HTTPベースの通信挙動に基づくRAT検知システムの試作

三村 守[1,2]　大坪 雄平[1,3]　田中 英彦[1]

概要：近年の標的型攻撃における RAT の通信は，通信先や特徴的な通信パターンが判明していない限り検知することは困難である．そこで，われわれは http ベースの受信量や時刻等のトラフィックパターンの挙動から，パターンマッチングを用いずに RAT を検知する手法を提案した．この手法では，Support Vector Machine および Random Forests に基づく分類器を用い，プロキシサーバのログからでも未知の RAT を検知することが可能である．本稿では，その検知手法とブラックリストと併用した運用構想を提案し，http ベースの通信挙動による RAT 検知システムを試作する．さらに，実運用のネットワークへの適用実験により，リアルタイムで運用可能であることを示す．

キーワード：標的型攻撃，RAT，通信挙動，Support Vector Machine，Random Forests，ブラックリスト

Implementation of a RAT Detection System using HTTP-Based Communication Behavior

Mamoru Mimura[1,2]　Yuhei Otsubo[1,3]　Hidehiko Tanaka[1]

Abstract: In recent APT campaigns it is hard to detect RAT activity, unless we know the C&C server address or the distinctive communication pattern. As a countermeasure, we proposed how to detect RAT activity without pattern matching, which uses the http-based behavior such as the sizes of the object returned to the client or the intervals of the logged time. Our method uses statistical classifiers based on support vector machine and random forests. Our method can detect unknown RAT activity in proxy server logs. This paper proposes the operation concept using the blacklist of C&C servers, and implements the RAT detection system using http-based communication behavior. Moreover, the experimental result on a real network shows the system can run in real time.

Keywords: APT, RAT, Communication Behavior, Support Vector Machine, Random Forests, Blacklist

1. はじめに

サイバー攻撃の脅威は年々深刻化しており，社会の関心も高まってきている．サイバー攻撃には様々な種類があるが，とりわけ特定の組織を執拗に狙う標的型攻撃による情報漏洩の被害は深刻である．多くの標的型攻撃では，RAT(Remote Access Trojan または Remote Administration Tool) と呼ばれる被害者のコンピュータを遠隔で操作するためのプログラムが用いられる．サイバー攻撃への対策として，大規模な企業や公共機関では，http 等の必要最小限のサービスの利用にアクセスを制限している場合も少なくない．このような大規模な組織を対象とした標的型攻撃では，通常の http 通信を用いて遠隔操作を実施する RAT が使用される．中には IDS (Intrusion Detection System) による検知を回避するために，特徴的な文字列を含まないで通信する RAT もある．このような RAT は，通信先や特徴的な通信パターンが判明していない限り検知することは困難である．また，既存の多くの振舞いによる RAT の検知手法では，パケット単位ですべてのトラフィックを傍受する必要があるため，プロキシサーバのログ解析には適用

[1] 情報セキュリティ大学院大学
Institute of Information Security
[2] 海上自衛隊幹部学校
JMSDF Command and Staff College
[3] 警察庁
National Police Agency

できないという課題があった．そこで，われわれはプロキシサーバのログから入手できる http ベースの受信量や時刻等のトラフィックパターンの挙動から，パターンマッチングを用いずに RAT を検知する手法を提案した [1]．この手法では，Support Vector Machine および Random Forests に基づく分類器を用い，プロキシサーバのログからでも未知の RAT を検知することが可能である．本稿では，その検知手法とブラックリストと併用した運用構想を提案し，http ベースの通信挙動による RAT 検知システムを試作する．さらに，実運用のネットワークへの適用実験により，リアルタイムで運用可能であることを示す．

以下，第 2 節では関連研究について示し，提案手法と比較してその違いを明確にする．第 3 節では，提案したプロキシサーバのログからパターンマッチングを用いずに RAT を検知する手法について説明する．第 4 節では提案手法を用いた運用構想を示し，http ベースの通信挙動による RAT 検知システムを試作する．第 5 節では試作した提案システムを実運用されているネットワークに適用する．第 6 節では提案システムの運用構想と実用性について考察し，最後にまとめと今後の課題を示す．

2. 関連研究

われわれの研究の目的は，大規模な組織をねらった標的型攻撃において，C&C サーバのアドレス，固定の文字列，正規表現等によるパターンマッチングで検知できない http ベースの RAT の通信を検知することである．不正な通信を検知することを目的とした手法は，これまでにも多く提案されている．しかしながら，これまでのほとんどの手法では，大規模な組織をねらった標的型攻撃における RAT を検知できるかどうかは明確ではない．既存の関連研究は，シグネチャベースによる C&C 検知，振舞いによる C&C 検知およびログ分析による C&C 検知に分類できる．以下，各分類毎に関連研究との違いについて示す．

