情シス担当のためのセキュリティコンテスト

MNCTF2016　解答例

マクニカネットワークス株式会社
セキュリティ研究センター
凌翔太

このたびは MNCTF2016 をチャレンジいただき、誠にありがとうございました。

本書について

本書は 2016 年 7 月 8 日に行われた MNCTF2016 で出題された問題の回答例を記したものです。今からでも、どなたでもチャレンジしていただけますので、まだ実施していない方は本書を読み進める前にチャレンジすることを強くお勧めいたします。

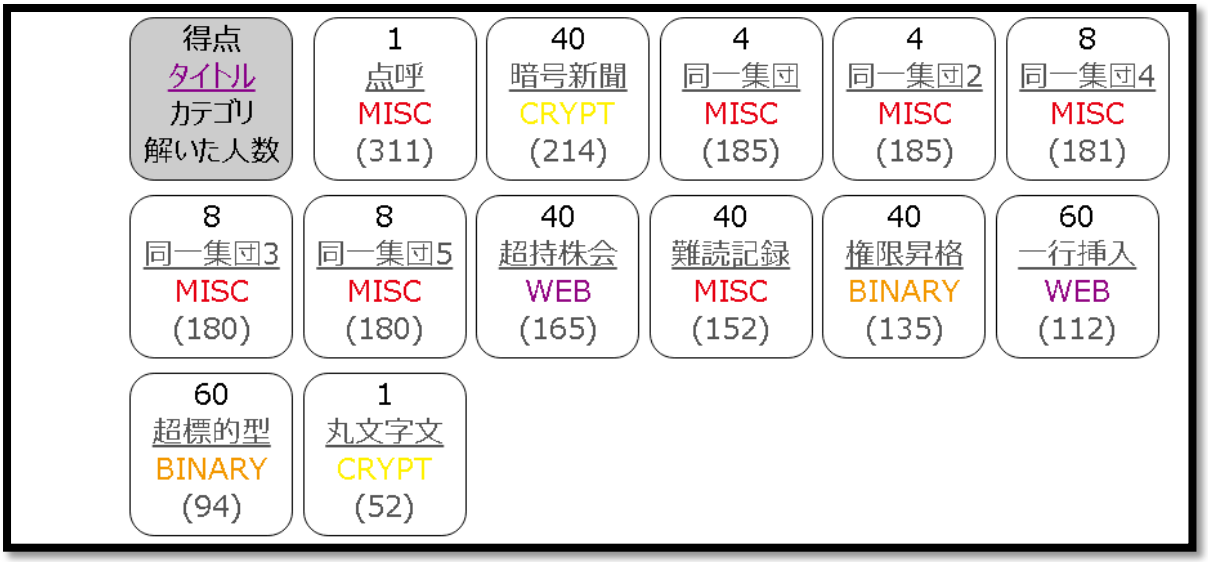
http://mnctf.info/mnctf2016/

一部の手順に攻撃手法を記載していますが、下記のネットワーク**以外**に対して、それらの手法を用いないでください。

1. MNCTF の問題サーバ（http://mnctf.info）
2. 1.に明示的に記載されている攻撃対象のネットワーク
3. 読者の管理対象のネットワーク

目次

問題一覧

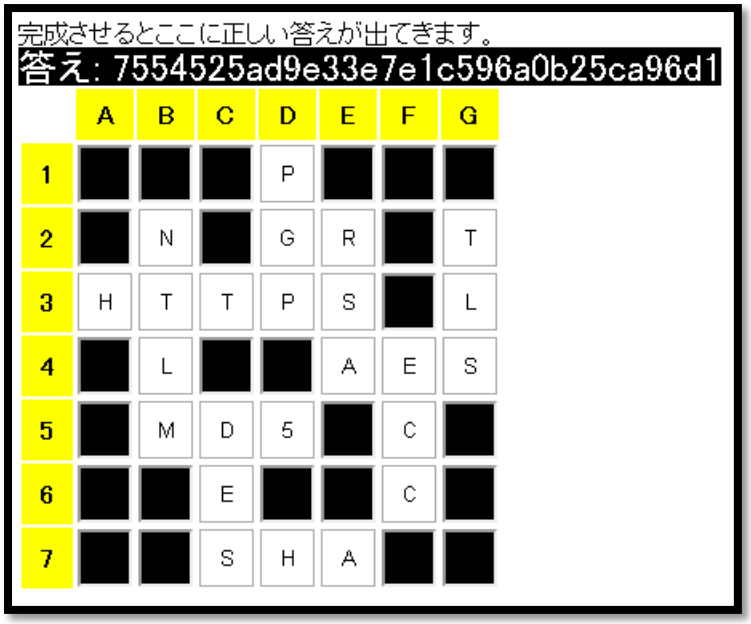


1. 点呼

問題文を読むだけの練習問題。

答え:MNCTF2016

暗号化に関するクロスワード。Web で検索したりして、すべてのキーワードを埋めると上部に答えが表示される。



答え: 7554525ad9e33e7e1c596a0b25ca96d1

出題者からのコメント：業務にもよりますが、細かい暗号アルゴリズムそのものは把握しなくても、それぞれの強度・特性を理解し、状況に応じて最適なものが選択できるようにするとよいでしょう。

与えられたドメイン名（shinobot.com）から、同じメールアドレスで取得された他のドメインを見つける問題。まずは、WHOIS を利用し、ドメイン取得者の情報を得る。

```
Registry Registrant ID:
Registrant Name: Shota Shinogi
Registrant Organization: Shota Shinogi
Registrant Street: 1-5-5 Teshigawara
