


# RawPOS Technical Brief

TrendLabs Security Intelligence Blog

April 2015



# Table of Contents

- Introduction .....3
  
- Threat Details .....3
  - Background on RawPOS .....3
  - Chain of Compromise .....4
  - Design of the Malware .....5
  - RawPOS Component Evolution .....14
  
- Methodology .....16
  
- Point-of-Sale Targets .....17
  
- Solutions and Recommendations .....20
  
- Indicators .....20



## Introduction

There is no denying that point-of-sale (PoS) malware have made their mark in the threat landscape. As we have noted in our [annual security roundup](#), 2014 was, in fact, [the year of PoS malware](#), with security incidents coming in at least once a month.

We have noted that new variants of PoS malware like Alina emerged as direct evolutions of older PoS RAM scraper families like Backoff. But this doesn't mean that older PoS malware families have been abandoned.

We noticed that casinos and resort hotels are the most recent victims of [an attack that used RawPOS](#), an old POS malware, to steal customer data. Touted as the earliest of its kind, very little research and documentation exists about RawPOS. As such, we will attempt to give light on this threat that may have been instrumental to previous credit card breaches documented and not previously attributed to this particular PoS threat.

## Threat Details

### Background on RawPOS

The earliest reference to RawPOS could be traced back to October 2008, with a [Visa Data Security Alert](#) that talks about debugging or parsing memory of point-of-sale systems to extract the full magnetic stripe data from volatile memory. Files of similar note have been mentioned in the following advisories as well:

- November 2008: [Visa – Malicious Software](#)
- May 2009: [First Data / Visa Data Security Alert – Targeted Hospitality Sector Vulnerabilities](#)
- June 2009: [First Data / Discover Data Security Alert - Hospitality Point-of-Sale Breaches](#)
- November 2009: [Visa – Targeted Hospitality Sector Vulnerabilities](#)

During our research, we also encountered recent articles and advisories that reference details related to this threat. This just shows that references to this PoS threat were made in the past and were well-documented—just not attributed to the malware family. Recent findings include the following:



- September 2014: [CK Systems – Data Compromise at our Managed Services Hosting Facility](#)
- February 2015: [Discover – Data Security Alert: Retail Data Security Breaches](#)
- March 2015: [Visa – “RAWPOS” Malware Targeting Lodging Merchants](#)

It is quite clear that, while RawPOS has been encountered over at least seven years ago, threat actors have been using it repeatedly - with success.

## Chain of Compromise

The [2008 Visa Data Security Alert](#) contained very important pieces of metadata, which we've added a few notes (in red) for clarity.

FILENAME	PURPOSE
csrsvc.exe	Memory dumper // generally found on the initial controlled host
MemPDumper.exe	Memory dumper // distributed to multiple hosts to dump credit card data
dnsmgr.exe	Track Data parser // file scraper, used to parse dumped data from the memory dumpers
dirmon.chm	Output file from track data parser programs
WinMgmt.exe	Calls csrsvc.exe and dnsmgr.exe and runs an interactive command shell on TCP Port 3373
install.bat	Batch file that installs WinMgmt as a windows service.
dump.bat	Batch file installs memory dumper program on a single computer
psexec.exe	Sysinternal tool used to run process on remote machines.
play.bat	File that calls install.bat file to install memory dumper on multiple systems.
Far.exe	Dos based file manager used by attacker
compenum.exe	Network scanner which outputs a list of accessible systems
shareenum.exe	Network scanner which outputs a list of accessible shares
lanst.exe	Given a (local or networked) location, parses and normalizes memory dumped text files

Table 1. Indicators listed from the October 2008 Visa advisory

While missing one file (*lanst.exe*, which would be found later), this file listing details a very accurate tool set of attackers that use RawPOS. This simple and short two-page report was really enough to imagine how attackers would've penetrated environments of interest.

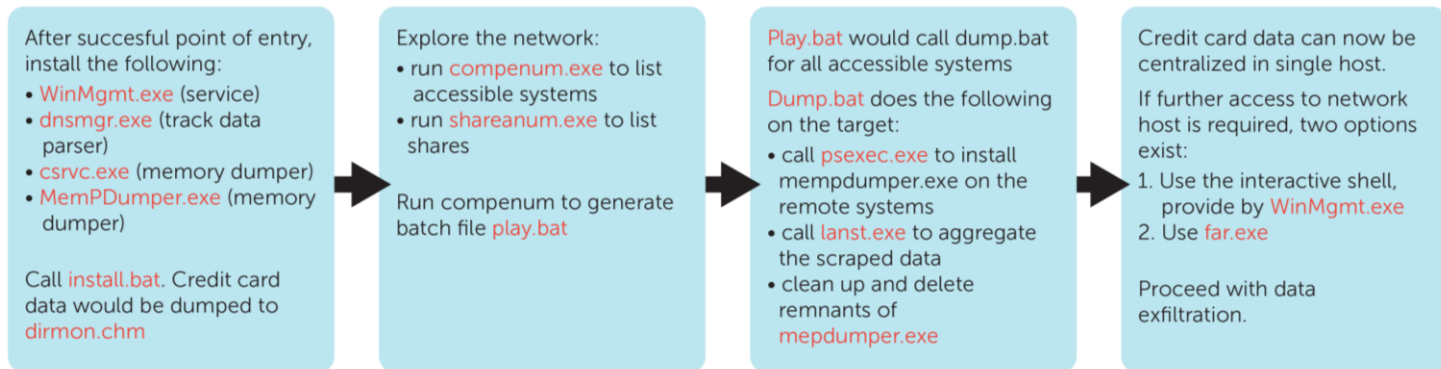


Figure 1. How a compromise would have happened, with the use of the indicators

## Design of the Malware

Before we start this section, we'd like to note that there are other indicators surrounding the actual compromise of the environment, including, but not limited to, *psexec.exe*, *far.exe*, *radminshare.exe*, *rstopservice.exe*, etc. While these are generally good files used for maligned intent, this section is dedicated to the malicious files detected by Trend Micro.