2.1 シグネチャによる C&C 検知

多くの研究者が，C&C サーバのアドレスと不正な通信を検知するために，固定の文字列や正規表現を用いたシグネチャに基づく手法に着目してきた．文献 [2] では，HTTP ベースのマルウェアを分類するために，リクエストの数，GET の数, POST の数，URL の平均の長さ，パラメータの平均数，POST で送信したデータの平均サイズ，平均の応答のサイズを使用している．この手法では，さらにクエリの内容も分析してクラスターに分類し，マルウェアのサンプルによるクラスターと類似性を比較し，シグネチャを自動生成している．この手法では，プロキシサーバのログからでも取得できる項目を用いており，その平均に着目して挙動を抽出している．しかしながら，これらの項目は分類のために用いられており，検知のために用いられているわけではない．文献 [3] では，既知の不正な通信の制御プロトコルのテンンプレートを作成し，未知の不正な通信を検知するシグネチャに基づく検知システムを提案している．このシステムでは，マルウェアのサンプルを必要としている．他のいくつかのシグネチャに基づく検知システムにおいても，シグネチャ作成のために動的解析を実施する必要がある [4], [5]．これらのシグネチャに基づく手法では，不正な通信を検知するためにすべてのパケットにアクセスする必要がある．また，これらの手法ではほとんどの場合，検知のためのシグネチャとして固定の文字列や正規表現を用いる．さらに，マルウェアのサンプルが必要な場合もある．われわれの手法は，不正な通信を検知するためにプロキシサーバのログから取得できる項目のみを用いており，すべてのパケットの監視を必要とせず，固定の文字列や正規表現も用いない．われわれの手法では，訓練データとして分析済みのプロキシサーバのログが必要であるが，マルウェアのサンプルは必ずしも必要としない．

2.2 振舞いによる C&C 検知

シグネチャを用いない振舞いによる検知手法としては，DNS の特徴的な通信のパターンを観測して不正なドメインを検知する手法が提案されている [6], [7], [8], [9]．たとえば，文献 [8] ではマルウェアがドメイン名を自動的に生成するアルゴリズムに着目し，感染した端末を検知する手法が提案されている．文献 [9] では，簡易で負荷の軽い不正なドメインを検知する手法が提案されている．この手法では，受動的に DNS の通信を監視し，大規模なプロバイダのネットワークにおいて，マルウェアの感染時の DNS クエリの振舞いを追跡する．これらの手法では，利用者の端末とそれらの DNS リゾルバの間の DNS の通信を監視する必要がある．われわれの手法では，DNS の通信を監視する必要はなく，標準のプロキシサーバのログのみを用いる．

ボットに感染した端末に焦点をあてた検知手法も多く提案されている [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20]．たとえば，BotHunter[10]，BotSniffer[11] や Botminer[12] 等は異常検知に基づくボット検知システムである．これらのシステムでは，感染した端末は C&C サーバを共有する同じボットネットに属するという特徴を利用し，ネットワークのホスト間における類似の振舞いを探す．Jackstraws[15] や BotFinder[16] では，マルウェアのサンプルから類似したパターンの接続の検知を試みる．これらの手法では，マルウェアのサンプルが必要である．DISCLOSURE[17] では，Net Flow から抽出した特徴を用いて C&C 通信を検知する．文献 [18] では，パケットサイズ，パケット数，到着間隔等の特徴量を用い，Ada Boost で通常の通信と不正な通信を区別することで，マルウェアへの感染を検知する手法を提案している．文献**?**では，パケット数，データサイズ，セッション時間，アクセス回

数およびアクセス時間の標準偏差を特徴ベクトルとして，Support Vector Machine によりC＆Cトラフィックを抽出する手法を提案している．多くの既存の手法では，機械学習を用いるためにパケットから収集した統計的な情報を用いる．このように，ほとんどの既存のボットに感染したホストを検知する手法では，すべてのパケット単位のネットワーク通信を監視できていることが前提となっている．そのため，プロキシサーバのログを解析する用途には用いることはできない．われわれの手法では，プロキシサーバの標準形式のログから取得できる http ベースの情報のみを用いるため，プロキシサーバのログを解析する用途にも適用することが可能である．また，いくつかの既存の手法ではマルウェアのサンプルが必要であるが，われわれの手法ではマルウエアのサンプルは必ずしも必要としない．

