ディープラーニングを用いたログ解析による悪性通信の検出

2.1 はじめに

ディープラーニングは、悪意ある通信を発見する手法としても活用可能です。ここでは、マルウェア感染及びExploit Kitの悪性通信を、一般的なファイアウォールやWebプロキシサーバの膨大なログから検出する手法を紹介します。

なお本稿は、国際的なセキュリティカンファレンス「Black Hat Europe 2018」のBriefingにて"Deep Impact: Recognizing Unknown Malicious Activities from Zero Knowledge"[*1]というタイトルで発表した内容を再構成したものです。

2.2 背景

マルウェア感染をはじめとする悪意のある活動を発見するには、現状では以下のいずれかの手法、もしくはこれらを組み合わせたソリューションで検出するケースがほとんどです。

- パターンマッチ(ブラックリスト、ホワイトリスト含む)
- 振る舞い分析
- イベント相関分析

しかし洗練された攻撃手法や未知の攻撃は、これらのソリューションをすり抜ける可能性があります。またそのような攻撃でなくとも、例えばパターンマッチによる検出は、少しパターンを変えられただけで検出ルールを変更する必要があります。これは攻撃者が容易に変更可能な情報、例えばC2サーバのドメイン名、IPアドレスや実行ファイルのバイナリパターンなどを基に検出しているからです。つまり、これら既存手法に依存せず、かつ攻撃者が変更しづらい攻撃の本質となる部分を検出の条件にすることができれば、既存手法と組み合わせることで、よりセキュリティレベルを上げることが可能になります。

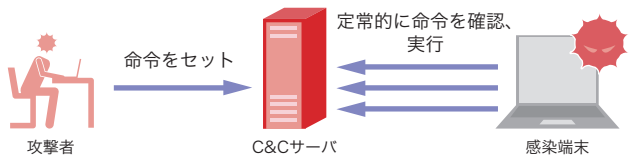

図-1 BOTやRATによるC&Cサーバへの定常的な通信

また、先に挙げたソリューションのいくつかは非常に高価で、必ずしもすべての組織が導入できるわけではありません。そこで今回は、多くの組織で汎用的に脅威を検出できるようにするため、特殊な装置やセキュリティデバイスではなく、Webプロキシサーバ、ルータやファイアウォールといった一般的なサーバやネットワーク機器のログから検出することを目指しました。これらのログは、従来は有効活用されることは稀で、例えば以下のようなケースに限られていました。

- 時間単位の通信量、回数などによる異常検知
- SIEMによるイベント相関分析
- IoC(Indicator Of Compromise)が入手できた場合[*2]

ディスク容量を圧迫しているこの種のログを利活用できれば、多くの組織が大きなネットワークの構成変更や追加投資をせずにセキュリティレベルを向上させることができます。

この種のログが有効活用されてこなかった要因の1つとして、システムや組織の規模にもよりますが、ログサイズが非常に大きく、複雑な処理を伴う解析が困難であるという点が挙げられます。しかしディープラーニングであれば、数十から数百キロバイト単位の情報量を持つ画像を数億件処理することができるなど、ビッグデータの解析に向いていることが知られています。よってログを適切に加工してディープラーニング向けに最適化できれば、この問題を解決できる可能性があるのです。

2.3 マルウェアのC&C(C2)サーバへの通信検知

BOTやRATなどのマルウェアは、C2サーバに定常的にアクセスを行い、攻撃者からの命令を取得して実行するという特性があります(図-1)。多くの場合、ポーリング間隔は数十秒から数分程度です。この間隔が長いほど、一命令あたりの待ち時間が長くなるため、攻撃者は活動しづらくなります。反対に、間隔が短ければ攻撃者は活動しやすくなりますが、単純に宛先ホスト別に通信回数を分析するだけで上位に来るため、防御側が発見

*1 Deep Impact:Recognizing Unknown Malicious Activities from Zero Knowledge(https://www.blackhat.com/eu-18/briefings/schedule/index.html#deep-impact-recognizing-unknown-malicious-activities-from-zero-knowledge-12276)。
*2 例えば異常検知やユーザからの報告などで発見した不審な端末から得られたホスト名や、外部ベンダーのレポートなどからIoCが得られる場合があります。

