RAT サーバの動作を遠隔操作者と同じ操作画面で観測する方法

高橋 佑典† 小林 大朗† 陳 悦庭† 米持 一樹†

吉岡 克成† 松本 勉†

†横浜国立大学
240-8501 神奈川県横浜市保土ケ谷区常盤台 79-7
{takahashi-yusuke-pw, kobayashi-masaaki-ny}@ynu.ac.jp, f9190yuki@gmail.com,
{yonemochi-kazuki-fj, yoshioka, tsutomu}@ynu.ac.jp

あらまし 近年問題となっている標的型攻撃ではRAT (Remote Administration Tool, Remote Access Trojan) が利用されることが多い. 一般にRATは攻撃対象ホスト内で動作するRATサーバと, それを遠隔操作するためのRATクライアントからなる. 近年のRATクライアントはGUIが用意されており直感的な遠隔操作が可能となっている. 本研究では, 攻撃者によるRATサーバの遠隔操作の状況を, 別途用意した監視用RATクライアントのGUI上で観測することを試みる. 具体的には, まず, 監視対象のRATサーバから攻撃者のRATクライアントとの通信に必要な認証情報を抽出し監視用RATクライアントに適用しておく. 次にRATサーバから攻撃者のRATクライアントへの通信を観測し, 同様の通信をプロキシにより監視用RATクライアントへ送信することで, 攻撃者による遠隔操作の様子を監視用RATクライアントのGUI上でリアルタイム観測する.

Observing RAT server's behavior through its client GUI

Yusuke Takahashi† Masaaki Kobayashi† YuehTing Chen†

Kazuki Yonemochi† Katsunari Yoshioka† Tsutomu Matsumoto†

†Yokohama National University
79-7 Tokiwadai, Hodogaya-ku, Yokohama-shi, Kanagawa, 240-8501 Japan
{takahashi-yusuke-pw, kobayashi-masaaki-ny}@ynu.ac.jp, f9190yuki@gmail.com,
{yonemochi-kazuki-fj, yoshioka, tsutomu}@ynu.ac.jp

Abstract Remote Administration Tool or Remote Access Trojan (RAT) is often used in targeted attacks. RAT generally consists of a server that runs on the targeted host and a client that remotely controls the server. Recent RAT client provides GUI so that attacker can intuitively control the server. In this study, we propose a method to observe behavior of RAT server controlled by an attacker through its client GUI. First, assuming that an active RAT server has been detected and identified in a network, we extract credential information from the server for connecting its client. The credential information is then applied to our monitor client. Next, we place a proxy between the RAT server and the

attacker to monitor the traffic between them. The proxy sends all traffic from the server to the monitor client, where all activity is observed through its GUI.

1 はじめに

近年，特定の企業や組織を狙った標的型攻撃が脅威となっている．この標的型攻撃には，しばしば RAT（Remote Administration Tool，Remote Access Trojan）が使用される．RAT は攻撃者に対象ホストのリアルタイム遠隔操作を提供するツールであり，操作対象ホスト上で動作する RAT サーバと，RAT サーバを遠隔操作するための RAT クライアントからなる．近年の RAT クライアントは，直感的な遠隔操作を行えるような GUI が提供されていることが多く，その使いやすさから様々な攻撃に悪用されている[10]．特に，関係者を装って RAT 感染をもたらす悪性ファイルを添付したメールを送る手法が広く採られており，最近では，ヨーロッパやアジアの企業や国会議員が，このような攻撃の被害にあったという報告がある[2, 3]．

標的型攻撃対策の重要性が広く認知されているにも関わらず，有効な対策が採られず上記のようなインシデントが多発している理由の１つに，攻撃側の振る舞いや動向に関する情報不足がある．特に RAT に関しては，機能的な側面だけでなく，その利用方法や運用実態を把握することが対策を行う上で重要といえる．

