TDSS part 1: The x64 Dollar Question

resources.infosecinstitute.com/tdss4-part-1/

Reverse engineering
April 19, 2011 by **ESET Team**

Introduction

In the two years since the Win32/Olmarik family of malware programs (also known as TDSS, TDL and Alureon) started to evolve, its authors have implemented a notably sophisticated mechanism for bypassing various protective measures and security mechanisms embedded into the operating system.

The fourth version of the TDL rootkit family (TDL4) is the first reliable and widely spread bootkit to target x64 operating systems (Windows Vista and Windows 7). Since TDL4 started to spread actively in August 2010, several versions of the malware have been released. By comparison with its predecessors, TDL4 is not just characterized by modification of existing code, but to all intents and purposes can be regarded as new malware. Among the many changes that have been applied as it developed, the most radical were those made to its mechanisms for self-embedding into the system and surviving reboot. One of the most striking features of TDL4 is its ability to load its kernel-mode driver on systems with an enforced kernel-mode code signing policy (64-bit versions of Microsoft Windows Vista and Windows 7) and perform kernel-mode hooks with kernel-mode patch protection policy enabled. This makes TDL4 a powerful weapon in the hands of cybercriminals. In this article, we consider the PPI (Pay Per Install) distribution model used by both TDL3 and TDL4, and the initial installation.

Distribution by Pay Per Install

In "TDL3: The Rootkit of All Evil?" Aleksandr Matrosov and Eugene Rodionov described how the DogmaMillions cybercrime group distributed the third version of the TDSS (a.k.a. TDL, Olmarik, or Alureon) rootkit using a PPI (Pay Per Install) scheme. It's both interesting and instructive to compare the ways in which TDL3 and TDL4 (as described in their more recent paper "The Evolution of TDL: Conquering x64″) have been distributed.

The TDL3 rootkit droppers were distributed using a Pay-Per-Install (PPI) scheme popular among cybercrime groups. The scheme is, in fact, similar to schemes commonly used for distributing browser toolbars. Toolbar distributors have a special build with an embedded identifier which allows for calculating the number of installations associated with that ID and therefore for determining their revenue. The same approach is used for distributing the rootkits: information about the distributor is embedded into the executable and special servers are used to calculate the number of installations.

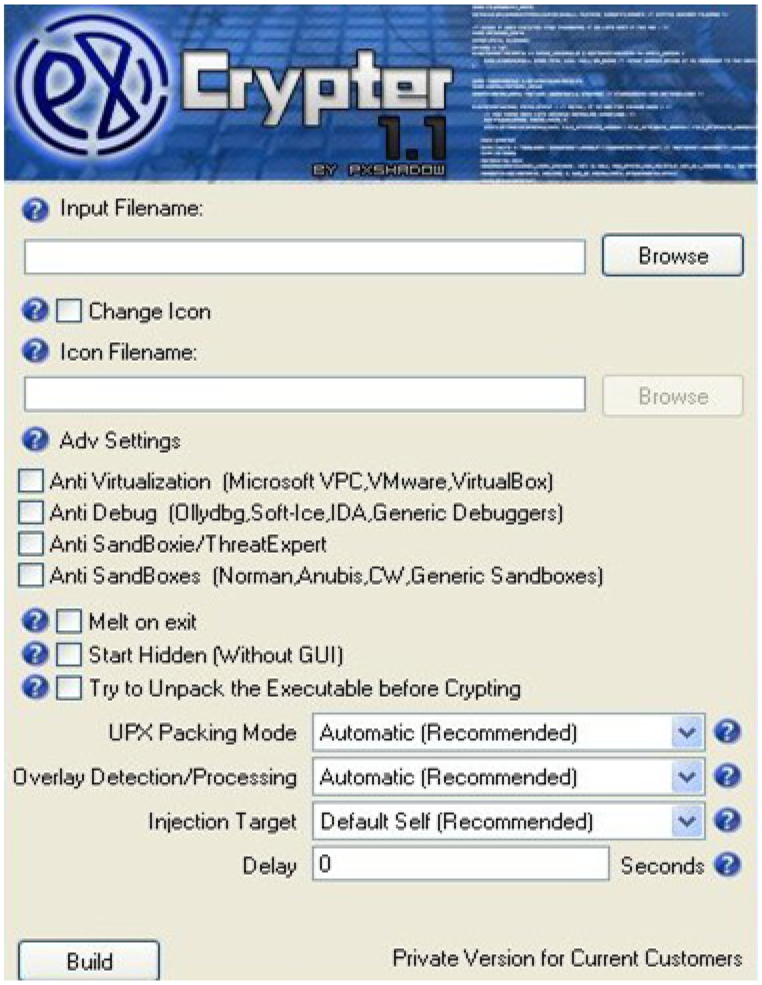
Anyone deciding to cooperate with the cybercrime group received a unique login and a password, identifying the number of installations per resource, like this:

```
hxxp://dogmamillions.com/download.html?
login=b0bah&amp;key=2b15ea4e5eb2bbd734081c051a14fa41&amp;affSid=0
```

The gang made use of a well-developed business infrastructure: for example, each affiliate had a personal manager who could be consulted in case of any problems.

Distributed malware was repacked every few hours (or even more frequently) using all-too-reliable and sophisticated packers and protectors in order to reduce the risk of detection by antivirus software, using sophisticated tools and techniques to detect debuggers and virtual machines. Partners were instructed not to check on whether the malware can be detected by AV by using resources like VirusTotal, and could even be “fined” for doing so.

You can see the user interface characteristic of one widely-used encryptor in the figure below. At that point, this was attracting a charge of around $500.

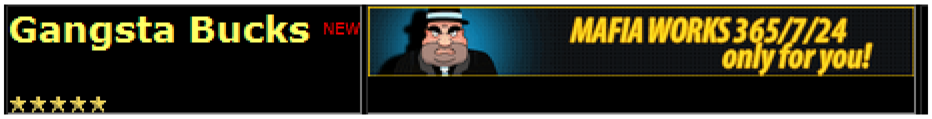

eX Crypter 1.1
BY EXSHADOW
Input Filename:
Browse
Change Icon
Icon Filename:
Browse
Adv Settings
Anti Virtualization (Microsoft VPC,VMware,VirtualBox)
Anti Debug (Ollydbg,Soft-Ice,IDA,Generic Debuggers)
Anti SandBoxie/ThreatExpert
Anti SandBoxes (Norman,Anubis,CW,Generic Sandboxes)
Melt on exit
Start Hidden (Without GUI)
Try to Unpack the Executable before Crypting
Build
Private Version for Current Customers

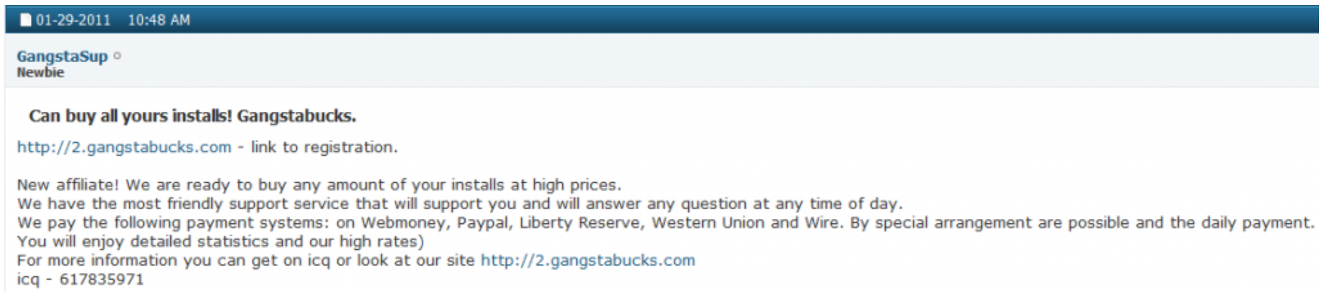
UPX Packing Mode: Automatic (Recommended)
Overlay Detection/Processing: Automatic (Recommended)
Injection Target: Default Self (Recommended)
Delay: 0 Seconds

Figure 1 – User Interface of a Packer

The Dogma Millions cybercrime group set up in business in the autumn 2009, placing many advertisements on public forums offering "easy money". However, it seems to have been somewhat uncomfortable with the copious attention it received last year, and shut down in the fall. Major affiliates to DogmaMillions could earn a cool $100,000 daily, so it's no surprise that TDL4, the latest generation of TDSS, quickly found similar distribution channels.