Registrant City: Takeshi
Registrant State/Province: NA
Registrant Postal Code: 2220000
Registrant Country: JP
Registrant Phone: +81.100100100
Registrant Phone Ext:
Registrant Fax:
Registrant Fax Ext:
Registrant Email:ryo_no_ryo-master@yahoo.co.jp
```

図、shinobot.com の WHOIS の抜粋

これで、取得した人のメールアドレスが手に入る。

「ryo_no_ryo-master@yahoo.co.jp」

今度は以下のような Reverse WHOIS を提供しているサイトを利用して、他のドメインを検索する。

https://www.passivetotal.org/
https://whoisology.com/
http://www.whoismind.com/
https://www.threatcrowd.org/
※いずれも無料で利用可能。

これらのサイトはドメインとその WHOIS のデータベースを持っており、通常 WHOIS ではできない逆引きができる。ただし、サイトによって、データを蓄積するタイミングが異なったり、情報量にばらつきがあったりするため、それぞれで試して、網羅性を高める必要がある。

答えは５つある。

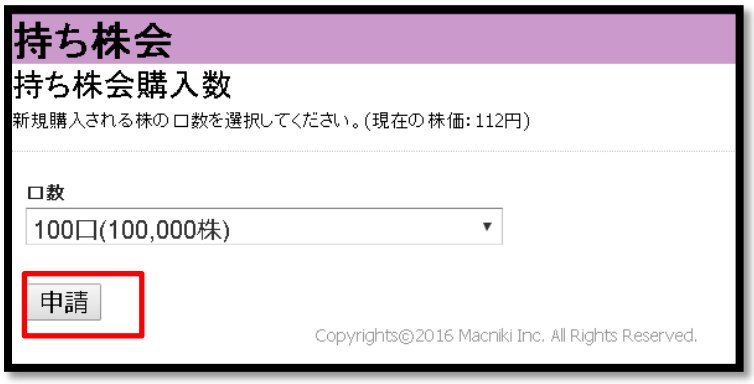
答え 1：mnctf.info
答え 2：mnd2015.info
答え 3：shinosec.com
答え 4：noitalumis.info
答え 5：shinolocker.com

出題者からのコメント：マルウェアから通信先を調べ、同じ攻撃者が持っている他のドメインを調べ、それらのドメインをあらかじめブロックする事によって、今後来る攻撃を予防することができます。ただし、マルウェアの通信先が侵害された正規の Web サーバである場合もあるため、ドメインの正当性を確認する必要があります。