2.3 ログ分析による C&C 検知

文献 [21] では，プロキシサーバのログからクライアントのアドレス，訪問先のアドレスおよびリクエストの数を用い，クライアントと共通するサーバに着目してグループに分類し，疑わしいドメインの検出を支援する手法を提案している．この手法では，疑わしいドメインの検出をブラックリスト等の他の手法に依存している．文献 [22] では，DNS のログ，プロキシサーバのログ等を使用し，内部ホストの訪問履歴とその User Agent から，組織全体の希少な訪問サイト，User Agent の傾向，ドメインの類似性等を分析し，通常状態と比較することで異常なドメインを検出する手法を提案している．この手法では，プロキシサーバ以外にも DNS のログを必要としている．また，疑わしいドメインを検出するために，ドメインの登録情報，ブラックリスト等の外部からの情報を必要としている．これらの手法は，プロキシサーバから取得できる項目を用いており，教師なし学習モデルを分類のために用いている．文献 [23] では，ポリシー違反や不正プログラムの配布等の企業ネットワークの設定における異常を認識する手法を提案している．この手法では，企業ネットワークにおける疑わしい活動を検知するために標準のログを用いている．このように，既存の多くのログを分析する手法では，クラスタリングのような教師なし学習モデルに着目している．われわれの手法では，教師あり学習モデルに着目しており，プロキシサーバの標準ログのみを用いる．

3. http ベースの通信挙動に基づく RAT 検知手法

3.1 前提条件

提案した http ベースの通信挙動に基づく RAT 検知手法を実現するための前提条件は，プロキシサーバのログに以下の項目が記録されていることである．

- 時刻
- リクエストの内容（メソッド，URL および User Agent を含む．）
- HTTP ステータスコード
- クライアントが受信したサイズ

これらの項目は，標準ログフォーマットに含まれており，多くのプロキシサーバで取得可能であると考えられる．この検知手法では，複数行のログから特徴となる挙動を抽出する．まず，HTTP ステータスコードが成功の行を対象として，URL に含まれるホスト毎にあらかじめ指定した行数のログを抽出する．次に，抽出した指定行数のログに含まれる時刻，リクエストの内容およびクライアントが受信したサイズから特徴ベクトルを作成する．

3.2 特徴ベクトル

この検知手法において，指定行数のログから作成する特徴ベクトルを以下に示す．

① 最頻出の受信サイズ
② 最頻出の受信サイズの数
③ 最頻出のリクエストの間隔
④ 最頻出のリクエストの間隔の数
⑤ 最頻出の path の長さ
⑥ 最頻出の path の長さの数
⑦ POST メソッドの数
⑧ User Agent の長さ

特徴ベクトルは，固有の文字列を用いずに，項目の頻度に着目して作成した．①から④は，受信サイズおよび間隔（直前のログの時刻との差分）のヒストグラムを**図 1** に示すように作成し，最も頻度が高い受信サイズおよび間隔とその数とした．⑤および⑥は，**図 2** に示すように RAT は同じ path に連続してアクセスするという特徴を数値化したものである．⑦は，一部の RAT は特定のメソッドを多用することに着目している．⑧は，User Agent の違いを考慮させるために選定した．

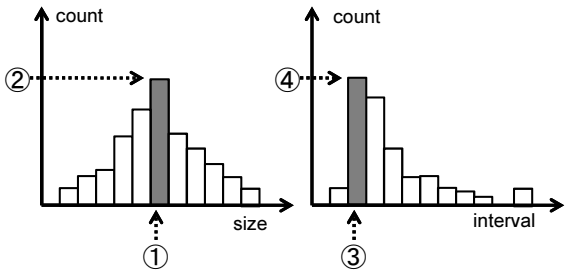

図 1 受信サイズと間隔のヒストグラム

3.3 学習と予測

この検知手法では，SVM (Support Vector Machine) と RF (Random Forests) を，教師あり学習モデルとして未

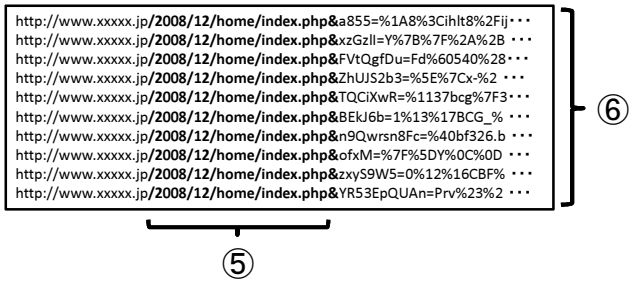

図 2 RAT のアクセスログの例

知のデータの予測に用いる．そのため，訓練データとして検知対象とする RAT の痕跡を含む既知のログが必要であり，かつそのログにおいて RAT の通信と通常の通信の区別がついている必要がある．この検知手法の動作は，学習フェーズと予測フェーズに分類される．

学習フェーズ

検知対象とする RAT の痕跡を含むログを読み込み，ホスト毎に指定行数のログから特徴ベクトルを作成する．次に，特徴ベクトルが RAT による通信であれば RAT の種類，それ以外であれば通常の通信のラベルを付与し，SVM または RF に学習させる．