しやすくなります[*3]。つまり、攻撃者にとって通信頻度の調整は重要かつ変更しづらい特徴であると言えます。一方で、組織内の一般ユーザがWebアクセスなど外部と通信を行う場合、頻繁に、かつ長時間アクセスすることは稀です。図-2は、1時間あたりにユーザが無害なWebサーバにアクセスした場合のイメージ図(左)とマルウェアがC2サーバと定期的に通信を行った場合のイメージ図(右)です。ほとんどの場合、このように通信パターンに違いが出るため、このパターンの違いを学習できればマルウェアの通信を検出できると考えました。この方法はDNS名、IPアドレスやURLなどに依存していないので、既存の検知手法で見逃しても本手法で検知することができます。

今回は、クライアントとサーバごとにログを分割し、1時間あたり1分ごとの通信回数をカウントします。それを60ドットの画像に見立てて、ディープラーニングの1つであるCNN(Convolutional Neural Network)を使い、画像認識を行いました。CNNはモデルによっては人間の認識率を上回る精度が出ることが知られているため、ログを画像化することで他のディープラーニングモデルよりも効果が出ることを期待して用いました。

学習データセットは、良性サンプルにはWebプロキシログを変換した150万枚を超える「画像」を使用し、悪性サンプルには実際のマルウェアの通信パターンは一切使わず、定期的な通信パターンをエミュレートしています。これにより、検体を入手できなくても想定可能な悪性パターンを生成し、学習するだけでマルウェアを検出できます。これがBlack Hatで発表した際の副題に含まれている「Zero Knowledge」にかかっています。エミュレートしたインターバルは3秒から12分まで幅広く取り(図-3)、それに加えて特殊ケースとして数分間連続で通信

	0	1	2	3	4	5	6	7	8	9	(分)
0	9	0	0	0	0	0	0	0	0	0	
10	0	0	0	0	8	1	0	0	0	0	
20	0	0	0	0	0	0	0	0	0	0	
30	0	0	0	0	0	0	0	0	0	0	
40	0	0	0	0	0	0	0	0	0	0	
50	0	0	0	0	0	0	0	0	0	0	

(1)正規のWebサーバ宛て

	0	1	2	3	4	5	6	7	8	9	(分)
0	1	0	1	0	1	0	1	0	1	0	
10	1	0	1	0	1	0	1	0	1	0	
20	1	0	1	0	1	0	1	0	1	0	
30	1	0	1	0	1	0	1	0	1	0	
40	1	0	1	0	1	0	1	0	1	0	
50	1	0	1	0	1	0	1	0	1	0	

(2)C2サーバ宛て

図-2 正規のWebサーバとの通信(左)とC&Cサーバへの定期的な通信(右)イメージ

	0	1	2	3	4	5	6	7	8	9	(分)
0	20	20	20	20	20	20	20	20	20	20	
10	20	20	20	20	20	20	20	20	20	20	
20	20	20	20	20	20	20	20	20	20	20	
30	20	20	20	20	20	20	20	20	20	20	
40	20	20	20	20	20	20	20	20	20	20	
50	20	20	20	20	20	20	20	20	20	20	

3秒に1回の通信をエミュレート

…

	0	1	2	3	4	5	6	7	8	9	(分)
0	1	0	0	0	0	0	0	0	0	0	
10	0	0	1	0	0	0	0	0	0	0	
20	0	0	0	0	1	0	0	0	0	0	
30	0	0	0	0	0	0	1	0	0	0	
40	0	0	0	0	0	0	0	0	1	0	
50	0	0	0	0	0	0	0	0	0	0	

12分ごとの通信をエミュレート

図-3 悪性通信のエミュレート

*3 最近のマルウェアは、C2サーバからスリープ時間を受け取り、攻撃者がアクティブなときだけ短時間スリープすることで頻繁に通信が発生し、それ以外の時は長時間スリープすることで、例えば1日平均あたりの単位時間で見ると異常検知には引っかかりづらくするなどの工夫が凝らされている場合もあります。