GangstaBucks appeared at the end of 2010 and was widely advertised in various forums in Russia and elsewhere, offering very similar terms and features to DogmaMillions, and a very similar mode of expression.




01-29-2011 10:48 AM

GangstaSup
Newbie

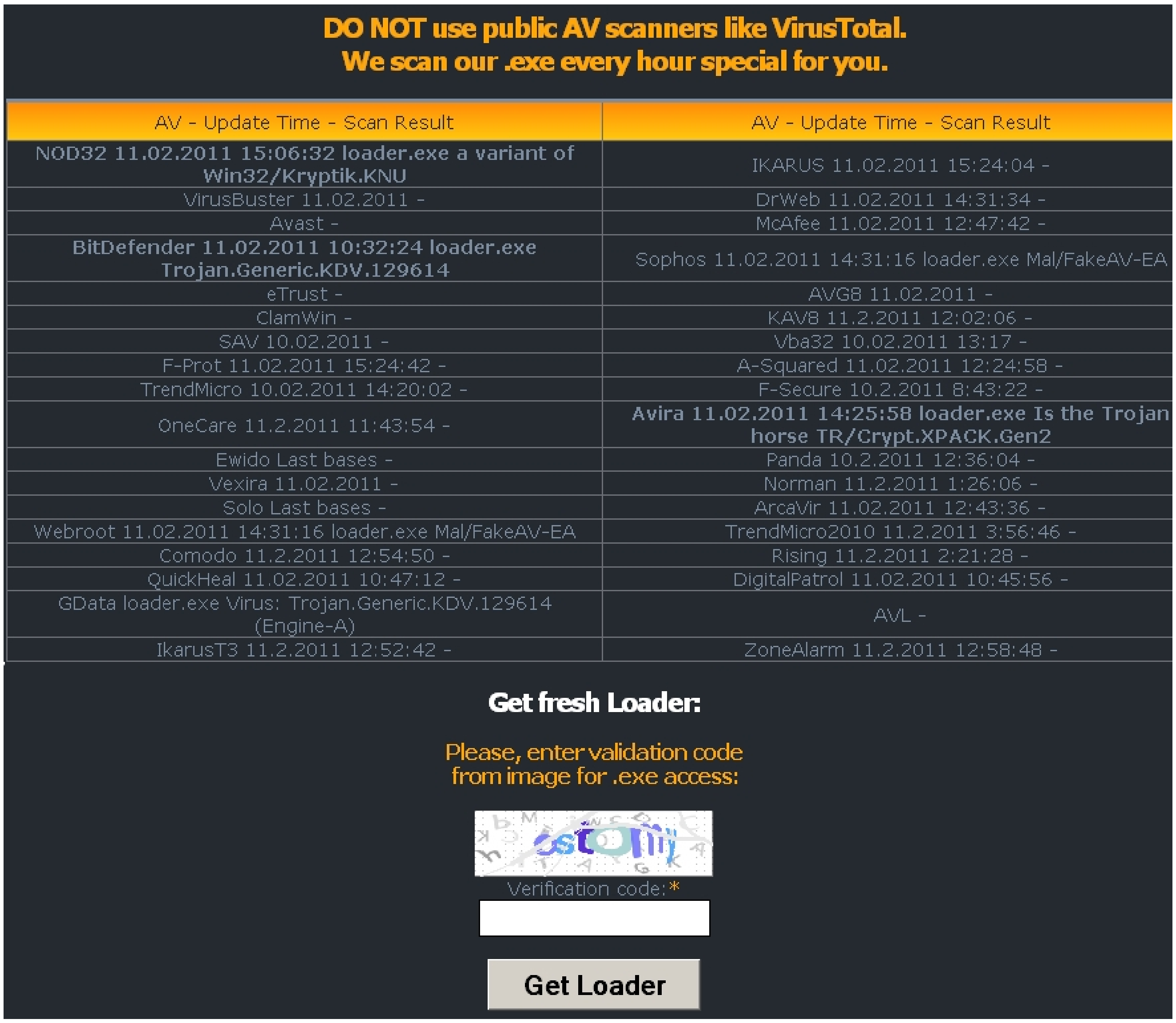
Can buy all yours installs! Gangstabucks.

http://2.gangstabucks.com - link to registration.

New affiliate! We are ready to buy any amount of your installs at high prices.
We have the most friendly support service that will support you and will answer any question at any time of day.
We pay the following payment systems: on Webmoney, Paypal, Liberty Reserve, Western Union and Wire. By special arrangement are possible and the daily payment.
You will enjoy detailed statistics and our high rates)
For more information you can get on icq or look at our site http://2.gangstabucks.com
icq - 617835971

Figure 2 – The GangstaBucks Adverts

Affiliates are able to download the current version of the Trojan downloader and to receive statistics relating to detection by antivirus software. This serves to dissuade the partner from submitting the current version to services such as VirusTotal that forward malicious samples to security companies.



DO NOT use public AV scanners like VirusTotal.
We scan our .exe every hour special for you.

AV - Update Time - Scan Result	AV - Update Time - Scan Result
NOD32 11.02.2011 15:06:32 loader.exe a variant of Win32/Kryptik.KNU	IKARUS 11.02.2011 15:24:04 -
VirusBuster 11.02.2011 -	DrWeb 11.02.2011 14:31:34 -
Avast -	McAfee 11.02.2011 12:47:42 -
BitDefender 11.02.2011 10:32:24 loader.exe Trojan.Generic.KDV.129614	Sophos 11.02.2011 14:31:16 loader.exe Mal/FakeAV-EA
eTrust -	AVG8 11.02.2011 -
ClamWin -	KAV8 11.2.2011 12:02:06 -
SAV 10.02.2011 -	Vba32 10.02.2011 13:17 -
F-Prot 11.02.2011 15:24:42 -	A-Squared 11.02.2011 12:24:58 -
TrendMicro 10.02.2011 14:20:02 -	F-Secure 10.2.2011 8:43:22 -
OneCare 11.2.2011 11:43:54 -	Avira 11.02.2011 14:25:58 loader.exe Is the Trojan horse TR/Crypt.XPACK.Gen2
Ewido Last bases -	Panda 10.2.2011 12:36:04 -
Vexira 11.02.2011 -	Norman 11.2.2011 1:26:06 -
Solo Last bases -	ArcaVir 11.02.2011 12:43:36 -
Webroot 11.02.2011 14:31:16 loader.exe Mal/FakeAV-EA	TrendMicro2010 11.2.2011 3:56:46 -
Comodo 11.2.2011 12:54:50 -	Rising 11.2.2011 2:21:28 -
QuickHeal 11.02.2011 10:47:12 -	DigitalPatrol 11.02.2011 10:45:56 -
GData loader.exe Virus: Trojan.Generic.KDV.129614 (Engine-A)	AVL -
IkarusT3 11.2.2011 12:52:42 -	ZoneAlarm 11.2.2011 12:58:48 -

Get fresh Loader:

Please, enter validation code from image for .exe access:

Figure 3 –Scanning Samples for Detection by AV Software

When the downloader is known to be widely detected, the partner receives a newly-repacked sample, so that release/detect cycle begins again.

When the downloader is launched it sends information about the compromised system to a C&C (Command and Control) server and pulls down a secondary downloader which in turn downloads and runs the main malware. The sequence of download events for the downloader which we analyzed is depicted in the following figure. As we can see, the first downloader obtains *Win32/Agent.QNF* which downloads and installs either *Win32/Bubnix* or *Win32/KeyLogger.EliteKeyLogger* malware onto the system.