予測フェーズ

対象とする未知のログを読み込み，学習フェーズと同様に，ホスト毎に指定行数のログから特徴ベクトルを作成する．次に，その特徴ベクトルを SVM または RF に予測させ，RAT の種類か通常の通信のラベルを出力させる．

この検知手法では以上の動作により，対象とするログから HTTP ベースの RAT の種類を検知する．

4. 検知システムの試作

4.1 運用構想

通信挙動に基づく RAT 検知手法は，ログを詳細に分析する用途とリアルタイムで通信を監視する用途に適用することが可能である．文献 [1] では，ログを詳細に分析する用途において，提案手法が有効かつ実用的であることを示した．本稿では，この検知手法をリアルタイムで通信を監視する用途に適用する手法を検討し，HTTP ベースの通信挙動に基づく RAT 検知システムを試作する．RAT 検知システムを試作するにあたり，まずその運用構想について説明する．

本稿で想定する運用構想を**図 3** に示す．図中の研究者は，この検知システムの仕組みに精通し，通信データを分析して新たな訓練データを作成したり，検知システムのメンテナンスを実施できる者を想定している．図中のオペレータは，いわゆる SOC (Security Operation Center) 等のオペレータを想定しており，インシデントの簡易な分析を行うアナリストもこれに含まれる．まず，研究者があらかじめ分析した RAT の痕跡を含むログを RAT 検知システムに入力し (①)，検知手法の学習フェーズを適用する．RAT 検知システムは監視対象とするネットワークからの通信内容を読み込み (②)，ログに相当するデータに加工した後，検知手法の予測フェーズを適用する．ネットワークを監視するオペレータは，検知システムが RAT の種類とその URL を検知した場合 (③) には，その内容を精査して正誤を判断する．オペレータはまた，外部の情報源から RAT の通信先 URL の情報を得た場合 (④) には，検知システムで精査した URL と合わせてブラックリスト検知システムに登録する (⑤)．ブラックリスト検知システムは監視対象とするネットワークからのすべての通信内容を読み込み (⑥)，登録されたブラックリストに合致した前後一定時間の通信内容を保存する．オペレータはその保存された通信内容を抽出し (⑦)，正誤の判断に活用するとともに，研究者にその分析結果を送付する (⑧)．分析の結果，誤検知と判断された URL については，ホワイトリストとして RAT 検知システムに登録して除外する (①)．

本稿では，以上に示した運用構想を前提に，通信挙動による RAT 検知システムを試作する．

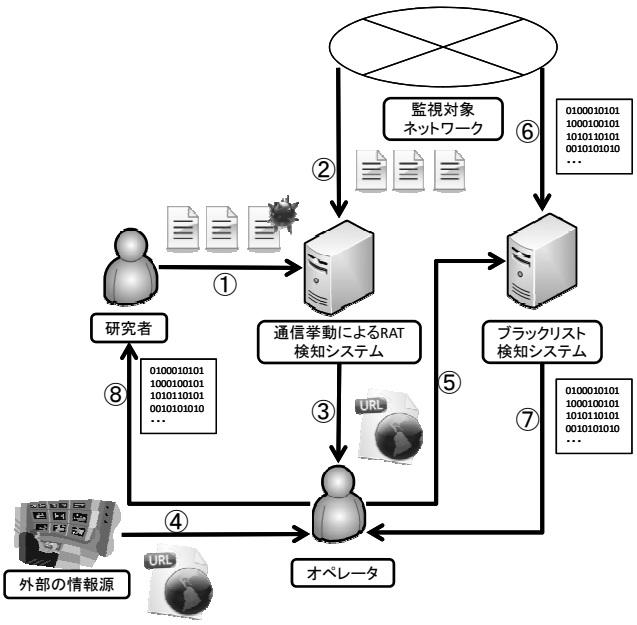

図 3 運用構想

4.2 機能と構成

次に，通信挙動による RAT 検知システムに必要な機能と構成を検討する．RAT 検知システムは，あらかじめ分析した RAT の痕跡を含むログを読み込み，監視対象とするネットワークからの通信内容を分析して RAT の種類とその URL を出力する．これらの機能を，学習，通信キャプチャ，ログ変換，予測の 4 つのモジュールに分割して実装

する．各モジュールの構成を**図 4** に示す．学習モジュールは，RAT の痕跡を含むログと，その分析結果である URL とラベルのセットを訓練データとして読み込み (①)，特徴ベクトルを作成して予測モジュールに入力する (②)．通信キャプチャモジュールは，監視対象とするネットワークからの通信内容をテストデータとしてキャプチャし (③)，ログ変換モジュールに入力する (④)．ログ変換モジュールは，キャプチャされた通信内容を読み込み (④)，ログに相当するデータに加工して予測モジュールに入力する (⑤)．予測モジュールは，あらかじめ学習モジュールから入力される特徴ベクトル (②) を SVM または RF に学習させた後，ログ変換モジュールから入力されるログ (⑤) から特徴ベクトルを作成し，これを SVM または RF で予測させる．予測の結果，正常な通信以外でホワイトリストに合致しない URL を検知した場合には，RAT の種類とその URL を出力する (⑥)．

