4-3　エンドホスト連携による不審プロセス分析

中里純二　津田　侑　高木彌一郎

本研究では普段利用することのない不審なプロセスを迅速に見つける手法を提案する。ホスト内部で動作するプロセスリストを取得し、特定ユーザが利用する不審通信を伴ったプロセスの抽出を行う。同一プロセスが動作するホスト数や頻度、ネットワーク状態からプロセスの出現特徴を算出し不審度を決定する。通信先などの監視を同時に行うことで不審な通信（C&C や RAT 通信など）を行うプロセスをいち早く検知することが可能になる。

1　はじめに

標的型攻撃は、まず標的とする相手や組織に関する情報をソーシャルネットワークなどのツールを巧みに利用し収集する。その後、メールなど一般的なコミュニケーションツールを用いて直接的な接触を試みる。このとき、組織内部に侵入するためのバックドアを設置する目的で、マルウェアやそれらをダウンロードするためのリンクなどをメールに添付する。添付されたマルウェアまたはリンクは、受信相手（標的となった相手）が自ら実行する必要があるが、攻撃者は事前に得られた情報を巧みに利用することで、相手に不信感を抱かせることなく添付ファイルを実行させる。また、利用されるマルウェア等はアンチウイルスソフトウェアによる検知を回避するなど、攻撃の事実を見つけることを困難にさせている。したがって、標的となった相手は攻撃されたこと自体に気付くことが遅れ、長期間にわたり組織内での不正活動を許してしまうことになる。実際に、国立研究開発法人宇宙航空研究開発機構（JAXA）では平成 23 年 3 月 17 日に標的型攻撃（メールによる侵入）によってマルウェアに感染し、翌年の平成 24 年 11 月 21 日にアンチウイルスソフトウェアによって発見されるまで、1 年半以上もの間マルウェアによる感染に気付くことなく、多くの情報が漏洩した可能性が高い事件が発生している [1]。このように、標的型攻撃に対してアンチウイルスソフトウェアなどでマルウェアの侵入を防ぐエンドポイント対策では、検知漏れにより情報漏洩が発生するなど重大なインシデントを招く恐れがあり、万全な備えになるとは言えなくなってきている。特に、標的型攻撃を検出できず、マルウェアが長期間にわたり感染した状態でいる場合は、攻撃者に組織内の情報収集や新たな攻撃を行う多くの機会を与えることになる。例えば、様々なサービスやそれらの管理者アカウント、特定の重要なシステムを探索する機会を与え、また侵入するチャンスを与えることとなり、より多くの機密情報にたどり着く可能性が高くなる。そこで、入口対策やエンドポイント対策のマルウェアの侵入自体を防止する対策はもちろんのこと、感染してしまうことを前提とした対策（内部対策）が重要となってきている [2]。

標的型攻撃の特徴として、標的組織内のホストの操作を制御するために RAT（Remote Administration Tool）や、OS などの情報を取得するための管理者用コマンドをはじめとした様々なツールが複合的に利用されることが知られている。すなわち、一般利用者が定常的に利用するアプリケーション（例えば Office Suite など）とは異なるプロセスが、攻撃の過程で標的となったホスト内で実行されることになる。そのため、前述したメールに添付されたファイルの実行などによりこれらのツールを含むマルウェアに感染した場合、その利用者や組織内の他の利用者が過去には実行したことの無い、初めて動作するプロセスが現れることが考えられる。また、RAT ツールの利用や新たにマルウェアをダウンロードする場合など、外部との通信が発生することが考えられるため、各プロセスの通信状態（待ち受けポートや通信先アドレスなど）を考慮したプロセス監視を行うことが重要となる。実際に、日本年金機構では標的型攻撃メールを開封したことによりマルウェアに感染し、不正な通信が発生していたことが知られている [3]。そのため、マルウェアの感染を前提とした対策では、通常では利用されることは少ないプロセスや通常とは異なる通信の発生をいち早く検出することが有効であると考えられる。特に、組織内などの利用アプリケーションが管理された（well-managed な）環境下においては、非常に有効な手段といえる。

そこで、本研究では利用者個々の端末内部で動作しているプロセス情報（プロセスリスト）を定期的に取

得し、利用者が普段利用しているプロセスか否かを判定する手法の提案を行う。プロセスの判定には、プロセス名とそのプロセスが実行された場所（実行パス）を同時に比較する。プロセスの比較に実行パスを含めることにより、正常プロセスに成りすました不正なプロセスの識別が可能になる。また、長期的に特定の利用者のみで動作するプロセスは、攻撃ツールが定期的に動作している可能性が高いため、不審度合いが非常に高いプロセスであることが考えられる。そこで、プロセスの不審度としてプロセスの出現頻度や同一プロセスを利用する利用者の数、そのプロセスの実行期間などを用いた出現特徴を定義する。また、各プロセスの通信状態を監視し、プロセスごとに見た通信先の出現頻度や、同一通信先へ通信を行っているホスト数などにより、不審通信を行うプロセスの抽出を行う。プロセスの不審度が一定以上のプロセスが発生した場合、そのプロセスを監視対象とし、より詳細な情報（例えば、利用する API 履歴など）を取得することで、マルウェアなどの悪意のあるプロセスを早期に発見し、迅速な対策につなげることが期待できる。