そこで本研究では，RAT の利用実態を把握するため，RAT サーバ検出時に，当該サーバが攻撃者に遠隔操作される様子を容易に観測する手法を提案する．具体的には，攻撃者が遠隔操作時に見ているはずの RAT クライアント側の GUI を，別に用意した監視用 RAT クライアントの GUI で再現することでリアルタイム監視を行う．まず，監視対象の RAT サーバから攻撃者の RAT クライアントとの通信に必要な認証情報を抽出し，監視用 RAT クライアントに適用しておく．次に RAT サーバから攻撃者の RAT クライアントへの通信を観測し，プロキシにより監視用 RAT クライアントへ同様の通信を送信することで，攻撃者による遠隔操作の様子を監視用 RAT クライアントの GUI 上でリアルタイム観測する．実証実験では，RAT の一種である，PoisonIvy と Cerberus についてテスト用 RAT サーバ検体を作成し，対応する RAT クライアントを通じてこれを遠隔操作した際に，その様子を監視用 RAT クライアント上で観測できることを確認した．また，実検体に対する有効性を調べるために VirusTotal [7]から提供を受けた RAT サーバ検体 27 体から認証情報の抽出を試み，そのうち 17 体から抽出が可能であることを確認した．

本稿では，まず第２章で RAT について述べる．第 3 章で提案手法について説明し，第 4 章で実証実験について記述し，第 5 章でまとめと今後の課題を述べる．

2 RAT

RAT は，RAT クライアントと RAT サーバで構成される．RAT サーバが動作するホストを RAT クライアントによって遠隔操作することが可能であり，ファイル操作やレジストリ操作，画面キャプチャなど，様々な機能がある．多くの RAT クライアントが GUI を持っており，上記の機能を直感的に使用することができる．標的型攻撃では RAT が多く使用されており，その中でも特に PoisonIvy が使用されるケースが多いという報告がある[4]．マルチプラットフォームで動作する RAT も現れており，RAT による被害はこれからも増加していくと考えられる[6]．以下では，PoisonIvy と Cerberus について詳しく述べる．

RAT サーバは実行可能ファイルであり，RAT クライアントのサーバ作成機能を用いて容易に作成できる．RAT サーバには，RAT クライアント側の IP アドレスやドメイン，ポート番号，接続時の認証に用いるパスワードなどが埋め込まれる．

図１に RAT サーバと RAT クライアントの通信

の流れを示す．侵入先のホスト上で RAT サーバが実行されると，RAT クライアント側と 3way ハンドシェイクを行ない，TCP セッションを確立する．その後，認証が行われ，RAT サーバに埋め込まれたパスワードと RAT クライアントに設定したパスワードが一致していなければ，セッションは切断される．認証に成功した場合は，上記のパスワードから生成された鍵による共通鍵暗号通信が行われる．この TCP セッションは RAT サーバ側から開始されるため，ファイアウォールで外部からの接続を制限している組織内ネットワーク内のホストであっても，外部から操作することができる．また，PoisonIvy については，組織内から外部への通信にプロキシサーバを利用する場合にも対応しており，プロキシサーバの設定を RAT サーバに埋め込む方法と，プロキシサーバの設定を RAT サーバ感染ホスト上で取得する方法がある．

RAT の通信の特徴として，リアルタイム操作を実現するための定期的な keep-alive 通信や，独自プロトコルによるペイロードの暗号化などがあり，この特徴を検知に用いる手法が提案されている[1, 5]．本研究では，これらの手法を含めて何らかの方法で活動中の RAT サーバを検知したという前提で，攻撃者が RAT を悪用してどのように不正活動を行なうかを観測する手法を提案する．なお，提案手法は，RAT サーバ検体のサンドボックス内での動的解析時にも利用が可能である．

3 提案手法

本章では，活動中の RAT サーバの挙動を，RAT クライアントの操作画面を用いて観測する手法について説明する．図 2 に提案手法の全体図を示す．提案手法は，監視対象の RAT サーバが動作する監視対象ホスト，監視用ホスト群，監視対象ホストと攻撃者の中間にあり，監視対象ホストから攻撃者への通信を観測し，監視用ホスト群に対して同様の通信を送信するためのプロキシ用ホストからなる．