There are generally 3 components of the malware itself.

1. The service used for persistence

The service is used to maintain access on the compromised system by offering a backdoor by installing itself as a service, and making sure that the memory dumper and the file data scraper are launched.

```

int __cdecl main(int argc, const char **argv, const char **envp)
{
    int result; // eax@11
    CHAR v4; // [sp+Ch] [bp-114h]@12
    LPCSTR ErrorString; // [sp+10Ch] [bp-14h]@12
    SERVICE_TABLE_ENTRYA ServiceStartTable; // [sp+110h] [bp-10h]@1

    memcpy(&ServiceStartTable, &ServiceTableEntry, 0x10u);
    if ( argc > 1 && (*argv[1] == '-' || *argv[1] == '/') )
    {
        if ( !_mbicmp("install", argv[1] + 1) )
        {
            InstallMalService();
            goto _exit;
        }
        if ( !_mbicmp("remove", argv[1] + 1) )
        {
            RemoveMalwareService();
            goto _exit;
        }
        if ( !_mbicmp("debug", argv[1] + 1) )
        {
            gbDebug = 1;
            MalDebugMode();
        }
    }
    _exit:
    exit(0);
}
printf("\nStartServiceCtrlDispatcher being called.\n");
printf("This may take several seconds. Please wait.\n");
result = StartServiceCtrlDispatcherA(&ServiceStartTable);
if ( !result )
{
    ErrorString = v4;
    w_ErrorFormating((int)"StartServiceCtrlDispatcher", &v4);
    LOWORD(result) = wReportEvent(1u, 0xC00003E9u, 1u, &ErrorString, 0);
}

```

Figure 2. Service handling functions for persistence

File samples collected for the service have gone through some obvious code changes as a good majority of them did have the capability to launch the backdoor routine to listen at Port 3373, but were not called. To further study this component, certain code changes to restore this functionality had to be done on the existing file sample and it presented us with a command-line functionality once connected to Port 3373, and it opened in a hard-coded location of *c:\winnt\system32* – a good indication that this had been designed to be used in an older environment since this path was more popular with Windows NT 4.0/2000.

```

Password: bartalamy!
Welcome, BOSS!
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

c:\winnt\system32>_

```

Figure 3. What the backdoor functionality would have looked like

To date, there exists another version of the service that had totally removed the backdoor component. This version was first seen used around 2012, and is the version being used to date.

```

if ( Msg == 1 )                                     // WM_CREATE
{
    if ( WSASStartup(0x101u, &WSAData) )
        return -1;
    gSocket = socket(2, 1, 0);
    if ( gSocket == -1 )
        return -1;
    name.sa_family = 2;
    *(_DWORD *)&name.sa_data[2] = htonl(0);
    *(_WORD *)&name.sa_data[0] = htons(3373u);
    if ( bind(gSocket, &name, 16) == -1 )
        return -1;
    if ( listen(gSocket, 1) == -1 )
        return -1;
    if ( WSAAsyncSelect(gSocket, hwnd, 0x401u, 43) == -1 )
        return -1;
    gSendPipe = 0;
}
else
{
    if ( Msg == 2 )                                   // WM_DESTROY
    {
        WSACleanup();
        closesocket(gSocketAccepted);
        closesocket(gSocket);
        PostQuitMessage(0);
    }
    else
    {
        if ( Msg != 0x401 )                           // WM_specific to this malware
            return DefWindowProcA(hwnd, Msg, wParam, lParam);
        MalPipeMessaging((int)hwnd, wParam, lParam);
    }
}
return 0;

```

Figure 3. The removed backdoor component of the installed service

The service component is worded very well to look like valid Windows services, with all characteristics about them looking very convincing, though there is some obvious reuse in some cases.

SERVICE NAME	DISPLAY NAME	DESCRIPTION
WinMgmtHelp	Windows Management Help Service	Additional services for Windows Management Instrumentation (WMI).
MonSync	Serial Port Synchronization Monitor	Monitors the synchronization of connected serial ports to this computer. If this service is stopped, serial port devices could not work properly.
wproxym	Window Network Switching Compatibility	Maintains date and time synchronization on all clients and servers in the network. If this service is stopped, date and time synchronization will be unavailable. If this service is disabled, any services that explicitly depend on it will fail to start.
sppt32	Microsoft Support	Maintains date and time synchronization on all clients and servers in the network. If this service is stopped, date and time synchronization will be unavailable. If this service is disabled, any services that explicitly depend on it will fail to start.
inetsrv	Internet Information Services	The Internet Information Services must be started to be able to communicate with the microsoft.com update servers.
CertSvc	Certificate Services	Creates, manages, and removes X.509 certificates for applications such as S/MIME and SSL. If this service is stopped, certificates will not be created. If this service is disabled, any services that explicitly depend on it will fail to start.
MSFSVC	Microsoft File Manager Services	Creates, updates, manages, and removes files for applications such as S/MIME and SSL. If this service is stopped, certificates will not be created. If this service is disabled, any services that explicitly depend on it will fail to start.

Table 2. Service component made to appear like legitimate Windows services

## 2. The memory dumper

This is used to dump the memory of specific processes. Two dumpers were usually used: one generic dumper that can be called to dump a specific process, and another dumper that is designed for specific processes that target specific PoS applications.



- The generic dumper, as shown below, can be configured to monitor a specific process to dump the memory to disk.

```
Installing windows updates...
Process Memory Dumper
Made By: DiabloHorn (Proud Member of: KD-Team)
Use as: memdump.exe -<options> [PID]
Options:
    -? = Show this help
    -l = List all running processes
    -s = show info on Process like Path
    -f = Dump private process memory by PID
    -f = Full private dump of all running processes
%s      (PID: %u)      Hex: %xh
```

```
Installing windows updates...
Proc3ss M3mory Dump3r
Made By: Diabl0H0rn (Proud Member of: KD-Te@m)
Use as: m3mdump.exe -<options> [PID]
Options:
    -? = Show this hlp
    -l = List all running processes
    -s = show info on Process like Path
    -f = Dump private process memory by PID
    -f = Full private dump of all running processes
%s      (PID: %u)      Hex: %xh
```

Figure 5. Options of the generic memory dumper, with the newer example on the right side

The string-wise impression of the file has references to *DiabloHorn* and *KD-Team* is visible in the vast majority of the samples collected. However, later examples have changed and stylized some characters on this text, presumably in attempt to evade prior signature detection.

