JPCERT CC®

「P2P 型ボット分析レポート」

有限責任中間法人 JPCERT コーディネーションセンター

平成 19 年 6 月 21 日

JPCERT CC®

目次

本レポートの一部は、悪用されることを回避する目的で公開を差し控えています。

1. はじめに(背景)

JPCERT/CC では、かねてよりボットの脅威を認識し過去 2 回にわたってその実態を調査してきた[1] [2]。特に 2005 年に行った調査「ボットネットの概要」では、ボットのソースコードを分析し、DoS 攻撃機能、スパムメール不正中継機能、バックドア機能など多くの機能が実装され、様々な金銭詐取の道具として使用されていることを明らかにした[2]。

従来のボットネットでは、C&C(Command and Control:指揮統制)サーバとして IRC(Internet Relay Chat)が利用されていた。その一方、近来 TCP ポート 80 番や 443 番を使う IRC サーバに接続するボットの急増が報告されている[3]。これは、ファイアウォール等によるアウトバウンドのトラフィック制御の影響を受けないポートを使用するようになったと想定できる。このように、情報セキュリティ関係機関と情報セキュリティベンダの綿密な連携の下での調査活動によって、ボットやボットネットの実態が次第に把握されつつある一方で、ハーダーやボット作成者は金銭詐取の道具を守るために、C&C サーバとの通信などを発見しづらくし、ボットネットの活動を「見えない化・潜行化」させていく可能性が高いことが予期できる。

このボットネットの活動を潜行化させる手段のひとつとして、命令の送受信に Peer to Peer(以下、P2P)型の通信を使用することが挙げられる。また、過去の調査の中で分析が行われた「Agobot/Phatbot」のソースコードの中に P2P メカニズムの実装コードが含まれていたことが判明している。さらに、実際に P2P 型の通信を使用して命令の受信を行うボットが確認されたとの報告[4]や、既存の P2P ファイル共有ソフトで使用されるプロトコルを利用するボットも確認されている[5]。

このような背景から、本調査では命令の受信に P2P 型の通信を使用するボットや、既存の P2P ファイル共有ネットワークと関連するボットやボットネット(以下、P2P 型ボット)の実態や、その特徴などについて調査を行い、それらが今後さらなる脅威となるかを考察する。

---

[1] Telecom-ISAC Japan 「ボットネット実態調査」
http://www.digitalforensic.jp/Resume2005/koyama.pdf

[2] JPCERT/CC 「ボットネットの概要」
http://www.jpcert.or.jp/research/2006/Botnet_summary_0720.pdf

[3] 「80 番ポートを使う IRC ボットが増加中」, セキュリティ組織が注意喚起
http://itpro.nikkeibp.co.jp/article/NEWS/20061117/254070/?ST=securityhole

[4] Websense、ボットをインストールするワーム「Nugache」を確認
http://japan.internet.com/webtech/20060502/4.html

[5] Uwaga na trojana tworzącego duży botnet P2P
http://www.zonep2p.pl/it/it123.html

2. 調査方法の検討

2.1. ソースコード解析による調査

2005 年に行った「ボットネットの概要」調査において使用した「Agobot/Phatbot」のソースコードから、P2P メカニズムを実装していると思われる部分のコードに着目して、その挙動を分析する。また、検索エンジンを用いて入手することができるボットのソースコードの中に P2P メカニズムの実装が含まれているか調査する。

2.2. 静的分析による調査

ウイルス対策ソフトベンダ等では、ウイルス検体の届出窓口の設置や、ハニーポットの運用といった手段を用いて検体を収集している[6]。収集された検体は専門の分析チームへ渡され、シグネチャの作成とパターンファイルへの反映が行われる。このウイルス分析には大きく「動的分析」と「静的分析」いう二つの分析手法がある。今回は、P2P 型の通信を行うといわれているボット検体を入手し、静的分析を実施することで、感染動作の詳細について調査する。

---

[6] トレンドマイクロ「ウイルス解析の仕組み」
http://www.trendmicro.co.jp/about/tl_virus/index.asp

3. 調査結果

3.1. ソースコード解析による調査結果

JPCERT/CC が 2005 年に実施した調査「ボットネットの概要」において「Agobot/Phatbot」のソースコード中に P2P メカニズムと思われるコードの存在が確認された。その際の調査では、P2P メカニズムの実装は実験的なものであるとされている。本調査では、P2P メカニズムが実装されていると考えられるソースコードを詳細に解析した。