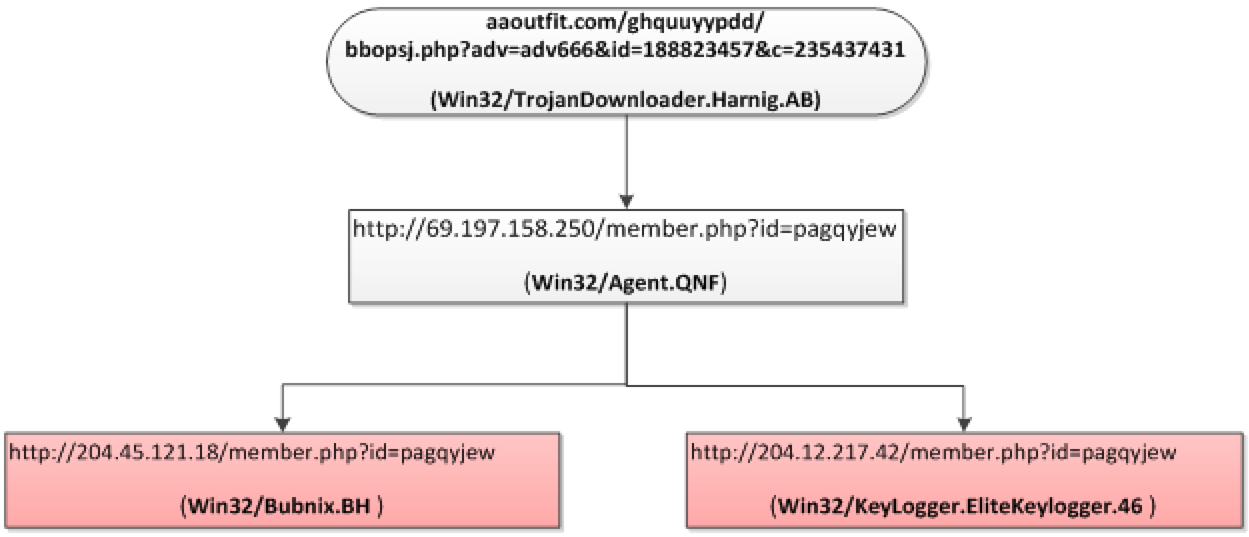

aaoutfit.com/ghquuyypdd/
bbopsj.php?adv=adv666&id=188823457&c=235437431
(Win32/TrojanDownloader.Harnig.AB)
http://69.197.158.250/member.php?id=pagqyjew
(Win32/Agent.QNF)
http://204.45.121.18/member.php?id=pagqyjew
(Win32/Bubnix.BH)
http://204.12.217.42/member.php?id=pagqyjew
(Win32/KeyLogger.EliteKeylogger.46)

Figure 4 – The Downloader at Work

Downloader packers and links are changed every few hours, so as to minimize the risk of detection by malware installation tracking systems.

In the middle of February we received a downloader (Win32/TrojanDownloader.Agent.QOF) that installs the latest version of the TDL4 bootkit onto the system. During the installation of the bootkit, as we can see from figure 5, the downloader reports back to the server to register the installation with the partner identifier.

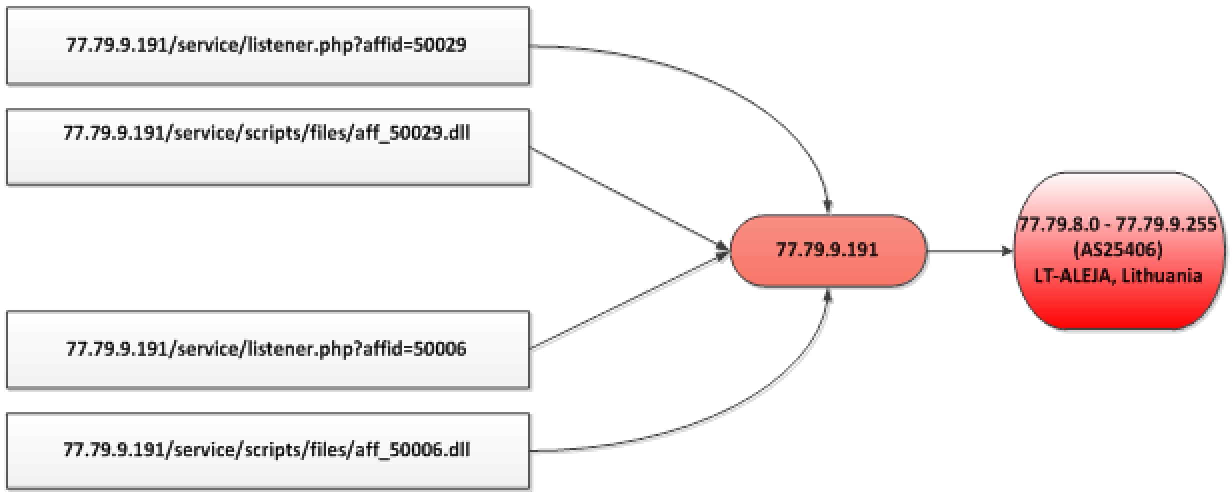
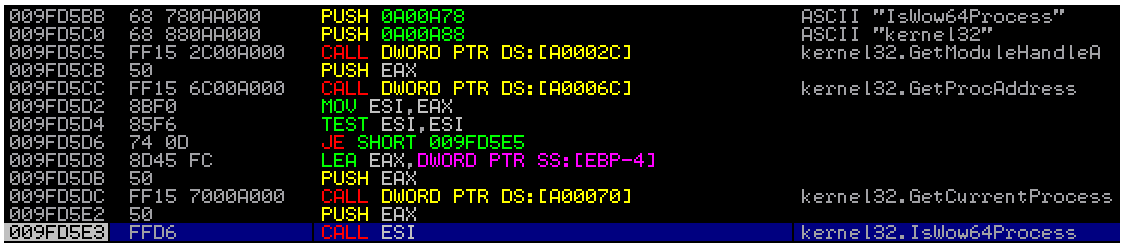

77.79.9.191/service/listener.php?affid=50029
77.79.9.191/service/scripts/files/aff_50029.dll
77.79.9.191/service/listener.php?affid=50006
77.79.9.191/service/scripts/files/aff_50006.dll
77.79.9.191
77.79.8.0 - 77.79.9.255
(AS25406)
LT-ALEJA, Lithuania

Figure 5 – Installation of GangstaBucks's TDL4

When conditions are mutually beneficial, services like DogmaMillions and GangstaBucks can accumulate hundreds of partners. In such a case the number of sites all over the world distributing the malicious software can reach several thousand.

Bootkit Installation

The installation of the bootkit is handled differently on x86 and x64 systems due to specific limitations on x64 platforms. As soon as the dropper is unpacked it checks whether it is running in Wow64 process and determines which branch of the code it should execute.