Another observation is that this binary is often time-bound as it is approximately timed not to execute about one month after its compile time. As such, this would ensure that, if the attacker is not able to get back to the environment in time, malicious activity would stop. This would deter most dynamic analysis of the file (i.e., sandbox execution) if run at a later date.

```

result = time(0);
if ( result <= 1429213820 )                               // Thu, 16 Apr 2015 19:50:20 GMT
{
  for ( i = 1; argc > i; ++i )
  {
    result = (int)argv[i];
    if ( *(_BYTE *)result == '-' )
    {
      charOption = (int)argv[i];
      switch ( *(_BYTE *)(charOption + 1) )
      {
        case 'L':
        case 'l':
          ListProcesses(result);
          exit(0);
          return result;
        case 'S':
        case 's':
          dwProcessId = atol(argv[i + 1]);
          ShowProcessInfo(dwProcessId);
          exit(0);
          return result;
        case 'D':

```

Figure 6. A recent example used, compiled last March 17, 2015

- The process-specific dumper, on the other hand, is derived from the generic dumper where the options listed above no longer apply as it had a hard-coded list of specific processes to look for. Two versions have been observed. The first one version directly wrote the memory to the file, and checked the internally-defined processes every 15 seconds.

```

system(make_mendump);                                     // mkdir mendump
while ( 1 )
{
  result = system(del_all_mendump);                       // del mendump\*.dmp
  doRamScrapping(result);
  sleep(15);
}

```

Figure 7. The earlier process-specific memory dumper waits for 15 seconds

The latter version, which seems to have started in 2012, is significantly different. After finding the target process, it finds the credit card information via regular expression. The change on this part of the process-specific memory dumper was probably an enhancement to make credit-card data scraping more efficient as the memory dumper would be more specific in matching the data it wanted to dump, and the file data scraper (discussed below) was enhanced to control the timing as to when memory dumper would execute.

```

if ( result )
{
  if ( EnumProcessModules(result, &hModule, 4, &v11) )
    GetModuleBaseNameA(hProcess, hModule, &szModuleName, 260);
  CloseHandle(hProcess);
  result = (void *)sprintf(s1, "%s", &szModuleName);
  dwCurrentPID = dwProcessId[dwPtr];
  if ( dwProcessId[dwPtr] != 4 )
  {
    result = (void *)stricmp(&szModuleName, "utg2.exe");
    if ( !result )
    {
      if ( bDump )
      {
        sprintf((char *)&buffer, "memdump\\%s-%d.dmp", &szModuleName, dwProcessId[dwPtr]);
        hObject = CreateFileA(&buffer, 0x40000000u, 0, 0, 2u, 0, 0);
      }
      result = (void *)DoRamScrapping(dwProcessId[dwPtr]);
      if ( bDump )
        result = (void *)CloseHandle(hObject);
    }
  }
}

```

Figure 8. A process-specific memory dumper that monitors for utg2.exe component of the Shift-4 credit-card processing application

It would seem that there is a builder for the memory dumper itself since the file construction seems consistent across the samples collected. This also gives the first indication of how customized RawPOS is as variations of the memory dumper can be quickly generated. A more detailed list of credit card applications that were targeted by this PoS threat is listed in an upcoming section titled *Point-of-Sale Targets*.

### 3. The file data scraper

The scraper, on the other hand, parses the dumped files from the memory dumper and scrapes the credit card data. The file data scraper is also in charge of encoding the dumped data. File samples surrounding this component are sparing, but the few samples that we do have do give insight on how this PoS threat has evolved.

This first example below had an explicit lifetime as specified by the *\$dietime* variable and, in this case, it was March 6, 2015. This version is quite simple as its sole purpose is to accept a file location and go through the indicated files to look for credit card data within the memory-dumped files. This early example was called by the service component and did not encrypt the data.

```

#!/usr/bin/perl# Подключаем основные модули
# 24.03.2008
# - При обнаружении в файле данных соответствующих строке поиска (regex)
#   прекратить обработку файла.
#perl2exe_include "Tie/Handle.pm";
#perl2exe_include "Math/BigInt/Calc.pm";
use strict;
use warnings;
#use DBI;
use FileHandle;
use Win32API::File::Time qw(:win);
use threads;
use POSIX qw(floor);
use Win32::Console::ANSI;
no warnings 'threads';
require "D:\\Secure\\Tools\\Include\\times.pm";
$|=1;

use vars '$dbh', '$url_start', '$dir_start', '@dir_filter', '@file_type_exclude','$version','$regex','$maxlifetime','$debug';
$version="Version 1.3 MultiThread from 25.03.2008";
#$regex = '([0-9]{15,19}(=|D)1[0-9]((0[1-9])|(1[0-2]))[0-9]{8,20})';
$regex =
'((B([0-9]{13,16})|([0-9]|\s){13,25})\^\^[A-Z\s0-9]{0,30}\|[A-Z\s0-9]{0,30}\^\^[0-9]{2}((0[1-9])|(1[0-2]))([0-9]|\s:
))[0-9]{8,30})|(<Field name="CardNumber">[0-9]{15,19}</Field>|(~CCM[0-9]{15,19}D[0-9]{4~}))';
$maxlifetime = 86400*30*6; # последнее обновление файла, примерно пол года
$debug = 'off';
if ( time > $dietime ) { die("Can't open Handle/Tie.PM!"); };