「Agobot/Phatbot」のソースコードは、Microsoft Visual C++ 6.0(以下、Visual C++)を用いて開発およびメンテナンスが行われていたと思われる。しかし、入手したソースコードには Visual C++のプロジェクトが組み込まれておらず、そのままではコンパイルされない状態であった。
ヘッダファイル「p2p.h」には、P2P メカニズムを実装していると考えられるいくつかのクラス(以下、P2P クラス)が定義されており、この中で定義されたクラスを実装している C++ソースファイル「p2p.cpp」の行数は、わずか 243 行であった。コード行数のみから機能や実装の完成度を正確に把握することはできないが、そのコード量から複雑な実装ではないと推測される。なお、P2P クラスの一部を利用するクラスが「ircgate.h」で定義され、「ircgate.cpp」で実装されていた。
以降、表 3-1 に示した各クラスについて解説する。

表 3-1 P2P メカニズムに関連するコードを含むファイルとクラス

＜非公開＞

まず、「CP2PConn」クラスについて、実装されていたメンバ変数およびメンバ関数を表 3-2 に示す。

表 3-2　CP2PConn クラスのメンバ関数とメンバ変数の概要

＜非公開＞

CP2PConn クラスには、サーバノードへの接続を行うクライアント動作が実装されていた。

メンバ変数に設定された接続先サーバノードの情報に従い、Run 関数で接続を行う。接続が確立された後、QueryServers 関数でサーバノードが保持しているノードリストを要求する”server_query”コマンドを送信し、その応答である”server_query_reply”コマンドを受信する。その後、ループでコマンド受信を行う。受信処理が実装されていたコマンドは”message”コマンドのみであり、”message”コマンドのペイロードに含まれるコンテンツの受信は、CP2PManager クラスの ForTargets 関数に委ねられている。

次に、「CP2PServer」クラスに実装されていたメンバ関数およびメンバ変数を表 3-3 に示す。

表 3-3　CP2PServer クラスのメンバ関数とメンバ変数の概要

＜非公開＞

CP2PServer クラスはその名称からも推測される通り、クライアントノードからの接続を受け付けるサーバ動作処理が実装されていた。

メンバ変数では、自身の IP アドレスおよび待ち受けポート、クライアントノードの IP アドレス、サーバ動作の実行状態などを保持する。
Run 関数が実行されるとクライアントノードからの接続を受け付けた後、ループでコマンド受信を行う。受信処理が実装されていたコマンドは”message”コマンドおよび"server_query"コマンドであった。"server_query"コマンドを受信した場合、応答としてサーバノードリストを含む”server_query_reply”コマンドを送信する。”message”コマンドに含まれるコンテンツの受信は、CP2PConn クラスと同様に、CP2PManager クラスの ForTargets 関数に委ねられている。

次に、「CP2PManager」クラスに実装されていたメンバ関数およびメンバ変数を表 3-4 に示す。

表 3-4　CP2PManager クラスのメンバ関数とメンバ変数の概要

＜非公開＞

CP2PManger クラスには、P2P ネットワークの維持管理を担う処理とメッセージの送受信を行う処理が実装されていた。

メンバ変数では、サーバノードへの接続情報、クライアントノードからの接続情報およびメッセージ送受信の対象(ターゲット)を保持する。
P2P ネットワークへの接続を維持する処理として、サーバノードへのコネクション数が上限値を下回った場合に、CP2PConn のインスタンスを生成して新しいサーバノードへの接続を行う。また、自ノードに接続しているクライアントの数が上限値を下回った場合に、CP2PServer のインスタンスを生成して新しいクライアントからの接続を受け付ける処理が Think 関数に実装されていた。接続先サーバノードおよび接続を受け付けるクライアントノードの選定方法は、リストに登録されたものを順に選ぶものであり特殊なアルゴリズムは使用されていなかった。
初期化を行う Init 関数では、TCP 接続を待ち受ける処理が実装されていた、デフォルトで使用されるポート番号は 24288 であった。サーバノード数およびクライアントノード数の上限値が、それぞれ 20 に設定されており 1 ノードが通信する最大ノード数が 40 ノードである。このため、形成される P2P ネットワークは既存の P2P ファイル共有ネットワークのような大規模なものではないと推測される。
また、メッセージ送受信のターゲットを管理する処理が、AddTarget 関数および DelTarget 関数で実装されていた。
Broadcast 関数は、特定のフラグが付与されたメッセージを、自身が保持している全てのクライアントノードに対して送信し、P2P ネットワークに拡散する。なお、メッセージがネットワーク内でループすることを防ぐ処理が実装されていた。
ForTargets 関数は、メンバ変数に登録されているターゲットからのメッセージを受信するため、CP2PTarget クラスの Recv 関数の呼び出しを行う。
Broadcast 関数、ForTargets 関数は、先に記載したクライアント動作を行う CP2PConn クラスおよびサーバ動作を行う CP2PServer のインスタンスから呼び出される。