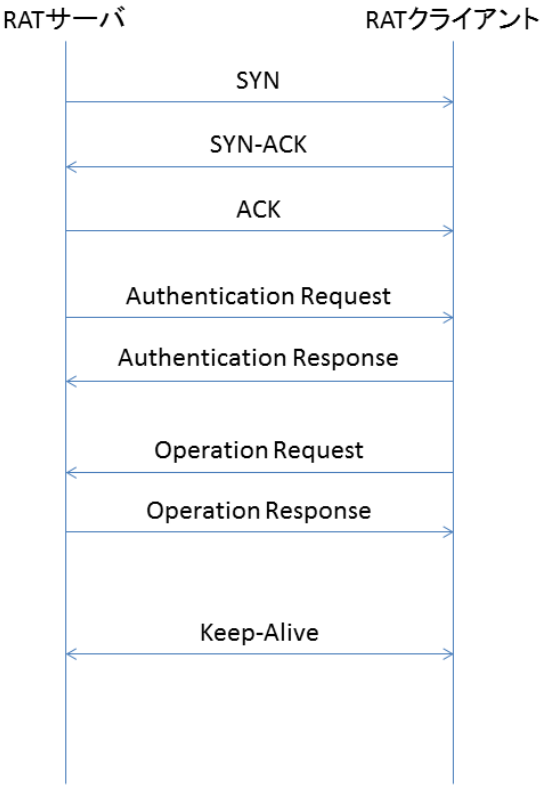

図 1 RAT 通信

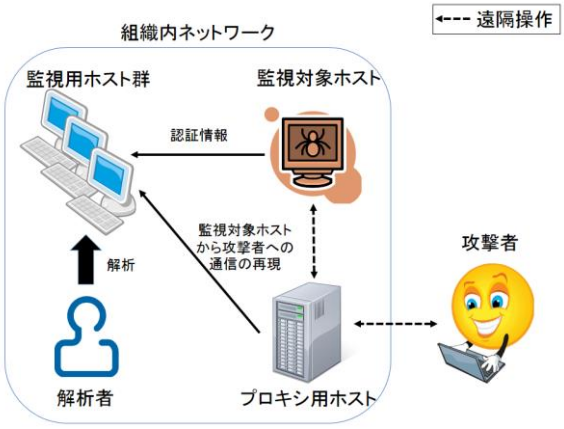

図 2 提案手法の全体図

3.1 認証情報の抽出

まず，監視対象の RAT サーバから，攻撃者の RAT クライアントとの接続に必要な情報の抽出を行なう．ここで抽出する情報は，アクセス先の IP アドレス，ポート番号，パスワードの 3 つである．これらの認証情報は，RAT サーバの実行可能ファイルの特定箇所に埋め込まれている

ため，シグネチャベースのパターンマッチにより，抽出する．実行可能ファイルが取得不可能の場合は，メモリフォレンジックツール(Volatility[9]等)を利用し，パスワードを抽出する．詳しくは4.3節にて述べる．

3.2 監視用クライアントの設定

監視用ホスト上で監視用 RAT クライアントを実行する．次に監視用 RAT クライアントに対して，3.1 節で抽出したパスワードを適用し，接続待機状態にする．なお，PoisonIvy クライアントのように 1 つのウィンドウ内で操作項目を選択して使用するタイプの GUIをもつ RAT クライアントでは，選択していない操作項目についてはリアルタイムで監視用ホストの GUI に反映がされない．そこで，監視したい項目分だけ監視用ホストを用意し，各監視用ホスト上で監視対象の項目を選択しておく．全ての監視用ホストにプロキシホストから同様の通信を送信することで複数項目の同時監視を行う．

3.3 RAT サーバ通信の再現

プロキシ用ホストは，監視対象ホストから攻撃者のRATクライアントへの通信を観測し，同様の通信を監視用ホスト群に送信する．

3.4 再接続対応

RATサーバとRATクライアントがTCPセッション確立後に行う認証時の通信を保存しておき，何らかの理由でプロキシ用ホストと監視用ホスト群とのセッションが切れた時には，この通信を再送することで再接続し，その後にRATサーバの通信を監視用ホスト群に送信する．