```

Figure 9. Earlier example of the file data scraper, circa 2008

Another example, as seen below in Figure 10, had taken off from a 2013 modification to include features that we had recently encountered.

```

#!/usr/bin/perl# Подключаем основные модули
# 24.03.2008
# - При обнаружении в файле данных соответствующих строке поиска (regex)
#   прекращать обработку файла.
# 24.09.2013
# - Шифрование кеша.
# - Обнуление mtimes для удаленных файлов (незначительный баг).
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="dxdiag32.dll";
my $command = "rptsvc32.exe";
my $commandruntimelimit = "300";
my $commandstarttime = 60;
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', '$dir_start', '@dir_filter', '@file_type_exclude', '$version', '$regex', '$maxlivetime', '$debug';
use vars '%mtimes', '%atimes';
$version="Version 1.3 MultiThread from 25.03.2008";
my $regex = '([0-9]{15,19}(=|D)1[0-9]((0[1-9])|(1[0-2]))[0-9]{8,20})';
my $maxlivetime = 86400*30*6; # последнее обновление файла, примерно пол года
$debug = 'off';
my $dietime = 5;
if ( time > $dietime ) { die("Can't open Handle/Tie.PM!"); };

```

Figure 10. More current example file data scraper

This modification was more sensitive to file times and time zone differences, aside from inheriting the encoding of the dumped credit card data with XOR. Furthermore, this version now also controls the execution of the memory dumper in specific configurations, like every five minutes during the day. This version that now calls the memory dumper is what we've seen as the latest fork to the 2008 version. In the example above, it is *rptsvc32.exe*, the name of the memory dumper.

These latter versions now include more verification on the data by comparing the values in the included file `D:\Secure\Tools\regex-t.pm`. The reference to `D:\Secure\Tools\Include\times.pm` still maintains the approximate time as to when to run its routines.

```
$regex = '(B{13,16}|([0-9]|\s){13,25})\^[A-Z\s0-9]{0,30}\/[A-Z\s0-9]{0,30}\^[0-9]{7-9}|1[0-9]|(1[0-2])|([0-9]|\s){3,50}|([0-9]{15,16}([A-Z]|=)(0[7-9]|1[0-9])|(0[1-9])|(1[0-2]))[0-9]{8,30}|(<Field name="CardNumber">[0-9]{15,19}</Field>|(~CCM[0-9]{15,19}D[0-9]{4~})|([0-9]{15,19}(=|D)1[0-9]([0-1-9])|(1[0-2]))[0-9]{8,20})';
D:\Secure\Tools\Include\regex-t.pm (END)

$dietime = 1404485023;
D:\Secure\Tools\Include\times.pm (END)
```

Figure 11. Regular expression is used to match credit-card values, and uses a finite time to execute

One of the interesting things about the file data scraper is a direct reference to *icverify.lrq*, which relates to [First Data's ICVERIFY software](#). Another is the fact that comments on the Perl script is in [Code Page \(CP866\)](#), which is required to support languages in the Cyrillic script. In fact, some lines of the Perl script make implicit references to the [Windows-1251 character encoding](#), and the comments are written in the Russian language.

## RawPOS Component Evolution

These three components have gone through extensive reuse over the years.

### 2010 and older

Finding files listed at the different advisories released by credit card companies would probably be the best reference for this. The earliest year that the files have been catalogued is around 2008, though more could have existed. Both Table 1 and Figure 1 are based on these indicators. This PoS threat, then unnamed, piqued interest in several write-ups, [such as this one](#), that documented how the memory dumper and file data scraper worked.

### 2012 – 2013

Seemingly, there was very little activity seen around 2011, in terms of file variations. Newer versions of the files were seen around 2012-2013, and perhaps the [only known reference](#) on how it worked was written in 2014 after the Goodwill breach. While still faithful to the three-component modular design, several changes were observed: the service no longer was bundled with the backdoor

component, and the file data scraper started to encode (via XOR) the dumped credit card information.

### 2014 - 2015

More indicators were gathered during this time. However, the service no longer called the memory dumper directly. Rather, the memory dumper is now being called upon by the file data scraper.