次に、「CP2ＰTarget」クラスに実装されていたメンバ関数およびメンバ変数を表 3-5 に示す。

表 3-5　CP2PTarget クラスのメンバ関数とメンバ変数の概要

＜非公開＞

CP2PTarget クラスは、先に示した 3 つのクラスによって形成される P2P ネットワーク上でメッセージを送受信するためのインターフェイスが実装されていた。
メンバ変数は、P2P ネットワークを管理するマネージャへのポインタ(CP2PManger のインスタンス)のみである。P2P ネットワークを利用した通信を実現するためには、このクラスを継承し拡張した別のクラスが必要と考えられる。
以上の解析結果から、P2P クラスは小規模な P2P ネットワークを形成し、その上でメッセージを送受信するインターフェイスを実装しているに過ぎず、この P2P クラスだけでは、従来の IRC を用いた C&C メカニズムを置換することができないと考えられる。

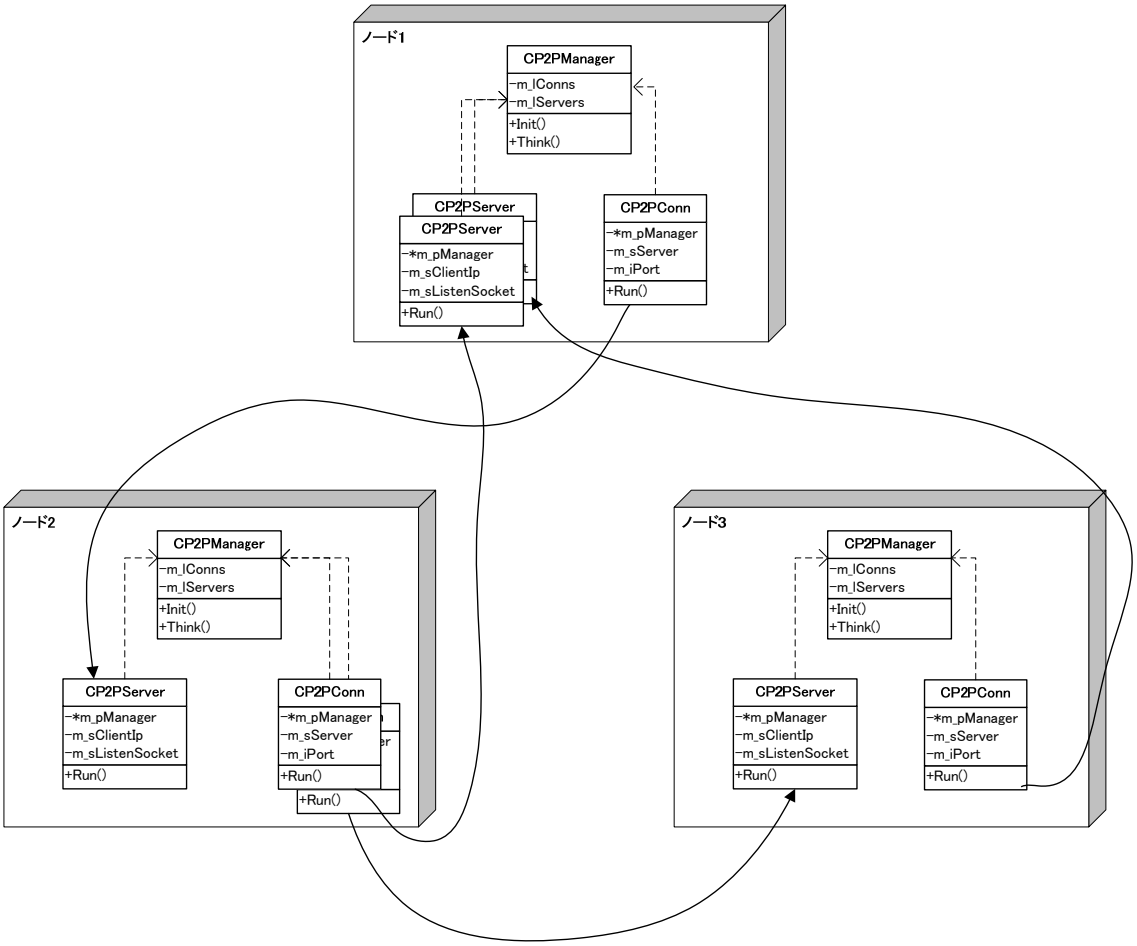

図 3-1　ノードとクラスの関係

次に、ヘッダファイル「ircgate.h」で定義され、「ircgate.cpp」に実装されていた CIRCGate クラスの解析結果を表 3-6 に示す。

表 3-6　CIRCGate クラスのメンバ変数とメンバ関数の概要

＜非公開＞

ヘッダファイル「ircgate.h」は、P2P クラスが定義されているヘッダファイル「p2p.h」をインクルードしており、「ircgate.cpp」に実装されている IRCGate クラスは CP2PTarget クラスを継承していた。しかし、そのファイルやクラスの名称から推察できる通り、実装の内容は TCP ポート 6667 番でクライアントからの接続を待ち受ける IRC サーバであった。また、CP2PTarget 以外のクラスを継承した実装は存在せず、P2P メカニズムとして未完成であり、動作しないものと考えられる。
メンバ変数として、クライアントの IP アドレスおよびニックネームを保持する変数が存在したが、これらはスカラ変数であり単一のクライアントの情報しか保持できないことから、通常の IRC サーバとして動作するように作成されたものではないと推測される。

以上の理由をふまえ、CIRCGate クラスは P2P クラスを用いて形成する P2P ネットワーク上で IRC プロトコルを動作させようとしたもと推測される。

3.2. 静的分析による調査結果

本調査では、以下のウイルスについて静的分析を実施した。