した後、数分間スリープするようなパターンも生成しています（図-4）。更に、想定するパターンから少しだけ異なるパターンが出現した場合やCNNに対する攻撃[*4]の耐性を上げるために、生成したパターンを1ドットずつずらすローテーション、もしくは既存の値をランダムで一部クリアするなどして、合計で約100万枚のパターンを生成しました。

また、テストデータセットには学習データと同様に、別の期間の約450万枚のWebプロキシログから変換した「画像」を良性サンプルに用い、悪性サンプルには実際のインシデントで得られたマルウェアの通信ログを画像化して検出できるかを試しています。今回は以下のマルウェアファミリについて調査しました[*5]。

- PlugX
- Asruex
- xxmm
- himawari/ReadLeaves
- ChChes
- Elirks
- Logedrut
- ursnif/gozi
- Shiz/Shifu
- Vawtrak
- KINS

結果は、今回構築したモデルを使用した場合、いずれのマルウェアも検出できました。また、以下のとおり良性サンプルを誤検知する確率も低いため、ホワイトリスト方式でこれらのFQDNを取り除けば、運用できそうなことが分かります。

- 良性サンプルセット1
  Accuracy：1,565,139/1,566,109（99.94%）
  誤検知したFQDNの数：64/246,190

- 良性サンプルセット2
  Accuracy：1,540,419/1,541,050（99.96%）
  誤検知したFQDNの数：72/243,106

- 良性サンプルセット3
  Accuracy：1,528,936/1,529,617（99.96%）
  誤検知したFQDNの数：65/243,185

ただし、様々な環境に適用する場合は、例えばWebメールやスポーツサイトなど、頻繁にリロードされるWebページなどを誤検知する可能性も考えられるため、それらをホワイトリストで消しこむか、多くのユーザから同一の宛先へのアラートが上がった場合は、正規のWebサーバへのアクセスとみなして検知しないなどの運用をして誤検知を減らします。

	0	1	2	3	4	5	6	7	8	9	(分)
0	1	1	0	0	1	1	0	0	1	1	
10	0	0	1	1	0	0	1	1	0	0	
20	1	1	0	0	1	1	0	0	1	1	
30	0	0	1	1	0	0	1	1	0	0	
40	1	1	0	0	1	1	0	0	1	1	
50	0	0	1	1	0	0	1	1	0	0	

2分間連続で通信後、2分間スリープ

	0	1	2	3	4	5	6	7	8	9	(分)
0	1	1	1	0	0	0	1	1	1	0	
10	0	0	1	1	1	0	0	0	1	1	
20	1	0	0	0	1	1	1	0	0	0	
30	1	1	1	0	0	0	1	1	1	0	
40	0	0	1	1	1	0	0	0	1	1	
50	0	0	0	0	1	1	1	0	0	0	

3分間連続で通信後、3分間スリープ

…

図-4 悪性通信のエミュレート（2）

＊4　Simple Black-Box Adversarial Perturbations for Deep Networks（https://arxiv.org/abs/1612.06299）。
＊5　単にマルウェア検体を入手して閉鎖環境で実行して採取したパターンではなく、実際のインシデントでC2サーバに接続して活動していたものを選んでいます。これは、＊3で記述したとおり、生存しているC2サーバに接続中だった場合と、閉鎖環境でただ動かした場合のスリープ時間に違いが出る可能性があるためです。

図-5から図-9は、実際のマルウェアの通信において検知に成功した通信パターンの例です。実際の検体は、完全な定期的通信でないのが見てとれますが、ディープラーニングによりその誤差を吸収して検知しています。Logedrutの通信(図-8)は12分に一度と頻度が低いですが、それでも良性サンプルとは区別して検知できています。またVawtrakの例(図-9)においては、最後の16分は通信が発生していませんが、このようなケースにおいても検出しています。

誌面の都合上、すべてのマルウェアファミリの例やモデルの詳細は本稿には載せていません。詳しくは、Black Hat Europe 2018のWebページの資料[*1]に掲載しています。今回は複雑なモデルを構築していないため、CPUでも十分に学習可能でかつ実用に耐えうる精度を確保できたと考えています。