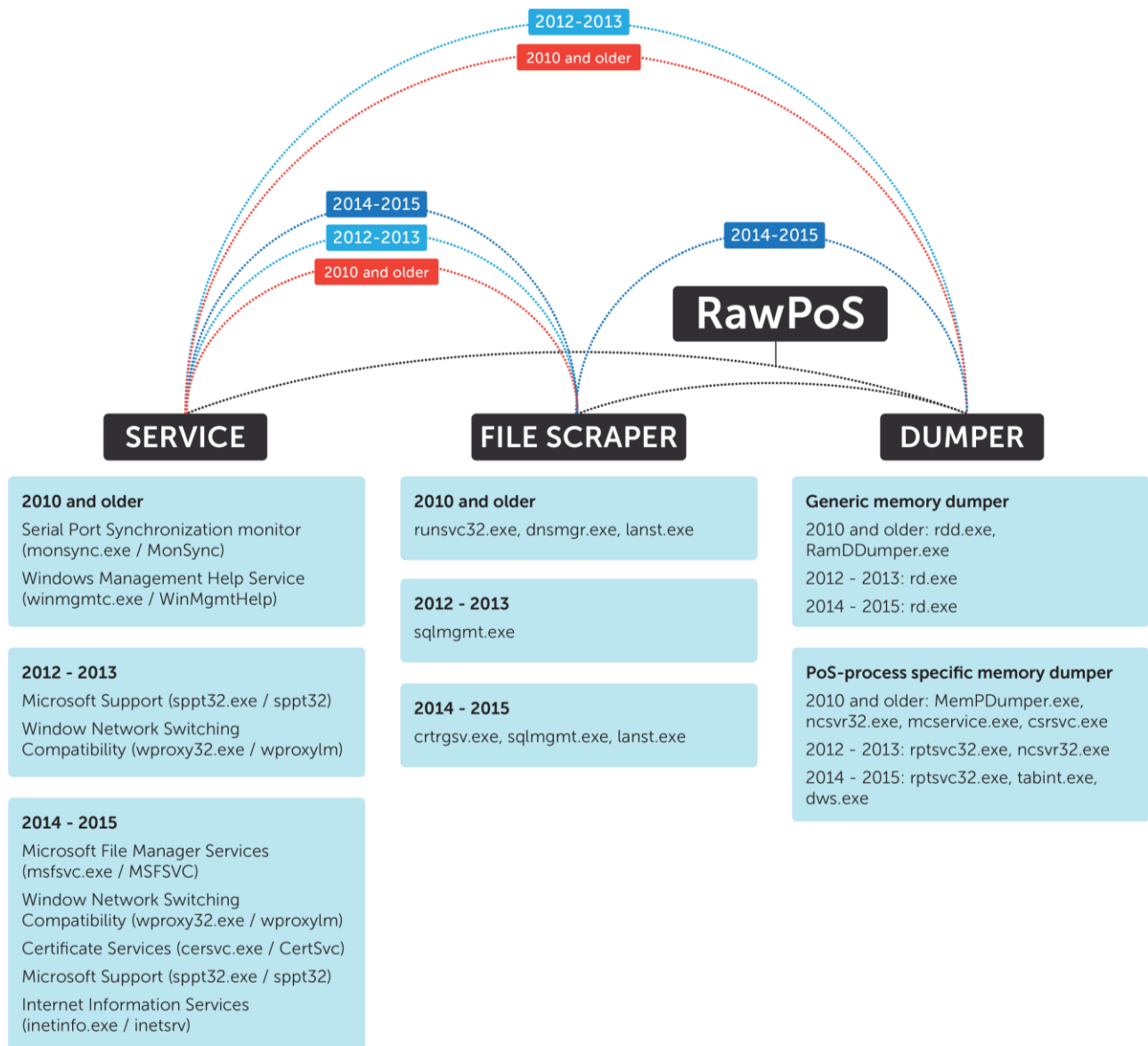


Figure 12. Evolution of the RawPOS components

As seen in Figure 12, the threat actors behind RawPOS has been able to use, reuse and re-engineer RawPOS over time to adapt to the changing times and often times adapting to the target environment.

## Methodology

In analyzing RawPOS, we can see that its modular design is highly configurable and has always been a multi-stage scraper. Brought about probably by pioneers in PoS malware threats, the design they chose has now proven to be enduring up to today:

- The multi-stage or multi-component strategy ensures a high success rate for the chosen environment, while making prevention and detection harder—no matter what type of solution.
- The threat is still successfully victimizing businesses, and the threat actors behind it are very familiar with how networks within small-to-medium business segments are designed.
- It is fault-tolerant, persistent and very specific – incident responders and threat investigators may chance upon a specific file that has only been deployed for that specific business.

This specific PoS threat's modular design enables the threat actors to tailor-fit the binaries for the target environment, and it would not be unusual to encounter different file hashes of the service, memory dumper, and file data scraper. It should be further noted that, if encountered in an environment, all three components may have been specifically built and compiled for the environment where they are found:

- The time references found in the memory dumper and the file data scraper would be approximately timed for one month upon initial infection.
- The memory dumper component would have a hard-coded list of processes to dump that is specific to the target PoS being used by the environment.

In the aforementioned report from Visa about RawPOS, file hashes were identified and we have mapped these files as part of the attacker's footprints for lateral movement.



SHA1	Original Filename	Notes
44375ddceb7f24a2e92a841e8275218dbb30401f7af9af15a682e353bc5fdf45e68151c23697b124	ent.exe netdom.exe	Essential NetTools 2.2 (build 51) NetDom 1.8 (Windows domain manager), tool to reset the secure channel between a workstation, server, or domain controller.
b399eaa47b6b1a6431a6ad84430185cfc019221d	cmdpause.exe	Limits and terminates the number of <i>cmd.exe</i> processes
24275065cc31b350752790a486890b0ca1e51a58	cmdpause.exe	Limits and terminates the number of <i>cmd.exe</i> processes
af572669accb64e54af626c3a7ef19b01c9edd64	compenum.exe	In <i>cmd</i> , enumerates network resources/existing connections with format Domain: { from WNetEnumResource}
efbc197aa2879f11cf440afc5351496803092755655b1293f322c862027bed69e8bc22eb4655d79c	psex.exe radminshare.exe	PsExec v1.72 Usage: NetShareAdd server. Re-shares admin\$
d35fff630352592dfc56f12c043fdab1f2881421	rstopservice.exe	Remote StopService. Ver 1.0 from 10/09/2005 Usage: rstopservice <computername> <servicename>
187beea1ed4738cd3af648b6ee51d631fc059a71	sdelete.exe	SDelete 1.51 by SysInternals. Securely deletes data.
823263c6f72a1733dd11f292103332631a4b446f	ud.exe	This program changes a file's date/time to July 15,2003 12:21:12 PM

Table 3. Files used for lateral movement

There was no command-and-control (C&C) module discovered and we can readily assume that the threat actors went back for the dumped credit card data at a later time, or had used other facilities to extract the data from the environment. After lapsing one month, the malware remnants would usually be left behind.

## Point-of-Sale Targets

Trend Micro has observed RawPOS to target casinos, resorts, and luxury hotels in the United States and Canada, with a few in Europe, Middle East and Latin America. This matches the information Visa shared in a [security alert in March 2015](#).

Since different business establishments would have different PoS software, threat actors that employ RawPOS have modified its code to support multiple PoS software over time. We also saw that attackers may have used the generic memory dumper to enhance the process-specific dumper as they encounter it, dropping or adding of specific processes as they learn about newer environments. This prevented repeat work of guessing which process within the PoS software suite to dump that may have the credit card information.

It should be noted that the list in the next page is compiled against what Trend Micro had seen in terms of file samples. While this PoS software listing tries to be as complete based on this file samples we have acquired, RawPOS and its components are highly configurable and we can certainly be sure that RawPOS has been modified to adapt to more PoS software.