- W32.Nugache.A@mm
- Trojan.Peacomm

3.2.1. W32.Nugache.A@mm

(1) 検体の概要

今回、静的分析の対象とした検体のファイルサイズと MD5 ハッシュ値は以下の通りである。

(ア) ファイルサイズ

177,152 バイト

(イ) MD5 ハッシュ値(128 ビット)

74600E5BC19538A3B6A0B4086F4E0053

(2) P2P 型通信の実装状況

一部報道とは異なり、W32.Nugache.A@mm は P2P 型通信を行わないことが判明した。当該検体は、あらかじめリストとして保持している特定の IP アドレス(以下、初期接続ノード)に接続する。W32.Nugache.A@mm に感染した PC は、これら初期接続ノードに TCP8 番ポートを使用して接続を行う。接続には IRC プロトコルが使用される。一方、W32.Nugache.A@mm の感染 PC 間での通信リンク確立といったコードは含まれていない。

(3) コマンド受信ならびに実行機能

当該検体は前述の IRC 通信によりハーダーからのコマンドを受信し、それを実行する機能を備えている。当該検体が受け付けるコマンド一覧を表 3-7 に示す。

表 3-7　W32.Nugache.A@mm のコマンド一覧

＜非公開＞

(4) IRC 通信の符号化

当該検体には、特定の符号化処理が実装されている。この符号化処理は IRC 通信の送受信前に必ず実施される。

3.2.2. Trojan.Peacomm

(1) 検体の概要

今回、静的分析の対象とした検体のファイルサイズと MD5 ハッシュ値は以下の通りである。

(ア) ファイルサイズ

29,379 バイト

(イ) MD5 ハッシュ値(128 ビット)

9BEE3B4AE3DA03EB3D5240B85372BCFA

(2) 隠匿機能

＜非公開＞

＜非公開＞

図 3-2 Trojan.Peacomm の持つ隠匿機能

(3) P2P 型通信の実装状況

静的分析を実施した結果、当該検体は高度な隠匿機能を有しており、処理の詳細を把握することはできなかったため、動的分析も合わせて実施した。アンチウイルスベンダの情報によれば、当該検体は overnet プロトコル[7]による P2P ネットワークを形成することが報告されている。overnet プロトコルは、ファイルの検索、ピアの確認等は UDP で行い、ファイルのダウンロードのみ TCP で行う。