	0	1	2	3	4	5	6	7	8	9	(分)
0	6	6	0	3	6	3	0	6	6	0	
10	3	6	3	0	6	6	0	3	7	2	
20	0	6	6	0	3	7	2	0	6	6	
30	0	3	8	1	0	6	6	0	3	8	
40	1	0	6	6	0	3	8	1	0	6	
50	6	0	4	8	0	0	7	5	0	5	

	0	1	2	3	4	5	6	7	8	9	(分)
0	96	95	92	96	95	97	96	97	101	95	
10	97	93	95	96	95	98	92	93	95	101	
20	96	95	94	93	88	98	95	97	97	96	
30	97	88	94	96	94	101	98	97	97	96	
40	95	95	91	93	91	101	96	100	97	89	
50	92	94	96	98	94	98	98	92	94	95	

図-5 PlugXの通信パターン

	0	1	2	3	4	5	6	7	8	9	(分)
0	0	0	0	0	0	0	0	0	0	4	
10	0	2	0	2	0	2	0	0	2	0	
20	0	2	2	0	2	0	0	2	0	2	
30	0	2	0	0	2	0	2	0	2	0	
40	0	2	0	0	2	0	0	2	0	2	
50	2	0	2	0	2	0	2	2	0	2	

図-6 Asruexの通信パターン

	0	1	2	3	4	5	6	7	8	9	(分)
0	1	0	1	1	0	1	1	0	1	1	
10	0	1	1	0	1	0	1	0	1	0	
20	1	0	1	0	1	0	0	1	1	0	
30	0	1	1	0	0	1	1	0	0	1	
40	1	0	0	1	0	1	0	1	0	1	
50	0	1	0	1	0	0	1	1	0	0	

図-7 Elirksの通信パターン

	0	1	2	3	4	5	6	7	8	9	(分)
0	0	0	0	0	0	0	1	0	0	0	
10	0	0	0	0	0	0	0	1	0	0	
20	0	0	0	0	0	0	0	0	0	1	
30	0	0	0	0	0	0	0	0	0	0	
40	1	0	0	0	0	0	0	0	0	0	
50	0	1	0	0	0	0	0	0	0	0	

図-8 Logedrutの通信パターン

	0	1	2	3	4	5	6	7	8	9	(分)
0	1	0	0	0	1	1	0	0	0	1	
10	1	0	0	1	1	0	0	0	1	1	
20	0	0	0	1	1	1	0	0	1	0	
30	1	0	0	1	0	1	0	0	1	0	
40	1	0	0	1	0	0	0	0	0	0	
50	0	0	0	0	0	0	0	0	0	0	

図-9 Vawtrakの通信パターン

2.4 Exploit Kitの検知

Webを閲覧しているPCがExploit kitに誘導されたとき、Exploit kitのサーバは図-10の順にコンテンツを送信します。

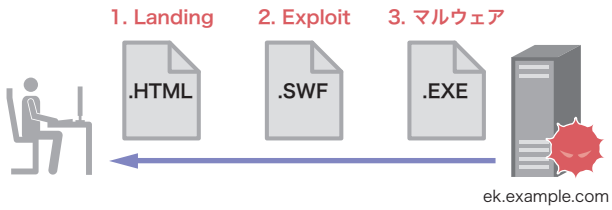

図-10 Exploit Kitサーバから送信されるコンテンツタイプの遷移

1. Landingページ：PCのWebブラウザ環境を識別し、次段のExploitコンテンツをロードさせる。Webブラウザ自体を対象としたExploitを含んでいる場合もある。コンテンツタイプはtext/html
2. Exploitコンテンツ：Webブラウザやそのプラグインを対象としたExploitを含むコンテンツファイル。コンテンツタイプはapplication/x-shockwave-flash、application/x-java-archive、application/x-silverlight-app、application/pdfなど
3. マルウェア：前段のExploitが成功すると、PCに感染させるマルウェア本体がロードされる。ほとんどの場合、コンテンツタイプはapplication/octet-streamまたはapplication/x-msdownload