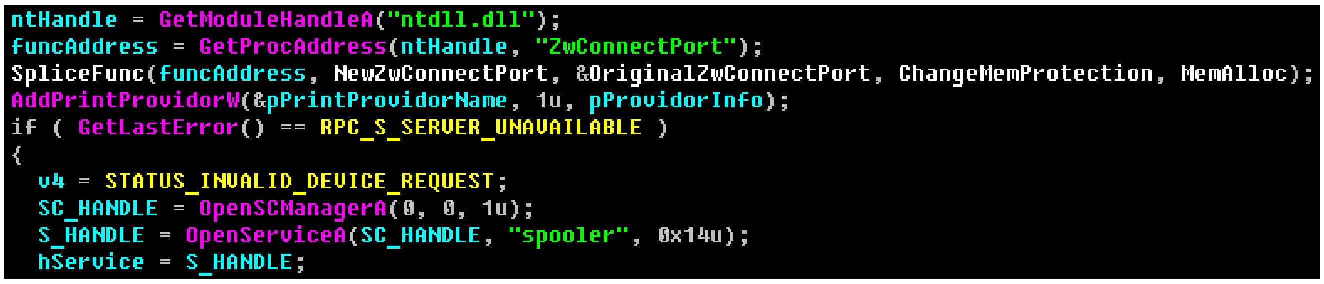
```
009FD5BB  68 780AA000   PUSH 0A00A78                 ASCII "IsWow64Process"
009FD5C0  68 880AA000   PUSH 0A00A88                 ASCII "kernel32"
009FD5C5  FF15 2C00A000 CALL DWORD PTR DS:[A0002C]   kernel32.GetModuleHandleA
009FD5CB  50            PUSH EAX
009FD5CC  FF15 6C00A000 CALL DWORD PTR DS:[A0006C]   kernel32.GetProcAddress
009FD5D2  8BF0          MOV ESI,EAX
009FD5D4  85F6          TEST ESI,ESI
009FD5D6  74 0D         JE SHORT 009FD5E5
009FD5D8  8D45 FC       LEA EAX,DWORD PTR SS:[EBP-4]
009FD5DB  50            PUSH EAX
009FD5DC  FF15 7000A000 CALL DWORD PTR DS:[A00070]   kernel32.GetCurrentProcess
009FD5E2  50            PUSH EAX
009FD5E3  FFD6          CALL ESI                     kernel32.IsWow64Process
```

Figure 6 –Determining OS Version

Infecting x86 Systems

On x86 systems the installation process looks the same as it does for TDL3/TDL3+, as described in an earlier paper (http://www.eset.com/resources/white-papers/TDL3-Analysis.pdf). To bypass HIPS the bootkit loads itself as a print provider into the trusted system process (*spooler.exe*) from whence it loads a kernel-mode driver (*drv32*) which infects the system.

The bootkit implements an additional HIPS bypassing technique which wasn't noticed in TDL3/TDL3+ droppers: it hooks the *ZwConnectPort* system routine exported from *ntdll.dll*.



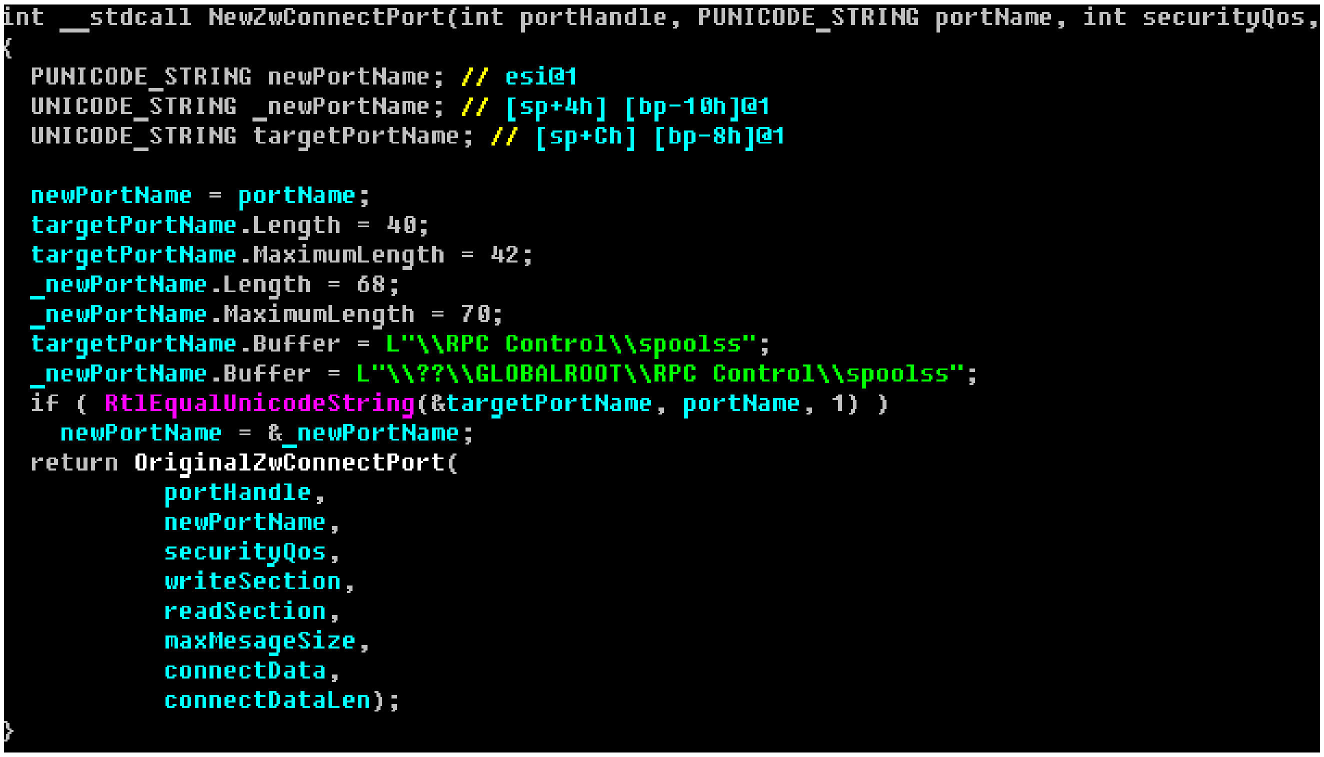
```
ntHandle = GetModuleHandleA("ntdll.dll");
funcAddress = GetProcAddress(ntHandle, "ZwConnectPort");
SpliceFunc(funcAddress, NewZwConnectPort, &OriginalZwConnectPort, ChangeMemProtection, MemAlloc);
AddPrintProvidorW(&pPrintProvidorName, 1u, pProvidorInfo);
if ( GetLastError() == RPC_S_SERVER_UNAVAILABLE )
{
  v4 = STATUS_INVALID_DEVICE_REQUEST;
  SC_HANDLE = OpenSCManagerA(0, 0, 1u);
  S_HANDLE = OpenServiceA(SC_HANDLE, "spooler", 0x14u);
  hService = S_HANDLE;
```

Figure 7 – Hooking ZwConnectPort

Here is the prototype of the function *ZwConnectPort.* Parameter *PortName* is set to the name of the target LPC port to connect to.

```
 NTSYSAPI
NTSTATUS
NTAPI
ZwConnectPort(
OUT PHANDLE PortHandle,
IN PUNICODE_STRING PortName,
IN PSECURITY_QUALITY_OF_SERVICE SecurityQos,
IN OUT PPORT_SECTION_WRITE WriteSection OPTIONAL,
IN OUT PPORT_SECTION_READ ReadSection OPTIONAL,
OUT PULONG MaxMessageSize OPTIONAL,
IN OUT PVOID ConnectData OPTIONAL,
IN OUT PULONG ConnectDataLength OPTIONAL
```

The routine is called during execution of *AddPrintProvidor* to connect to the print spooler LPC port. As shown here the hook prepends to the target port name *"??GLOBALROOT"* string in an attempt to connect to the print spooler service.



```
int __stdcall NewZwConnectPort(int portHandle, PUNICODE_STRING portName, int securityQos,
{
  PUNICODE_STRING newPortName; // esi@1
  UNICODE_STRING _newPortName; // [sp+4h] [bp-10h]@1
  UNICODE_STRING targetPortName; // [sp+Ch] [bp-8h]@1

  newPortName = portName;
  targetPortName.Length = 40;
  targetPortName.MaximumLength = 42;
  _newPortName.Length = 68;
  _newPortName.MaximumLength = 70;
  targetPortName.Buffer = L"\\RPC Control\\spoolss";
  _newPortName.Buffer = L"\\??\\GLOBALROOT\\RPC Control\\spoolss";
  if ( RtlEqualUnicodeString(&targetPortName, portName, 1) )
    newPortName = &_newPortName;
  return OriginalZwConnectPort(
           portHandle,
           newPortName,
           securityQos,
           writeSection,
           readSection,
           maxMesageSize,
           connectData,
           connectDataLen);
}
```

Figure 8 – The Code of ZwConnectPort Hook

When the driver is loaded into kernel-mode address space it overwrites the MBR (Master Boot Record) of the disk by sending SRB (SCSI Request Block) packets directly to the miniport device object, after which it initializes its hidden file system. The bootkit's modules are written into the hidden file system from the dropper by means of the *CreateFile* and *WriteFile* API functions.

