


 **codypierce** / **hackers-grep**

 Watch

4

 Star

18

 Fork

0

hackers-grep is a utility to search for strings in PE executables including imports, exports, and debug symbols

 6 commits

 1 branch










 0 releases

 1 contributor

 Branch: **master** ▾

hackers-grep / +



README edits		
	codypierce authored a day ago	latest commit 0456a15f48 
 .gitignore	chchchchanges	a day ago
 README.md	README edits	a day ago
 dislib.py	Initial add	2 days ago
 file_details.py	Initial add	2 days ago
 hackers-grep.py	README edits	a day ago
 msdia80.dll	Initial add	2 days ago
 pdbsymbols.py	Initial add	2 days ago

 **README.md**



hackers-grep

hackers-grep is a tool that enables you to search for strings in PE files. The tool is capable of searching strings, imports, exports, and public symbols (like `woah`) using regular expressions.

I have used this tool to find functionality across Windows that I want to investigate further. It has allowed me to answer questions like "How many libraries process XML?".

It's also fun to poke around with.

Installation

1. Install comtypes (<https://pypi.python.org/pypi/comtypes>)
2. Install pywin32 (<http://sourceforge.net/projects/pywin32/files/pywin32/>)
3. Install Microsoft debugging symbols (<https://msdn.microsoft.com/en-us/windows/hardware/gg463028.aspx>)


Usage

The options for hackers-grep are listed below. Keep in mind that all regular expression strings should use python's "re" module syntax and are case insensitive. For more information please see <https://docs.python.org/2/library/re.html>.


```
Z:\hackers-grep>hackers-grep.py -h
Usage: hackers-grep.py [options] <search path> <file regex> <string regex>

Options:
  -h, --help                show this help message and exit
  -d MAX_DEPTH, --max-depth=MAX_DEPTH
                             Maximum directory recursion depth (default: 1)
  -x, --exports-only        Only search Export section strings
```


<> Code


 Issues


0

 Pull requests

0


 Wiki


 Pulse


 Graphs


HTTPS clone URL

https://github.com



You can clone with [HTTPS](#), [SSH](#), or [Subversion](#). 

 Clone in Desktop

 Download ZIP

```

-n, --imports-only    Only search Import section strings
-a, --all-the-things  Search strings, import, exports
-s, --symbols         Include symbols in search
-p SYMBOL_PATH, --symbol-path=SYMBOL_PATH
                        Symbol path (default: C:\Windows\Symbols)
-e EXPORT_FILTER, --export-filter=EXPORT_FILTER
                        Search modules matching this Export regex
-i IMPORT_FILTER, --import-filter=IMPORT_FILTER
                        Search modules matching this Import regex
-f, --show-info       Display file details size and modification time
-v, --verbose         Verbose output

```

Examples

Search for every dll that imports wininet!InternetOpen

This example demonstrates how to search imports. It will search C:\Windows\System32 for any file ending in .dll that imports InternetOpenA or InternetOpenW.

```

Z:\hackers-grep>hackers-grep.py -n c:\windows\system32 *.dll "InternetOpen[A,W]"
c:\windows\system32\msidcr130.dll  WININET.dll!InternetOpenA
c:\windows\system32\msscp.dll     WININET.dll!InternetOpenA
c:\windows\system32\mssign32.dll  WININET.dll!InternetOpenA
c:\windows\system32\msnetobj.dll  WININET.dll!InternetOpenA
c:\windows\system32\oleprn.dll    WININET.dll!InternetOpenW
c:\windows\system32\urlmon.dll    WININET.dll!InternetOpenW
c:\windows\system32\winethc.dll   WININET.dll!InternetOpenW
c:\windows\system32\wuwebv.dll    WININET.dll!InternetOpenW

