tokyo 184-8795 Japan
ruo@nict.go.jp

Abstract Recently free P2P software is widely used. However, sometimes support and version up is not provided for new vulnerabilities and attacks. In this paper we present a modification of P2P software without dealing with source code. We apply debugging technology, called DLL injection for tracing behavior of P2P application. Proposed software modifiles import table to inject hook routine into target process. By doing this, snapshot of process memory is analyzed and malicious behavior of P2P system is filtered and prevented. Our system makes it possible to modify P2P software without source code or updating which means we can protect ourselves by proposed method.

1　はじめに

１台のコンピュータがサーバとクライアントを兼ねるＰ２Ｐでは、ユーザ同士の情報交換が簡単に行え、ネットワークトラフィックの急増などの変化にも耐性があるため、急速に普及が進んでいる。その反面、Ｐ２Ｐネットワークを介した著作権侵害や、Ｐ２Ｐ経由でのウィルス

感染による情報漏洩が問題になっている。Ｐ２Ｐソフトウェアはフリーで公開される場合が多い反面、商用ソフトに比べ開発コミュニティの発足と発展の段階でバージョンアップや修正が行われなくなることが多い。また、諸般の事情で発展が止まる場合もある。そのため、新たな脆弱性やP2Pアプリケーションによって形成されるオーバーレイネットワークへの攻撃が判明しても、対応する修正やパッチなどが用意されないことがある。

2　Ｐ２Ｐソフトウェアの構造と特徴

Ｐ２Ｐソフトウェアは、一般に３つの形態に分けられる。ノード同士が直接データやファイルを交換するのがＰ２Ｐの特徴であるが、ネットワーク上のどのノードがどういうファイルを持っているかはサーバが管理するものを第一世代、純粋Ｐ２Ｐといわれているのが、サーバを必要とせず、完全にノード同士がデータ交換をするのが第二世代である。さらに、匿名性を増すためにキャッシュ機能を備えたものは第３世代のＰ２Ｐソフトウェアと言われる。

2.1　Ｐ２Ｐソフトウェアの構造

Ｐ２Ｐソフトウェアは、ノード管理、クエリ管理、キー管理、タスク管理の４つのモジュールで構成されることが代表的である。図１は、P2P ソフトウェアの一般的な構造を示したものである。このうちP2P ソフトウェアの挙動を追跡するのに重要な情報はキーテーブル、ノードテーブル、送受信ファイルテーブルであるが、本論文では、仮想メモリ上に展開されているこれらのデータを、パケットを送信時のAPI 呼び出しをフックし、キャプチャし、解析、復号とフィルタを行うシステムを構築した。フックを行う方法にはついては３節で詳解する。

2.2　Ｐ２Ｐソフトウェアの特徴

P2P アプリケーションは、複数のユーザの利用を前提としている以上、プログラムを気軽に作りかえることはできない。また、バージョンアップや修正が保障されているわけではなく、諸般の事情で開発がとまってしまう事がある。このような特徴を持つソフトウェアを安全に使うためには、本論文で示すデバッグの技術を用いた修正を施すことが有効である。また、第二の重要な特徴に、P2P アプリケーションでは、構成されるネットワークのトポロジーを把握できないことが上げられる。これは、特定のプローブシステムを P2P ネットワーク上に設置すると、それが攻撃の対象になる可能性があり、ブロードキャスティングも同様の利用で行われないことが多いためである。そのため、サーバベースでのモニタを行うことは難しく、トポロジーやトラフィックフローを把握するためには各プローブでのプロセスメモリの情報を得る必要がある。

3　提案手法

現在、多数のＰ２Ｐソフトウェアが公開されているが、特にMicrosoft Windows 上で動くアプリケーションの場合、ソースコードまで公開されるケースはあまりない。また、諸般の事情で開発が止まってしまうことがある。そのため、Ｐ２Ｐソフトウェアへの攻撃や情報漏洩などに対応するためには、P2P の不正な動作を外部から監視修正するプログラムを作成すると有効である。通常の OS では、メモリ空間はプロセスごとに割り当てられ、それらは保護される。あるプロセスが他のプロセスのメモリを制約なしに操作することはできないが、他のプログラムのインスタンスをサブクラス化したい場合、ソフトウェアの動作やパラメータの遷移を検証したい場合、共有メモリを利用する場合、他のプロセスのメモリ空間を操作することができる。このような機能を使うソフトウェアには、プロセスメモリエディタやデバッガがあり、本論文で提案するシステムはこれらの一種であると言える。