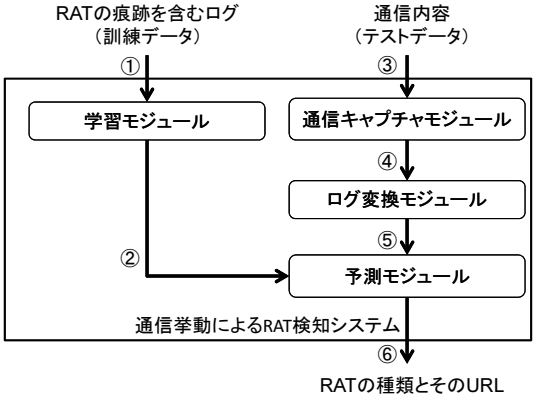

図 4 各モジュールの構成

4.3 実装

次に，各モジュールの実装手法について説明する．

学習モジュールおよび予測モジュールは，Python-2.7 と機械学習のライブラリを活用して実装した．SVM については libsvm-3.2[24] の C-SVC（ソフトマージン識別器）を用いた．カーネル関数については，分類するデータに関する事前知識がないことから，汎用的な用途で用いられる RBF（ラジアル基底関数）カーネルを選択した．RF については scikit-learn-0.16.1[25] の RandomForestClassifier を用いた．SVM，RF ともに，その他のパラメータについては，デフォルトの値とした．学習モジュールは，予測モジュールを実行する前にあらかじめ実行する．また，分析した RAT の痕跡を含むログにアップデートがあった場合にも再度実行する．予測モジュールについては，監視対象とするログが入力される毎に実行する．

通信キャプチャモジュールについては，httpry[26] をそのまま用いることとした．httpry は，Web のトラフィックを解析するために設計されたオープンソースのプログラムであり，特徴ベクトルの作成に必要な内容をすべてテキストで出力することが可能である．通信キャプチャモジュールは，指定した時間毎に通信内容をキャプチャし，特徴ベクトルの作成に必要な内容を出力する．

ログ変換モジュールは，Python-2.7 を用いて実装した．ログ変換モジュールは，通信キャプチャモジュールがテキストデータを出力する毎に実行する．

5. 実運用ネットワークへの適用

5.1 実験環境

実装した各モジュールを，実運用のネットワークの全トラフィックを傍受できるサーバに設置した．このサーバの OS は Ubuntu14 であり，仮想マシン上で動作している．実験に用いた仮想マシンのスペックは，**表 1** に示すとおりである．監視対象のネットワークの帯域は 1Gbps であり，そのほとんどの用途は Web およびメールである．このネットワークを利用するユーザは 25 万人程度であり，Web のトラフィックの容量は数分程度で数十 G 単位となる．なお，このネットワークは 24 時間体制で厳重に監視されており，IDS やサンドボックス等の最新の標的型攻撃対策機材を導入している．

表 1 実験環境

CPU	Core i5-3450 3.1GHz
Memory	DDR3 SDRAM 8GB
HDD	Serial ATA 600
OS	Ubuntu14
NIC	1Gbps

5.2 実験内容

今回の実験では，まずあらかじめ入手した RAT の痕跡を含むログから作成した特徴ベクトルを訓練データとして学習させた．その後，それを試作した RAT 検知システムに読み込ませ，実運用のネットワークトラフィックをテストデータとして約 1 か月間動作させた．このログは，2015 年に大規模なサイバー攻撃を受けたある組織の 1 日分のプロキシサーバのログであり，その容量は約 250MB である．このログには，通常の http 通信を用いる 2 種類の RAT による遠隔操作の痕跡が記録されている．通信キャプチャの時間は 1 時間毎とし，分類器については SVM と RF の双方を用いた．特徴ベクトルを作成するためのログの行数は 30 とした．

5.3 実験結果

実験結果の概要を**表 2** に示す．表中の RT は 1 時間分のログを予測モジュールで処理するのに要した時間を示す．FV は 1 時間分のログから生成された特徴ベクトルの数で

あり，DC はその中で RAT と判定された数を示す．表には各項目ごとに SVM と RF の平均値，最大値および最小値を示している．処理時間については，SVM と RF の双方において 1 時間分のログを平均 1 分強の時間で処理しており，最大でも 10 分に満たない時間で処理できていることが確認できる．特徴ベクトルの数については，時間帯によって大きな差があることが確認できる．RAT と判定された数については，SVM は 0 であり，RF では 1 時間で平均 10 弱，最大で 142 件となった．今回の実験期間においては，24 時間の監視において RAT による外部への不正通信は確認できなかったことから，RF を用いた場合の検知は誤検知の数ということになり，SVM では誤検知は発生していない．RF でも False Positive の割合は平均で 1%未満にとどまった．誤検知が発生した RAT は上記 2 種類のうちの 1 種類のみであり，この RAT は，同一の長さの通信を短時間に頻繁に繰り返すという特徴がある．