4 実証実験

本章では，提案手法の実証実験について記述する．実証実験では，まずテスト検体を用いて提案手法が意図通りに働くかを確認した．次に，実検体に対する適用可能性を調べた．

4.1 実験環境

1 台の実マシン上に，監視対象ホスト，監視用ホスト A と監視用ホスト B，攻撃者ホストをそれぞれ仮想マシンとして用意し，攻撃者ホストは仮想ネットワーク NW1 に，監視対象ホストと監視用ホスト A，B は同一の仮想ネットワーク NW2 にそれぞれ属するようにする．また，NW1 と NW2 の間は，プロキシホストを介して接続する．実験環境のネットワーク構成を図 3 に示す．監視対象ホスト上で， RAT サーバが実行され，攻撃者が攻撃者ホスト上の RAT クライアントを用いてこれを遠隔操作する様子を，監視用ホスト A，B により観測するというシナリオとする．

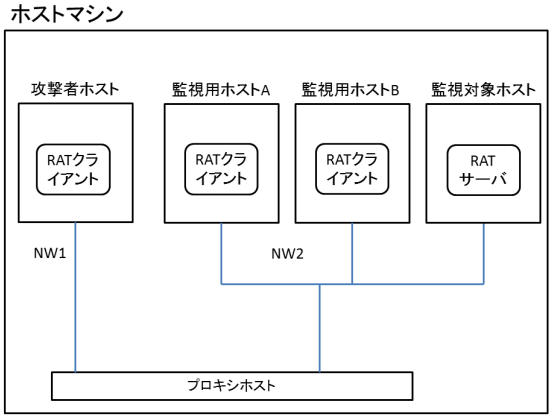

図 3 実験環境

プロキシホストで各ホストの通信の監視を行い，RAT サーバから攻撃者の RAT クライアントへ送られたパケットを観測した場合は，即座に同内容のパケットを監視用ホスト A，B へも送信する．プロキシホストはホスト OS 上で動作する iptables と自作の python スクリプトにより実装した．

テスト用検体として，PoisonIvy と Cerberus の RAT サーバをそれぞれ作成し，これらを用いて実験を行った．

4.2 実験結果

PoisonIvy と Cerberus のいずれも監視用 RAT クライアントの GUI において攻撃者ホストの遠隔操作内容をリアルタイム表示させることがで

きた．監視用ホストを 2 台用意することで，2 つの異なる操作項目について同時に監視可能であることも確認した．実験の様子は参考文献[11]において動画として公開している．