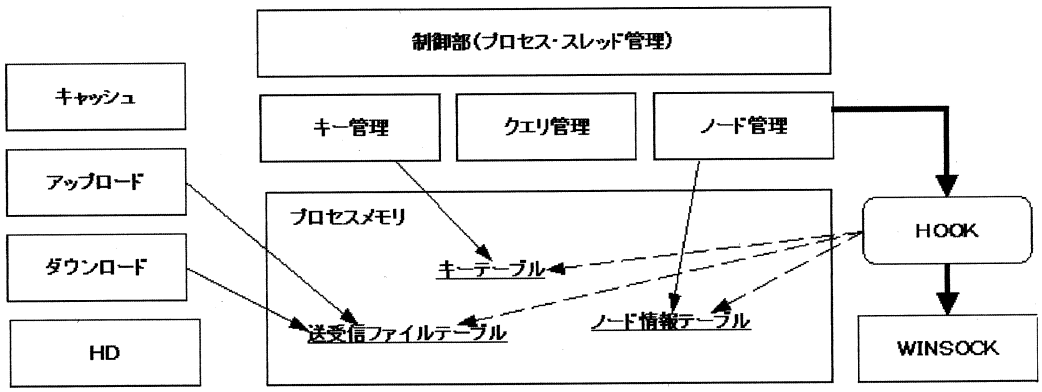

図 1: P2P ソフトウェアの構造。P2P アプリケーションは、ファイルテーブル、クエリ、ノード、キー管理モジュール、それらを同期する制御部から構成される。これらの情報がプロセスメモリ上で遷移する。

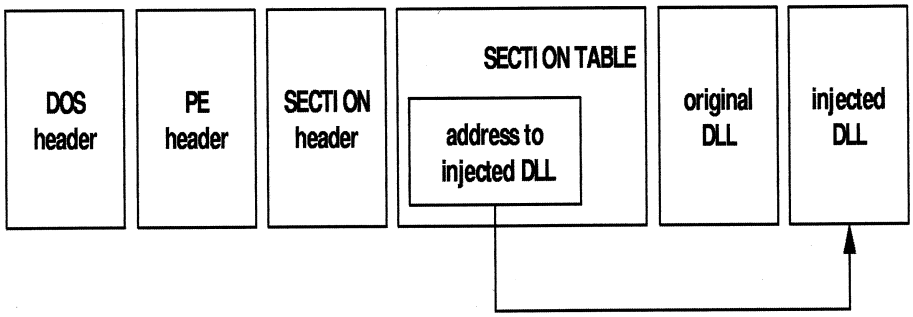

図 2: DLL Injection。実行ファイルのセクションテーブルのインポートモジュールセクションに格納されているアドレスを注入した DLL のアドレスに変更することで、API のフックを行う。

3.1 DLL injection

以上のような開発の要請から、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、Microsoft 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、P2P 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 2 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。

3.2 適用APIとアドレスの検索

提案システムでは、GetModuleHandleなどを使って、他プロセスのハンドラを取得する。他プロセスのアドレス空間内に（共有）メモリをアロケートする方法は、VirtualAllocEXなどがある。実際にリモートプロセスメモリには、WriteProcess Memoryを使ってデータを書き込む。アドレスの検索は、GetModuleHandleを使って、プロセスのハンドラ（アドレス）を取得し、Microsoft Windowsの実行ファイル形式を参照しながら、インポートセクション内のフックしたい関数のアドレスを探す。以下にこの概略を示す。

```
pointer=GetModuleHandle
currentPfn=GetProcAddress
/*pointer を移動して、インポートセクションを探す*/
/*インポートセクション内*/
if(Npointer==CurrentPfn)
  /* 発見 */
```

ただし、対象となるプロセスが特定のLoadLibraryやcreateProcessの使い方をしている場合、上のコードに追加操作を加える必要がある。関数をフックする場合、プロセスメモリのスナップショットを取得し、フィルタの動作に移行することができる。[1]