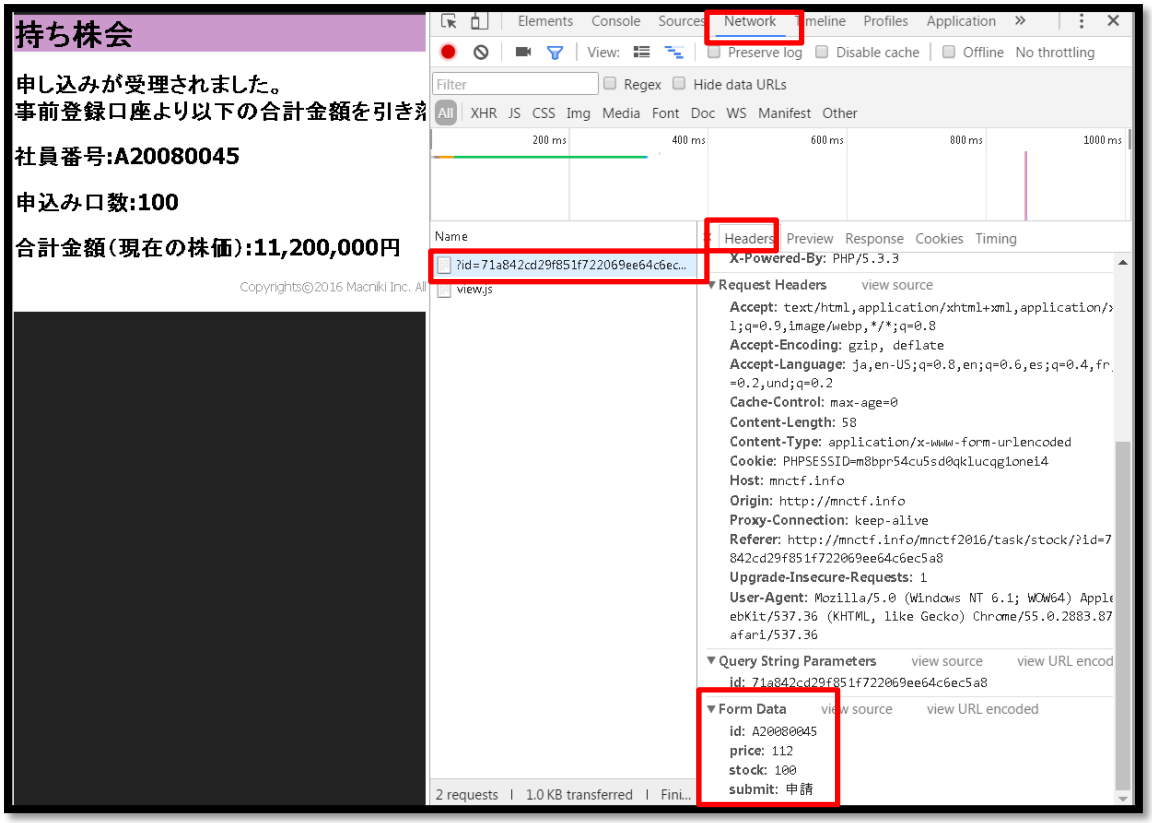
4. 超持株会

Web フォームのパラメータを改ざんする問題。

ブラウザの開発者（デベロッパー）ツールの機能のみで解くことができる。Google Chrome を用いて解説する。デベロッパーツール[F12]を起動し、まずは与えられたページで[申請]ボタンを押してみる。