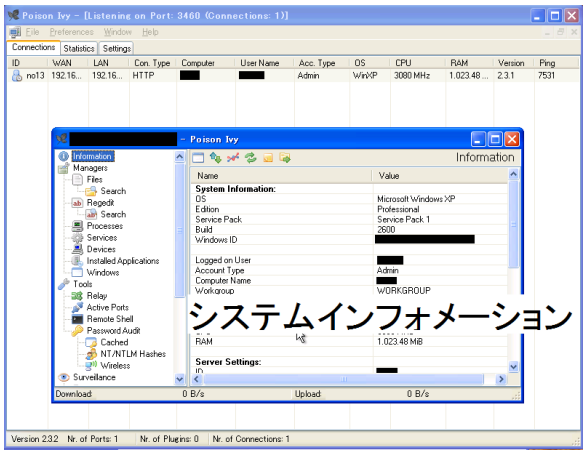

図 4　PoisonIvy 監視用ホスト A

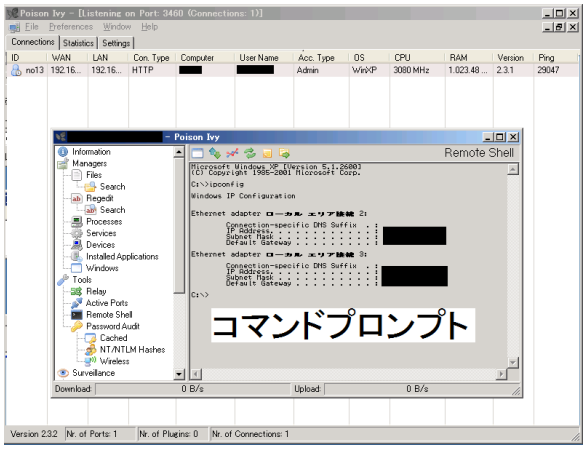

図 5　PoisonIvy 監視用ホスト B

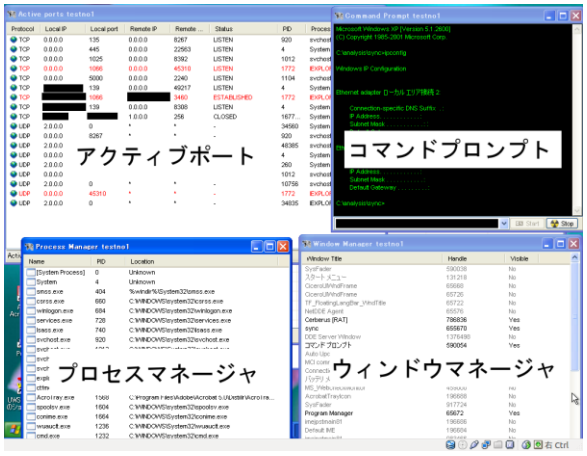

図 6　Cerberus 監視ホスト

監視対象ホスト上で RAT サーバが実行され，攻撃者の RAT クライアントと接続されると，監視用 RAT クライアントにも接続成功の表示がされた．図 4，図 5 はそれぞれ異なる操作画面を表示させた PoisonIvy の監視用ホストの画面キャプチャ画像である．それぞれ異なる操作項目について，攻撃者が操作した結果が反映されていることがわかる．図 6 は Cerberus の監視用ホストの画面キャプチャ画像である．図のように Cerberus は複数の操作項目を1つのウィンドウで同時に表示させることができる．

RAT の機能の 1 つに RAT サーバが動作しているマシンの画面キャプチャ機能がある．PoisonIvy と Cerberus のいずれも画面キャプチャ機能があるが，攻撃者ホストでこの機能を使用すると，監視用ホストとの接続が切れてしまう問題が判明したが，PoisonIvy については，3.4 節で説明した再接続対応により，継続監視が可能であった．このことから PoisonIvy の認証時の通信内容は同一のサーバにおいて固定であり，再送により容易に認証に成功することがわかった．Cerberus については再接続対応が未実装であり，今後の課題としたい．

4.3　実検体からのパスワード抽出

提案手法では，RAT サーバのアクセス先や接続の際に使われるパスワードを事前に把握している必要がある．そこで，VirusTotal に投稿されている PoisonIvy と Cerberus の検体を取得し，パスワードが抽出できるか調査した．

まず，自作した PoisonIvy サーバ検体ファイル中に設定したパスワードが含まれるかを確認したところ，図 7 のように，パスワードが平文のまま実行可能ファイルの一部として埋め込まれていることが分かった．

図 7 の中ほどの「test-id」，「test-group」，「127.0.0.1」は，それぞれ管理 ID，管理グループ，実行後に接続する IP アドレスである．その後の「E」という文字のあとに出てくる赤線部分が RAT サーバのパスワードであり，この RAT サーバのパスワードは「password」である．このような特定の文字列をシグネチャとし，実検体から

パスワードを抽出した．一部の検体についてはパッカー等による難読化が施されていたため，実行可能ファイルからの直接抽出はできなかったが，メモリフォレンジックツールである Volatility のパスワード検知機能も併用することでパスワードの抽出が行えた．表 1 はパスワードの抽出に成功した検体数である．

```
0001620: 0408 0053 7475 6250 6174 6818 0428 0053  ...StubPath..(.S
0001630: 4f46 5457 4152 455c 436c 6173 7365 735c  OFTWARE\Classes\
0001640: 6874 7470 5c73 6865 6c6c 5c6f 7065 6e5c  http\shell\open\
0001650: 636f 6d6d 616e 6456 0435 0053 6f66 7477  commandV.5.Softw
0001660: 6172 655c 4d69 6372 6f73 6f66 745c 4163  are\Microsoft\Ac
0001670: 7469 7665 2053 6574 7570 5c49 6e73 7461  tive Setup\Insta
0001680: 6c6c 6564 2043 6f6d 706f 6e65 6e74 735c  lled Components\
0001690: fa0a 0700 7465 7374 2d69 64f9 0b0a 0074  ....test-id....t
00016a0: 6573 742d 6772 6f75 7090 010d 0009 3132  est-group.....12
00016b0: 372e 302e 302e 3100 840d 8c01 0400 0000  7.0.0.1.........
00016c0: 0000 c102 0400 ffff ffff 4501 0800 7061  ..........E...pa
00016d0: 7373 776f 7264 090d 0100 0112 0e04 0074  ssword........t
00016e0: 6573 74f6 0301 0001 6501 2600 7b44 3336  est.....e.&.{D36
00016f0: 4538 3642 352d 3034 3445 2d44 3041 442d  E86B5-044E-D0AD-
0001700: 3138 3834 2d30 3139 3631 4335 3831 4435  1884-01961C581D5
0001710: 417d 4104 0100 0142 040b 006d 736e 6d73  A}A....B...msnms
0001720: 6772 2e65 7865 fb03 0900 2921 566f 7141  gr.exe....)!VoqA
0001730: 2e49 342d 010c 0063 6f70 7966 696c 652e  .I4-...copyfile.
0001740: 6578 65f7 0301 0002 f803 0100 01f9 0301  exe.............
```