内容	コンテンツタイプ	URLパス&パラメータ
Landingページ	text/html	/?NTI0OTU5&RCDUIv&oJhtJNm=dGFraW5n&wouMDc=Y2FwaXRhbA==&JgtXjOEttIAHrl=Y2FwaXRhbA==&TKCcodYFxdiy=dGhpbmdz&tNDodvGjF=Y2FwaXRhbA==&pHtonQrvp=bG9jYXRlZA==&kl345dfdfg234fsd=UDQTpjkGELQNmyN9ZAF1G9P2s3EeBzhWZiMHT-RTZZA4QrZSQR7Rt3VzyxrckQPskg1TH6ml&pWjLlCBUlUSRlw=Y2FwaXRhbA==&nR45dsgd54lsCs=xXrQMvWfbRXQDJ3EKvjcT6NAMVHRGUCL2YqdmrHXefjaf1WkzrfFTF_3ozKATASG6_ZtdfJ

内容	コンテンツタイプ	URLパス&パラメータ
Flash Exploit	application/x-shockwave-flash	/?NTQ0NjEw&zWuWFX&lskPeVWn=dW5rbm93bg==&NCDmQdmxCxapA=dW5rbm93bg==&eLCxfNVxDhHqBH=Y29uc2lkZXI=&nzZHzkCNdL=cmVwb3J0&HZELKhjPUenym=cG9wdWxhcg==&nR45dsgd54lsCs=wnrQMvXcKxXQFYbDKuXDSKZDKU7WG0aVw4-dhMG3YpjNfynz1ezURnL1tASVVFiRrbMdKL&kl345dfdfg234fsd=VYOQfk20LUKgEzm9sJVFhBo66tjUmDmBCd1JLX-UeLMg9DqZOSHblL0Vz0zLMRQlgigECy&rZpDUeqxIDnMQL=bG9jYXRlZA==&LENxPZQZ=cmVwb3J0

内容	コンテンツタイプ	URLパス&パラメータ
マルウェア	application/x-msdownload	/?MjEwNzA1&mTONXmiGJttk&nR45dsgd54lsCs=wXrQMvXcJwDQDobGMvrESLtGNknQA0KK2lv2_dqyEoH9fWnihNzUSkr16B2aCm3W&UEiQzsUEYQeeS=Y2FwaXRhbA==&jeeGWAgbhZSFoHh=bG9jYXRlZA==&KRssZN=bG9jYXRlZA==&BWeciQaXKEgAey=bG9jYXRlZA==&SOymAmL=cG9wdWxhcg==&uLNyyCiGt=cG9wdWxhcg==&wlNBeZFOQXgP=dW5rbm93bg==&kl345dfdfg234fsd=_fcpKeRXaVKziULVLwczyllbUVJFpqj6i0SAmxDPhcGD_hKEUQ1M-5KREYFmmF7F