```

By using the option "-f" we can also print information about the file, including the useful "File description" property.

```

Z:\hackers-grep>hackers-grep.py -f -n c:\windows\system32 *.dll "InternetOpen[A,W]"
c:\windows\system32\msidcr130.dll  [ 468 KB] [07/14/2009] [Microsoft Corporation] [IDCRL Dynamic
c:\windows\system32\msscp.dll     [ 492 KB] [02/03/2015] [Microsoft Corporation] [Windows Media
c:\windows\system32\mssign32.dll  [  38 KB] [07/14/2009] [Microsoft Corporation] [Microsoft Tr
c:\windows\system32\msnetobj.dll  [ 259 KB] [02/03/2015] [Microsoft Corporation] [DRM ActiveX I
c:\windows\system32\oleprn.dll    [ 104 KB] [07/14/2009] [Microsoft Corporation] [Oleprn DLL]
c:\windows\system32\urlmon.dll    [1280 KB] [07/16/2015] [Microsoft Corporation] [OLE32 Extens
c:\windows\system32\winethc.dll   [  81 KB] [07/14/2009] [Microsoft Corporation] [WinInet Help
c:\windows\system32\wuwebv.dll    [ 169 KB] [07/20/2015] [Microsoft Corporation] [Windows Upda

```

Search for every dll that exports DllGetClassObject

This example demonstrates how to search exports. It will search C:\Windows\System32 for any file ending in .dll that exports the OLE function DllGetClassObject

```

Z:\hackers-grep>hackers-grep.py -x c:\windows\system32 *.dll DllGetClassObject
c:\windows\system32\accessibilitycpl.dll  <export>!DllGetClassObject
c:\windows\system32\ActionCenterCPL.dll   <export>!DllGetClassObject
c:\windows\system32\activeds.dll          <export>!DllGetClassObject
c:\windows\system32\AdmTmpl.dll           <export>!DllGetClassObject
c:\windows\system32\adsldp.dll            <export>!DllGetClassObject
c:\windows\system32\adsmsext.dll          <export>!DllGetClassObject
c:\windows\system32\amstream.dll          <export>!DllGetClassObject
...

```

Search for every occurrence of the string "xmlns"

This example will search C:\Windows\System32 for every occurrence of the string "xmlns" in a dll.

```
Z:\hackers-grep>hackers-grep.py c:\windows\system32 *.dll xmlns.*
...
c:\windows\system32\mscms.dll      xmlns:wcs='http://schemas.microsoft.com/windows/2005/02/co
c:\windows\system32\mscms.dll      xmlns:gmm='http://schemas.microsoft.com/windows/2005/02/co
c:\windows\system32\mscms.dll      xmlns:cam='http://schemas.microsoft.com/windows/2005/02/co
<snip>
c:\windows\system32\msmpeg2vdec.dll  xmlns:c
c:\windows\system32\msmpeg2vdec.dll  xmlns:c
c:\windows\system32\mstscax.dll      xmlns:psf='http://schemas.microsoft.com/windows/2003/08/pr
c:\windows\system32\mstscax.dll      xmlns:psf='http://schemas.microsoft.com/windows/2003/08/pr
<snip>
c:\windows\system32\wlanpref.dll     xmlns:p='http://www.microsoft.com/networking/WLAN/profile/
c:\windows\system32\wlanpref.dll     xmlns:p='http://www.microsoft.com/networking/WLAN/profile/
c:\windows\system32\WlanMM.dll       xmlns:an='http://www.microsoft.com/AvailableNetwork/Info'
c:\windows\system32\WlanMM.dll       xmlns:an='http://www.microsoft.com/AvailableNetwork/Info'
<snip>
```