表 2 実験結果の概要

	SVM			RF		
	Ave	Max	Min	Ave	Max	Min
RT	1m7s	7m38s	5s	1m12s	8m9s	2s
FV	24582	113161	698	24582	113161	698
DC	0	0	0	9.7	142	0

図 5 は特徴ベクトルの数を時系列でグラフにしたものである．縦軸は特徴ベクトルの数，横軸は時間を示しており，左端から右端で 1 か月に相当する．60,000 以上のピークが発生している箇所は，平日の日中の通信量が多い時間である．10,000 に満たない小さなピークは，休日の日中を示している．このような実際のネットワークでは，時間帯や曜日によって通信量が大きく異なることが確認できる．

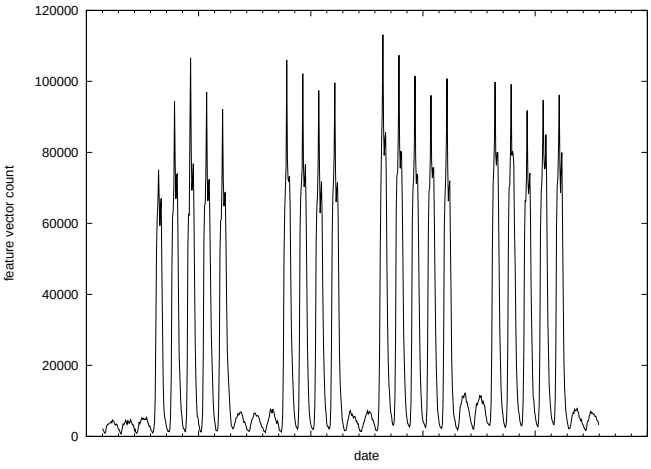

図 5 特徴ベクトル数

次に，検知数を時系列でグラフにしたものを**図 6** に示す．縦軸は検知数，横軸は時間を示しており，左端から右端で 1 か月に相当する．先に述べたとおり，今回の場合は図中の検知数は誤検知の数となる．SVM については誤検知は発生しなかった．図 5 と図 6 を比較すると，RF の誤検知は必ずしも特徴ベクトルが多い場合に発生しているわけではないことが確認できる．

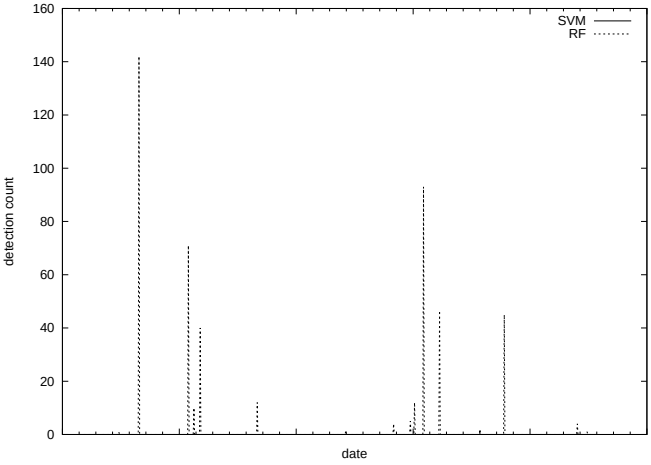

図 6 検知数

6. 考察

6.1 正確性

今回の実験では，大規模な実運用のネットワークにおいて試作した検知システムを 1 か月の間動作させることができた．動作期間中の 24 時間の監視において，RAT による外部への不正通信は確認できなかったことから，不正な通信の検知に関しては新たな知見を得ることはできなかった．この検知システムにおいて，未知の RAT による類似の不正通信を検知できるかどうかについては今後も検証の余地がある．誤検知に関しては，SVM ではまったく発生せず，RF でも最大で 1%未満にとどまった．この結果から，実運用のネットワークにおいても SVM の場合は誤検知がほとんど発生せず，RF の場合はやや誤検知が発生するということを確認することができた．この誤検知数の傾向は，別の組織における RAT の痕跡を含むログの分析結果と同様であった [1]．誤検知の数は，RF の場合においても 1 時間で最大 142 件であった．この数量であれは，RF の場合でも 1 名のオペレータで誤検知か否かの判定を時間内に実施することができるものと考える．より大規模なネットワークや複数のネットワークの監視に用いる場合には，SVM を用いることでオペレータの負荷を軽減することが可能である．また，誤検知が発生するサイトをあらかじめホワイトリストとして登録し，除外することも可能である．したがって，今回試作した検知システムは，実運用のネットワークにおいても監視の業務に実用的に適用可能であるものと考える．