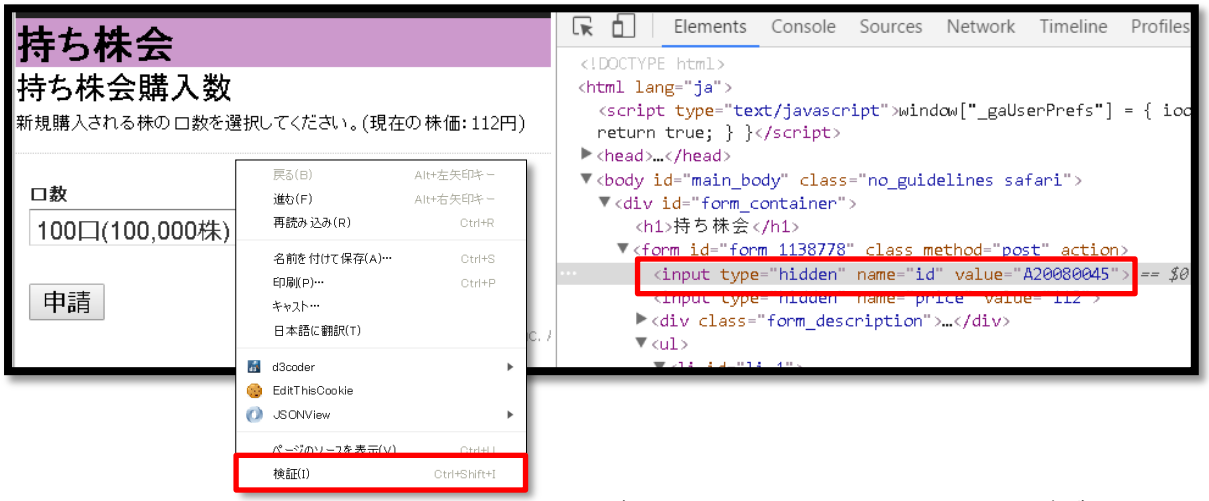


ページが表示されたら、デベロッパーツールの[Network]タブを開き、「?id=」から始まるファイルを選択し、[Headers]タブをクリックし、[Form Data]欄に着目する。



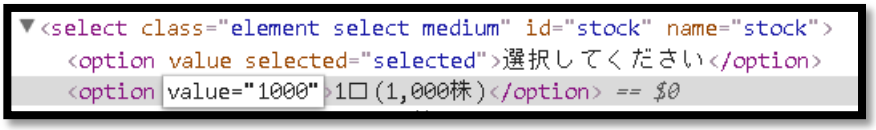
[id]および[stock]パラメータが改ざんの対象となる。入力ページに戻り、これらのパラメータがどこで生成されているかを確認する。

フォーム部分を右クリックし、[検証]をクリックすると、HTML が表示される。

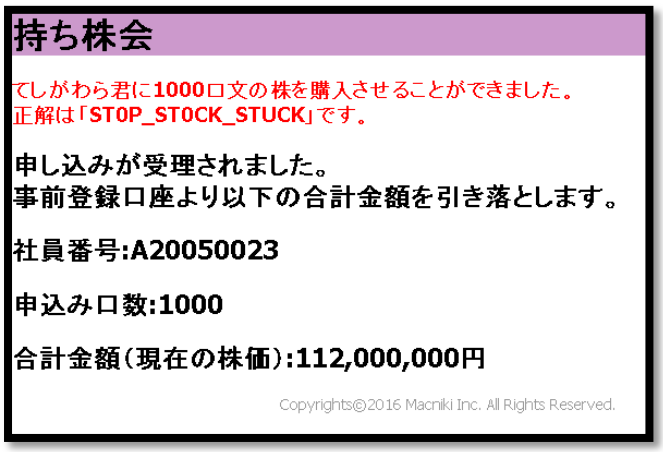


右側の赤枠の通り、id パラメータは hidden タグで指定されている。value 欄をダブルクリックし、書き換える。

同様に stock パラメータは select/option タグで指定されているので、value 欄をダブルクリックし、書き換える。



この状態で 1 口を選択し、[申請]ボタンをクリックする。



答え: ST0P_ST0CK_STUCK

出題者からのコメント: パラメータ改ざん攻撃の典型的な例です。通常は Fiddler、BurpProxy、OWASP ZAP など脆弱性診断の専門家も使うようなツールで解きますが、本書ではあえて、ブラウザの機能のみで解く方法を紹介しました。ちなみにこの問題を作った背景にはとある企業の社内システム（外注）で当脆弱性があり、それを利用した出題者が人事部にこっぴどく怒られたという逸話があるとかないとか。

