F-28

P2P ネットワークの共有ファイル検索についての一提案

A proposal of Searching Files Shared in P2P Network

薄田 昌広†　　上原 哲太郎‡
Masahiro Susukita　　Tetsutaro Uehara

1. はじめに

クライアント・サーバモデルと異なり、個々の PC が送信者にも受信者にもなる P2P ネットワークによるファイル共有は、適切に利用することで通信負荷を分散させて大量のコンテンツ配信ができるなどの利点があるため注目されている[1]。しかし一方で組織内機密文書の漏洩のセキュリティ事故のきっかけともなっている。特に Winny や Share といったソフトウェアで構築される P2P ネットワーク上には、暴露ウィルスと呼ばれるマルウェアが多数流通しており、これらに感染することで個人情報や企業機密を含む PC 内部の電子文書が漏洩して広域に拡散する事故が社会問題となっている。

このような事故への対応としては、企業等での電子文書管理を厳格にするという防止対策に加え、P2P ネットワークへの流出を想定とした事後対策も重要となる。P2P ネットワーク上への電子文書流出自体を防ぐことが出来なくとも、早期に発見できれば、当該文書が広くネットワーク上で拡散する前に削除できる場合があり、被害想定などの対応も早くできる。しかし、P2P ネットワークでは各ファイルの所在の管理も分散して行われているため、全体の文書検索が困難である。

本研究では、P2P ネットワーク内のファイル検索を行う検索方式を提案し、P2P ネットワークによるファイル共有ソフトウェアである Winny を対象として検索クローラの試作評価を行った。

2. P2P ネットワークでのファイル検索の課題とクローラの提案

P2P ソフトウェアが構築する P2P ネットワークでは、ファイル共有を行う場合にファイルを集中管理するサーバを置かず、ファイルはネットワーク内のノードが分散して管理する。同様に、ネットワークに参加しているノードの構成情報も集中管理されていない。

組織内から流出した文書が P2P ネットワーク内に存在するかどうかを確実に確認するためには、ネットワーク内にあるすべてのファイルを把握することが望ましいが前述のように、P2P ネットワークの構造上、すべてのファイルを把握することはすべてのノードを知るということに等しく、P2P ソフトウェアの機能では実現できない。

そこで、本研究では、P2P ソフトウェアの機能によらず、別のプログラムから P2P ネットワークにアクセスしてすべてのファイルを把握する仕組みを提案する。機能としては、インターネット上で Web の検索サービスが稼動させている Web クローラに相当するため、本稿ではこれを P2P ネットワーククローラと呼ぶ。Web クローラがサイト間のリンクをたどって範囲を拡大しながらコンテンツをキャッシュしてゆくのと同様に、P2P ネットワークノードの保有するノード情報を元に範囲を拡大し、必要に応じてファイル内容をキャッシュする。

3. Winny を対象としたクローラの設計

本稿では、実際のクローラ試作の対象ソフトウェアを Winny とした。Winny は匿名性が高いファイル共有ソフトウェアであり、国内での利用者が多く、ウィルスによるものを含めて複数の流出事故に関連している。また、Winny はクローズド開発であるが、ソフトウェアの動作やプロトコルのかなりの部分が開発者により公開されている[2]。

3.1 Winny プロトコル

Winny プロトコルの特徴を以下に示す。

(1) Winny は一般的なピュア P2P の特徴を持ち、管理サーバを持たない
(2) ノード構成はフラットではなく、接続している回線の速度や外部からの直接アクセスの可否などの条件を元に階層化されている。
(3) ファイル情報とノード情報をあわせて一つのレコードとした「キー」を定期的に交換しノードで分散管理する。
(4) クエリ要求には特定のファイル検索のクエリ要求とキー情報を交換するための拡散クエリ要求がある
(5) ダウンロードの際は、キーに記述されている、ファイルに対応したノードの IP アドレスへダウンロード要求を行う
(6) NAT 内にあるノードなど外部からアクセスできないノードは別ノードを経由してファイル交換を行うことができる。(Port0 ノードと呼ばれる)
(7) 要求の多いファイルをキャッシュして拡散すると同時に、ファイル保有ノードの特定を避けて匿名性を高めるため、ファイル要求に対してある確率で所有ノードを偽った回答をする