本稿では、実際に利用者から取得したプロセス情報を用いて、プロセスの出現頻度、同一プロセスの利用者数、さらに、通信を伴う場合はその頻度などの関係を示す。出現頻度の高いプロセスほど多くの利用者が存在していることから、出現頻度が高く使用者数が少ないプロセスや、出現頻度が低いが使用者数が少なく実行期間が長いプロセスは、不審なプロセスである可能性が高いことが示されている。そこで、さらに各プロセスの通信状態を考慮することで不審な通信を伴ったプロセスの抽出を行う。実際に、出現特徴が高いプロセスの多くは出現特徴の低い通信（他でも多く利用されている通信）を伴っていることを示す。最後に、日本年金機構のインシデントで明らかとなった情報から本方式の有用性をシミュレーションする。

2　関連研究

標的型攻撃の対策を大別すると、攻撃の侵入を防ぐ入口対策、万が一侵入を許してしまった場合でもそれ以上の被害拡大を防ぐ内部対策、重要な情報などが外部に漏洩することを防ぐ出口対策の 3 つがある。

入口対策では、そもそもユーザの手元までマルウェアが届かないようにするファイアウォールや IPS/IDS があり、万が一ユーザの手元までマルウェアが届いた場合でもその感染を防ぐため、アンチウイルスソフトウェアなどにより悪意のあるソフトウェアの侵入を防ぐ境界防御が行われる。しかし、特に標的型攻撃では、これらの外部からのマルウェア侵入を防ぐ対策を回避するなど、侵入を完全に防ぐことは難しい。そこで、内部対策や出口対策が非常に重要となっている。

2000 年初めより、一般社団法人 JPCERT コーディネーションセンター（JPCERT/CC）[4] や独立行政法人情報処理推進機構（IPA）[5] などによる標的型攻撃調査が行われている [6]–[8]。情報窃取を目的とした攻撃では、攻撃者のリスクを低減するためにステルス性を高め、確度の高い攻撃が行われることが示されている。ステルス性を高めることにより、攻撃者にとって攻撃が見つかるリスクを抑え、更に長期的な攻撃が可能になるとされている。

文献 [9] では、メールなどに添付されて侵入を試みる攻撃に対する対策を提案している。遠隔操作されたコンピュータから標的型攻撃メールを送信する場合に、事前に抽出した本人の行動特性と攻撃者によって操作されたときの特性を比較し、送信されたメールが攻撃か否かの判別を行う方式の提案を行っている。

内部対策では、標的型攻撃を受けて感染した犠牲ホストが行うネットワーク活動に着目した対策が、多く研究されている [10]–[12]。文献 [10] では、送信元 IP アドレスや送信先 IP アドレス、あて先ポート番号に従って抽出した時系列の特徴からデータの傾向の変化をとらえ、不審な通信を抽出する。従来方式である ChangeFinder[13] に比べ早い段階で不審な通信を発見することが可能となっている。文献 [11]–[12] では、攻撃基盤を拡大する過程で標的ノードのリモート制御など、攻撃者が内部的に共通して使わざるを得ない攻撃手法（チョークポイント）に着目している。

チョークポイントを利用することで、システムの振る舞いに矛盾・異常がないかを判定し、正規プログラムに成りすました攻撃や正規通信に紛れる攻撃、亜種・未知・難読化などのアンチウイルスソフトウェアを回避するマルウェアによる攻撃を検出することに成功している。これらは、感染ホストが行うネットワーク通信などの対外的な活動を観測しているが、文献 [14] では、端末内部で動作するプロセスの親子関係に着目し、新規プロセスの特定法を提案している。特に、文献 [15] で誤判定の原因となった実行パス情報を正規化することで精度向上を行った。本稿ではプロセスの実行頻度や実行期間に加え、ネットワーク状態に着目した不審プロセス判定方式の提案を行う。

3　プロセスの特徴

ここでは、各端末で動作するプロセスの出現頻度、利用者数、実行期間からプロセスの出現特徴を算出する方法を示す。

3.1　プロセスリストの取得

各端末で動作するプロセスの取得には文献 [15] と同様に行う。各端末からプロセスリストを取得するシステムの概要を説明する。各端末に情報取得用のエージェントツールをインストールし、定期的に管理サーバに対してプロセスリストを送信する。図 1 に情報収集を行うエージェントの概要を示す。

エージェントツールは、各端末内の監視対象となっているプロセスの動作を定期的に記録する。具体的には、プロセス ID、プロセス名、CPU 使用率、メモリ使用量、プロセス状態、親プロセス ID、プロセス実行パス（実行されたプロセスの保存場所）、プロセス生成時間、ネットワーク状況の情報を取得する。親プロセス ID とは、当該プロセスを実行したプロセス（親プロセス）のプロセス ID を表し、この情報によりプロセスツリー（プロセスの親子関係）を再現することが可能となる。