図 7　RAT サーバの設定部分

表 1　RAT サーバ検体からのパスワード抽出

RAT の種類	成功数(体)	解析数(体)
PoisonIvy	11	17
Cerberus	6	10

4.4　考察

提案方式適用のシナリオ　提案方式を適用するには，監視対象の RAT サーバに対応する RAT クライアントプログラムを入手し，かつ監視対象の RAT サーバから認証情報を取得する必要がある．また，実運用ネットワークにおいて，検出された RAT サーバを放置したまま観測を続けるのは，現実的ではない．このような前提が成り立つ観測のシナリオとしては，RAT サーバ検知後に検疫ネットワーク等に移動し，観測を続ける方法や，実運用ネットワークから隔離されたサンドボックス上で観測を行う方法が考えられる．

遠隔操作内容の独立性　攻撃者側の RAT クライアントは，攻撃者からのローカルな操作入力と RAT サーバからのデータ受信との両方に依存して状態が遷移するのに比べて，監視用 RAT クライアントは，ローカル操作を伴わずにデータ受信のみにより状態が遷移するため，2 つの RAT クライアントの状態に差異が生じてしまうという本質的な問題点が提案方式には存在する．しかし実際に評価実験を行った際には，この状態の差異による大きな不具合は確認されず，監視用 RAT クライアントがシャットダウンした場合には，再起動し，再接続処理を行うことで監視を継続することができた．これは，RAT クライアントの操作は，そもそも，それ以前の状態への依存度が低く，比較的独立しているものが多いためと推測される．実際の不正活動時にも何らかの理由でコネクションが切断されるケースは想定されるため，このような作りになっていると思われる．

認証情報抽出について　今回評価対象とした PosonIvy と Cerberus については，検体ファイルおよび実行時のメモリから比較的容易にパスワード等の認証情報を抽出できた．これには RAT サーバの作成手順が関係していると思われる．PoisonIvy では，ビルダの GUI から認証情報を入力し，作成ボタンを押下することで簡単に RAT サーバを作成可能となっているが，その実態は，コンパイル済みの RAT サーバの実行可能ファイルに後から認証情報を挿入する内部処理となっている．そのため，ソースコードレベルの難読化により認証情報を秘匿することは困難であり，比較的容易にこれが抽出できたものと思われる．一方，認証情報埋め込み後であっても，ランタイムパッカー等の利用は可能であるため，既存のアンパック手法の適用やメモリ展開後のダンプデータに対してパスワード抽出処理を行う必要がある．4.3 節の実験では，パスワード抽出には実行可能ファイルに対して直接マッチングを行うか，既存のメモリフォレンジックツールを用いるのみであったため，これ以外の方法の適用については今後の課題としたい．

ホストベースの監視との比較　RAT サーバの動作の監視は，サンドボックス解析等で広く用いられている API フックやエミュレータなどのホストベースの挙動観測手法を監視対象ホストに適用することによっても可能である．しかしながら，これらのホストベースの手法と提案手法は少な

くとも2つの点で異なる. まず, 提案手法は攻撃者の操作内容自体を観測するのに対して, 上述のホストベースの監視は操作の結果を観測する方法である. 遠隔操作の実態を把握する点では, いずれの情報も重要であるため2つの観測手法を相補的に併用することも考えられる. また, API フックやエミュレータによる監視機構を検知される可能性がある[12, 13, 14]のに対して, 提案手法は一度 RAT クライアント接続のための認証情報さえ取得すれば, 観測自体はネットワークベースで可能であり, 適切に実施すれば検知される可能性は低い.