該当文字列を抽出するための正規表現を書く問題。

抽出すべき文字列の条件：

・先頭は「AD¥」から始まる（ドメイン名）

・そのあとに 16 進数が 6 文字続く

・英字はすべて大文字

AD¥¥[0-9A-F]{6}

ただし、これでは、ログ内に含まれるコンピュータ名でない「AD¥AABBCCDD」が引っかかってしまうため、これを除外しなければならない。ログ内のコンピュータ名を観察すると、後ろに必ずスペースを含んでおり、さらに抽出時にスペースはトリムされると記載されていることから、スペースを指定する

AD¥¥[0-9A-F]{6}¥s

正規表現	意味
AD	そのまま「AD」にマッチする
¥¥	「¥」は正規表現では特殊な意味を持つため、「¥¥」と記載することでリテラルの「¥」とマッチさせることができる。
[0-9A-F]	0〜9、A〜F の文字にマッチする。
{6}	前のグループ（[0-9A-F]）を 6 回繰り返す。
¥s	半角スペース、タブなどの空白文字にマッチする

答え：R3G3X_M4573R

出題者からのコメント：ログから特定の文字列を抽出するというニーズは高く、その際正規表現を用いることは珍しくありません。セキュリティエンジニアであれば、正規表現を書くスキルは必須と言えるでしょう。

脆弱性を突いて、権限昇格をする実行ファイルから突く脆弱性を調べる問題。

ZIP を解答すると実行ファイル（.exe）が解凍されるので、strings で文字列を抽出する。

```
ZnVuY3Rpb24gSW52b2tlLU1TMTYtMDMyIHsNCjwjDQouU1lOT1BTSVMNCiAgICANCiAgICBQb3dlclNoZWxsIGltcGxlbWVudGF0aW9uIG9mIE1TMTYtMDMyLiBUaGUgZXhwbG9pdCB0YXJnZXRzIGFsbCB
```

Base64 のような長い文字列が表示されるので、デコードをすると、以下が表示される。

```
function Invoke-MS16-032 {
<#
.SYNOPSIS
      PowerShell implementation of MS16-032. The exploit targets all
```

Web で検索すると、MS16-032 の脆弱性を突く PowerShell スクリプトだと判明する。つまり、脆弱性を突く PowerShell スクリプトを生成する実行ファイルであることがわかる。

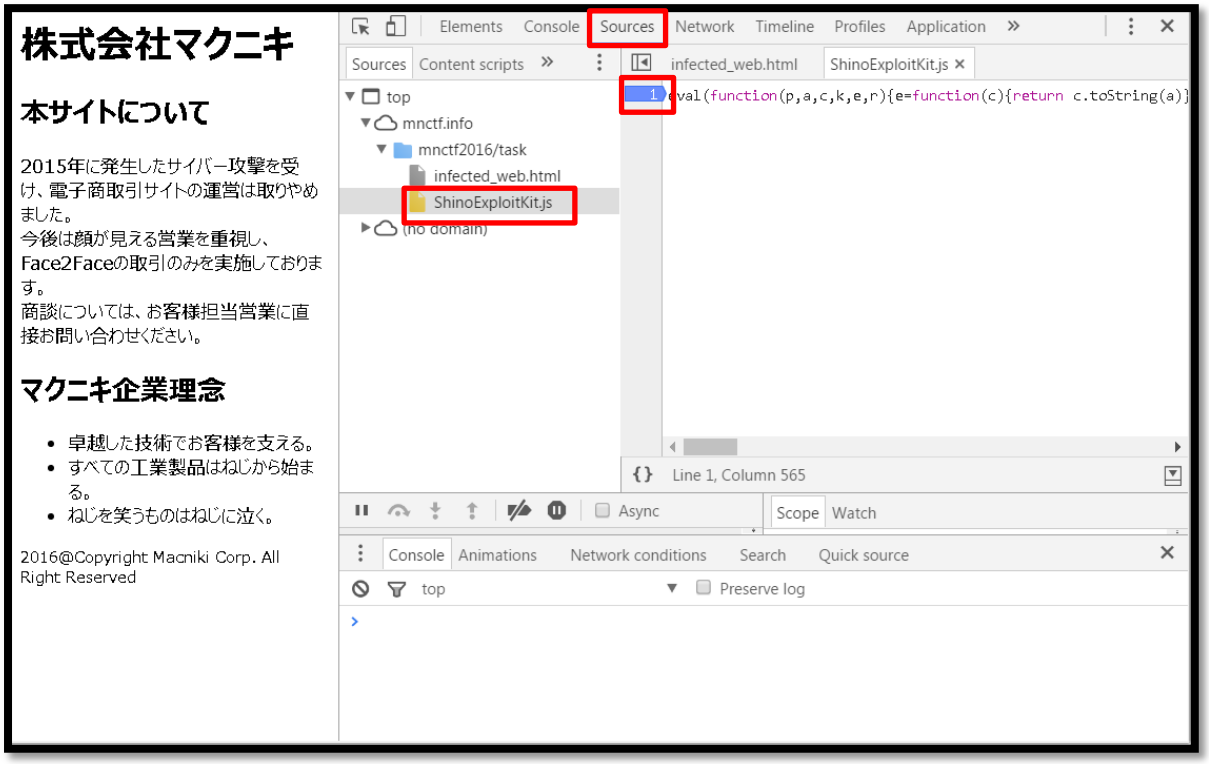
MS16-032 で検索すると CVE 番号は「CVE-2016-0099」であることがわかる。