The algorithm for infecting x86 operating systems is presented in Figure 10. It is important to mention that the TDL4 dropper exploits the now-patched MS10-092 vulnerability in the Microsoft Windows Task Scheduler service in order to elevate privileges and successfully load its own driver. The vulnerable systems include all Windows operating systems starting from Microsoft Windows Vista (both x86 and x64 versions). If it fails to exploit the vulnerability it copies itself into a file into TEMP directory with the name *"setup_xxx.exe"* and creates a corresponding manifest file requesting administrative privileges to run the application. After that, it runs the copied dropper by calling *ShellExecute* and a dialog box message requesting administrative rights is displayed to the user.

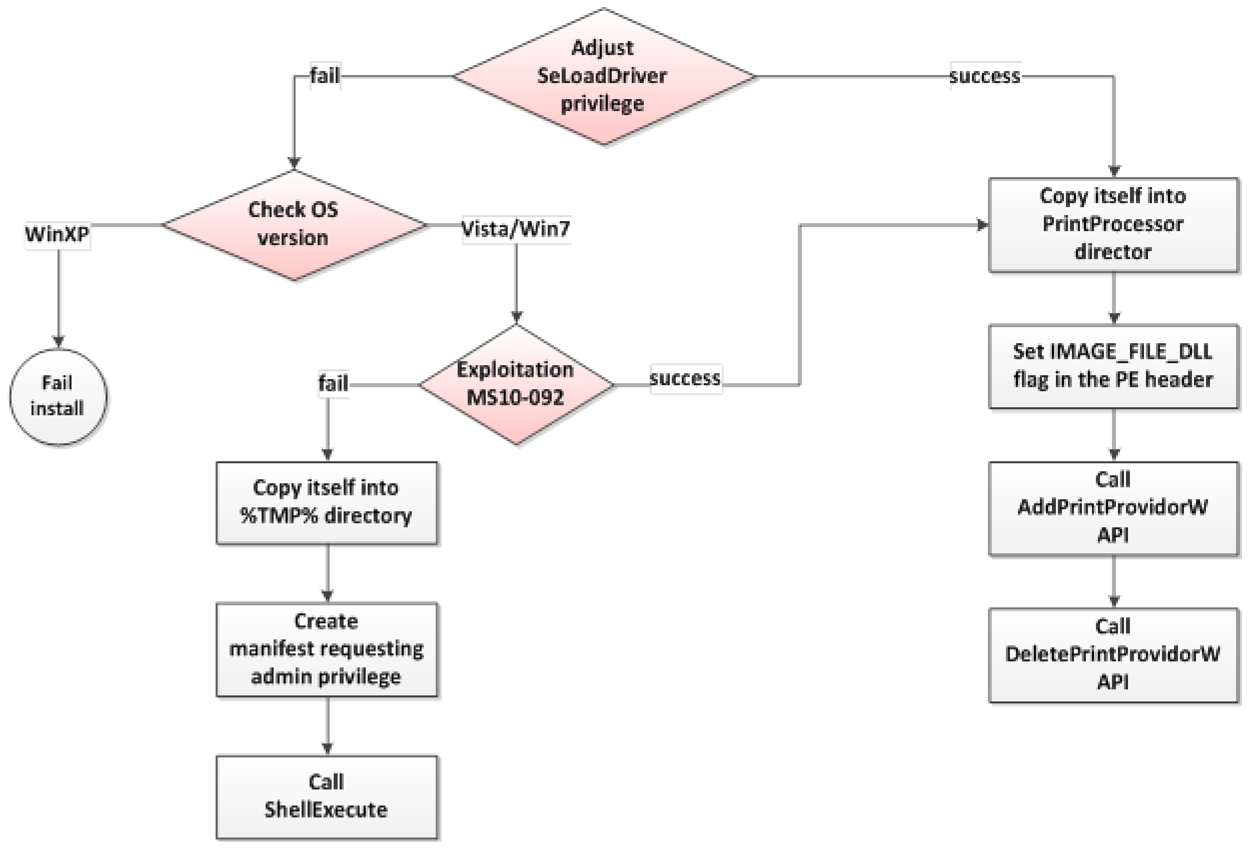

Adjust SeLoadDriver privilege
fail
success
Check OS version
WinXP
Vista/Win7
Fail install
Exploitation MS10-092
fail
success
Copy itself into PrintProcessor director
Set IMAGE_FILE_DLL flag in the PE header
Call AddPrintProvidorW API
Call DeletePrintProvidorW API
Copy itself into %TMP% directory
Create manifest requesting admin privilege
Call ShellExecute

Figure 9 – The Algorithm of Infecting x86 System

Infecting x64 Systems

When the dropper is run on x64 operating systems it is unable to load the kernel-mode driver, as 64-bit systems require it to be signed. To overcome this restriction the dropper writes all its components directly to the hard drive by sending IOCTL_SCSI_PASS_THROUGH_DIRECT requests to a disk class driver. It obtains the disk's parameters and creates the image of its hidden file system in the memory buffer which is then written onto the hard drive at a certain offset. We'll discuss the way in which the hidden file system is maintained in a future article, but an earlier article for Virus Bulletin (Rooting around in TDSS) is also relevant to the discussion. When the image is written the dropper modifies the MBR of the disk to get its malicious components loaded at boot time. After that, the dropper reboots the system by calling the *ZwRaiseHardError* routine, passing as its fifth parameter *OptionShutdownSystem.* This instructs the system to display a BSOD (Blue Screen Of Death) and reboot the system:

```
 NTSYSAPI
NTSTATUS
NTAPI
NtRaiseHardError(
IN NTSTATUS ErrorStatus
IN ULONG NumberOfParameters,
IN PUNICODE_STRING   UnicodeStringParameterMask OPTIONAL,
IN PVOID *Parameters,
IN HARDERROR_RESPONSE_OPTION ResponseOption,
OUT PHARDERROR_RESPONSE Response );
```

In the next figure we present a diagram depicting process of infecting x64 system.