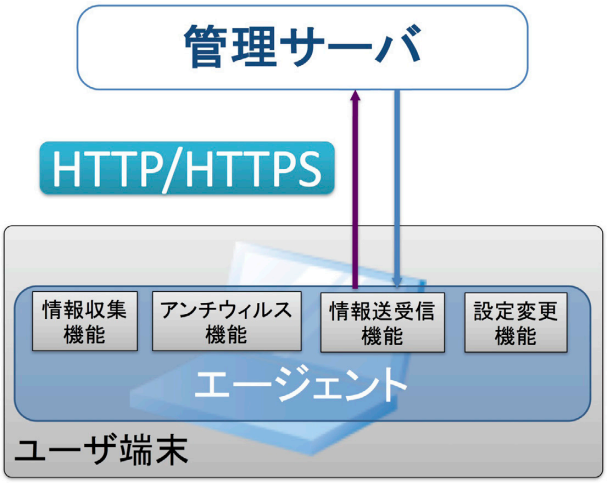

図 1　エージェント概要

本システムでは、エージェントにより取得した情報を HTTP により定期的に管理サーバへ送信する。企業などの管理されたネットワークでは、クライアントのネットワークセグメントからサーバのネットワークセグメントへの接続性は確保されているが、逆の場合は直接接続できないことが多い。そのため、エージェント側からの定期的なポーリング方式とすることでネットワーク環境による接続性の影響を最小限にしている。

各エージェントは、ポーリングにより送信する情報を図 2 に示す XML フォーマットにより管理サーバへと送信する。

図 2 では、2015 年 8 月 10 日の 18 :00 に行われたポーリングの結果を示している。各端末で動作しているプロセス情報は、<process_information_list> タグに記録される。この例では、プロセス ID が 2,016（id="2016 "）の CcmExec プロセス（name="CcmExec"）をプロセス ID が 568（parent_id="568 "）の親プロセスが実行し、実行時間は同日の 9 :00 :38（creation_time="2014 -08 -10 09 :00 :38.656 "）であることがわかる。また、実行されたプログラムの実態は“C:\Windows\System32 \CCM\CcmExec.exe”に存在し

```
<?xml version="1.0" encoding="utf-8”?>
<nirvana_request message_type="1" version="2" request_datetime="2015-08-10 18:00:00">
  <host_information id="a799ebba9388...cb825ae9d">
    <!-- ネットワークインターフェース情報 -->
    <network_interface macaddr="**:**:**:**:**:**">
      <ipaddress addr="***.***.***.***"/>
    </network_interface>
    <!-- 利用ユーザ名 -->
    <logon_user user="******"/>
    <!-- システム情報 -->
    <os_information os="Windows 7" service_pack="Service Pack 1" architecture="x86"/>
  </host_information>
  <!-- 既にマルウェアの検出が行われているか -->
  <detected_state detected="true"/>
  <!-- プロセス情報 -->
  <process_information_list>
    <process_information id="2016" name="CcmExec" cpu="0.0" mem="34148352” status="Execute"¥
        parent_id="568” path="C:¥Windows¥System32¥CCM¥CcmExec.exe" creation_time="2015-08-10 09:00:38.656">
      <tcp_state ip_src="***.***.***.***" port_src="49296" ip_dst="***.***.***.***” port_dst="80" state="ESTABLISHED"/>
      <udp_state ip_src="127.0.0.1" port_src="58798"/>
    </process_information>
          :
  </process_information_list>
          :
</nirvana_request>
```

図 2　プロセスの状態報告 XML 例

ていることを示している。また、このプロセスはネットワーク接続を行っており、49,296 番ポートから 80 番ポートに TCP プロトコルを用いた通信を行っていることがわかる。さらに、58,798 番ポートで UDP の接続を待ち受けていることが確認できる。エージェントから送信された情報は、管理サーバ上のデータベースに格納され管理される。

3.2　プロセスの頻度取得

プロセス頻度は一定期間中に新たに動作した、同じプロセス名、同じ実行パスの 2 つの要素が一致するプロセスを同一プロセスとしてカウントする。プロセスの比較に実行パスを含めることにより、同一名を持つプロセスが存在した場合でも実行パスが異なれば違うプロセスであることが識別できる。例えばプロセス名を偽装し、正常プロセスに成りすます不正なプロセスが実行された場合、プロセスの実行パスが異なるため正常プロセスとは異なることが識別できる。本システムでは、エージェントからの定期的なポーリングによりプロセス情報を収集しているため、異なるタイミングでプロセス ID、親プロセス ID、プロセス名、実行パス、プロセス生成時刻の 5 要素全てが同一なプロセスが報告される。これらのプロセスは、継続的に動作しているプロセスのため、プロセス数のカウントにはプロセス ID やプロセス生成時刻などが異なるプロセスを新たに実行されたプロセスとして利用する。

