IIR

Internet
Infrastructure
Review

Mar.2019
Vol.42

定期観測レポート

SOCレポート

フォーカス・リサーチ(1)

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

フォーカス・リサーチ(2)

大規模メールシステムの設計

IIJ
Internet Initiative Japan

Internet Infrastructure Review

March 2019 Vol.42

エグゼクティブサマリ

従来のDDoS攻撃は、マルウェアに感染した大量のPCによるBotnetを悪用し、いっせいに攻撃対象に通信を発生させるものが主流でした。それが2013年頃から、ホームルータやインターネットカメラなどIoT機器の不適切な設定を利用したり、それらの脆弱性を悪用してマルウェアに感染させることによる大規模なDDoS攻撃が観測されるようになりました。

インターネットに接続されるIoT機器の数は今後も大きく増加すると想定され、それらの機器がDDoS攻撃に利用されるのを防ぐことは、安全なインターネットを保つ上でとても重要です。そのような脅威に対応するため、日本では2月1日に総務省、国立研究開発法人情報通信研究機構（NICT）、電気通信事業者により、IoT機器調査と利用者への注意喚起を促す取り組み「NOTICE（National Operation Towards IoT Clean Environment）」が発表されました。

NOTICEでは、NICTがインターネット上でサイバー攻撃に悪用されるおそれのあるIoT機器を調査し、当該機器の情報を電気通信事業者に通知します。通知を受けた電気通信事業者は、当該機器の利用者を特定し、注意喚起を行うことになっています。これは官民が協力して、インターネットに接続されるIoT機器の安全性を高める取り組みであり、IIJもその一員として積極的に参加しています。

「IIR」は、IIJが研究・開発している幅広い技術の紹介を目指しています。私たちが日々のサービス運用から得られる各種データをまとめた「定期観測レポート」と、特定テーマを掘り下げた「フォーカス・リサーチ」から構成されています。

1章の定期観測レポートでは、SOCレポートを取り上げます。IIJのSOCでは、サービスとして提供しているセキュリティ機器のログをはじめ、様々なログを情報分析基盤に集約・分析し、観測した脅威情報を「wizSafe Security Signal」でタイムリーに発信しています。ここでは「wizSafe Security Signal」で発信したレポートのなかから、情報分析基盤を活用して明らかになった特筆すべき3つの活動と、情報分析基盤を活用した機械学習について紹介します。

2章のフォーカス・リサーチ（1）では、IIJの社員がBlack Hat Europe 2018で発表した内容を「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」として再構成・掲載しました。特殊な装置やセキュリティ機器を用いるのではなく、一般的なサーバやネットワーク機器のログから、汎用的に脅威を検出できるような仕組みを検討しました。これらの膨大なログは複雑な処理が必要ですが、適切に加工してディープラーニング向けに最適化すれば、有効活用できる可能性があることを確認しています。

3章のフォーカス・リサーチ（2）では、IIJが提供しているメールゲートウェイサービス「IIJセキュアMXサービス」のリニューアルについて紹介します。本サービスは提供開始から10年以上が経過していますが、今もなお契約数が大きく伸びているIIJの代表的なサービスの1つです。とはいえ、10年も経つと利用環境が変化すると同時に、システムも陳腐化し、様々な課題を抱えていました。それらの課題を解決するためのアーキテクチャの見直しや、システムの自社開発という判断など、実際に開発に携わったエンジニアのレポートが参考になれば幸いです。

IIJでは、このような活動を通じて、インターネットの安定性を維持しながら、日々改善・発展させていく努力を続けています。今後も、企業活動のインフラとして最大限にご活用いただけるよう、様々なサービス、ソリューションを提供し続けてまいります。

島上 純一（しまがみ じゅんいち）
IIJ 取締役 CTO。インターネットに魅かれて、1996年9月にIIJ入社。IIJが主導したアジア域内ネットワークA-BoneやIIJのバックボーンネットワークの設計、構築に従事した後、IIJのネットワークサービスを統括。2015年よりCTOとしてネットワーク、クラウド、セキュリティなど技術全般を統括。2017年4月にテレコムサービス協会MVNO委員会の委員長に就任。

SOCレポート

1.1 はじめに

IIJでは、2016年10月31日にセキュリティ事業の新ブランド「wizSafe(ウィズセーフ)」を発表し[*1]、お客様が安全にインターネットを利用できる社会に向けて日々活動しています。その一環として、SOCで観測したセキュリティに関する脅威情報をwizSafe Security Signal[*2]を通してブログ形式でタイムリーに発信しています。この中では、IIJの情報分析基盤を通して解明した脅威情報の一部を公開しています。情報分析基盤の概要については、過去のInternet Infrastructure Review(IIR) Vol.38[*3]をご覧ください。

ここでは、情報分析基盤を活用した分析の概要を紹介します。情報分析基盤には、IIJサービスとして提供しているファイアウォールやIPS/IDS、アンチウイルスなどのセキュリティ機器のログをはじめ、DNSクエリやWebアクセス、メール送受信ログなど、様々なログが集約されています。これらのログには、大量の正常通信の中にごくわずかな異常(脅威)通信が含まれるという特性があります。そのため、脅威が明確に確認できるよう、集計方法や可視化の検討が必要となります。

1.2節では2018年に情報分析基盤を通して明らかになった脅威情報について紹介し、1.3節では情報分析基盤を活用した新たな取り組みについて紹介します。なお、2018年の観測情報はwizSafe Security Signalにまとめています[*4]。

1.2 観測情報

まずは、昨年wizSafe Security Signalで報告した内容の中から、情報分析基盤を活用して明らかになった特筆すべき活動について取り上げます。

1.2.1 仮想通貨に関連する攻撃

2018年は、攻撃者が仮想通貨を用いて攻撃の収益化を試みる事例が注目を集めた年でした。IIJの情報分析基盤を用いた分析でも、攻撃者が仮想通貨の悪用を試みる事例を複数観測しています。

まず、Webサイトの改ざんに伴うマイニングスクリプトの埋め込み事例です。SOCの観測では、Webサイトに埋め込まれたマイニングスクリプトの中に、管理者が意図して埋め込んだもの

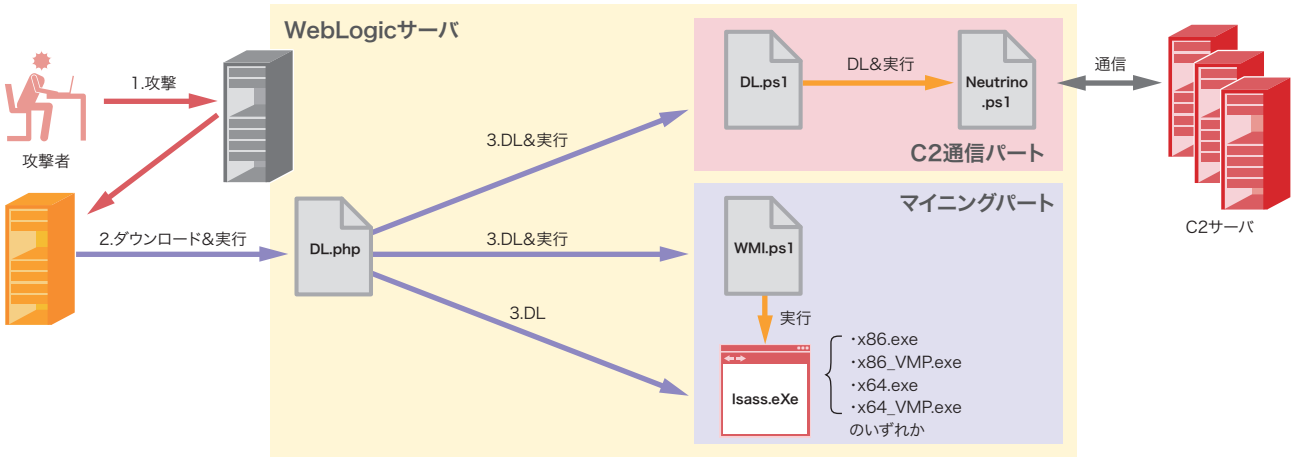


図-1 GhostMinerによる攻撃の流れ

＊1　IIJ、セキュリティ事業の新ブランド「wizSafe(ウィズセーフ)」を発表(https://www.iij.ad.jp/news/pressrelease/2016/1031.html)。
＊2　wizSafe(https://wizsafe.iij.ad.jp)。
＊3　Internet Infrastructure Review(IIR)Vol.38(https://www.iij.ad.jp/dev/report/iir/038/01.html#anc02)。
＊4　wizSafe、「wizSafe Security Signal 2018年 年間サマリ」(https://wizsafe.iij.ad.jp/2019/03/601/)。