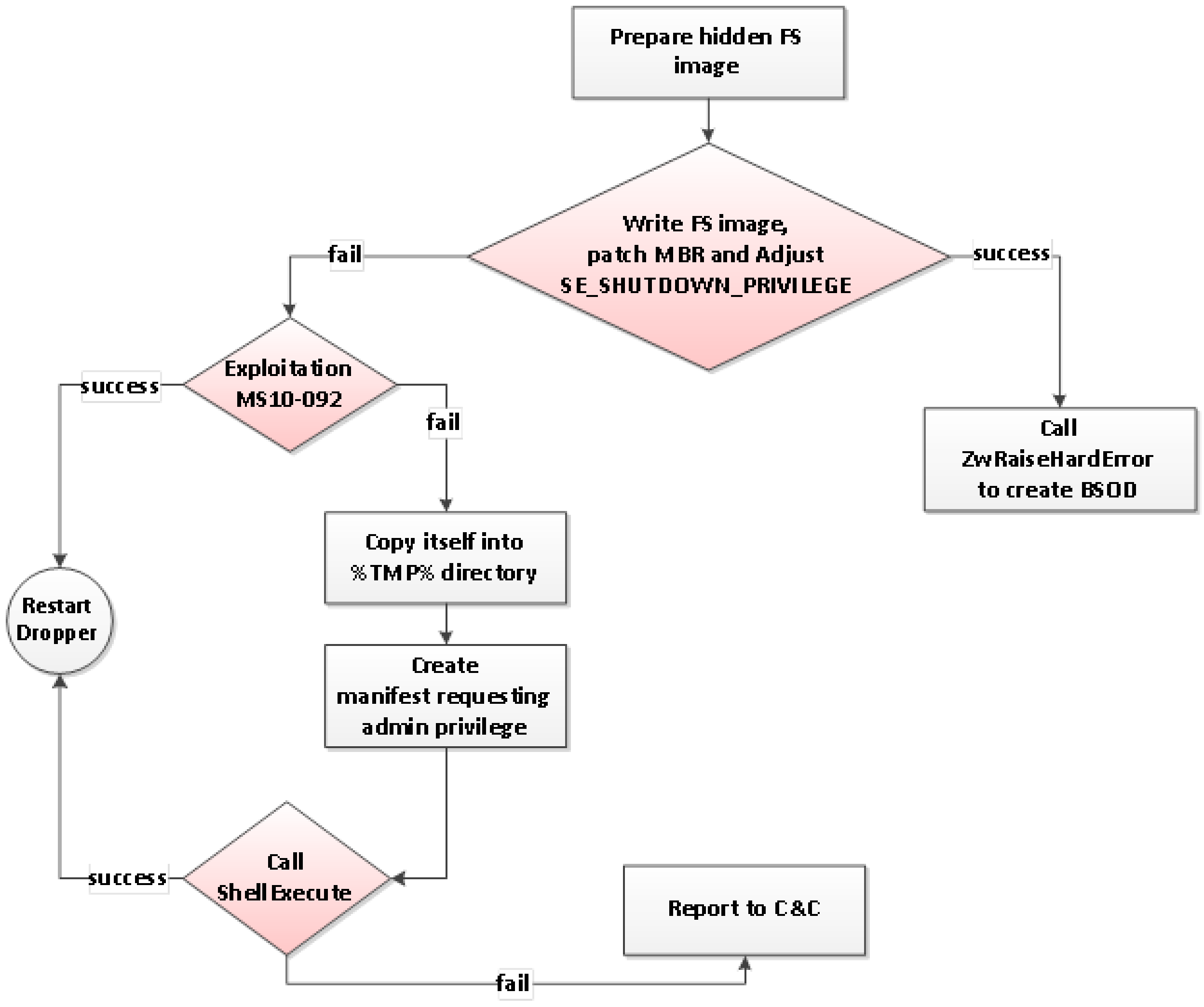

Prepare hidden FS image
Write FS image, patch MBR and Adjust SE_SHUTDOWN_PRIVILEGE
fail
success
Exploitation MS10-092
success
fail
Call ZwRaiseHardError to create BSOD
Copy itself into %TMP% directory
Restart Dropper
Create manifest requesting admin privilege
Call ShellExecute
success
Report to C&C
fail

Figure 10 – The Algorithm for Infecting x64 Systems

The Dropper’s Payload

The bootkit’s components are contained inside the “.config” section of the dropper (the layout of the section is described below as detailed in our report on TDL3).

The rootkit dropper is encrypted. The decryption routine is slightly obfuscated and varies between different droppers. During unpacking, the dropper performs some simple anti-debugging checks and also checks that it isn’t running inside a virtual machine. The next figure shows the structure of the dropper.

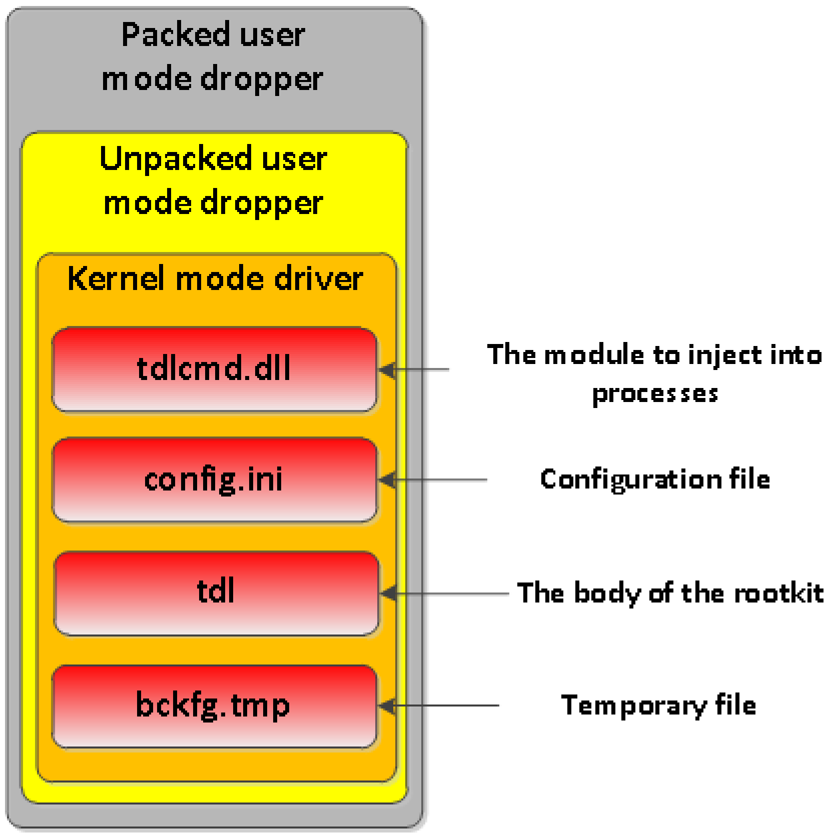


Figure 11 – The Dropper Structure

Here is the list of modules that are dropped into the hidden file system:

Dropped modules	Description
mbr	original contents of the infected hard drive boot sector
ldr16	16-bit real-mode loader code
ldr32	fake *kdcom.dll* for x86 systems
ldr64	fake *kdcom.dll* for x64 systems
drv32	the main bootkit driver for x86 systems
drv64	the main bootkit driver for x64 systems
cmd.dll	payload to inject into 32-bit processes
cmd64.dll	payload to inject into 64-bit processes
cfg.ini	configuration information
bckfg.tmp	encrypted list of C&C URLs

Table 1 – Dropped Modules

Comparison with TDL3/TDL3+

Here is the table summarizing the major differences between the TDL3/TDL3+ and TDL4 droppers which include bypassing HIPS, escalating privileges, installation mechanism and number of installed modules.

	TDL3/TDL3+	**TDL4**
Bypassing HIPS	AddPrintProcessor/AddPrintProvidor	AddPrintProvidor, ZwConnectPort
Privilege Escalation	–	MS10-092
Installation mechanism	By loading kernel-mode driver	By loading kernel-mode driver,Overwriting MBR of the disk
Number of installed modules	4	10

Table 2 – Comparison of TDL Droppers

TDL4 and Glupteba

At the beginning of March 2011 we received another interesting sample of TDL4: this time, one that downloads and installs another malicious program, Win32/Glupteba.D. This was the first instance the authors had come across of TDL4s being used to install other malware. It is important to mention that this is *not* a plug-in for TDL4: it is standalone malware, which can download and execute other binary modules independently. A sample of Win32/Olmarik.AOV was obtained from a server at vidquick.info. After what looked at first like a standard TDL4 installation, at any rate by comparison with the most recent versions analyzed, Win32/Olmarik.AOV received a command from the C&C server to download and execute another binary file.

Win32/Glupteba.D uses the customary blackhat SEO (Search Engine Optimization) methods to push clickjacking contextual advertising as used by the ads network Begun (http://www.begun.ru/), which has a high profile in Russia. Clickjacking algorithms have been developed for crawling web sites pushing typical content for specified context ads. All the affected web sites are hosted by a single provider: “Masterhost.ru” is, in fact, the biggest Russian hosting-provider.