3.3　プロセスの出現特徴

文献 [16] [17] では、通常のプロセス（悪意の無いプロセス）は、

- プロセスの出現頻度が高く、利用者も高い
- プロセスの出現頻度が低くとも利用者数が高い
- プロセスの出現頻度が高く、利用日数も高い
- プロセスの出現頻度が低くとも利用日数が高い

の 4 つの条件を満たしていることが示されている。そこで、利用者 $\mathcal{U}_i$ が利用するプロセス P_j の出現特徴 $F_{i,j}$ を以下の通り定義する。

$$pf_{i,j} = \frac{p_{i,j}}{\sum_k p_{i,k}} \quad (1)$$

$$uf_j = \frac{|\{u:u \ni P_j\}|}{|\mathcal{U}|} \quad (2)$$

$$df_{i,j} = \frac{|\{d:d \ni P_j\}|}{|D_i|} \quad (3)$$

$$F_{i,j} = pf_{i,j} \times \left(\log_2 \frac{1}{uf_j}\right)^{df_{i,j}} \quad (4)$$

ここで、式 (1) はプロセス P_j の出現頻度を示す。は利用者 $\mathcal{U}_i$ が利用したプロセス P_j の出現回数を示し、$\sum_k p_{i,k}$ は、$\mathcal{U}_i$ が利用した全プロセス数を示す。式 (2) はプロセス P_j の利用者頻度（プロセス P_j を利用するユーザの割合）を示す。$|\{u:u \ni P_j\}|$ は、プロセス P_j を実行した利用者の数を示し、$|\mathcal{U}|$ はアクティブな全利用者数を示す。式 (3) はプロセス P_j の利用頻度（プロセス P_j を実行した日数の割合）を示す。$|\{d:d \ni P_j\}|$ はプロセス P_j を実行した日数を示し、$|D_i|$ は利用者 $\mathcal{U}_i$ がアクティブになった日数を示す。

例えば、利用者 $\mathcal{U}_i$ が過去 14 日のうち 14 日間（$|D_i|$ =14）アクティブになり「chrome」プロセスを 7 日（$|\{u:u \ni P_j\}| = 7$）で 7 回（$p_{i,j} = 7$）利用し、それ以外のプロセスを期間中に 3 回（全プロセス $\sum_k p_{i,k} = 10$）利用していた場合、また、「chrome」プロセスを 10 人の利用者（$|\mathcal{U}| = 10$）のうち 3 人（$|\{u:u \ni P_j\}| = 3$）が利用していた場合、出現頻度 $pf_{i,j}$ は 7 /10 = 0.7、利用者頻度 uf_j は 3 /10 = 0.3、利用頻度 $df_{i,j}$ は 7 /14 = 0.5 となる。したがって、利用者 $\mathcal{U}_i$ が実行したプロセス（$p_{i,j}$）の出現特徴 $F_{i,j}$ は、

$$F_{i,j} = 0.7 \times \left(\log_2 \frac{1}{0.3}\right)^{0.5} = 0.92$$

となる。

図 3 に、あるユーザが 2015 年 12 月 1 日の 9 時から 10 時の 1 時間の間に利用したプロセス（$\sum_k p_{i,k} = 146$ プロセス）の過去 14 日間（うち $|D_i| = 9$）における出現特徴を示す。

図 3 より、文献 [16] で不審なプロセスとした出現特徴が 0.05 以上のプロセスが 6 プロセスあることがわかる。文献 [16] では、出現特徴が高くなる原因としてプロセスの実態がユーザディレクトリ内に存在するなど、ユーザ環境により実行パスが異なることが原因とされている。実際に、出現特徴が 0.05 以上となった 6 プロセスのうち 4 プロセスは、プロセスの実態がユーザディレクトリ内（c:¥ %USERPROFILE %）に存在していた。

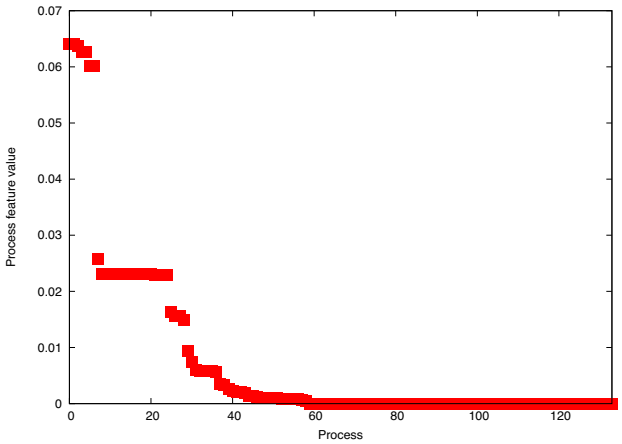

図 3　出現特徴分布

4 ネットワークの特徴

ここでは、各端末で動作するプロセスが行う通信の頻度、同一アクセスホスト数からネットワークの出現特徴を算出する方法を示す。

4.1 通信頻度

プロセスが通信を伴う場合、**3.1** で示したプロセスリストにより通信情報を取得する。通信情報には、ネットワーク接続が確立している場合（図 2 中の state="ESTABLISHED" のとき）通信先 IP アドレスやポート番号が、待ち受けの場合（図 2 中の state="LISTEN" または UDP プロトコルの通信のとき）待ち受けポートの情報を得ることができる。