ではないと考えられるケースが複数見つかっています。攻撃者は、Webサイトに存在する脆弱性を悪用するなどの方法で、ユーザが閲覧するWebページにマイニングスクリプトを埋め込みます。これにより、被害サイトを閲覧したユーザのコンピュータで仮想通貨がマイニングされ、収益が攻撃者の手に渡ります。

上記の事例はクライアントを狙ったものですが、サーバに仮想通貨をマイニングさせる攻撃も観測しています。具体例の1つとしては、GhostMiner(図-1)と名付けられた攻撃キャンペーン[*5]が挙げられます。GhostMinerキャンペーンは2018年3月に観測しており、Oracle WebLogic Serverの脆弱性(CVE-2017-10271)が悪用されました。脆弱性はリモートコード実行が可能となるもので、攻撃者は最終的にWebサーバに対し仮想通貨をマイニングさせようと試みます。なお、この他にもリモートコード実行の脆弱性を利用して、サーバに仮想通貨をマイニングさせる試みを多数観測しています[*6][*7]。

また、マイニングではなく不正送金を目指す試みとして、2018年12月にはEthereumのクライアントが備えるJSON-RPCを狙ったスキャン活動(図-2)を観測しました[*8]。スキャン活動では、設定の不備によりインターネットからアクセスできる状態にあるEthereumのクライアントを探索していました。なお、実際に不正な送金を成立させるには、複数の条件を満たす必要があります。

仮想通貨は、攻撃を直接的に収益化でき、種類によっては匿名性も高いという、攻撃者にとって都合の良い特性を備えています。また、仮想通貨のマイニングを意図した攻撃では、攻撃対象としてクライアントとサーバの事例が混在しているように、計算リソースさえあればどのような環境も標的となります。今後も、仮想通貨は攻撃を収益化する方法の1つとして用いられていくと考えられます。

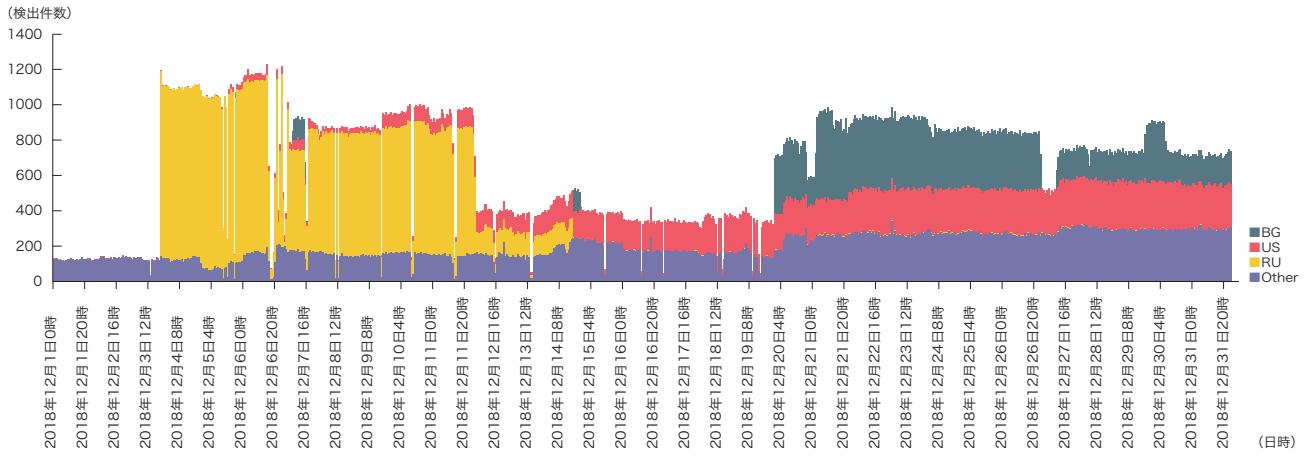


図-2 8545/TCPに対するスキャン活動(2018年12月)

*5 wizSafe、「GhostMinerの感染拡大」(https://wizsafe.iij.ad.jp/2018/04/323/)。
*6 wizSafe、「wizSafe Security Signal 2018年1月 観測レポート」(https://wizsafe.iij.ad.jp/2018/02/247/)。
*7 wizSafe、「wizSafe Security Signal 2018年2月 観測レポート」(https://wizsafe.iij.ad.jp/2018/03/286/)。
*8 wizSafe、「EthereumのJSON RPCにおけるスキャン活動の観測」(https://wizsafe.iij.ad.jp/2019/01/541/)。

1.2.2 SYN/ACKリフレクション攻撃

2018年にSOCで観測したDDoS攻撃の中で特徴的だった事例は、wizSafe Security Signal 2018年9月[*9]に掲載した80/TCPを利用したSYN/ACKリフレクション攻撃です（図-3）。送信元アドレスを偽装したTCPのSYNパケットを多数のアドレスに同時に送信し、その応答であるSYN/ACKパケットを利用して送信元アドレスに対してDDoS攻撃を行うものです。

このSYN/ACKリフレクション攻撃をSOCで観測したのは2018年9月26日でしたが、10月以降も小規模ながら観測しており、情報分析基盤を通して日々、同種の攻撃を検知しています。9月26日に観測したDDoS攻撃の1つの特徴は、攻撃の送信元がインターネット上に80/TCPを公開している各サーバに対して少量のSYNパケットを送信して攻撃を実現している点です。仮に攻撃者が単一のサーバに対してSYNパケットを大量に送信してしまうと、SYNパケットを受け取ったサーバの管理者がTCP SYN Flood攻撃[*10]を受けていると判断して通信を遮断してしまう可能性があります。その場合、攻撃者が想定している攻撃規模を実現できない可能性があります。また、サーバ1台あたりに届くSYNパケットの量が少量であるため、攻撃者は広範囲にSYNパケットを送信していると考えられます。

前述の攻撃で利用されていたポートは80/TCPであり、80/TCPを公開しているサーバは一般的にWebサーバであると考えられます。そのため、正常なWebアクセス通信の中にSYN/ACKリフレクション攻撃で利用するSYNパケットが少量送信されていたところで、そのパケットが攻撃に利用されていると判断するのは困難です。この事例の場合は、情報分析基盤上に存在する複数のお客様のファイアウォール・ログを横断的に分析することで検出しました。

ファイアウォール・ログを用いた検知では、内外部からのアクセス情報が確認できるため、複数のファイアウォール・ログで、80/TCPの応答通信が特定のIPアドレス（攻撃対象となっている偽装されたIPアドレス）に対して発生している場合に検知可能となります。ただし、この特徴はDDoS攻撃ではなくスキャン活動である可能性もあります。そのため、ファイアウォール・ログから集計された送受信バイト数や継続時間などを基にDDoS攻撃とスキャン活動を区別しています。