6.2 リアルタイム性

今回の実験では，1時間分のログを平均1分強の時間で処理しており，最大でも10分に満たない時間で処理できていることが確認できた．したがって，試作した検知システムは，大規模な実運用のネットワークにおいてもほぼリアルタイムに適用することが可能であると考える．処理時間にはまだ余裕があるため，さらに広い帯域や利用者の多いネットワークにも適用することが可能であると考えられる．

6.3 スケーラビリティ

試作した検知システムでは，既存の多くの振舞いによる検知手法のように，複数の端末の通信内容を傍受する必要はなく，単一の端末の通信内容から特徴ベクトルを作成する．そのため，ISP レベルのネットワークや複数のネットワークを監視する場合には，そのネットワーク毎に容易に処理を分散することが可能である．したがって，監視対象の規模や通信量に応じて処理を分散すれば，スケーラビリティにおける制約はなくなるものと考える．

6.4 制限

試作した検知システムでは，アウトバウンドの通信を対象としており，攻撃者の RAT による遠隔操作を検知することに焦点をあてている．したがって，この検知システムは出口対策の一つであり，最終的に攻撃者が攻撃目標を達成する段階における対策に位置づけることができる．マルウェアを添付したメールやドライブバイダウンロード攻撃等に対しては，サンドボックス等の別の入口対策が必要である．この検知システムでは複数行のログから特徴ベクトルを作成するため，1時間強のタイムラグが発生する．大規模な組織への標的型攻撃においては，数十日以上が経過してから攻撃が発覚することも珍しくない．したがって，この1時間強のタイムラグは実用に足る範囲内であると考える．今回の実験では，ログを集約する時間を1時間に設定したが，この時間は短縮することも可能である．しかしながら，標的型攻撃に用いられる RAT には，発見を困難にするために通信の頻度が極めて少ないものも存在するため注意が必要である．提案手法では，訓練データとして RAT の痕跡を含むプロキシサーバのログが必要という制約がある．今回提案した運用構想では，ブラックリスト方式の運用と併用することで，新たな RAT の痕跡を含む通信データの取得を可能としている．

7. おわりに

本稿では，http ベースの受信量や時刻等の通信の挙動から，パターンマッチングを用いずに RAT を検知する手法を用いた http ベースの通信挙動による RAT 検知システムを試作し，大規模な実運用のネットワークに適用して実用的な監視の業務に適用可能であることを示した．また，その RAT 検知システムと通信先のブラックリストを組み合わせ，新たな訓練データの取得を可能とする実用的な運用構想を示した．

今後の課題としては，他の実運用のネットワークへの適用や他のプロキシサーバのログへの適用が挙げられる．今回の実験では，実運用のネットワークにおける RAT による外部への不正通信が確認できなかったため，その効果については検証の余地がある．また，他の種類の RAT に対する効果も明確ではない．ほとんどの場合，RAT の痕跡を含むプロキシサーバのログは共有されることがないため，これを新たに入手することは現実的に期待できない．しかしながら，提案するブラックリストを併用した運用構想を，実運用のネットワークに適用することができれば，ブラックリストから他の種類の RAT を検知し，新たな RAT とその特徴ベクトルを訓練データに加えられる可能性もある．したがって，運用構想を含めた検知システムの実運用のネットワークへの適用が，これらの課題解決の糸口になるものと考えられる．

参考文献

[1] 三村 守, 大坪 雄平, 田中 英彦: プロキシのログからの機械学習による *RAT* の検知方式, Proc. CSS 2015, (2015).

[2] Perdisci, R., Lee, W. and Feamster, N.: *Behavioral Clustering of HTTP-based Malware and Signature Generation using Malicious Network Traces*, Proc. 2010 USENIX Symposium on Networked Systems Design and Implementation, 2010.

[3] Nelms, T., Perdisci, R. and Ahamad, M.: *ExecScent: Mining for New C&C Domains in Live Networks with Adaptive Control Protocol Templates*, Proc. 22nd USENIX Security Symposium, (2013).

[4] Rafique, M.Z. and Caballero, J.: *FIRMA: Malware Clustering and Network Signature Generation with Mixed Network Behaviors*, Proc. 16th International Symposium on Research in Attacks, pp.144-163, (2013).

[5] Kamiya, K., Aoki, K., Nakata, K., Sato, T., Kurakami, H. and Tanikawa, M.: *The Method of Detecting Malware-infected Hosts Analyzing Firewall and Proxy Logs*, Proc. 10th Asia-Pacific Symposium on Information and Telecommunication Technologies, (2015).