利用者 $\mathcal{U}_i$ が利用するプロセスの通信 N_l の出現頻度を

$$nf_{i,l} = \frac{n_{i,l}}{\sum_k n_{i,k}}$$

と定義する。ここで、$n_{i,l}$ は $\mathcal{U}_i$ が実行したプロセスのうち N_l と同一の通信情報を持つプロセスの出現回数を示し、$\sum_k n_{i,k}$ は、$\mathcal{U}_i$ が使用した通信を伴う全プロセス数を示す。したがって、通信を伴うプロセスのうち同一通信情報を持つプロセスの割合を示す。例えば送信先 IP アドレスが 203.0.113.13 に対して HTTP（80 /TCP）接続しているプロセスが 2 プロセス（$n_{i,l} = 2$）存在し、ある期間中に実行したプロセスのうち 10 プロセスが何らかの通信を伴う場合（$\sum_k n_{i,k} = 10$）、出現頻度 $nf_{i,l}$ は、2 /10 = 0.2 となる。

図 4 に、あるユーザが 2015 年 12 月 1 日の 9 時から 10 時の 1 時間の間に利用したプロセス、の過去 14 日間におけるプロセスの出現特徴と通信頻度の分布を示す。

図 4 より、プロセスの出現特徴が大きなプロセスにおいてネットワーク通信が発生している物がいくつかあることがわかる。特に、プロセスの出現特徴が $F \leq 0.05$ の場合不審なプロセスである可能性が高いため注意する必要がある。

4.2 通信利用者頻度

ここでは、プロセスの出現特徴と通信利用者頻度の分布を示す。あるプロセスの通信 N_l と同一な通信を行うユーザ（ホスト）の割合 hf_l を

$$hf_l = \frac{|\{h:h \ni N_l\}|}{|\mathcal{U}|}$$

と定義する。ここで、$|\{h: h \ni N_l\}|$ は、通信情報 N_l と同じ通信を行う利用者の数を示し、$|\mathcal{U}|$ はアクティブな全利用者数を示す。したがって、例えば 10 人の利用者（$|\mathcal{U}| = 10$）のうち 3 人（$|\{h: h \ni N_l\}| = 3$）の利用者が送信先 IP アドレス 203.0.113.13 に対して HTTP（80 /TCP）接続していた場合、通信利用者頻度は $hf_l = \frac{3}{10} = 0.3$ となる。

図 5 に、あるユーザが 2015 年 12 月 1 日の 9 時から 10 時の 1 時間の間に利用したプロセスの、過去 14 日間におけるプロセスの出現特徴と通信利用者頻度の分布を示す。

図 5 より、プロセスの出現特徴が大きい場合でも、そのプロセスと同一の通信が多くの利用者により発生しているものがある。つまり、特定の利用者が利用するプロセスでも、他の利用者が利用するプロセスの通信と同様な通信を発生させていることがわかる。特に、多くの利用者が通信を行っている場合、その通信先（または待ち受けポートなどの通信情報）は一般的なアクセス先の可能性が高いため危険性が低いことが考えられる。一方で、プロセスの出現特徴が大きく、同一通信を行う利用者が少ない場合、例えば C&C 通信などの不正な通信の可能性が高い。そのため、このように多くの利用者が通信を行っていない通信を行うプロセ