2018年9月に掲載したSYN/ACKリフレクション攻撃については、IIJのハニーポットにおいても観測しており、IIJ SECTブログの「IoT機器を踏み台として利用するSYN/ACKリフレクション

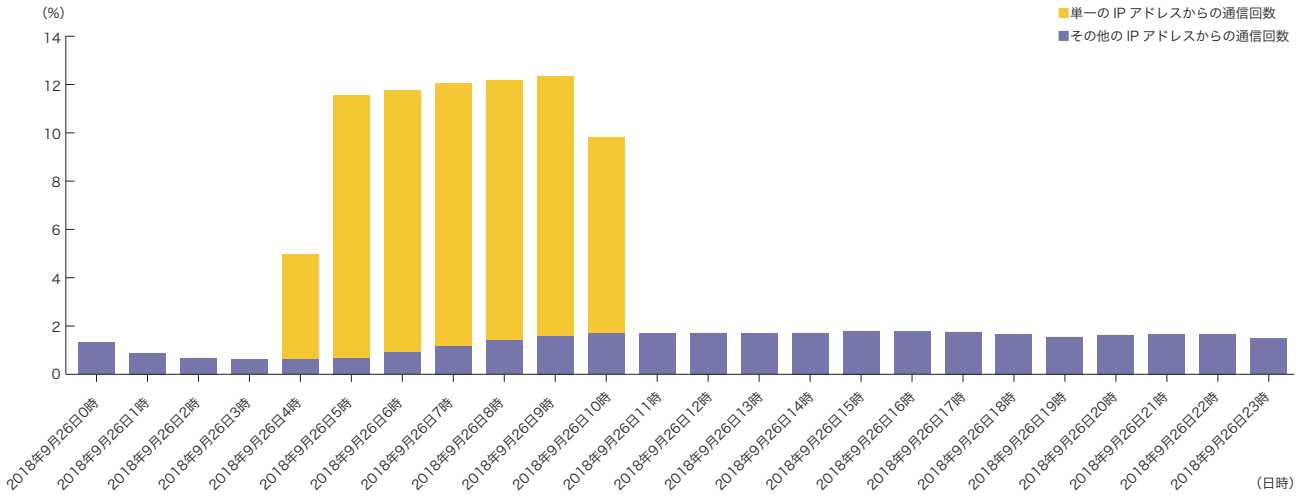


図-3 単一のIPアドレスから80/TCPへの通信の増加

＊9　wizSafe、「wizSafe Security Signal 2018年9月 観測レポート」（https://wizsafe.iij.ad.jp/2018/10/470/）。

＊10　TCP SYN flood攻撃は、TCP接続の開始の呼を示すSYNパケットを大量に送付することで、攻撃対象に大量の接続の準備をさせ、対象の処理能力やメモリなどを無駄に利用させる。

攻撃」[*11]で詳しく報告しています。攻撃に利用されているポートの変化やUDPプロトコルを利用した複合型のDDoS攻撃であることが解説されていますので併せてご覧ください。

1.2.3 既知の脆弱性を狙った攻撃の再興

2018年の情報分析基盤を用いた分析で印象的だったのが、過去に発表されて既に問題を修正したパッチが提供されている脆弱性が、時間をあけて再び狙われる事例です。具体例の1つとしては、Microsoft Officeの数式エディターの脆弱性(CVE-2017-11882)を悪用したマルウェアが挙げられます。

マルウェアが悪用した脆弱性は、Microsoft Officeの数式エディターにバッファオーバーフローの問題があり、リモートコード実行が可能になるというものです。Microsoft社は、この脆弱性を修正するパッチを2017年11月に提供しています。また、パッチを適用する以外の回避策として、数式エディターの機能を無効化する、という方法もありました。

情報分析基盤では、修正から1年近く経った2018年9月に、この脆弱性を狙った攻撃を観測しました(図-4)[*12]。攻撃者は、脆弱性を悪用するマルウェアをメールに添付して送付しています。脆弱性に関する意識が世間から薄れたタイミングで、未対策あるいは一時的に機能を無効化することで問題を回避していた環境を意図的に狙ったと考えられます。

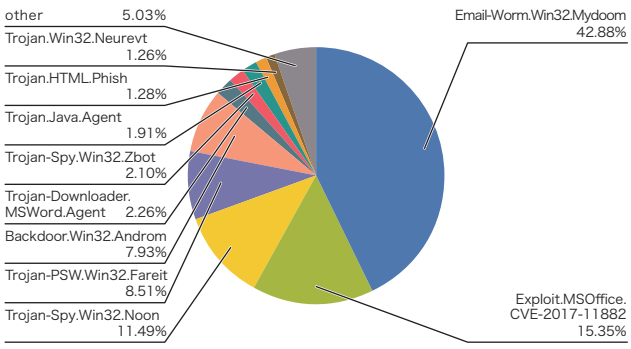


図-4 メール受信時に検出したマルウェア種別の割合(2018年9月)

このような事例は、今回取り上げたMicrosoft Officeの脆弱性以外にも情報分析基盤で複数観測しています[*13]。脆弱性について、根本的な対策や様々な事情から回避策を用いる場合には継続的な対応が要となることが、教訓として得られる事例といえます。

1.3 機械学習を用いた悪性通信の検出

情報分析基盤が取り扱うデータには、大量の正常通信の中にごくわずかな異常(脅威)通信が含まれるという特性があります。これを機械学習によって発見する取り組みが進んでいます。タスクとして主に扱うのは、不均衡データ(Imbalanced Data)からの異常検知(Anomaly Detection)です。ここでは、進行中の2つのプロジェクトとその課題について紹介します。

1.3.1 DNSクエリデータへの適用

マルウェアは、C2(Command and Control)サーバの通信先として、DGA(Domain Generate Algorithm)と呼ばれる手法で機械的に生成したドメインを用いる場合があります。DGAで生成されるドメインは、アルゴリズムの種類や動作する際のパラメータによって大きく異なります。そのため、あらかじめマルウェアの通信先をブラックリストとして管理することや、検知するシグネチャを表現することが困難な場合があります。

*11 IIJ-SECT Security Diary、「IoT機器を踏み台として利用するSYN/ACKリフレクション攻撃」(https://sect.iij.ad.jp/d/2019/02/128021.html)。
*12 wizSafe、「wizSafe Security Signal 2018年9月 観測レポート」(https://wizsafe.iij.ad.jp/2018/10/470/)。
*13 wizSafe、「wizSafe Security Signal 2017年11月 観測レポート」(https://wizsafe.iij.ad.jp/2017/12/184/)。

そこで、このプロジェクトでは情報分析基盤のDNSクエリデータと機械学習を組み合わせることで、その問題の解決を目指しています。人間によるルール化が難しいタスクであっても、機械学習を用いることで解決できる場合があるためです。機械学習のアルゴリズムには、識別に有効な特徴量を含むデータを与えることで、異常データを分類する能力を自律的に獲得できるという、望ましい特性があります。例えば、前号のIIR Vol.41でURL文字列に着目してニューラルネットワークで詐称サイトを識別する取り組みについて紹介しました[*14]。

DGAを機械学習で検出する試みは、既に実用化されているものも含めて、いくつかの研究が知られています。現在は、その中でもUSENIX Security '18で発表されたFANCI(Feature-based Automated NXDomain Classification and Intelligence)[*15]という手法の追試に着手しています。この手法について書かれた論文では、先行研究の知見などを基にしたドメインの特徴量とランダムフォレストと呼ばれる機械学習のアルゴリズムを組み合わせることで、高い汎化性能が得られるとされています。