2010 and older	2011-2013	2014-2015
<p>Aloha POS Table Service                      Cherry USB driver in HP POS                      Clip Consultants Digital File Management                      Concept Software Systems                      Fidelio XPRESS                      Firebird POS                      Infineon Technologies                      InfoGenesis / Shift 4 software                      InnQuest Software roomMaster                      Intrix SuperCharge                      MICROS 9700 and RES                      Menusoft Systems Corporation DDCCSrv                      Micros Fidelio                      Midnite Express Slipstream                      Navision Software Financials                      Novacom NovaTouch POS                      Passepartout Retail POS                      Executables for Oracle Forms                      Profit Series Hospitality Solutions (also MICROS)                      Sequiter Software Inc                      Shift4 Corporation Credit-card Processing / Transaction Gateway                      Siriusware Protobase                      Wincor-Nixdorf TradePos Pos32</p>	<p>911 Software CCV Server                      AGILSYS POS                      Accelerated Payment Technologies                      Action Systems Restaurant Manager                      Aldeo for Restaurants                      Aloha POS Table Service                      Blackboard Inc Dining Services                      CAM Commerce Solutions                      COSMOS POS                      Chase Payment Tech Internet Gateway                      Cherry USB driver in HP POS                      Data Business Systems POS                      Datacap Systems NetePay                      Datawire Network Access Module                      Fiducial Informatique                      Financial Payments, LLP Card Issuance                      FirstData Payment                      Future POS                      HIS Secure Payment Gateway                      IBM POS                      ISS Retail OmniPOS                      Image Technology Systems                      InfoGenesis / Shift 4 software                      Interactive Designs Salon Touch                      Interbase POS                      Micros 9700 and RES                      Mauve System3                      Menusoft Systems Corporation                      Micros Fidelio                      Microsoft Store Operations                      Midnite Express Slipstream                      Oracle Forms and Application Server                      PixelPoint POS                      QA/1 Data Systems Central Credit Card                      Quest Retail Technology                      RDC Inc. POSitouch                      Radiant/NCR Counterpoint POS                      Retail Pro 8/9 International                      Rocket Software                      SKIDATA AG                      Sage Software                      Shift4 Corporation Credit-card Processing / Transaction Gateway                      Siriusware Protobase                      Verifone PCCharge Payment Server                      pcAmerica Desktop POS</p>	<p>InfoGenesis / Shift 4 software                      Micros 9700 and RES                      Micros Fidelio                      Executables of Microsoft                      Oracle Forms and Application Server                      Profit Series Hospitality Solutions (also MICROS)                      Radiant/NR Counterpoint POS                      Retail Pro 8 International                      Shift4 Corporation Credit-card Processing / Transaction Gateway                      Siriusware Protobase</p>

Table 4. Supported PoS software

## Solutions and Recommendations

Attackers who employ RawPOS as their means of stealing credit card data may have modified RawPOS according to the target environment. That being said, dealing with RawPOS is not an easy task since the files could have been generated specifically for that environment – the service names may have changed, as well as any file name reference and timed execution events for the memory dumper and file data scraper. Also, the attackers use standard administrative tools (like *psexec.exe*, *far.exe*, *radminshare.exe*, *rstopservice.exe*, etc.) for lateral movement and data collection.

Trend Micro has released a [paper](#) that discusses possible combinations that business may decide to employ for protection against PoS threats. For endpoint monitoring and validation for possibly active infections, [Trend Micro Deep Discovery Endpoint Sensor](#) is a context-aware endpoint security monitor that is designed to speed up the discovery, investigation, and can be used to assess endpoints for possible compromise – including the use of the YARA rule shared below.

## Indicators

While Trend Micro detects a lot more files than the ones listed below, these are the files that we have based our analysis from.

SHA1	Compile Time	Size	Detection	Notes
608543398f1ee27c12ea1fcc583a1952dfd8829b	3/23/2015	64,000	TSPY_RAWPOS.SM1	Service
4ba983396ecab355d3e4c84fd7d13ca28dfd9af4	3/19/2015	64,000	TSPY_RAWPOS.SM1	Service
4a240bf2192a9d2cbdbf28d05cc4edb2524e9834	3/17/2015	197,632	TSPY_RAWPOS.SM	Memory dumper
b7149a491f35ab045bb14974bcbb32e7bdc083dc	2/23/2015	64,512	TSPY_RAWPOS.SM1	Service
66f671d27a36a970698de1e97a4e1f69e4d85b3b	2/18/2015	197,632	TSPY_RAWPOS.SM	Memory dumper
23988632314c4739a63b252efe6ef8ed64756d77	2/10/2015	197,632	TSPY_RAWPOS.SM	Memory dumper
8e5cafaf3ed6b4f2675dd287a98882f8b85028c5	2/10/2015	64,000	TSPY_RAWPOS.SM1	Service
747faf9eb98f4e8cc13fb1bd3204b9584b326d6f	1/26/2015	197,632	TSPY_RAWPOS.SM	Memory dumper
91f9631fced951ebff9877a8e97e0ce84fb7eb58	1/19/2015	197,632	TSPY_RAWPOS.SM	Memory dumper
a5a34e1d280c27de33823a1b0282b4d9cfd815b8	12/30/2014	197,632	TSPY_RAWPOS.SM	Memory dumper
e6eb1c8a7c01450f7c6a850dd0345611929db418	12/29/2014	64,000	TSPY_RAWPOS.SM1	Service
ec3a3bb760ef5bad58557600d592792e67c272b6	12/5/2014	197,632	TSPY_RAWPOS.SM	Memory dumper
06fcbab63ffd8fccff7527a38d69d65dbe20feae6	11/28/2014	197,632	TSPY_RAWPOS.SM	Memory dumper
eab3fe9f70dae82a7e4663b65348df0337cd94cf	11/28/2014	64,000	TSPY_RAWPOS.SM1	Service
878064491316dd7ef9f9c4e274fd14d639e4de33	11/3/2014	197,632	TSPY_RAWPOS.SM	Memory dumper
993ed91023f1927ba0bb9542926a8688d702db47	10/23/2014	197,632	TSPY_RAWPOS.SM	Memory dumper