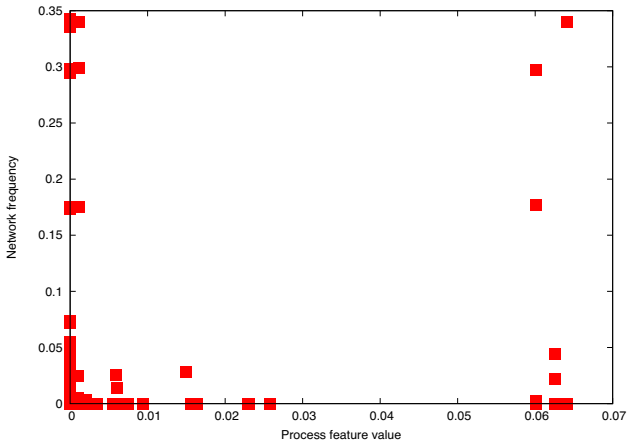

図 4　プロセスの出現特徴と通信頻度分布

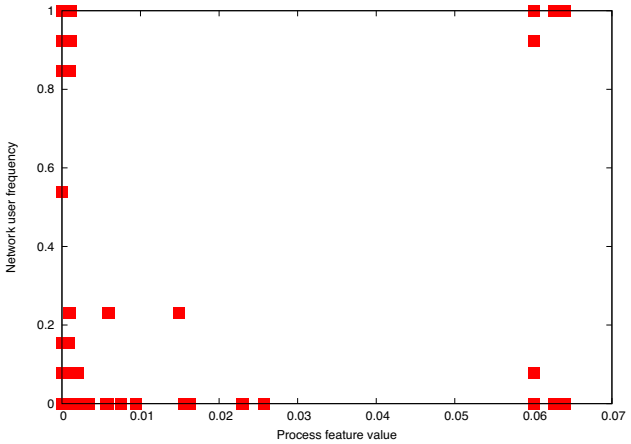

図 5　プロセスの出現頻度と通信利用頻度分布

スを抽出することが重要となる。

4.3　ネットワークの出現特徴

まず、**3.3** と同様にネットワークの出現特徴を定義する。悪意のあるプロセスが動作していた場合、ネットワークアクセスの発生は攻撃の拡大、更なるマルウェアのダウンロード、さらには、情報の外部への流出など重大なインシデントの発生を意味する。そのため、不審な通信をいち早く検出し対策を行う必要がある。そこで、利用者 $\mathcal{U}_i$ が行う通信 $N_{i,l}$ の出現特徴 $NF_{i,l}$ を

$$inf_{i,l} = \log_2 \frac{1}{nf_{i,l}} \tag{5}$$

$$ihf_l = \log_2 \frac{1}{hf_l} \tag{6}$$

$$NF_{i,l} = inf_{i,l} \times ihf_l \tag{7}$$

と定義する。式 (5) は同一の通信情報を持つ通信が少ないほど大きな値になる。式 (6) は式 (2) と同様に、同じ通信を行う利用者が少ないほど高い値になる。したがって、$NF_{i,l}$ は特定の利用者が初めて行う通信を発生した場合に大きな値をとるようになる。

5　プロセスの不審度

ここでは、プロセス出現特徴とネットワーク出現特徴からプロセスの不審度を定義する。また、日本年金機構で発生したインシデントの情報を基に、不審度の妥当性評価を行う。

5.1　プロセスの不審度

プロセスの不審度 $S_{i,j}$ をプロセスの出現特徴とネットワークの出現特徴（$F_{i,j}$，$NF_{i,l}$）より

$$S_{i,j} = \mathrm{MAX}\left(F_{i.j} \times NF_{i,l}\right)$$

と定義する。各プロセスは複数の通信を発生している場合があるため、通信先ごとに不審度を算出しその最大値をプロセスの不審度とする。また、通信を伴わないプロセスの場合、$S_{i,j} = F_{i.j}$ としてプロセスの出現特徴を不審度とする。

図 6 に、あるユーザが 2015 年 12 月 1 日の 9 時から 10 時の 1 時間の間に利用したプロセスの、過去 14 日間におけるプロセスの不審度の分布を示す。

図 6 より、ネットワーク状態を考慮することでプロセスの出現特徴が高いプロセスでも不審度が低くなることがわかる。つまり、プロセスの出現特徴が高い場合でも多くの利用者がアクセスする通信を行い、通信の出現特徴が高いプロセスでも多くの利用者が実行しているプロセスであることがわかる。

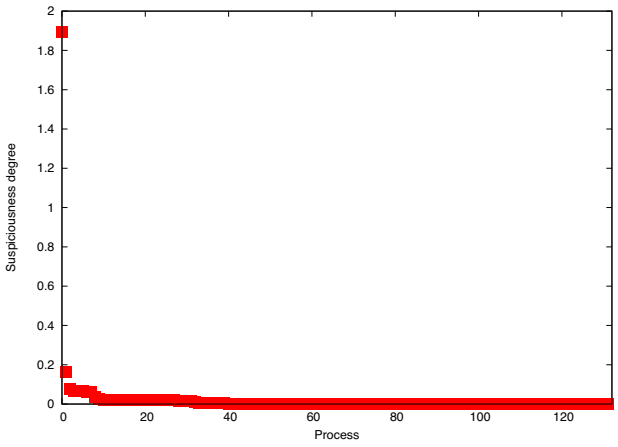

図 6　プロセス不審度

一方で、プロセスの不審度が約 1.9 となるプロセスが存在することがわかった。このプロセスは、13 人中 4 人のごく限られた利用者のみが利用するプロセスで、プロセスの出現特徴も約 0.06 と高めであった。また、通信を伴う 367 プロセスのうち同じ通信情報をもつプロセスは 1 プロセスのみで、利用者も 1 人であったため通信の出現特徴が 31.5 と非常に高い値となった。実際には、UDP ソケットを生成するプロセスであり、ソケット生成毎に割当ポート番号（送信元ポート番号）が変更されることが原因となっていた。この結果、通信の出現特徴が非常に高いが特に不正なプロセスではないことがわかった。