5 まとめと今後の課題

RAT サーバの挙動監視, 攻撃者の行動監視の手法として, RAT クライアントの GUI を用いる手法の提案を行なった. RAT サーバから攻撃者の RAT クライアントへの通信を監視用に用意した RAT クライアントへ送信することで, 実験環境において攻撃者の操作結果のリアルタイム監視に成功した. また, VirusTotal に投稿されている検体を解析し, 本手法に必要な認証情報の抽出が多くの検体について可能であることがわかった. 今後は, テスト用の RAT サーバ検体ではなく実際の攻撃に使用される実検体を用いての観測実験を行う.

謝辞 本研究の一部は, JSPS 科研費 24680006 の助成により行われた.

参考文献

[1] 鳥居 悟, 清水 聡, 森永 正信, "RAT 通信監視手法の提案, " コンピュータセキュリティシンポジウム 2012 論文集, 2012 巻, 3 号, pp. 571-578, 2012.
[2] Frutas RAT による標的型攻撃 | Symantec Connect コミュニティ, http://www.symantec.com/connect/ja/blogs/frutas-rat-0 (Last Visit: 2013/08/26).
[3] ウイルスメール:国会議員2人に届く 朝日新聞記者装い一 毎日jp(毎日新聞), http://mainichi.jp/select/news/20130724k0000m040107000c.html (Last Visit: 2013/08/26).
[4] TrendMicro, "トレンドマイクロにおけるスレットリサーチの現状と課題, " MWS2012, 2A2-1, 2012.
[5] 山田 正弘, 森永 正信, 海野 由紀, 鳥居 悟, 武仲 正彦, "組織内ネットワークにおける標的型攻撃の検知方式, " 研究報告セキュリティ心理学とトラスト, 2013-SPT-6 巻, 53 号, pp. 1-6, 2013.
[6] エフセキュアブログ : 最近気になる Remote Administration Tool(RAT), http://blog.f-secure.jp/archives/50684057.html (Last Visit: 2013/08/26).
[7] VirusTotal, http://www.virustotal.com/jp/ (Last Visit: 2013/08/26).
[8] 2012 年、日本国内の持続的標的型攻撃を分析 ～自組織に有効な対策を行うには～ | トレンドマイクロ セキュリティ ブログ (ウイルス解析担当者による Trend Micro Security Blog), http://blog.trendmicro.co.jp/archives/6717 (Last Visit: 2013/08/26).
[9] volatility - An advanced memory forensics framework - Google Project Hosting, https://code.google.com/p/volatility/ (Last Visit: 2013/08/26).
[10] アジア圏で確認された標的型攻撃、RAT「RARSTONE」を利用 | トレンドマイクロ セキュリティ ブログ (ウイルス解析担当者による Trend Micro Security Blog), http://blog.trendmicro.co.jp/archives/7401 (Last Visit: 2013/08/26).
[11] 情報・物理セキュリティ研究拠点:攻撃者による RAT の遠隔操作をリアルタイム監視, http://ipsr.ynu.ac.jp/ratmon/
[12] Xu Chen, Andersen, J., Mao, Z.M., Bailey, M., Nazario, Jose, "Towards an understanding of anti-virtualization and anti-debugging behavior in modern malware, " Dependable Systems and Networks With FTCS and DCC, 2008. DSN 2008. IEEE International Conference

on, pp. 177 – 186, 2008.
[13] T. Raffetseder, C. Kruegel, E. Kirda, "Detecting System Emulators, " Proceeding ISC'07 Proceedings of the 10th international conference on Information Security, pp. 1-18, 2007.
[14] x86 API Hooking Demystified | Development & Security, http://jbremer.org/x86-api-hooking-demystified/ (Last Visit: 2013/08/26).