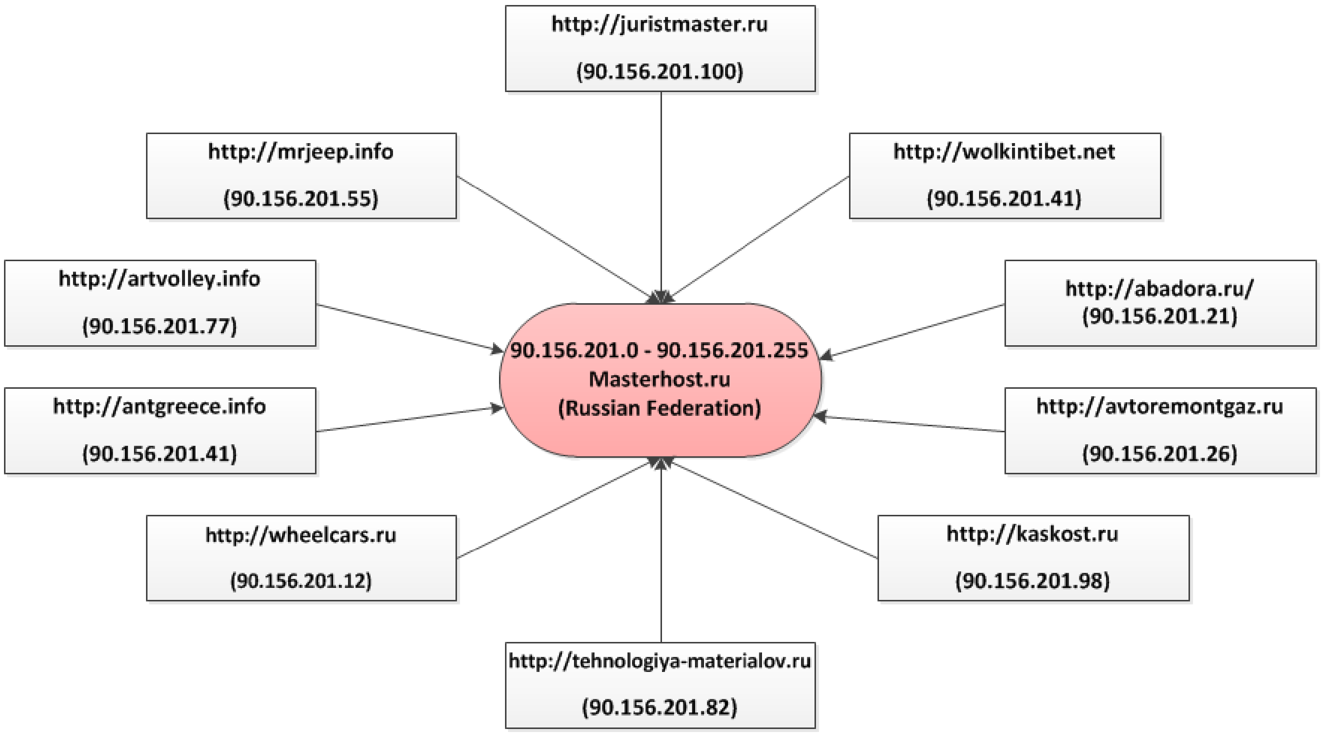
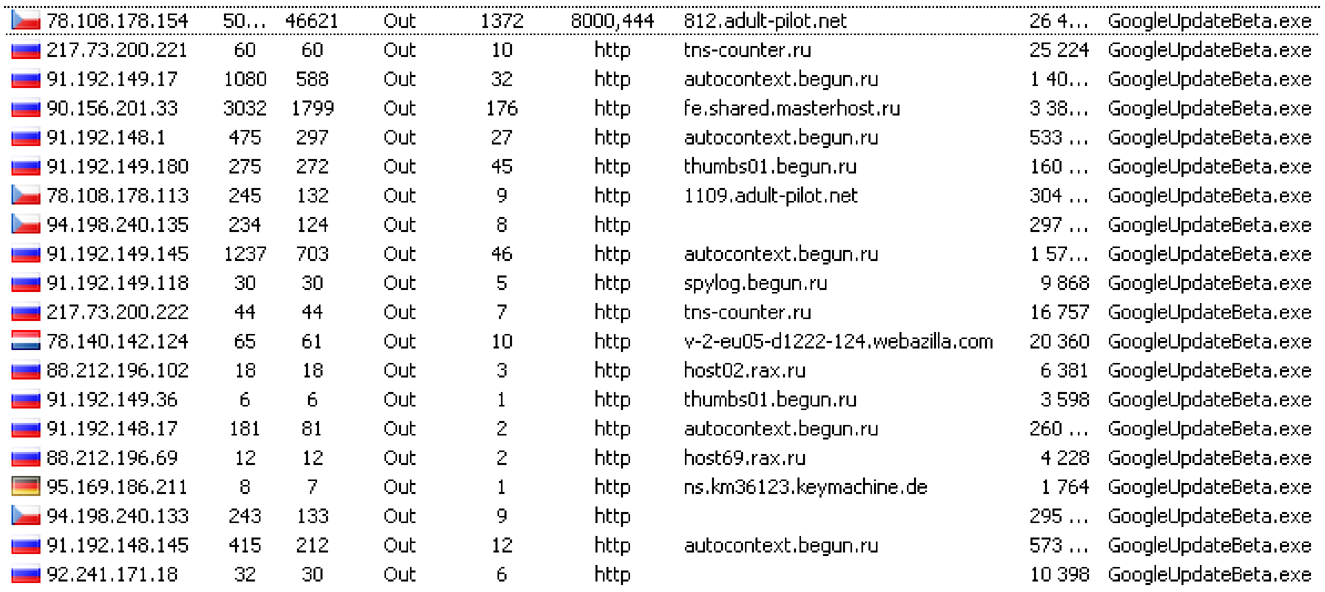

http://juristmaster.ru (90.156.201.100)
http://mrjeep.info (90.156.201.55)
http://wolkintibet.net (90.156.201.41)
http://artvolley.info (90.156.201.77)
http://abadora.ru/ (90.156.201.21)
90.156.201.0 - 90.156.201.255 Masterhost.ru (Russian Federation)
http://antgreece.info (90.156.201.41)
http://avtoremontgaz.ru (90.156.201.26)
http://wheelcars.ru (90.156.201.12)
http://kaskost.ru (90.156.201.98)
http://tehnologiya-materialov.ru (90.156.201.82)