7d3556c6cebc15cb57a357721a00dc21fa928212	8/5/2014	197,632	TSPY_RAWPOS.SM	Memory dumper
ad1b093e3ea4178f38559e92a061212cb3844bb0	7/4/2014	197,632	TSPY_RAWPOS.SM	Memory dumper
40fea895857a1257635ad773ef7d01340594512d	7/2/2014	64,000	TSPY_RAWPOS.SM1	Service
146bf418caaa73a62cd8121dd453774b22b59794	6/18/2014	197,632	TSPY_RAWPOS.SM	Memory dumper
9e2f682a81d9dc654500da763e64db533fa124ad	6/18/2014	64,000	TSPY_RAWPOS.SM1	Service
edb557cd1e79401537910eeb892d33bf31d333e0	6/11/2014	197,632	TSPY_RAWPOS.SM	Memory dumper
b8b73c88b6684e8e03d1e3b46e0d54cfbb1a58bd	6/4/2014	197,632	TSPY_RAWPOS.SM	Memory dumper
c88fb6ae34813b1f2b5074658ccc0a73be5ebb78	5/28/2014	197,632	TSPY_RAWPOS.SM	Memory dumper
b8af714be5869d1efaae08674cb5187a467958fc	4/26/2014	197,632	TSPY_RAWPOS.SM	Memory dumper
e32bbbd2337b4f5ff89564dfe8fe72edc566d2f9	4/25/2014	197,632	TSPY_RAWPOS.SM	Memory dumper
4fe427ed761670e3271ad278b56cff3629e20524	4/21/2014	64,000	TSPY_RAWPOS.SM1	Service
b395124e9013ce9f0374c1897bc3faa8df3605a8	1/11/2014	197,632	TSPY_RAWPOS.SM	Memory dumper
172f0b8186213f1e3f915303a318dcd16a3dfd47	12/13/2013	64,000	TSPY_RAWPOS.SM1	Service
b93c0346aac2679f73e7dbae5833e4e88cc90fc2	11/14/2013	197,632	TSPY_RAWPOS.SM	Memory dumper
aec4fd09e003d76570186c0d6f7bfbc90aa542e4	10/18/2013	197,632	TSPY_RAWPOS.SM	Memory dumper
5022b7ad076ee84bd53586e36087108fb985fe74	10/18/2013	197,632	TSPY_RAWPOS.SM	Memory dumper
57e3df25d0f2b70b0f1c585e04b49177d959d48b	10/7/2013	197,632	TSPY_RAWPOS.SM	Memory dumper
9ced23a36404180f358dd30fdcdc46d08202a7c1	9/17/2013	198,144	TSPY_RAWPOS.SM	Memory dumper
eeab9b95b532838e27b4d9d3a02d0602dfd3492c	9/12/2013	197,632	TSPY_RAWPOS.SM	Memory dumper
ec15afbb142b47a8a9572ff895790f4c5d80c859	9/1/2013	197,632	TSPY_RAWPOS.SM	Memory dumper
cafb96334eb53760fec329bef099035c748ef35a	6/19/2013	197,632	TSPY_RAWPOS.SM	Memory dumper
ede7b6251d5a8d91e7c1f053278b9df7af5ea400	6/13/2013	197,632	TSPY_RAWPOS.SM	Memory dumper
abccdf07186438cb89e81199526be35fd705445f	6/4/2013	64,512	TSPY_RAWPOS.SM1	Service
81a4024c83967667340e778fa8a27aa8cfdc6442	6/4/2013	64,000	TSPY_RAWPOS.SM1	Service
4b5bcd5a748d3aee55ac335ef01a3f9410a2511a	3/20/2013	197,632	TSPY_RAWPOS.SM	Memory dumper
16e9832dac1f4c9489ffb683d419a2a9f0c3ebd0	3/6/2013	197,632	TSPY_RAWPOS.SM	Memory dumper
f8381738e6035704b4396414148a646c0fe7a530	2/12/2013	197,632	TSPY_RAWPOS.SM	Memory dumper
7d2bb1b3f8e5818059d2e2c71e7886bc99e61de7	1/21/2013	199,680	TSPY_RAWPOS.SM	Memory dumper
ef0b6a818a59681d73d57d4f077c512f87efc3aa	1/13/2013	199,680	TSPY_RAWPOS.SM	Memory dumper
b87894cd92dc5e6003cdb5a0ee701691379d6298	1/4/2013	197,632	TSPY_RAWPOS.SM	Memory dumper
600985fd6ba013a4e512f50912dd242ad9926356	12/23/2012	199,680	TSPY_RAWPOS.SM	Memory dumper
197459391ce6d95808637e0033dfb2bd5c14260a	11/16/2012	197,632	TSPY_RAWPOS.SM	Memory dumper
90087927e924dfb433b3d1d809630d84e797aaaae	9/27/2012	199,168	TSPY_RAWPOS.SM	Memory dumper
9d04e6d0ce614a4a67a73e7400388f04fcb34c0c	7/13/2012	198,656	TSPY_RAWPOS.SM	Memory dumper
02e1763d48ba1f2ce12dc2bf47bcaa53a274cba2	6/27/2012	197,632	TSPY_RAWPOS.SM	Memory dumper
cb53751f3cd1f336c0cbc4c461e8742254708d55	4/16/2012	197,632	TSPY_RAWPOS.SM	Memory dumper
b9711ab81e4695c901983e48ac80fcb918c4d094	10/3/2011	1,776,847	TROJ_RAWPOS.A	File scraper
704954be63d0ca62d088a4cbdacc81178d0c514	10/3/2011	2,294,446	TROJ_RAWPOS.A	File scraper
6c8f872d9d2f506562733e185ed930ec9c093696	10/3/2011	1,776,765	TROJ_RAWPOS.A	File scraper
6f7e9e23d30cb74903b152a84dac25ce2a68bfa2	10/3/2011	2,294,450	TROJ_RAWPOS.A	File scraper
1f11e35fb0b4e179c17bcabfbd5f2ef3e05d1cb2	10/3/2011	1,800,036	TROJ_RAWPOS.A	File scraper
b06a5856e6ed48df957d0bef81c09ab9d4d29565	6/24/2011	75,776	TSPY_RAWPOS.SMO	Memory dumper
48c0730ac86babe08ad78e3eed1a91adf327c742	12/3/2010	66,048	TSPY_RAWPOS.SM1	Service
02efe49d18b5120f661f8be48a03a357a957f0a5	6/1/2010	74,752	TSPY_RAWPOS.SMO	Memory dumper
0a8300183eaafdb8b1d3724652c1b794a1e35d54	11/28/2009	197,120	TSPY_RAWPOS.SM	Memory dumper
290b885b662c134998ee3b8bb6b940b0ae9fbbce	9/27/2009	197,120	TSPY_RAWPOS.SM	Memory dumper
7638021a205c4766909dc265debeef48554f2b0f	2/24/2009	74,752	TSPY_RAWPOS.SMO	Memory dumper
d315ebe8f7881b501ccaec460d22d3d5c3125862	2/19/2009	66,048	TSPY_RAWPOS.SM1	Service
0512d1bef690af3c3c50420bd4f55fc663cefc88	12/9/2008	76,288	TSPY_RAWPOS.SMO	Memory dumper
711d29846e75afcf19f1f720f00c6f051e0232	11/21/2008	76,800	TSPY_RAWPOS.SMO	Memory dumper