追試において目指している最初のステップは、論文の手法はなるべくそのままで、データとして情報分析基盤から得られたDNSクエリデータを用いるというものです。このステップでは、情報分析基盤のデータに対して手法がそのまま適用できるのかを見極めます。なぜなら、使用するデータが異なれば、同じ手法を用いても得られる結果が同じになるとは限らないからです。その上で、適用できないと判断した場合には原因の調査と解消を、できると判断した場合には更なる性能向上を含む実用化に向けた検討を進める予定です。性能向上の余地については、例えば近年盛んに用いられる勾配ブースティング決定木(Gradient Boosting Decision Tree)の適用や、アンダーサンプリング(Undersampling)とバギング(Bagging)を組み合わせたアンサンブル学習(Ensemble Learning)の応用が挙げられます。

ただし、情報分析基盤で処理するデータの流量が多いことから、現実的な問題として、モデルには高いスループットも求められます。モデルやワークフローの複雑化による計算量の増大と、得られる性能向上の間でバランスを取りながら、様々なチューニングを通して、最終的には異常検知の仕組みの1つとして、情報分析基盤に組み込むことを目指しています。

1.3.2 Webプロキシデータへの適用

もう1つのプロジェクトでは、Webプロキシのデータからマルウェアが発生させるC2サーバへの通信の検知を目指しています。これについては、Black Hat Europe 2018でIIJエンジニアが発表した内容[*16]を情報分析基盤のログへ適用するため、現在、検証を重ねています。なお、Black Hat Europe 2018で発表した内容については、後述する「フォーカス・リサーチ(1)〜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」で解説しています。

*14 IIJ、技術レポート「Internet Infrastructure Review(IIR)」(https://www.iij.ad.jp/dev/report/iir/041/02.html)。
*15 USENIX、"FANCI : Feature-based Automated NXDomain Classification and Intelligence"(https://www.usenix.org/conference/usenixsecurity18/presentation/schuppen)。
*16 Black Hat、"Deep Impact:Recognizing Unknown Malicious Activities from Zero Knowledge"(https://www.blackhat.com/eu-18/briefings/schedule/#deep-impact-recognizing-unknown-malicious-activities-from-zero-knowledge-12276)。

プロジェクトでは、Convolutional Neural Networkという画像認識でよく利用されるディープラーニングの一種を用いて、正常通信と異常通信（C2通信）の傾向を判断します。ここで重要となるのが、学習モデルの性能とその評価です。

仮に、機械学習のモデルとして95%以上の精度（Accuracy）を発揮できるモデルがあるとしたら、一般的に性能が良いと感じられます。しかし、情報分析基盤に収集されるログの量は膨大であるため、1%に対する量は人間が目で見て処理できる量ではありません。誤検知が発生したとしても、運用上、耐えうるだけの精度を提示しなければなりません。もちろん、これは機械学習だけで実現する場合であり、機械学習以外のシステム的な処理で誤検知数を減らすアプローチも考えられます。

また、精度の他に、扱うデータセットの分布の違いにも注意しなければなりません。各種カンファレンスや学会などで発表されている優秀とされている機械学習モデルで扱うデータセットの分布と、SOCで扱うデータセットの分布の特性が異なる可能性があるため、追試をする必要があります。

以上のことから、SOCでは、機械学習モデルの追試や精度向上をはじめ、機械学習モデルを利用するシステムの全体設計や運用を考え、現状のセキュリティ業務に負荷を与えずに品質を向上させるシステムの構築に注力しています。

精度向上の試みとしては、特徴量エンジニアリングを実施しています。特徴量エンジニアリングというと、あらかじめデータ分析に基づいて有効と考えられるものを追加していくイメージが強いですが、それ以外のアプローチもあります。例えば、既にある特徴量に対して様々な統計量を計算し、その元となったデータと結合したものを学習・評価に用いるというものです。この他にも様々な手法を用いて、特徴量の追加と評価を繰り返しながら、モデルの性能を向上させていきます。

1.4 おわりに

今回は、情報分析基盤を活用した分析の概要と、2018年の具体的な観測事例、そして機械学習に対する取り組みについて紹介しました。最後に紹介した機械学習を用いたアプローチは、従来の手法では検知が困難もしくは限界があった脅威に対して、現在よりも更に検知範囲を拡大できる可能性を秘めています。これらの取り組みが実現できているのは、ひとえにお客様から頂戴した通信に関するログを同意に基づいて情報分析基盤で活用できているためです。機械学習によるアプローチは特に大量のデータを必要とすることから、情報分析基盤を通して通信ログが活用できてこその取り組みといえます。今後も、wizSafe Security SignalやIIJ SECTブログを通して脅威情報をタイムリーに発信すると共に、お客様がより安全にインターネットを利用できる社会の実現に向けて邁進します。