図-11 Rig Exploit Kitのコンテンツタイプ遷移の例

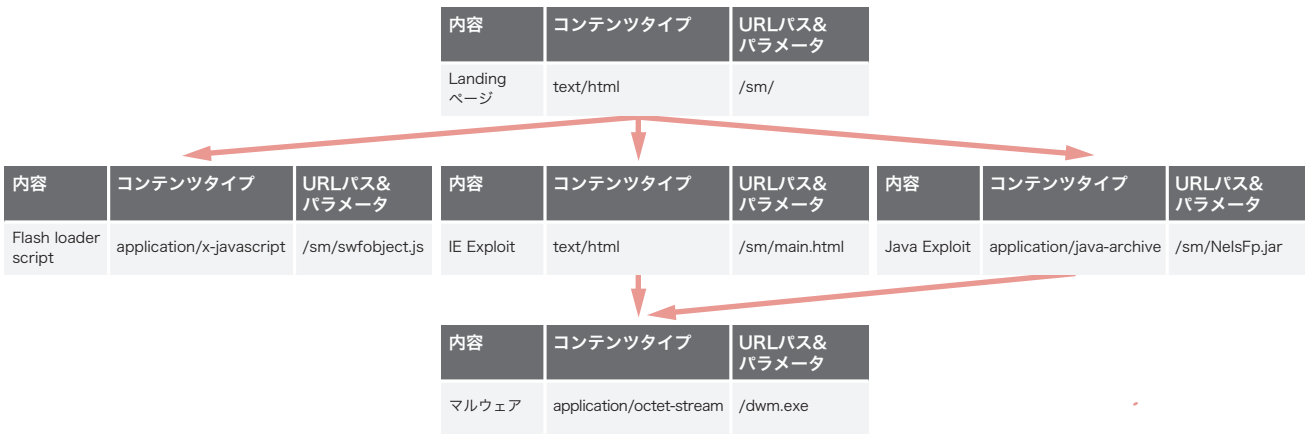

図-12 KaiXin Exploit Kitのコンテンツタイプ遷移の例

図-11、図-12はそれぞれRig Exploit Kit、KaiXin Exploit Kitの観測時のコンテンツタイプ遷移の例です。前述の通りに遷移していることが確認できます。

一方、通常のWeb閲覧時は、任意のWebサーバから送信されるコンテンツタイプがこのように遷移するケースはほとんど考えられません。例えば、大規模なWebサービスなどではコンテンツタイプに応じてサーバが用意されることが多いため、図-13のようにExploitとして悪用されるFlashやJavaなどのコンテンツと、Landingページのような HTMLコンテンツは異なるサーバから送信される傾向にあります。また、単一サーバで全コンテンツをホストしている場合は、図-14のように近年のExploit kitではあまり用いられない画像やCSSなどが同じサーバから送信されることになります。

これらを踏まえた上で、Webプロキシログなどを解析してExploit kitサーバから送信されるコンテンツタイプの遷移と、通常のWebサーバ接続時のコンテンツタイプの遷移を識別することができれば、パターンマッチなどに依存することなくExploit kitを検知できるのではないかと考えました。前述のExploit kit特有のコンテンツタイプの遷移は、「WebブラウザにExploitコンテンツを実行させ、PCをマルウェアに感染させる」というExploit kitの仕組みそのものを表しているため、未知のExploit kitであっても同様に検知できるはずです。また、同じ理由から、Exploit kit作成者がこの遷移を変更して検知を逃れることは簡単ではないものと考えられます。

遷移の識別には、自然言語処理やビデオ/オーディオストリームなどの時系列データの処理に使われるRNN(Recurrent Neural Network)という手法を用います。そのため、最初にWebプロキシログをRNNで処理できる形式に変換する必要があります。ここでは、ログを個々のクライアントPC–宛先サーバごとに分解して一連のセッション[*6]を1つの単位(以後「シーケンス」)とします。更に、ノイズ耐性を高める目的で、各シーケンス内で