5.2　評価

ここでは、日本年金機構で発生した標的型攻撃の情報を基に不審プロセスの抽出評価を行う。日本年金機構に対して標的型攻撃は 4 回行われ、3 回の攻撃が成功していたことが報告されている [3]。2015 年 5 月 8 日の 1 回目の攻撃では 1 台の端末が、2 回目の 5 月 18 日の攻撃では 3 台の端末が、5 月 20 日に 4 回目の攻撃では最初に 1 台の端末が感染し、その後 2 日間で 20 台以上の端末に感染を拡大したとされている。

そこで、各攻撃発生時におけるプロセスの不審度をシミュレートする。日本年金機構では平成 27 年 4 月 1 日現在で、正規・準職員数が 12,000 人であることから、12,000 台の端末が利用されていると仮定する [18]。また、シミュレーションは 1) 攻撃 1：第 1 回目の攻撃成功時、2) 攻撃 2：第 2 回目の攻撃成功時、3) 攻撃 3：第 4 回目の攻撃成功後 20 台に感染が広がった時点の 3 通りを行う（感染後 2 日）。攻撃 1、攻撃 2 では、感染時の場合を想定するため利用頻度は 1 日となり、利用者頻度は攻撃 1 が 1 人、攻撃 2 が 3 人としている。攻撃 3 は、感染が広がるのに 2 日を要してい

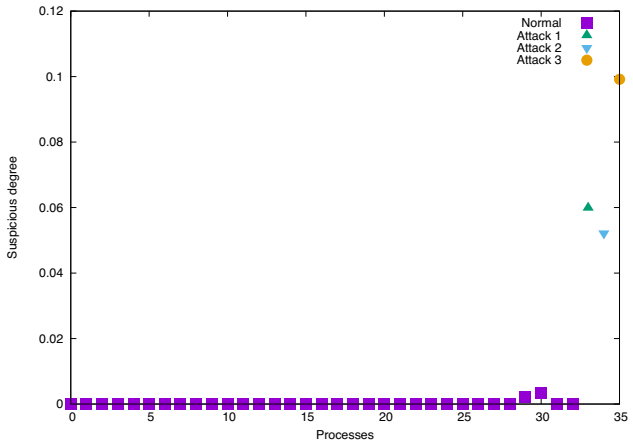

図 7　シミュレーションによるプロセス不審度

るため、利用頻度が 2 日で利用者頻度は 20 とした。図 7 に各攻撃におけるプロセスの不審度を示す。

ここで、**5.1** で示したユーザが特定の時刻において動作していたプロセスの不審度を比較として示す（図 7 中の Normal）。図 7 より、攻撃に利用されたプロセスの不審度は他のプロセスの不審度に比べ非常に大きくなることがわかった。特に、攻撃 1 や攻撃 2 のように攻撃が成功した時点（利用頻度が少ない場合）でも大きな不審度を得られ、不審なプロセスの検出に非常に有用であることがわかった。また、攻撃 3 は、ほかの 2 つの攻撃よりもプロセスの不審度が高くなった。これは、利用頻度と利用者頻度などが大きくなり、プロセスの不審度も大きくなったことが考えられる。さらに、プロセスの不審度は感染が広がるにつれて高い値へと変化していくことから、いち早く不審なプロセスとして検出可能なことが期待できる。

6　おわりに

本稿では、利用者個々の端末内部で動作しているプロセス情報（プロセスリスト）を定期的に取得し、利用者が普段利用しているプロセスか否かをプロセスの不審度を用いて判定する手法の提案を行った。プロセスの判定には、プロセス名と共にそのプロセスが実行された場所（実行パス）を同時に比較することで、正常プロセスに成りすます不正なプロセスの識別を可能とした。

頻繁に利用されるプロセスは多くの利用者が利用し、プロセスの動作日数（利用頻度）も複数日動作することから、利用者数が少なく利用日数が長いプロセスを抽出可能なプロセスの出現特徴 $F_{i,j}$ を定義した。また、プロセスの出現特徴と通信頻度を比較した結果、プロセスの出現特徴が大きく通信頻度も大きなプロセスが存在することがわかった。さらに、同一の通信を伴うプロセスの利用割合を比較した結果、一部の通信は限られた利用者のみ発生していることがわかった。

以上の結果より、通信の頻度が少なく、利用者が少ないプロセスを抽出可能な通信の出現特徴 $NF_{i,l}$ を定義し、プロセスの不審度を各出現特徴から決定した。実際に 1 時間に動作した多くのプロセスが不審では無いプロセスであることを示した。一方で、1 つの不審プロセスを抽出し、詳細解析の結果不正なプロセスでは無いことを示した。また、日本年金機構における個人情報流出事案の報告書をもとにシミュレーションを行った結果、通常利用するプロセスと不審なプロセスの不審度は大きく異なることが示され、提案方式が十分有用であることを示した。