執筆者：
小林 智史（こばやし さとし）
IIJ セキュリティ本部 セキュリティビジネス推進部 セキュリティオペレーションセンター データアナリスト。

執筆者：
守田 瞬（もりた しゅん）
IIJ セキュリティ本部 セキュリティビジネス推進部 セキュリティオペレーションセンター データアナリスト。

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。

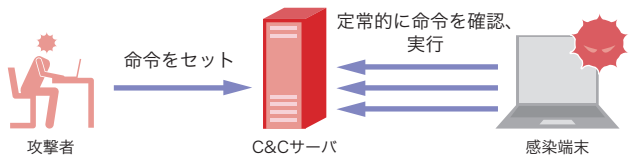


図-1 BOTやRATによるC&Cサーバへの定常的な通信

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。

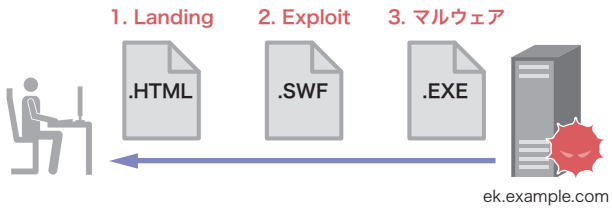


図-10 Exploit Kitサーバから送信されるコンテンツタイプの遷移

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

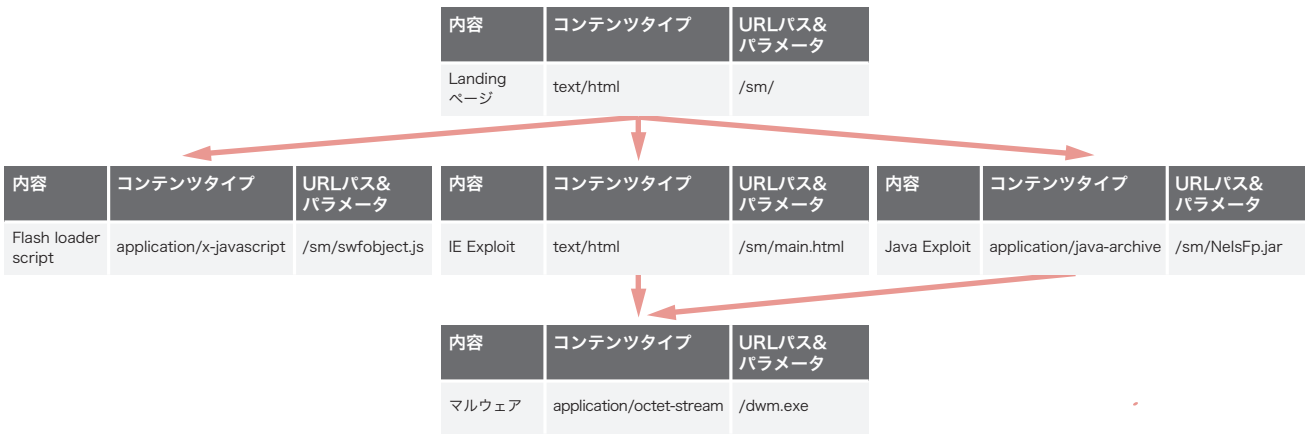


図-12 KaiXin Exploit Kitのコンテンツタイプ遷移の例

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

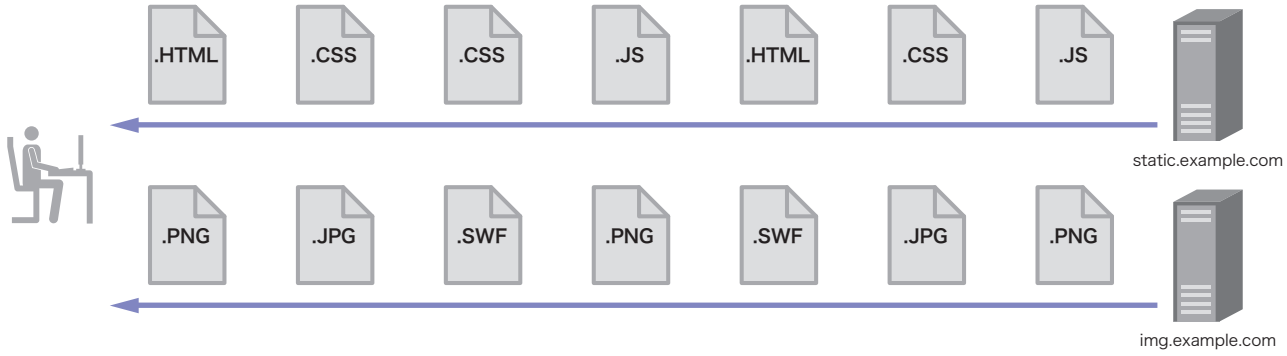


図-13 コンテンツタイプに応じたサーバを備えたWebサービス利用時のコンテンツタイプ遷移の例

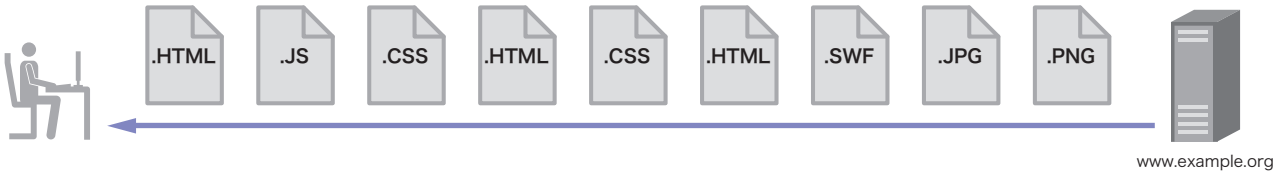


図-14 単一のWebサーバでホストされるサービス利用時のコンテンツタイプ遷移の例

*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

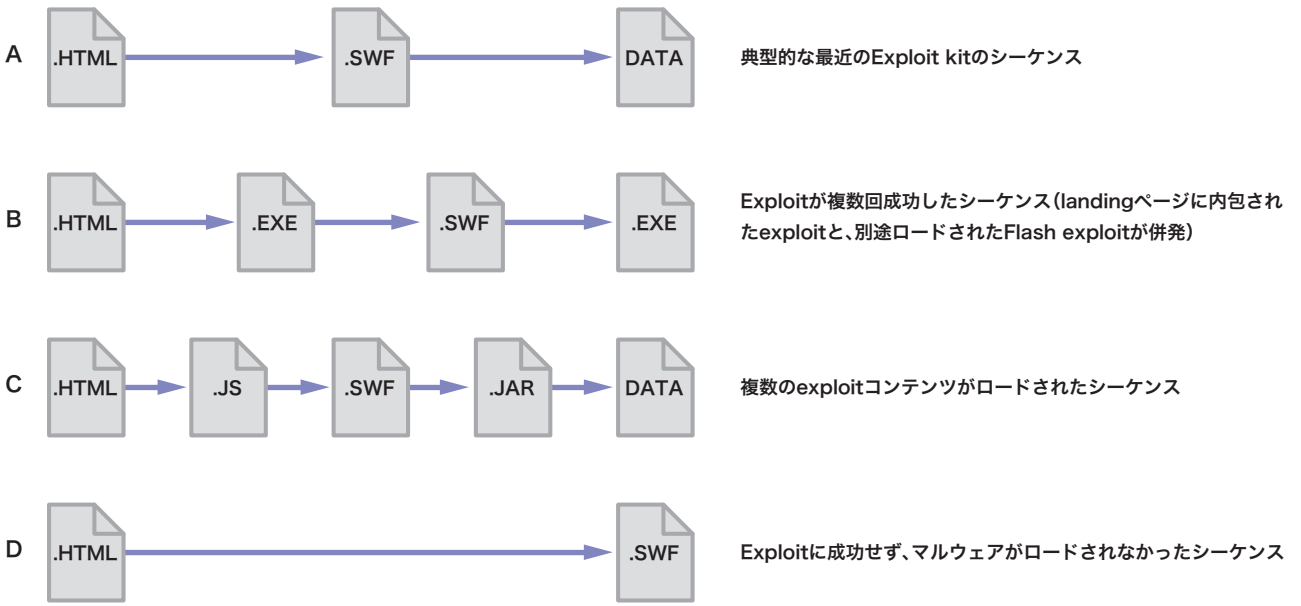


図-15 生成した擬似Exploit Kitシーケンスの例

*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。

大規模メールシステムの設計

3.1 はじめに

2006年10月に提供を開始した「IIJセキュアMXサービス(以下SMX)」のシステムを、2017年4月に全面リニューアルしました。SMXは「メールゲートウェイ」と分類される機能を中心としたサービスです。お客様のメールシステムがメールを受け取る前に、一旦IIJのメールサーバでメールを受け取り、ウイルスフィルタ・迷惑メールフィルタ・送信ドメイン認証フィルタ・バックスキャッターフィルタ・サンドボックスフィルタなど、様々な技術で脅威メールを防ぎ、安全なメールのみをお客様のメールシステムにお届けします。

サービス提供開始から10年以上が経過し、その間にメールの通数やサイズなどの流量にはじまり、サーバのスペックからメールシステムの要件に至るまで、メールサービスを取り巻く環境は大きく変わりました。SMXのシステムもそのような変化に合わせて拡張に拡張を重ねてきたため、サービス開始当初とは大きく異なるシステム構成となっていました。しかし、アーキテクチャの根本はサービス開始当初から引き継いでいるものも多く、長い間システムに限界を感じていました。

そこで今回のリニューアルでは、システムをアーキテクチャから全面的に見直しました。本稿では、SMXの新しいメールシステムの特に配送系の設計について、見直しの経緯と共に紹介します。

3.2 リニューアルに向けた課題と目標

リニューアルにあたり、まず、古い配送系の課題を踏まえていくつかの目標を定めました。

3.2.1 アーキテクチャの見直し

1点目は、無理に拡張を重ねた古いアーキテクチャの見直しです。古典的な大規模メールシステムでは、図-1のように単機能のメールサーバ(Message Transfer Agent、以下MTA)を直列に並べて配送系を構成するアーキテクチャが一般的であり、リニューアル前のSMXの配送系もこれに類するアーキテクチャを採用していました。

このような、多段MTA構成のアーキテクチャの最大のメリットは拡張性の高さです。既存の配送系に対して、追加したい機能を持つMTAを連結することで、手軽に配送系を拡張できます。また、MTA間はメール配送の標準プロトコルであるSMTPで接続するため、異なるベンダーの製品を組み合わせる際もインタフェースの互換性を心配する必要がなく、製品の組み合わせが原因となるトラブルは起きにくい傾向にあります。