静的分析の結果、TCP、UDP それぞれによるソケット処理を行っていることが確認できた。また動的分析では、プログラムの初期動作として system32 下に peers.ini ファイルを生成し、当該ファイル中に初期接続ノードの IP アドレスをエンコードした文字列を保存することが分かった。この処理は、静的分析により該当する部分を発見することができた。また、動的分析では初期接続ノードへの UDP によるデータ送信と、services.exe による UDP ポート 4000 番の使用が確認された。

(4) コマンド受信ならびに実行機能

静的分析を実施した結果、当該検体は高度な隠匿機能を有しており、P2P 型の通信処理の詳細を把握することはできなかったが、HTTP によるファイルのダウンロード、および実行を行う処理が確認された。services.exe に注入された不正コードはファイルの download および実行を行うコードと推測される。

(5) IRC 通信の符号化

静的分析を実施した結果、IRC 通信の実装は確認できなかった。

---

[7] https://opensvn.csie.org/mlnet/trunk/docs/overnet.txt

4. 考察

「Phatbot/Agobot」のソースコードを分析した結果、同ボットのソースコードに含まれる P2P メカニズムは、小規模な P2P ネットワークを形成するが、ボットネットを制御するための処理が未実装であった。加えて、実際に出現した W32.Nugache.A@mm ならびに Trojan.Peacomm を静的分析した結果、これらも、従来の IRC を用いた C&C メカニズムを P2P 型の通信で置換するものではなかった。

DDoS 攻撃やスパムメール送信などを効率的に実施するためには、ボット感染コンピュータにコマンドを同時かつ一斉伝達する必要がある。現状のボットに P2P ネットワークを構築する機能が実装されていない理由の一つは、同報性・同期性に優れている IRC ネットワークのほうが、P2P ネットワークよりも DDoS 攻撃やスパムメール送信に適しているためと推測される。
また、P2P ネットワークの特長としてネットワークの高可用性が挙げられるが、安定した P2P ネットワークの構築には一定数以上のノードが必要と推測される。加えて、一度構築された P2P ネットワークは、その規模にもよるが管理が難しいという性質も考えられる。
IRC を C&C サーバとする現状のボットネットは、C&C サーバを容易に変更でき、十分な可用性をもっており、またその管理がしやすいというメリットもある。
このような理由が、P2P メカニズムへ移行されない二つめの理由と考えられる。

昨今、企業などのネットワークでは、ファイアウォールなどによるアウトバウンドの通信制御が浸透してきており、それを回避する目的で C&C サーバとの通信に TCP ポート 80 番や 443 番などを使用する場合や、Web サーバが C&C サーバとなって、HTTP プロトコルで制御されるボットネットが拡大しているという懸念もある。
1 つのボットネットを形成するボットの数を小規模に抑えるボットネットの小規模化や内部ネットワークを標的とした感染活動、むやみな攻撃活動(スパムメールの不正中継や DDoS 攻撃)の抑制という報告もある[8]。このように、ボットネットの活動は潜行化の気配が見られ、既存のプロトコルと容易に判別可能な P2P 通信が行われる可能性は低い。

---

[8] ボットネットは“目立たない”ように工夫を凝らす
http://itpro.nikkeibp.co.jp/article/NEWS/20061116/254007/

以上のような背景や理由をふまえ、独自の P2P 型の通信メカニズムを実装したボットによるボットネットの拡大よりも、恒常的に発生しうるトラフィックである HTTP や HTTPS に扮した通信、「Winny」などの既存の P2P ファイル共有ネットワーク、「Skype」などの P2P 型の VoIP ネットワークや IM に利用されている既存の大規模な P2P ネットワークを利用したマルウェアの拡散が、直近の脅威として懸念される。

5. 参考資料

5.1. 「Agobot/Phatbot」P2P 関連ソースコード

ヘッダファイル「p2p.h」

＜非公開＞

ソースファイル「p2p.cpp」

＜非公開＞

ヘッダファイル「ircgate.h」

＜非公開＞

ソースファイル「ircgate.cpp」

＜非公開＞

5.2. W32.Nugache.A@mm の初期接続ノード一覧

＜非公開＞

5.3. Trojan.Peacomm の初期接続ノード一覧

＜非公開＞