4 実装例

本論文では、前節で述べた方法を用いて、適用対象システムにWinnyを用いて、P2Pソフトウェア追跡システムの実装を行った。図3に、実装システムの実行画面を示した。実装システムでは、ディレクトリ内に、フック用のDLL、P2Pソフトウェア、環境設定ファイルなどを用意し、P2Pソフトウェアを起動したあと、そのプロセスにフックモジュールをマップする。[2]今回は、パケットの送受信関係の関数をフックし、（ネットワーク経由での）コンテンツ入出力時にP2Pソフトウェアのプロセスメモリのスナップショットを取り、パケットヘッダや、コンテンツの情報を解析し、フィルタリングや停止をおこなうシステムを作成した。また、ウィンドウへの出力形式はTcpDumpの形式に準拠した。同システムを用いると、Microsoft Windowsやアンチウィルスソフトが用意している機能よりも粒度の細かいフィルタリングを行うことが可能である。また、どのようなコンテンツが処理されているのか、また、重要なデータが格納されている特定のディレクトリを指定し、入出力されているトラフィックを停止し、情報漏洩の対策に用いることが可能である。

5 まとめと今後の課題

Microsoft Windows上で実行されるP2Pソフトウェアの不正な利用による情報漏洩やセキュリティ事案が問題になっている。P2Pソフトウェアは、フリーで公開／利用されるケースが多い反面、ソースコードが公開されず、新たな攻撃方法に対して、バージョンアップやサポートがされない場合も多い。また、開発側の対応が諸般の事情で止まってしまう場合もある。本論文

[1] スナップショットを取るものには、CreateToolhelp32snapshotなどがある。

[2] P2Pソフトウェアの中には、作成者が暗号アルゴリズムやコマンドフォーマットを公開しているものがある。

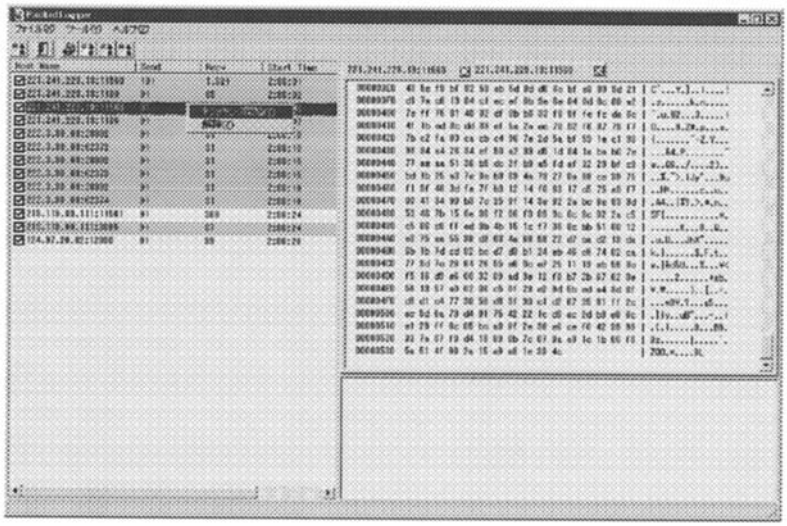

図 3: 実装したソフトウェアの表示画面。P2P ソフトウェアのネットワーク上の挙動を把握するため、プロセスメモリ内のパケットのデータを TcpDump 形式で表示させた。

では、このような現状の上で、ユーザが P2P ソフトウェアを安全に利用するためのソフトウェア修正技術を議論した。提案システムでは、対象 P2P ソフトウェアのインポートテーブルを変更し、フィルタ／プロテクト用の DLL（独自の関数）をプロセスに注入することで、ソフトウェアの挙動を追跡し、情報漏洩の防止を行う。これにより、開発やバージョンアップの行われない P2P ソフトウェアにおいても、新たな不正コードや情報漏洩法に対して対策を行うことが可能になった。

謝辞

提案システムの考案と構築に協力して頂いた（株）コムラッドの高根英哉氏と開発チームの方々に謝意を記す。

参考文献

[1] "A Distributed Decentralised Information Storage and Retrieval System", Ian Clarke, Division of Informatics University of Edinburgh Dissertation, 1999
http://freenet.sourceforge.net/freenet.pdf

[2] Programming Applications for Microsoft Windows Forth Edition, Jeffrey Ritcher, Microsoft Press, 1999

[3] Symantec white paper, "Attacks on Win32 - PARTII"
Virus Bulletin Conference, 2000

[4] Gary Nebbett, Windows NT/2000 Native API Reference, Sams Publishing; ISBN:1578701996,2000

[5] Richard J. Simon, Michael Gouker, Brian Barnes, Windows 95 API Bible: Win32 Programmer's Reference, Pearson Education Ltd. 1995.

[6] 金子勇, Winny の技術, アスキー書籍編集部（編）, 2005