ただし、この拡張方法には副作用もあるため、過剰な適用は禁物です。最も懸念すべき副作用は、MTAの段数の増加に伴うストレージI/Oの増加と運用コストの増加です。多段MTA構成の配送系では、メールがMTAを通過するたびに受け取ったデー

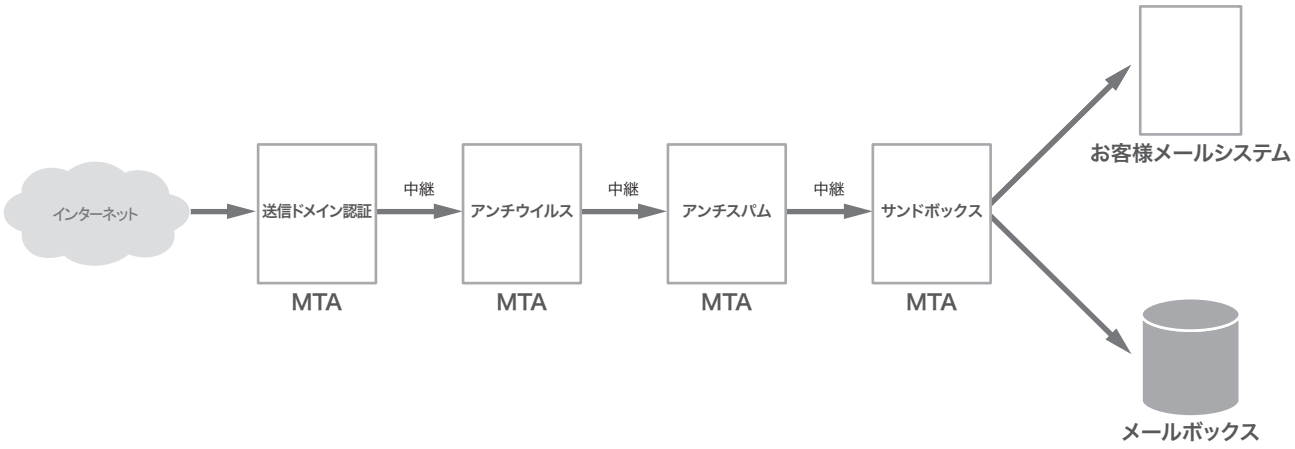


図-1 古典的なメールシステムのアーキテクチャ

タをストレージに書き出すため、メールサイズの何倍ものストレージI/Oが発生します。ほぼ同じ内容のメールが何度も書き出されるため、配送系全体として見ると無駄なストレージI/Oの多いアーキテクチャでもあります。NASやSANなどのネットワークストレージを用いる構成の場合、これらのI/Oだけでストレージネットワークを逼迫させてしまい、配送系全体のボトルネックになっていきます。従来のSMXの配送系でもストレージがボトルネックであった上に、CPUやメモリは遊んでしまい、高速化・大容量化が進むハードウェアの恩恵を十分に生かしきれていない状況でした。

更に、異なるMTAを直列に並べることは、運用方法の異なる様々なMTAが配送系に混在することを意味します。ログの見方・操作方法・障害対応手順・配送系増強の際の構築手順など、必要な知識が増えるだけでなく、それぞれのMTAが配送経路の分岐や通知メールの送信などを行うため、配送系内におけるメールの流れも複雑になり、配送系全体を把握することが困難になっていきます。

SMXでは、規模の拡大と共に配送系の拡張を繰り返したため、複雑で高コストな配送系になっており、これ以上の拡張は難しい状態でした。

3.2.2 フィルタ機能の向上

2点目の目標は、ウイルスフィルタ(アンチウイルス)及び迷惑メールフィルタ(アンチスパム)機能の精度向上です。メールセキュリティサービスにとって、アンチウイルス及びアンチスパムは双璧とも言うべき重要な機能です。

アンチウイルス、アンチスパム共に、数多くのセキュリティベンダーがサービスや製品を提供しているのですが、セキュリティ業界は変化が激しく、次々に新しい攻撃手法が生まれては、それに対抗する新しい技術の開発が日々行われています。そのため、ベンダーAの製品の検知精度が高い日もあれば、ベンダーBのサービスの検知精度が高い日もあり、ベンダーCが素晴らしい検知精度の新製品をリリースし、状況が一変することもあります。反対に、特定のベンダーのエンジンを一途に使い続けていると、そのエンジンが採用している技術が陳腐化し、検知精度が下がってしまうリスクがあるということでもあります。このような状況の中で、ウイルスメールや迷惑メールに対し、高い検知精度を維持するための仕組みの必要性を感じていました。

3.2.3 過度なベンダー依存の回避

3点目の目標は、特定ベンダーへの過度な依存の回避です。SMXは幅広い機能を提供するために、ベンダーの製品やサービスをシステムに数多く組み込んでいます。ベンダーの提供する製品やサービスは魅力的なものも多いですが、いわゆる「ベンダーロックイン」と呼ばれる状態にならないよう、依存度をコントロールする必要を感じていました。

特に海外のベンダーに多いのですが、ある日突然、競合他社に会社を丸ごと買収されて、提供中の製品やサービスが終了することも決して珍しくありません。組み込んでいる製品が急に使えなくなると、組み込んでいる側のインパクトは決して小さくはありません。インパクトをゼロにはできなくとも、最小限に抑えるための対策が必要でした。

3.3 単段MTA構成によるハードウエアリソースの有効利用

以上のような目標に沿って配送系をリニューアルするにあたり、まず、コモディティサーバの高性能化によって余裕のできていたCPUやメモリを活用して、システム全体のボトルネックであり、コスト要因にもなっているストレージI/Oの削減を目指すこととしました。

辿り着いたアーキテクチャは、従来とは正反対のものになりました。つまり、MTAは一段のみで無駄な中継はせず、単一の

高機能なMTAの中ですべての処理を完結させます(図-2)。このアーキテクチャでは、メールをストレージに書き出す処理は1回のみになるため、同じ内容を何度も繰り返し書き出していた従来の配送系に比べ、ストレージI/Oを大幅に削減することが可能です。

また、従来の配送系では、ストレージI/Oがボトルネックになり大半のサーバのCPUリソースが遊んでいる一方、一部のサーバでのみ、CPUに負担のかかるウイルススキャンなどによってCPUの稼働率が高くなっていました。均一な構成のMTAを並列に配置することで、これまで遊んでいたCPUリソースを負担のかかる処理に回せるようになり、配送系全体におけるCPUリソースの利用効率の向上も見込めます(図-3)。

SMXではアンチウイルス、アンチスパム共に複数のエンジンを採用しているため、単段MTA構成のアーキテクチャを実現するためには、1つのサーバ内でいくつものエンジンをメモリにロードする必要があります。一般に、アンチウイルスエンジンやアンチスパムエンジンは大量のデータをメモリに保持するため、大量のメモリを消費する傾向にあります。そのエンジンをいくつもロードするため、合計するとひと昔前のサーバでは収容できない量のメモリが必要になるのですが、前述のコモディティサーバの高性能化により、このような構成を取ることが可能になりました。

3.4 アンチウイルス／アンチスパムエンジンを交換可能に

次に、アンチウイルスエンジン及びアンチスパムエンジンは「いつでも入れ替えられる」ようにシステム全体を設計しました。

IIJでは、アンチウイルスエンジンやアンチスパムエンジンを自社開発しておらず、セキュリティベンダーの提供するエンジンを配送系に組み込んでこれらの機能を提供しています。そのため、検知精度に問題があっても自社で直接対応することはできません。しかし、逆の視点で捉え、陳腐化した技術を切り離し、旬な技術をタイムリーに組み込めることを強みとして活かせるよう、特定のアンチウイルスエンジンやアンチスパムエンジンに密に結合しない設計を目指しました。