3.2 クローラの設計

Winny ネットワークの特徴を考慮し、クローラの設計を検討する。必要な機能は、ネットワーク内のファイル情報の収集と、ファイル本体のダウンロードおよび検査である。Winny ネットワークの特徴を考慮し、前者はキーの収集によって実現し、後者については、誤ったノードからのダウンロードによるファイル拡散の防止についても検討した。

3.2.1 キークローラ

Winny クローラの目的は全てのノードの持つファイル情報を収集することである。Winny では、ノードの情報はやりとりされるキーの中に含まれている。そこで、キーを

† 関西電力株式会社, The Kansai Electric Power Co. Inc,
‡ 京都大学, Kyoto University

収集してノード情報を取り出し、未アクセスの新ノードがあれば接続してキーを要求するという一連の作業を繰り返すことで、キーを収集しながらネットワーク全体を巡回する。

3.2.2 ファイルダウンローダ

保有ノードに対してファイルを指定してダウンロード要求を送信することでダウンロードを行う。ただし、Winny の機能として確率的に保有ノードを偽るという動作がある。保有していないノードにダウンロード要求を行うと、そのノードがファイルを新たにキャッシュして不必要にファイルが拡散する。そのため、1 個のファイルについて複数のキーを照合し、保有ノードの可能性が高いところからダウンロードを行う。

3.2.3 キーデータベース

ファイル情報はノードと同じくキーに含まれる。ネットワーク内のすべてのキーを収集し、ファイルのハッシュ値、ファイル名などのファイル情報を取り出せば、巡回完了時にファイルの一覧を得ることができる。ただし、ファイルがアーカイブファイルである場合は内部に複数のファイルが含まれるため、個々のファイルについてファイルデータベースにデータを追加する。

3.2.4 ファイルチェッカ

ファイルキャッシュに置かれたファイルについて、ファイルが流出したものであるかを検査する。検査の実施状況についてはファイルデータベースへ反映する。

クローラ全体の構成は図 1 の通り

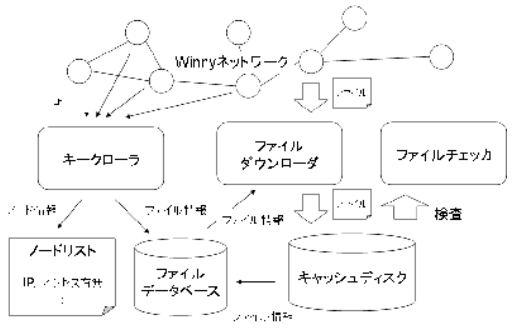

図 1: Winny クローラの構成

4. クローラの実装

設計に基づいて Winny ネットワークのクローラを実装した。すべてのプログラムは Java で作成し、拡張性を考慮してマルチスレッドで動作するようにした。キーデータベースは Barclay DB で作成した。

4.1 キークローラ

キークローラは、Winny ネットワーク内のノードを巡回し、キーを収集しながら、キーに含まれるファイルを所有している Winny ノードを推測する。ファイルを所有している Winny ノードを推測できたキーはファイルリストデータベースを参照し、未判定キーであった場合はファイルリストデータベースに記録するとともに、ダウンローダに引き渡してダウンロードと検査を行う。

具体的な動作手順を図 2 に示す。

候補ノードのアドレス集合を C、既に接続を試みたノードの集合を R とする。
1. C を初期ノード群とする
2. C が空になるまで 2.1～2.2 を繰り返す
 2.1. C からひとつのノードを取り出し、R に加える
 2.2. 2.2.1～2.2.2 を指定回数繰り返してキーを収集する
  2.2.1. 2.1 で取り出したノードに対し拡散クエリ送信要求を送信
  2.2.2. 拡散クエリの応答としてキーを受信
 2.3. 収集したキーのアドレスが R に含まれていなければ C に加える

図 2:キークローラの動作手順

Winny ではキーが Winny ネットワーク内を拡散していく際に、キーに含まれるアドレス情報がある確率でキーを別ノードに拡散しようとしているノードのアドレスに書き換えられる。この書き換えられたアドレスを用いてダウンロードを行った場合、そのアドレスのノード上にキャッシュが作成されることとなり、ファイルが拡散されてしまう。このようなことは極力避ける必要があるため、キークローラでは以下の方法でファイルの所有ノードの推測を行う。