c3918542074c7548fba6a3b246712f45e8534f10	5/28/2008	75,776	TSPY_RAWPOS.SM0	Memory dumper
e7b8a70ef8e45fd3a4fa412850b59032c0468318	5/27/2008	19,7120	TSPY_RAWPOS.SM	Memory dumper
fa64e1e1894274f080431523b19297ab99be4fca	5/7/2008	66,048	TSPY_RAWPOS.SM1	Service
ab2d82cdc856e86ad15407208ba375e2fe5e273c	4/7/2008	66,048	TSPY_RAWPOS.SM1	Service

Here are the YARA rules to cover both the service and the memory dumper. No YARA rule was generated for the file data scraper due to the nature of the file.

```
rule PoS_Malware_RawPOS2015_service : RawPOS2015_service
{
meta:
  author = "Trend Micro, Inc."
  date = "2015-03-10"
  description = "Used to detect RawPOS RAM service, including 2015 sample set"
  sample_filetype = "exe"
strings:
  $string0 = "OpenService failed - %s"
  $string1 = "OpenSCManager failed - %s"
  $string2 = "Unable to install %s - %s"
  $string3 = "File already exists"
  $string4 = "Stopping %s."
  $string5 = "This may take several seconds. Please wait."
  $string6 = "%s failed to stop."
  $string7 = "%s removed."
  $string8 = "Debugging %s."
  $string9 = "Could not create registry key"
  $string10 = "\\\\.\\pipe\\susrv"
  $string11 = "SYSTEM\\CurrentControlSet\\Services\\EventLog\\Application\\%s"
condition:
  all of ($string*)
}
```

```
rule PoS_Malware_RawPOS2015_dumper : RawPOS2015_dumper
{
meta:
  author = "Trend Micro, Inc."
  date = "2015-03-10"
  description = "Used to detect RawPOS RAM dumper, including 2015 sample set"
  sample_filetype = "exe"
strings:
  $string1 = "(1[0-2])([0-9]"
  $string2 = "(1[0-2])[0-9]{8,30}"
  $string3 = "((B(([0-9]{13,16})"
  $mess1 = "Found track data at %s with PID %d"
```

```

$mess2 = "Enter Process Id: "
$mess3 = " Dump private process memory by PID"
$mess4 = "Dumping private memory for pid %s to %s.dmp..."
$mess5 = " Full private dump of all running processes"
$memd1 = "memdump\\%s-%d.dmp"
$memd2 = "mkdir memdump >NUL 2>NUL"
condition:
  (all of ($memd*)) and (all of ($mess*)) and (any of ($string*))
}

```

```

rule PoS_Malware_RawPOS2015_dumper_old : RawPOS2015_dumper_old
{
meta:
  author = "Trend Micro, Inc."
  date = "2015-03-10"
  description = "Used to detect RawPOS memory dumper, pre-2012"
  sample_filetype = "exe"
strings:
  $string0 = " Full private dump of all running processes"
  $string1 = " show info on Process like Path"
  $string2 = " Show this help"
  $string3 = " List all running processes"
  $string4 = "Dumping private memory for pid %s to %s.dmp..."
  $string5 = "%s-%d.dmp"
  $string6 = "memdump\\%s-%d.dmp"
  $string7 = "del memdump\\"
  $string8 = "Process Memory Dumper"
  $string9 = "Base size: %u"
  $string10 = "Module ID: %u"
  $string11 = "Hex: %xh"
condition:
  all of ($string*)
}

```

Trend Micro Incorporated, a global leader in security software, strives to make the world safe for exchanging digital information. Our innovative solutions for consumers, businesses and governments provide layered content security to protect information on mobile devices, endpoints, gateways, servers and the cloud. All of our solutions are powered by cloud-based global threat intelligence, the Trend Micro™ Smart Protection Network™, and are supported by over 1,200 threat experts around the globe. For more information, visit [www.trendmicro.com](http://www.trendmicro.com).

©2015 by Trend Micro, Incorporated. All rights reserved. Trend Micro and the Trend Micro t-ball logo are trademarks or registered trademarks of Trend Micro, Incorporated. All other product or company names may be trademarks or registered trademarks of their owners.



225 E. John Carpenter Freeway  
Suite 1500  
Irving, Texas  
75062 U.S.A.

Phone: +1.817.569,8900