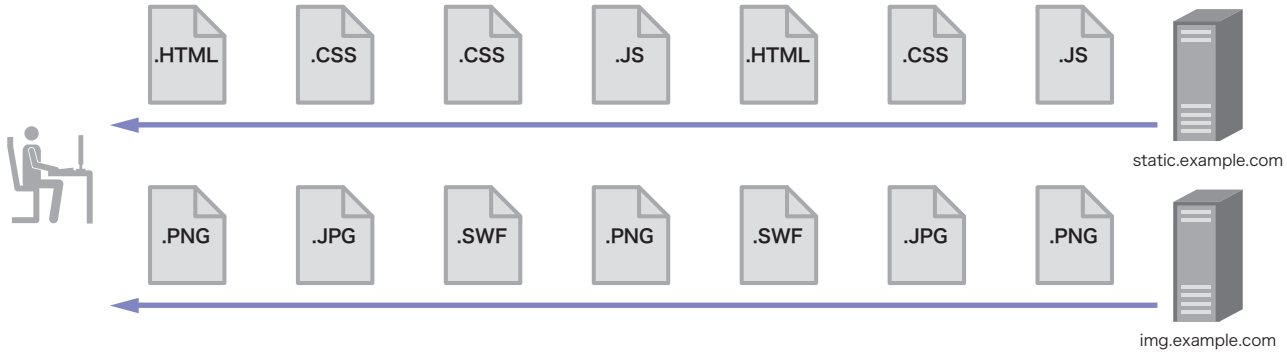

図-13 コンテンツタイプに応じたサーバを備えたWebサービス利用時のコンテンツタイプ遷移の例

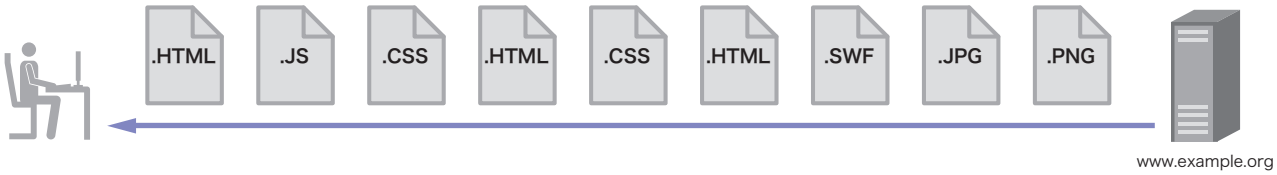

図-14 単一のWebサーバでホストされるサービス利用時のコンテンツタイプ遷移の例

*6 正確には、クライアントPC-宛先サーバごとに分解した上で、更にWebブラウジングセッションごと(Webブラウザなどで任意のURLへアクセスした際に、追加の操作なしで連続して読み込まれるファイル群への一連のリクエストおよびレスポンス)に分解します。今回使用したWebプロキシログは個々のセッションを識別できる環境で取得したものですが、一般的なWebプロキシ環境であってもタイムスタンプなどを利用して個々のセッションを推測できます。

同じコンテンツタイプが連続する場合には、それらを削除しています。また、シーケンス長の上限は5とし[*7]、それを超える場合は、以降を削除しました。最後に、シーケンス内の各行をそれぞれ84次元のベクトルに変換します。これは、One–Hotエンコーディングで変換したコンテンツタイプを示す83次元と、「リファラとリクエストURLが同じドメインを含むか否か」のフラグによって構成されています。

学習データセットには、約390万行のWebプロキシログを変換した約58万シーケンスの良性サンプルと、想定されるExploit kitの遷移パターンをエミュレートした約30万シーケンスの悪性サンプルを使用しました。実際に観測したパターンではなく、Exploit kitのコンテンツ遷移として想定されるパターンを網羅的に生成したものを悪性サンプルとしています。図-15は生成した擬似シーケンスの一例です。複数種のExploitコンテンツがロードされるケースやExploitが複数回連続して成功するケース、Exploitが成功せずマルウェアのダウンロードが生じないケースなどを想定したシーケンスを含んでいます。

テストには、次の14種類のExploit kitの実際の通信データを変換した悪性サンプルと、学習データセットとは異なる期間のWebプロキシログを変換した約170万シーケンスの良性サンプルを使用しました。

- Rig
- Nebula
- Terror
- Sundown
- KaiXin
- Neutrino
- Angler
- Nuclear
- Magnitude
- Fiesta
- SweetOrange
- Goon
- Infinity
- Astrum