今後、不審なプロセスが発見された場合、そのプロセスの詳細情報（API 履歴など）を取得するなど、更なる監視下に置くことで、不正な動作（不審なネットワーク接続など）をいち早くとらえ、迅速な対応を行う必要がある。不審プロセスの判定にも API 履歴や通信情報などの詳細情報を取得することで、より精度高い不正プロセスの特定を行うことが可能となり、より早い確実な対策を行うことが期待できる。

謝辞

本研究を遂行するにあたり、支援端末へのエージェントツールのインストールや管理に、情報システムグループの皆様に多大な協力を頂いたことを深く感謝する。

【参考文献】

1　宇宙航空研究開発機構プレスリリース，“JAXA におけるコンピュータウィルス感染に関する調査結果について，” http://www.jaxa.jp/press/2013/02/20130219_security_j.html（2016 年 4 月現在）
2　独立行政法人 情報処理推進機構 (IPA)，“[高度標的型攻撃] 対策に向けたシステム設計ガイド，” https://www.ipa.go.jp/security/vuln/newattack.html (2016 年 4 月現在)
3　サイバーセキュリティ戦略本部，“日本年金機構における個人情報流出事案に関する 原因究明調査結果，” (PDF)，第 4 回会合（平成 27 年 8 月 20日)，http://www.nisc.go.jp/active/kihon/pdf/incident_report.pdf (2016 年 4 月現在)
4　一般社団法人 JPCERT コーディネーションセンター (JPCERT/CC)，http://www.jpcert.or.jp/（2016 年 4 月現在)
5　独立行政法人 情報処理推進機構 (IPA)，http://www.ipa.go.jp/（2016 年 4 月現在)
6　JPCERT/CC，“標的型攻撃について，” (PDF)，https://www.jpcert.or.jp/research/2007/targeted_attack.pdf（2016 年 4 月現在)
7　JPCERT/CC，“標的型攻撃対策手法に関する調査報告書，” (PDF)，http://www.jpcert.or.jp/research/2008/inoculation_200808.pdf（2016 年 4 月現在)
8　IPA，“標的型サイバー攻撃の事例分析と対策レポート，” https://www.ipa.go.jp/security/fy23/reports/measures/index.html (2016 年 4 月現在)
9　片山 佳則，寺田 剛陽，津田 宏，“利用者の行動特性を用いたサイバー攻撃における成りすまし対策技術，” 人工知能学会全国大会，4G1-4, 2014.
10　北澤 繁樹，祢宜 知考，河内 清人，榊原 裕之，藤井 誠司，“標的型攻撃検知システムの評価，” マルウェア対策研究人材育成ワークショップ (MWS 2009)，A6-3, 2009.

11 海野 由紀 , 森永 正信 , 山田 正弘 , 鳥居 悟 , “標的型サイバー攻撃におけるシステム内部の諜報活動検知の提案, ” コンピュータセキュリティシンポジウム (CSS 2012) , pp.360–367, 2012.
12 Satoru Torii, Masanobu Morinaga, Takashi Yoshioka, Takeaki Terada, and Yuki Unno, “Multi-layered Defense against Advanced Persistent Threats (APT) ,” FUJITSU Sci. Tech. J., vol.50, vo.1, pp.52–59, 2014.
13 Jun-ichi Takeuchi and Kenji Yamanishi, “A Unifying Framework for Detecting Outliers and Change Points from Time Series, ” IEEE Transactions on Knowledge and Data Engineering, vol.18, Issue 4, pp.482–492, 206.
14 中里　純二，津田　侑，高木　彌一郎，衛藤　将史，井上　大介，中尾　康二，“ホスト型 IDS を用いた不審プロセスの特定, ” 暗号と情報セキュリティシンポジウム (SCIS 2015)，2A1-5，2015.
15 中里　純二，津田　侑，高木　彌一郎，衛藤　将史，井上　大介，中尾　康二，“ホスト型 IDS を用いた標的型攻撃対策, ” コンピュータセキュリティシンポジウム (CSS 2014)，2B2-3，2014.
16 中里　純二，津田　侑，衛藤　将史，井上　大介，中尾　康二，“プロセスの出現頻度を用いた不審プロセス特定, ” 信学技報，vol.115，no.334，pp.61–66，2015.
17 中里　純二，津田　侑，衛藤　将史，井上　大介，中尾　康二，“プロセスの出現頻度や通信状態に着目した不審プロセス判定, ” 信学技報，vol.115，no.488，pp.77–82，2015.
18 日本年金機構，日本年金機構について，http://www.nenkin.go.jp/info/index.html (2016 年 4 月現在)

中里純二　（なかざと　じゅんじ）
サイバーセキュリティ研究所
サイバーセキュリティ研究室
研究員
博士 (工学)
サイバーセキュリティ

津田　侑　（つだ　ゆう）
サイバーセキュリティ研究所
サイバーセキュリティ研究室
博士 (情報学)
サイバーセキュリティ、標的型攻撃対策

高木彌一郎　（たかぎ　やいちろう）
サイバーセキュリティ研究所
サイバーセキュリティ研究室
研究技術員
ネットワークセキュリティ