まず、セキュリティベンダー各社のアンチウイルス及びアンチスパムエンジンの評価を行いました。IIJの管理するハニーポットで受信したウイルスメールや迷惑メールに対して、ウイルススキャン及び迷惑メールスキャンを行い、数ヵ月間かけて検知性能の統計を取り、比較を行いました。ウイルスメールや迷惑メールには流行のようなものがあり、短期間の検証では、評価期間中にたまたま行われていたスパムキャンペーン*1に対する検知性能の影響が大きく影響し、長期的な視点で見た場合の検知性能を適切に評価できない恐れがあります。そのため、検証期間は長めに設定しました。検証に際して、様々なセキュリ

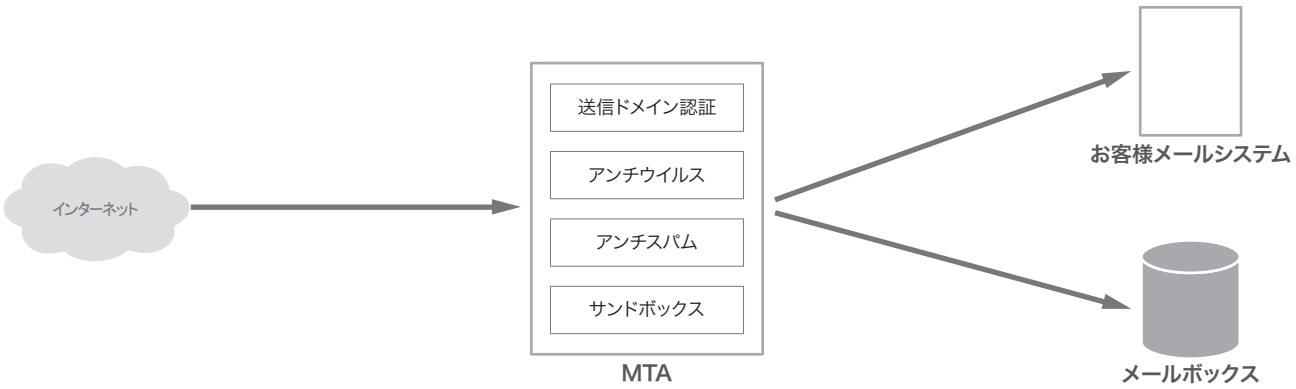


図-2 リニューアル後のメールシステムのアーキテクチャ

＊1　同一あるいは似通った迷惑メールの大量送信行為。

ティベンダーから製品説明を受けたのですが、特にアンチスパムエンジンについては製品によってアプローチに特色があり、検証結果もそのアプローチの差を反映した大変興味深いものでした。手間のかかる検証でしたが、おかげで各エンジンの検知精度や傾向を把握することができました。

また、更なる検知精度向上のため、アンチウイルスエンジン及びアンチスパムエンジンはそれぞれ複数組み込み、それぞれの処理結果を取りまとめたものを最終結果として扱うこととしました。

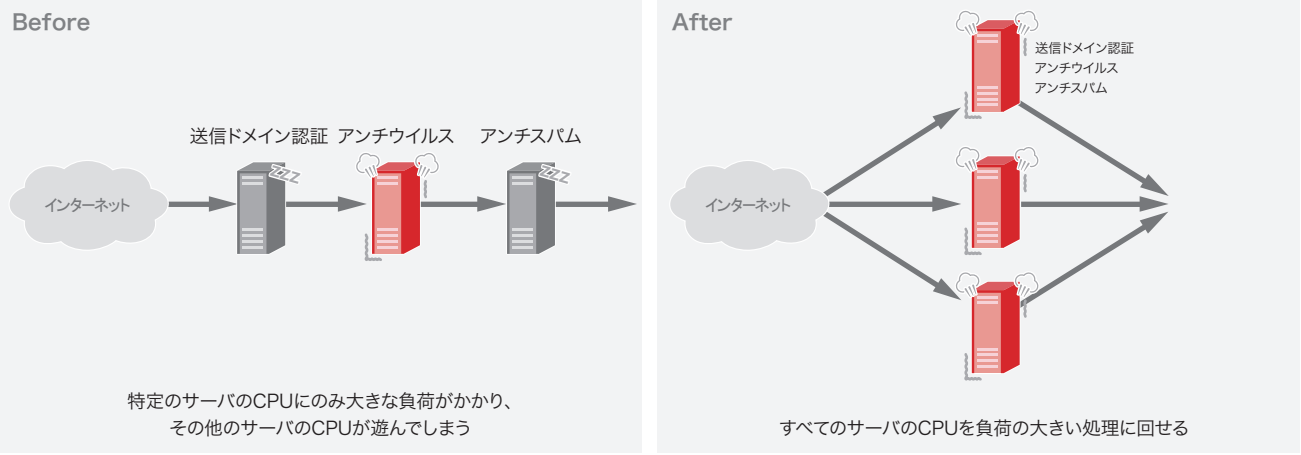
検証の結果から、そして普段運用する中の肌感覚からも言えると思うのですが、ウイルスメールや迷惑メールが撒かれ始めてからエンジンが検知するようになるまでの時間は、いずれかのエンジンが飛び抜けて優秀というものではありません。あるキャンペーンではエンジンAの検知が早く、別のキャンペーンではエンジンBの検知が早い、といった具合です。複数のエンジンを並べることにより、キャンペーンの初期段階のすり抜けを減らすことができるようになります。特にアンチスパムエンジンについては、あるエンジンの弱点を他のエンジンで補えるように、異なるアプローチのエンジンを組み合わせています。

複数のエンジンを組み合わせたのは、主に検知精度の向上を狙ったものでしたが、副次的な効果もありました。1つはスキャンエラーの削減です。ウイルスメールや迷惑メールでは、メールのヘッダや添付ファイルが破損していたり、意図的な細工が施されているために、スキャンを正常に完了できないケースも珍しくありません。複数のエンジンでスキャンすることにより、全くスキャンできないメールの数を大幅に抑えられるようになりました。また、ごくまれに特定のメールや添付ファイルをスキャンすると、アンチウイルスエンジンやアンチスパムエンジンがクラッシュしてしまうケースもあるのですが、そういった場合の緊急対応として、メールの疎通を優先し、問題の起きているエンジンを切り離すという選択肢を持つことができます。更に、セキュリティベンダーが会社ごと買収されて製品が使えなくなるような場合にも、インパクトを最小限に抑えることができます。

3.5 MTAの自社開発を決断

配送系のリニューアルにあたっての最大の問題は、この設計をどう実装するかでした。結果的に、MTAを自社開発することにしたのですが、他にも、Postfixやsendmailに代表されるオープンソースのMTAを組み合わせる方法や、MTAベンダー製のMTAを採用する方法がありました。

Postfixやsendmailでは、milterという高度な機能を実現するためのインタフェースが提供されており、手軽かつ安全にメールの制御や書き換えを実装可能です。その反面、milterによる



多段MTAアーキテクチャ　　単段MTAアーキテクチャ

図-3 ハードウェアリソースの有効活用

拡張では、アーキテクチャ上、特にI/Oのオーバーヘッドが大きくなってしまいます。また、機能面でもSMXのような複雑なシステムを実現するには機能不足であると言わざるを得ません。Postfixやsendmail本体を直接改造するアイデアもあるのですが、本家のアップデートに追従し続けるコストは想像よりずっと大きなものです。