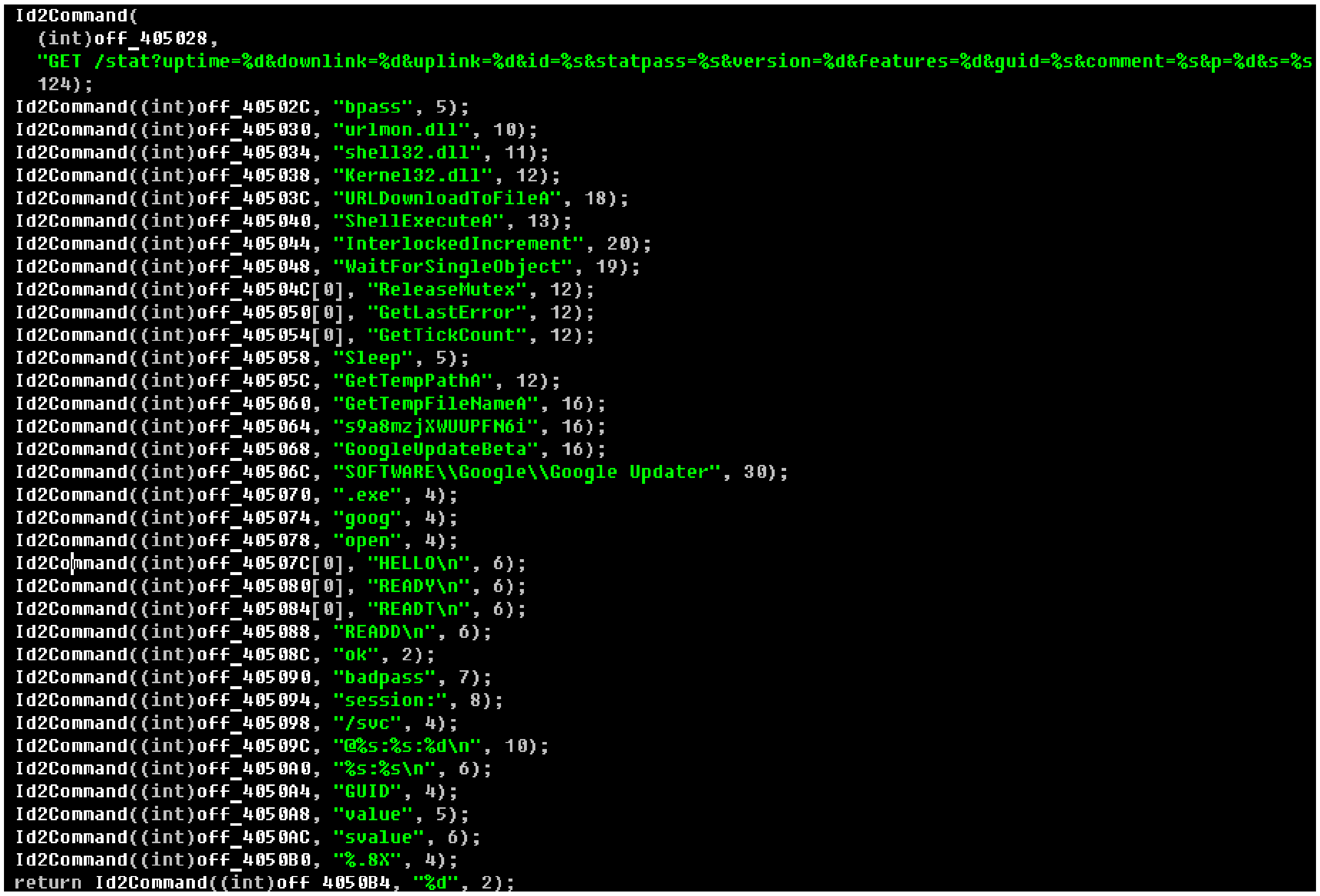
Figure 12 – the Masterhost.ru Empire

Network activity from Win32/Glupteba.D is shown in the following screendump:



🇨🇿 78.108.178.154	50...	46621	Out	1372	8000,444	812.adult-pilot.net	26 4...	GoogleUpdateBeta.exe
🇷🇺 217.73.200.221	60	60	Out	10	http	tns-counter.ru	25 224	GoogleUpdateBeta.exe
🇷🇺 91.192.149.17	1080	588	Out	32	http	autocontext.begun.ru	1 40...	GoogleUpdateBeta.exe
🇷🇺 90.156.201.33	3032	1799	Out	176	http	fe.shared.masterhost.ru	3 38...	GoogleUpdateBeta.exe
🇷🇺 91.192.148.1	475	297	Out	27	http	autocontext.begun.ru	533 ...	GoogleUpdateBeta.exe
🇷🇺 91.192.149.180	275	272	Out	45	http	thumbs01.begun.ru	160 ...	GoogleUpdateBeta.exe
🇨🇿 78.108.178.113	245	132	Out	9	http	1109.adult-pilot.net	304 ...	GoogleUpdateBeta.exe
🇨🇿 94.198.240.135	234	124	Out	8	http		297 ...	GoogleUpdateBeta.exe
🇷🇺 91.192.149.145	1237	703	Out	46	http	autocontext.begun.ru	1 57...	GoogleUpdateBeta.exe
🇷🇺 91.192.149.118	30	30	Out	5	http	spylog.begun.ru	9 868	GoogleUpdateBeta.exe
🇷🇺 217.73.200.222	44	44	Out	7	http	tns-counter.ru	16 757	GoogleUpdateBeta.exe
🇳🇱 78.140.142.124	65	61	Out	10	http	v-2-eu05-d1222-124.webazilla.com	20 360	GoogleUpdateBeta.exe
🇷🇺 88.212.196.102	18	18	Out	3	http	host02.rax.ru	6 381	GoogleUpdateBeta.exe
🇷🇺 91.192.149.36	6	6	Out	1	http	thumbs01.begun.ru	3 598	GoogleUpdateBeta.exe
🇷🇺 91.192.148.17	181	81	Out	2	http	autocontext.begun.ru	260 ...	GoogleUpdateBeta.exe
🇷🇺 88.212.196.69	12	12	Out	2	http	host69.rax.ru	4 228	GoogleUpdateBeta.exe
🇩🇪 95.169.186.211	8	7	Out	1	http	ns.km36123.keymachine.de	1 764	GoogleUpdateBeta.exe
🇨🇿 94.198.240.133	243	133	Out	9	http		295 ...	GoogleUpdateBeta.exe
🇷🇺 91.192.148.145	415	212	Out	12	http	autocontext.begun.ru	573 ...	GoogleUpdateBeta.exe
🇷🇺 92.241.171.18	32	30	Out	6	http		10 398	GoogleUpdateBeta.exe

Commands for Win32/Glupteba.D directed to the C&C server look like this:



```
Id2Command(
  (int)off_405028,
  "GET /stat?uptime=%d&downlink=%d&uplink=%d&id=%s&statpass=%s&version=%d&features=%d&guid=%s&comment=%s&p=%d&s=%s
  124);
Id2Command((int)off_40502C, "bpass", 5);
Id2Command((int)off_405030, "urlmon.dll", 10);
Id2Command((int)off_405034, "shell32.dll", 11);
Id2Command((int)off_405038, "Kernel32.dll", 12);
Id2Command((int)off_40503C, "URLDownloadToFileA", 18);
Id2Command((int)off_405040, "ShellExecuteA", 13);
Id2Command((int)off_405044, "InterlockedIncrement", 20);
Id2Command((int)off_405048, "WaitForSingleObject", 19);
Id2Command((int)off_40504C[0], "ReleaseMutex", 12);
Id2Command((int)off_405050[0], "GetLastError", 12);
Id2Command((int)off_405054[0], "GetTickCount", 12);
Id2Command((int)off_405058, "Sleep", 5);
Id2Command((int)off_40505C, "GetTempPathA", 12);
Id2Command((int)off_405060, "GetTempFileNameA", 16);
Id2Command((int)off_405064, "s9a8mzjXWUUPFN6i", 16);
Id2Command((int)off_405068, "GoogleUpdateBeta", 16);
Id2Command((int)off_40506C, "SOFTWARE\\Google\\Google Updater", 30);
Id2Command((int)off_405070, ".exe", 4);
Id2Command((int)off_405074, "goog", 4);
Id2Command((int)off_405078, "open", 4);
Id2Command((int)off_40507C[0], "HELLO\n", 6);
Id2Command((int)off_405080[0], "READY\n", 6);
Id2Command((int)off_405084[0], "READT\n", 6);
Id2Command((int)off_405088, "READD\n", 6);
Id2Command((int)off_40508C, "ok", 2);
Id2Command((int)off_405090, "badpass", 7);
Id2Command((int)off_405094, "session:", 8);
Id2Command((int)off_405098, "/svc", 4);
Id2Command((int)off_40509C, "@%s:%s:%d\n", 10);
Id2Command((int)off_4050A0, "%s:%s\n", 6);
Id2Command((int)off_4050A4, "GUID", 4);
Id2Command((int)off_4050A8, "value", 5);
Id2Command((int)off_4050AC, "svalue", 6);
Id2Command((int)off_4050B0, "%.8X", 4);
return Id2Command((int)off_4050B4, "%d", 2);
```

Conclusion

In the next article in this series, we'll consider the bot, kernel mode and bootkit functionality of the malware.

TDSS part 2: Ifs and Bots

Posted: April 19, 2011

Author

ESET Team

VIEW PROFILE

Aleksandr Matrosov is a Senior Malware Researcher at ESET. He is also a Lecturer at the Cryptology and Discrete Mathematics at National Nuclear Research University MEPh. He specializes in the analysis of malicious threats and cybercrime activity. Eugene Rodionov is a malware researcher for ESET. Rodionov also holds the position of Lecturer at the National Nuclear Research University MEPhI in Russia. His interests include kernel-mode programming, anti-rootkit technologies, reverse engineering and cryptology. David Harley is a Senior Research Fellow at ESET. He is a Director of the Anti-Malware Testing Standards

Organization, Chief Operations Officer at AVIEN, and CEO of Small Blue-Green World. He is a Fellow of the BCS Institute and holds qualifications in security management, service management (ITIL), BSI security audit and medical informatics.