答え：**CVE-2016-0099**

出題者からのコメント：問題を作った当初は、パッチが出た直後ということもあり、VirusTotal にアップしてもマルウェア判定されず、そこまで簡単には解かれないと考えていましたが、すぐにシグネチャが作成され、VirusTotal で CVE 番号を簡単に得ることができるものとなりました。なお、こういう OS の脆弱性をついて、権限昇格が可能な脆弱性は APT の攻撃者にとって恰好のターゲットとなるため、社内システムを含む、全台のパッチ適用をおすすめします。

改ざんされた Web サイトから挿入された JavaScript を解析する問題。開発者ツールを利用して、JavaScript をデバッグする。

開発者ツールを起動した状態で、改ざんされたページを表示し、開発者ツールの[Sources]タブを開く。



ファイルリストから「ShinoExploitKit.js」をダブルクリックし、右ペインの行番号[1]をクリックする（これで、ブレークポイントを置くことができる。

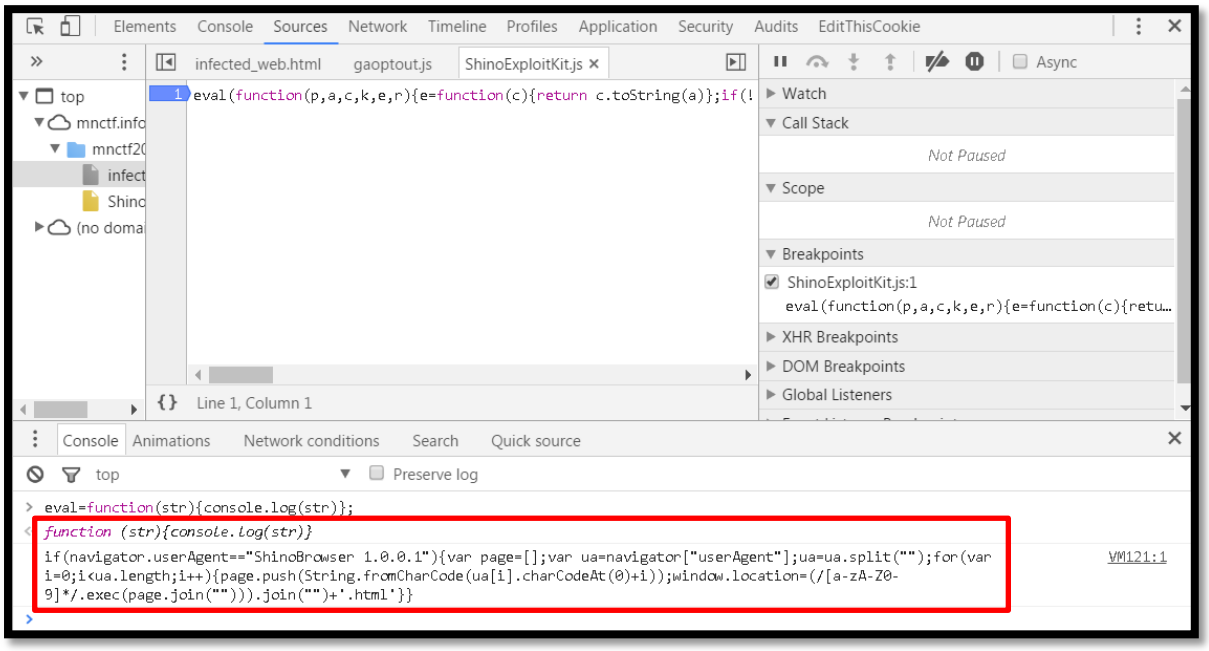
[F5]キーを押して、ページを更新する。

設置したブレークポイントで処理が止まるため、画面下部の Console（見えていない場合は[Esc]キーを押すと表示される）で以下のコードを貼り付けて、[Enter]キーを押す。

```
eval=function(str){console.log(str)};
```