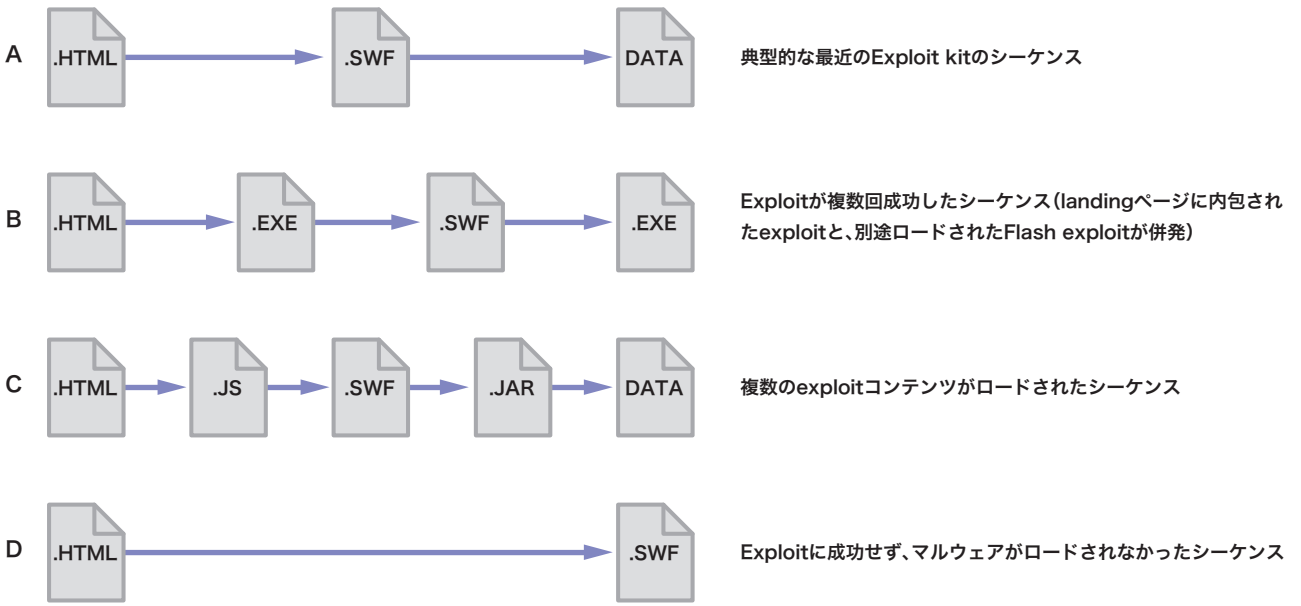

図-15 生成した擬似Exploit Kitシーケンスの例

*7 昨今観測されるExploit kitでは、サーバからのコンテンツ送信が5回を超えることはほとんどありませんが、今後そのようなケースが増えた場合は、この上限を増やす必要があると考えられます。

今回構築したモデルは上記すべてのExploit kitを検知することができました。また良性サンプルに関しても、以下のとおり比較的低い誤検知率を示すことを確認しました。

・良性サンプルセット1
シーケンス数:562,390
誤検知数:642
Accuracy 0.9988

・良性サンプルセット2
シーケンス数:574,452
誤検知数:681
Accuracy 0.9988

・良性サンプルセット3
シーケンス数:576,294
誤検知数:639
Accuracy 0.9988

なお、15行程度のホワイトリストを適用することで、上記の誤検知数が半減することを確認しています。実環境に適用する場合は、このようなホワイトリストやホストレピュテーション、異常検知、サンドボックスによる自動解析といった他の手法を併用してアラートを絞り込むことを推奨します。

また、誌面の都合上、掲載していませんが、Black Hat Europe 2018のWebページに公開している資料[*1]では、MLP(Multilayer Perceptron)モデルを用いてWebプロキシログからRig Exploit Kitを識別する手法も公開しています。この手法は、個々のExploit kitが備えるURLの特徴に注目するため、新出Exploit kitの検出には不向きですが、既知のExploit kitを識別したり亜種を追跡したりする用途には適しています。ここで紹介したRNNを用いるExploit kit検知手法と併用することで、既知のExploit kitに関する検知精度を高めることが可能です。

執筆者:
鈴木 博志(すずき ひろし)
IIJ セキュリティ本部 セキュリティ情報統括室 マルウェア&フォレンジックアナリスト。
IIJ-SECTのメンバーで、主にマルウェア解析とフォレンジック調査を担当。
Black HatやFIRST TCなどの国際カンファレンスや、セキュリティキャンプ(GCC含む)、サイバーコロッセオなどで、講演やトレーニングを行う。
Black Hat USAでは日本人として初めてトレーニング講師に選ばれた。

執筆者:
梨和 久雄(なしわ ひさお)
IIJ セキュリティ本部 セキュリティ情報統括室 スレットアナリスト。
IIJ-SECTメンバー。
社内外のインシデント対応やWebクローラを用いた調査、マルウェア解析、フォレンジック調査などの業務に従事。それらの知見に基づき、Black Hat、FIRST TCなどの国際カンファレンスで講演やトレーニングを行う。