1. あるノード A から、ノード A のアドレスを記述したキーが得られた場合、その理由は以下の 3 つの場合のいずれかである。
 (A) ノード A がファイルまたはキャッシュを所有している。
 (B) ノード A に接続している Port0 ノードがファイルまたはキャッシュを所有している。
 (C) ノード A は、ファイルもキャッシュも所有しておらず、ノード A がキーのアドレスを書き換えた。
2. (C)のアドレス書き換えが発生する確率は、ノード A がキャッシュが全く持っていない場合において 0.04 である。
3. 従って、M 回連続してノード A のアドレスを記述したキーが得られた場合、その確率は 0.04 の M 乗となる。

そこで、キークローラは M 回連続してノード A のアドレスを記述したキーが得られた場合に、(A)または(B)であると推測して、ダウンローダにキーを渡し、ノード A からダウンロードさせる。
デフォルトの M=3 の場合、この推測が誤る確率は 0.000064 である。

図 3:ノード推測の手順

4.2 ダウンローダ

ダウンローダはキークローラより漏出疑いのファイルに対応するキーを受け取り、そのキーに記されたファイル、Winny ノードアドレスに基づきダウンロードを行う。ダウンロードされたファイルは独自のキャッシュ形式で保存された後、ファイルチェッカにより取り出されて検査される。

ダウンロード先の Winny ノードに接続する際、最初は自身を非 Port0 ノードとして宣言して接続する。ダウンロードしようとしているファイルの本来の所有ノードが Port0 ノードであった場合、接続しようとしているノードは中継ノードである。自身を Port0 ノードと宣言していた場合、この中継ノードを介してダウンロードされるため

ファイルを拡散させてしまうことになる。これを防ぐため、一旦自身を非 Port0 ノードとして宣言して接続する。一方、ファイルの本来の所有ノードが接続しようとしているノードである場合、そのノードから BadPort0 と判断され、切断される場合がある。この場合は自身を Port0 ノードとして再接続することでダウンロードを行う。

ダウンロードしようとしているファイルの本来の所有ノードが Port0 ノードであった場合、接続先のノードからはコマンド 23 により Port0 ノードの情報が通知される。この場合、一旦接続を切断した後、指定された範囲内のポート用いてファイルの本来の所有ノードである Port0 ノードからの接続を待ち、その後 Port0 ノードよりダウンロードを行う。

4.3 キーデータベース

キーデータベースの構成を表 1 に示す

表 1:キーデータベースの構成

エントリ名	説明
Key	Winny キー （ハッシュをキーとして参照する）
Date	データベースにエントリが記録、あるいは更新された時刻
State	ファイルの検査状況を示す。 (0)ダウンロード未了または未検査 (1)検査済みで流出ファイルではない (2)流出ファイル、もしくは流出ファイルを含む
fileList	ダウンロードされたファイルがアーカイブファイルであった場合、アーカイブに含まれる漏出ファイルのリスト
localFileName	ダウンロードされたファイルのローカルホスト上でのファイル名

4.4 ファイルチェッカ

ファイルチェッカはダウンロードしたファイルを検査する。今回の実装ではファイルに特定の文字列が含まれているかどうかという判定を行った。

5. 動作検証

模擬環境において小規模な Winny ネットワークを再現し、Winny ノードのアップロードディレクトリに置かれた流出ファイルをクローラにより検出できるか確認を行った。

5.1. 検証環境

5.1.1 Winny ノード実行環境

Winny ノードは 1 台のホストマシン上で複数のゲスト仮想マシンを動作させ、そこで実行した。表 2 に仮想マシンを動作させるホストマシンの構成、表 3 に個々の仮想マシンの構成を示す。個々の仮想マシンでは 1 ノードずつ Winny を起動し、全体で 8 ノードからなる Winny ネットワークを構成した。Winny ネットワークの構成については 5.1.2 にて記す。

表 2 ホストマシン構成

CPU	AMD Athlon 64X2 3800+
メモリ	6GB
VM ホスト	VMWare Server 2.0
仮想マシン数	8

表 1仮想マシン構成

CPU 数	1
メモリ割当	192MB
OS	Windows2000

5.1.2 Winny ネットワークの構成

Winny に与える初期ノード情報を調整し、図 3 に示す初期構造を持つ Winny ネットワークを構成した。図中の楕円が各 Winny ノードであり、楕円中の数値は表 4 のノード番号に相当する。ただし、図 の構造は恒常的なものではなく Winny 自身がノード間でノード情報を交換することでネットワーク構造は逐次変化する。