これにより、ビルトインの関数である eval が上書きされて、コンソールにパラメータが表示されるようになる。

※eval 関数は難読化された JavaScript でよく使われる関数で、パラメータの文字列を JavaScript として実行する関数である。



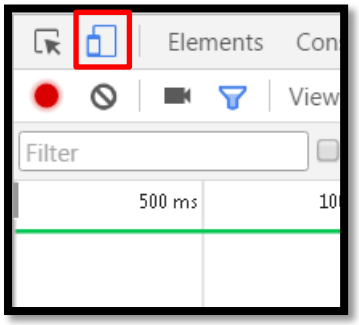
[F5]キーを押して、止めていた処理を続行する。eval 関数が実行されて、その結果、Console 欄に実行される予定だったコードが表示される（上記赤枠）。

整形したものが以下である。

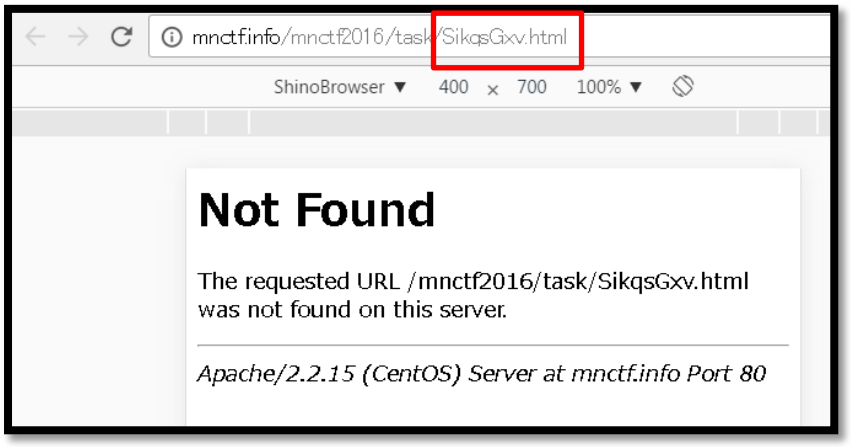
```
if(navigator.userAgent=="ShinoBrowser 1.0.0.1"){
    var page=[];var ua=navigator["userAgent"];ua=ua.split("");
    for(var i=0;i<ua.length;i++){
        page.push(String.fromCharCode(ua[i].charCodeAt(0)+i));
        window.location=(/[a-zA-Z0-9]*/.exec(page.join(""))).join("")+'.html'
    }
}
```

1 行目で、UserAgent が「ShinoBrowser 1.0.0.1」であるかどうかを確認しており、その UserAgent を元に遷移するページを決定している。UserAgent を上記の通りにする。