ベンダー製MTAを採用する選択肢はとても現実的でした。いくつかのMTAベンダーがISPや大規模メール送信事業者向けのMTA製品を開発しています。これらの製品でも、これまでに説明したような、一段ですべての処理を行うことをコンセプトにしたものが主流になっています。また、様々なセキュリティベンダーのアンチウイルスエンジンやアンチスパムエンジンを差し替えて組み込めるようになっていたり、大規模な配送系で必要とされる細かい要求を満たす、柔軟で極限まで細かなカスタマイズが可能になっているなど、ベンダー製MTAはオープンソースのMTAとは比べ物にならないほど多機能で強力です。我々にとっても目指していた配送系を実現するのに最も「手っ取り早い」選択肢でした。実際、多くのISPや大規模メール送信事業者ではベンダー製MTAを採用しており、SMXの従来の配送系でも部分的にではありますが、ベンダー製MTAを組み込んでいました。

ベンダー製MTAを導入することの唯一の、そして最大の懸念は、SMXのすべてがそのMTAと一蓮托生になる点です。単段MTA構成のアーキテクチャでは、MTAは配送系そのものです。また、配送系以外も、サービス仕様から運用手順に至るまで、システム全体がMTAに深く依存することになります。つまり、「MTAでできることがSMXでできること」になり、その逆も然りです。

IIJでは、お客様の声だけでなく、メール業界やセキュリティ業界の最新動向も詳しくウォッチしながら、積極的なSMXの機能拡張を続けています。そのため、ときには他の事業者では必要としないような機能が配送系に要求される場合もあります。このようなケースで、MTAが備えていない機能をベンダーに追加してもらうのは一般的に困難です。MTAベンダーも数多くの顧客を抱えていますので、多くの顧客に必要とされる機能や、彼らにとって重要な顧客が必要とする機能の開発が優先されるのは必然であり、独自性の強い機能やニッチな機能の追加の優先度は低くならざるを得ません。ベンダー製MTAは、要求する仕様が明確かつ将来の仕様変更が少ないケースには向いていると思いますが、SMXの積極的な拡張方針を支え切れるかどうかは未知数でした。

また、ベンダー製MTAにも会社買収のリスクがあります。MTAにはシステム全体が大きく依存しますので、買収が実際に起きた場合の影響はアンチウイルスエンジンやアンチスパムエンジンに比べると甚大です。実際、ここ数年間で、MTAベンダーや製品の買収が数件ありました。MTAを開発しているベンダーは絶対数がそれほど多くないため、割合にするとかなり高く、見過ごせないリスクです。

最後の自社開発という選択ですが、こちらも決して容易い選択肢ではありません。ISPレベルの膨大な流量を支えるMTAには、極めて高い安定性、堅牢性及びパフォーマンスが求められます。それに加えて、SMXの多様な機能と柔軟性を実現しなくてはなりません。更に、将来に渡る機能拡張を支える技術力も求められます。

そのようなMTAをもしゼロから開発するのであれば、自社開発という決断はとてもできなかったかもしれません。しかし、IIJではメールシステムのコンポーネントの多くを自社開発してきた経験とノウハウがありました。そして頼りになる開発チームが揃っていたため、リスクは高そうだが、得るものも多い自社開発を決断することができました。

3.6 リニューアルの成果

3.6.1 開発目標の達成

MTAをはじめとしたシステム全体のリニューアルプロジェクトは最初のリリースを迎えるまでに丸1年以上を要し、筆者が経験した中でも最も大規模な開発プロジェクトとなりました。

長い開発期間と度重なるテストの末に完成したシステムでは、あらかじめ定めた目標は軒並み達成できました。柔軟かつ多機能なMTAによって、単段MTA構成アーキテクチャの配送系を実現することができました。従来の配送系で課題となっていたストレージI/Oは設計のとおり大幅に削減され、配送系全体のパフォーマンスも向上しました。ウイルスフィルタや迷惑メールフィルタは、大幅に見直したことで検知精度が一段と向上しました。また、それぞれのエンジンの検知率を継続的にウォッチすることで、検知率に変化があった場合にすぐにアクションを取れるようにしました。

このようにMTAを自社開発したことで、MTAベンダーが買収されるリスクから解放され、エンジンの入れ替えを可能にすることでセキュリティベンダーが買収された際の影響も最小限にとどめられるようにしました。今回のリニューアルで最も重要だったのは、すごく当たり前のようですが、IIJがベンダーの買収劇にふりまわされることなく、自身でサービスを主体的に動かすことができる基盤にできたことだと考えています。

3.6.2 副次的なメリット

当初から定めていた目標以外にも、いくつかの副次的なメリットがありました。

まず、不具合対応のスピードが改善されました。ベンダー製MTAの場合、不具合が発生するとその発生条件を洗い出してベンダーに報告し、修正を依頼するのですが、再現条件が不明だったり、再現条件にお客様の情報が含まれているためベンダーに渡すことができなかったりするため、不具合を確認してもらうまでに時間を要したり、そもそも不具合の確認ができなかったりすることもしばしばです。一方で、自社開発の場合は、運用チームと開発チームが緊密に連携できるため、特に不具合の原因の特定が圧倒的に早く、暫定対応・恒久対応ともに素早く適切に進めることができます。

また、直接比較することはできないのですが、結果的に自社開発の方が、チームが高いモチベーションを保った状態で開発を進められたのではないかと感じています。MTAまたはその他にベンダーの製品を組み込んだシステムを開発する場合、開発チームではベンダーの製品とのインタフェースの開発を行うのですが、製品の不明瞭な仕様と格闘する不毛な作業が多く、個人的にあまり楽しくない作業であることがしばしばです。一方、自社開発では作業量が圧倒的に多く、それが開発チームの負担になることが大きな懸念でした。しかし、忙しくこそあったものの、決してチームが疲弊していくという雰囲気ではなかったように感じています。やはり、自分たちの手で大規模なシステムを作り上げ、それが徐々に動く面白さがあるのではないかと思っています。

新システムは2017年4月にリリースし、丸1年かけて旧システムから移行しました。リリース後も様々な意見や要望をいただき、機能の追加や不具合の修正を行いましたが、矢継ぎ早の改修を実現できたのも、システムを自社開発したからこそ達成できたものと思っています。

補足をすると、ベンダーの製品を全面的に活用したシステムを否定するつもりは全くありません。どちらにもメリットとデメリットがあるため、状況に応じて適切なバランスを選択することが必要だと考えています。SMXでも海外を含む数多くのベンダー製品をシステムに組み込んでおり、日ごろからベンダー各社と緊密に連携を図りながらサービスを提供していることを最後に付け加えておきます。

今回は、SMXの配送系の設計について紹介しました。SMXは新しく手に入れたアーキテクチャを活かして、これからも進化を続ける所存です。

執筆者:
鈴木 高彦（すずき たかひこ）
IIJ ネットワーク本部 アプリケーションサービス部 サービス開発課 シニアエンジニア。
2004年IIJ入社。以降、一貫してメールサービスの開発に従事。
オープンソース送信ドメイン認証フィルタプログラム yenma の開発者。

March 2019 Vol.42

IIJ
Internet Initiative Japan

株式会社インターネットイニシアティブ(IIJ)について

IIJは、1992年、インターネットの研究開発活動に関わっていた技術者が中心となり、日本でインターネットを本格的に普及させようという構想を持って設立されました。
現在は、国内最大級のインターネットバックボーンを運用し、インターネットの基盤を担うと共に、官公庁や金融機関をはじめとしたハイエンドのビジネスユーザに、インターネット接続やシステムインテグレーション、アウトソーシングサービスなど、高品質なシステム環境をトータルに提供しています。
また、サービス開発やインターネットバックボーンの運用を通して蓄積した知見を積極的に発信し、社会基盤としてのインターネットの発展に尽力しています。

本冊子の情報は2019年3月時点のものです。

IIJ-MKTG019-0042

株式会社インターネットイニシアティブ
〒102−0071 東京都千代田区富士見2-10-2　飯田橋グラン・ブルーム
E-mail：info@iij.ad.jp　URL：https://www.iij.ad.jp