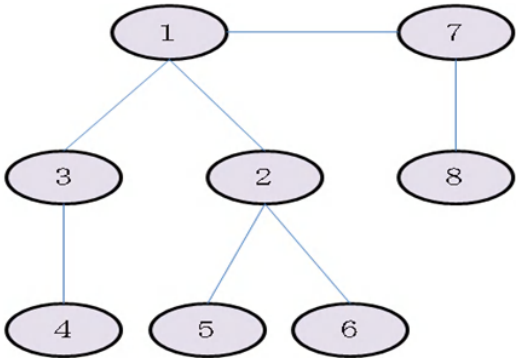

図 3:ノード構成

各 Winny ノードの回線速度設定とアップロードディレクトリに置いたファイルは表 4 の通りとした。ノード 4 は PORT0 設定としており、ノード 7 に流出した想定のファイルである samp.zip を配置している。

表 4

ノード番号	回線速度設定	アップロードファイル	
1	999	large-deer.jpg	3.8 MBytes
		ubuntu-ja-9.10-desktop-i386.iso	669 MBytes
2	980	Win2.txt	24 Bytes
3	970	Win3.txt	19 Bytes
4	800 (PORT0)	xx.txt テストドキュメント.txt	3 Bytes 9 Bytes
5	800	Win5.txt	19 Bytes
6	800	Win6.txt	19 Bytes
7	1100	Win7.txt	19 Bytes
		samp.zip （流出ファイル）	4.26 KBytes
8	1000	Win8.txt	19 Bytes

5.1.3 クローラ実行環境

クローラを実行させたマシンの構成を表 5 に示す。

表 5:クローラの実行マシン構成

CPU	Pentium M 1.4GHz
メモリ	512MBytes

また、クローラに与えたパラメータは表 6 の通り。クローラが Winny ネットワークに接続するために用いる初期ノードはノード 1 のみとした。キークローラのスレッド数は最大 10、ファイルダウンローダのスレッド数は最大 3 とした。

表 6:パラメータ

項目	値
初期接続ノード	ノード 1
キークローラスレッド数の最大	10
ダウンローダスレッド数の最大	3

5.2 実行結果

検証環境においてクローラを実行し、(1)クローリングに要する時間、(2)流出ファイルを発見するまでに要した時間、(3)大容量ファイル（ubuntu-ja-9.10-desktop-i386.iso）の検査完了まで要した時間、(4)大容量ファイルを除いた全ファイルの検査完了まで要した時間、(5)クローラのネットワークトラフィック量、を記録することとした。

結果としてはファイルを発見することができ、(1)~(5)については以下の通りとなった。

(1)全ノードを 1 回ずつ訪れるのに要した時間

約 30~60 秒

(2)流出ファイルを発見するまでに要した時

3 分 14 秒

(3)大容量ファイルの検査完了までの時

10 時間 13 分 38 秒

(4)大容量ファイルを除いたファイル検査完了までの時間

11 分 45 秒

(5)クローラのネットワークトラフィック

最大　約 80KBytes/sec

平均　約 30KBytes/sec

5.3 考察

クローラはすべてのノードを発見して巡回し、ZIP アーカイブに含まれる流出ファイルを発見するという一連の動作を正しく完了した。ただし、実行時間に関しては、ノード数以上のスレッドを並列実行したにもかかわらず、全ノードの巡回にかかる時間が長くなるという結果となった。さらに、大容量ファイルでは、ネットワーク性能から期待される値よりもはるかに長い時間をダウンロードに要している。これは、Winny プロトコルの特徴で、拡散クエリへの応答に時間がかかることと、大容量ファイルのダウンロードを途中で一時切断することが原因であると推測される。

キークローラやダウンローダの実行スレッド数を増やすことで巡回時間とダウンロード時間を短縮する可能性は高いが、トラフィック量などネットワークへの影響は大きくなる。

6. まとめと課題

本稿では、組織内の電子文書が P2P ネットワークに流出したことを想定し、流出を早期に発見して被害の拡大を防ぐため、P2P ネットワーククローラによるファイル検索を提案した。さらに、Winny を対象にクローラを試作し、模擬ネットワークで正しく動作することを検証した。

今後の課題としては、ファイル検査の方式の検討および、実ネットワークでの運用を想定した、検出性能とトラフィックへの影響の評価などが挙げられる。

参考文献

1) 江崎浩:”P2P 教科書”, インプレス R&D(2008-01)

2) 金子勇:”Winny の技術”, アスキー(2005-10)