ここでは割愛するが、[Device Tool Bar]を用いることで他のブラウザをエミュレーション（サイズや UserAgent など）できる。自分でデバイスを追加することもできるため、それを用いて、UserAgent を変更する。



UserAgent を変更した状態で、再度同じページを[F5]キーで読み込むと、攻撃ページに遷移する。



答え：SikqsGxv.html

出題者からのコメント：実際の Exploit Kit にもよくある、アクセスしてきたブラウザの情報、プラグインのバージョンなどを確認した上で、脆弱性のあるブラウザのみ、エクスプロイトのページに誘導するタイプの JavaScript を意識して作成しました。ブラウザの開発者ツールは非常に強力ですが、実際に同様の解析をされる際は、感染しても影響のない端末上で実施してください。

マルウェアを解析し、発動条件を調べる問題。
FLOSS という難読化された文字列を抽出する無償ツールを用いることで回答が導き出せる。
以下はその実行結果である。

```
>floss MNCTF2016_MalwareBinary.exe
FLOSS static ASCII strings
!This program cannot be run in DOS mode.
Rich
.text
`.rdata
(中略)
%p @
%t @
Computer Name: %s
This host is not the target.
Operation Aborted.
Targeted host found. Continue operation.
RSDS
3:¥.pdb
GCTL
.text$mn
(中略)
GetComputerNameA
KERNEL32.dll
(中略)

FLOSS extracted 1 stackstrings
TESHIGAWARA-PC

Finished execution after 4.671000 seconds
```

上部は Strings というツールでも抽出可能な文字列で、その中に「Computer Name」や「This host is not the target」という文字列に加えて、「GetComputerNameA」という WinAPI が利用されていることから、コンピュータ名の照合を行っていることが伺える。

そして、FLOSS の特徴である「stackstrings」の文字列から「TESHIGAWARA-PC」が確認できる。

答え：TESHIGAWARA-PC

出題者からのコメント：APT 攻撃でも特定の端末あるいは特定のユーザがログインしていないと発動しないマルウェアが存在します。そういうマルウェアを意識してこの問題を作成しました。FLOSS というツールはマルウェアに含まれるデコードルーチンをヒューリスティック解析により見つけ出し、そのルーチンを用いて、文字を解読してくれるため、strings よりも多くの文字が抽出できます。

9. 丸文字文

暗号文を解読する問題。丸囲み文字がずらっと並んでいるが、英数字と「/」、「+」のみで構成されていることから Base64 であると推測できる。

また、HTML のソースコードは 10 進数の実体参照となっていることからそれを変換する Python スクリプトを書く。

```
def circled2plain(x):
    if 0x2460 <= x and x <= 0x2468:     #1~9
        return chr(x-0x2460 + ord('1'))
    elif 0x24b6 <= x and x <= 0x24cf:   #A~Z
        return chr(x-0x24b6 + ord('A'))
    elif 0x24d0 <= x and x <= 0x24e9:   #a~z
        return chr(x-0x24d0 + ord('a'))
    elif 0x24ea == x:                   #0
        return '0'
    elif 0x2295 ==x:                    #+
        return '+'
    else:
        return '/'                     #/


text='''&#9430;
&#9315;
&#9313;
&#9401;
&#9419;
&#9406;

(中略)

&#9421;
&#9312;
&#9319;
&#9439;'''

decoded=''
lines=text.split('¥n')
for l in lines:
    decoded=decoded + circled2plain(int(l[2:6]))
print decoded

file=open('marumojibun.bin',"wb")
file.write(decoded.decode('base64'))
file.close
```

出来上がった「marumojibun.bin」をバイナリエディタで開くと Shift JIS のテキストファイルだとわかる。

※見極めるポイントは 8x xx の文字が多い。

拡張子を.txt に変更し、メモ帳で開くと内容が正しく表示される。

最後に「フラグはこの人の名前」とあるので、この文章に該当する人を Web 検索で調べると、「Kevin Mitnick」だと分かる。

答え：Kevin Mitnick

出題者からのコメント：大会が行われた際、全問正解者が早々と出たため、途中から追加したおまけ問題です。

以上