[6] Antonakakis, M., Perdisci, R., Dagon, D., Lee, W. and Feamster, N.: *Building a Dynamic Reputation System for DNS*, Proc. 19th USENIX Security Symposium, (2010).

[7] Antonakakis, M., Perdisci, R., Lee, W., Vasiloglou II, N. and Dagon, D.: *Detecting Malware Domains at the Upper DNS Hierarchy*, Proc. 20th USENIX Security Symposium, (2011).

[8] Antonakakis, M., Perdisci, R., Nadji, Y., Vasiloglou, N. Abu-Nimeh, S., Lee, W. and Dagon, D.: *From Throw-Away Traffic to Bots: Detecting the Rise of DGA-Based Malware*, Proc. 21th USENIX Security Symposium, (2012).

[9] Rahbarinia, B., Perdisci, R. and Antonakakis, M.: *Segugio: Efficient Behavior-Based Tracking of New Malware-Control Domains in Large ISP Networks*, Proc. 2015 IEEE/IFIP International Conference on Dependable Systems and Networks , (2015).

[10] Gu, G., Porras, P., Yegneswaran, V., Fong, M. and Lee, W.: *BotHunter: Detecting Malware Infection Through IDS-Driven Dialog Correlation*, Proc. 16th USENIX Security Symposium (2007).
[11] Gu, G., Zhang, J. and Lee, W.: *BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic*, Proc. 15th Symposium on Network and Distributed System Security, (2008).
[12] Gu, G., Perdisci, R., Zhang, J. and Lee, W.: *BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-Independent Botnet Detection*, Proc. ACM 17th Conference on Security Symposium, (2008).
[13] Yen, T.F. and Reiter, M.K.: *Are Your Hosts Trading or Plotting? Telling P2P File-Sharing and Bots Apart*, Proc. 2010 IEEE 30th International Conference on Distributed Computing Systems, pp.241-252, (2010).
[14] Zhang, J., Perdisci, R., Lee, W., Sarfraz, U. and Luo, X.: *Detecting Stelthy P2P Botnets Using Statistical Traffic Fingerprints*, Proc. 2011 IEEE/IFIP 41st International Conference on Dependable System&Network, pp.121-132, (2011).
[15] Jacob, G., Hund, R., Kruegel, C. and Holz, T.: *JACKSTRAWS: Picking Command and Control Connections from Bot Traffic*, Proc. 20th USENIX conference on Security, pp.29-29, (2011).
[16] Tegeler, F., Fu, X., Vigna, G. and Kruegel, C.: *BotFinder: Finding Bots in Network Traffic Without Deep Packet Inspection*, Proc. 8th International Conference on Emerging Networking Experiments and Technologies, pp.349-360, (2012).
[17] Bilge, L., Balzarotti, D., Robertson, W., Kirda, E. and Kruegel, C.: *DISCLOSURE: Detecting Botnet Command and Control Servers Through Large-Scale NetFlow Analysis*, Proc. 28th Annual Computer Security Applications Conference, (2012).
[18] 市野 将嗣，市田 達也，畑田 充弘，小松 尚久：トラヒックの時系列データを考慮した AdaBoost に基づくマルウェア感染検知手法，情報処理学会論文誌，Vol.53, No.9, pp.2062–2074 (2012).
[19] 岩野 透，吉浦 裕，畑田 充弘，市野 将嗣：LPC ケプストラム分析を利用したマルウェアの感染検知，情報処理学会論文誌，Vol.56, No.9, pp.1716–1729 (2015).
[20] 山内 一将，川本 淳平，堀 良彰，櫻井 幸一：C＆C トラフィック分類のための機械学習手法の評価，情報処理学会論文誌，Vol.56, No.9, pp.1745–1753 (2015).
[21] Tran, M. and Nakamura, Y.: *A Supplementary Method for Malicious Detection*, Journal of Communications, Vol.9, No.12, pp.923-929, (2014).
[22] Oprea, A., Li, Z., Yen, T., Chin, S. and Alrwais, S.A.: *Detection of Early-Stage Enterprise Infection by Mining Large-Scale Log Data*, CoRR, Vol.abs/1411.5005, (2014).
[23] Yeganeh, S.H. and Ganjali, Y.: *Beehive: Towards a Simple Abstraction for Scalable Software-Defined Networking*, Proc. 13th ACM Workshop on Hot Topics in Networks, pp.13-22, (2014).
[24] libsvm (online), 入手先 ⟨https://www.csie.ntu.edu.tw/~cjlin/libsvm/⟩ (2016.07.27).
[25] scikit-learn (online), 入手先 ⟨http://scikit-learn.org/⟩ (2016.07.27).
[26] httpry (online), 入手先 ⟨http://dumpsterventures.com/jason/httpry/⟩ (2016.07.27).