Search for every RPC server that also imports CreateFile

This example shows how to use the import filter "-i" option. In this case, we want to find any dll that imports CreateFile, but filter only those that import RpcServerListen as well.

```
Z:\hackers-grep>hackers-grep.py -n -i RpcServerListen c:\windows\system32 *.dll CreateFile
c:\windows\system32\authui.dll      KERNEL32.dll!CreateFileW
c:\windows\system32\certprop.dll    KERNEL32.dll!CreateFileW
c:\windows\system32\FXSAPI.dll      KERNEL32.dll!CreateFileW
c:\windows\system32\FXSAPI.dll      KERNEL32.dll!CreateFileMappingW
c:\windows\system32\bthserv.dll     KERNEL32.dll!CreateFileW
c:\windows\system32\iscsiexe.dll     KERNEL32.dll!CreateFileW
c:\windows\system32\msdtcprx.dll    KERNEL32.dll!CreateFileW
c:\windows\system32\RpcEpMap.dll     API-MS-Win-Core-File-L1-1-0.dll!CreateFileW
c:\windows\system32\rpcss.dll        KERNEL32.dll!CreateFileMappingW
c:\windows\system32\bdesvc.dll       KERNEL32.dll!CreateFileW
c:\windows\system32\SCardSvr.dll     KERNEL32.dll!CreateFileW
c:\windows\system32\tapisrv.dll      KERNEL32.dll!CreateFileMappingW
c:\windows\system32\tapisrv.dll      KERNEL32.dll!CreateFileW
c:\windows\system32\termsrv.dll      KERNEL32.dll!CreateFileW
c:\windows\system32\wbiosrv.dll      API-MS-Win-Core-File-L1-1-0.dll!CreateFileW
c:\windows\system32\wiaservc.dll     KERNEL32.dll!CreateFileW
c:\windows\system32\wiaservc.dll     KERNEL32.dll!CreateFileMappingW
c:\windows\system32\tbssvc.dll       KERNEL32.dll!CreateFileA
c:\windows\system32\wscsvc.dll       KERNEL32.dll!CreateFileW
c:\windows\system32\wscsvc.dll       KERNEL32.dll!CreateFileMappingW
c:\windows\system32\rdpcore.dll      KERNEL32.dll!CreateFileW
c:\windows\system32\wlansvc.dll      KERNEL32.dll!CreateFileW
c:\windows\system32\wwansvc.dll      KERNEL32.dll!CreateFileW
```

Search symbols like a boss

This example shows my favorite use of hackers-grep. The ability to combine some of the previously shown features and the verbose symbol information makes this a nice tool for finding that next bug :)

In this example we want to use debugging symbols (.pdb) to search for every occurrence of hfont.

```
Z:\hackers-grep>hackers-grep.py -s c:\windows\system32 *.dll ".*HFONT.*"
c:\windows\system32\advpack.dll      struct HFONT__ * g_hFont
c:\windows\system32\advpack.dll      struct HFONT__ * g_hFont
<snip>
c:\windows\system32\dxtrans.dll      public: virtual long __stdcall CDXTLabel::GetFontHandl
c:\windows\system32\dxtrans.dll      public: virtual long __stdcall CDXTLabel::SetFontHandl
c:\windows\system32\dxtrans.dll      public: virtual long __stdcall CDX2D::GetFont(struct H
c:\windows\system32\dxtrans.dll      public: virtual long __stdcall CDX2D::SetFont(struct H
c:\windows\system32\dxtrans.dll      public: virtual long __stdcall CDXTLabel::GetFontHandl
c:\windows\system32\dxtrans.dll      public: virtual long __stdcall CDXTLabel::SetFontHandl
c:\windows\system32\dxtrans.dll      public: virtual long __stdcall CDX2D::GetFont(struct H
c:\windows\system32\dxtrans.dll      public: virtual long __stdcall CDX2D::SetFont(struct H
```

```
<snip>
c:\windows\system32\XpsGdiConverter.dll  protected: void __thiscall std::list<struct std::pair<
t ,class std::tr1::shared_ptr<struct xgc::GDIResource<struct HFONT__ *> > >,class std::allocator
tagLOGFONTW const ,class std::tr1::shared_ptr<struct xgc::GDIResource<struct HFONT__ *> > > > :
c:\windows\system32\XpsGdiConverter.dll  public: virtual void * __thiscall std::tr1::_Ref_count
e<struct HFONT__ *> >::`vector deleting destructor'(unsigned int)
<snip>
```

This example searches symbols for IStream::Read

```
Z:\hackers-grep>hackers-grep.py -s c:\windows\system32 *.dll ".*IStream::Read.*"
c:\windows\system32\apss.dll      public: virtual long __stdcall CStream::CImpIStream::Read(void
c:\windows\system32\apss.dll      public: virtual long __stdcall CStream::CImpIStream::Read(void
c:\windows\system32\avifil32.dll  public: virtual long __stdcall CPrxAVIStream::Read(long,long,'
c:\windows\system32\avifil32.dll  public: virtual long __stdcall CPrxAVIStream::ReadFormat(long
c:\windows\system32\avifil32.dll  public: virtual long __stdcall CPrxAVIStream::ReadData(unsigned
c:\windows\system32\avifil32.dll  public: virtual long __stdcall CPrxAVIStream::Read(long,long,'
c:\windows\system32\avifil32.dll  public: virtual long __stdcall CPrxAVIStream::ReadFormat(long
c:\windows\system32\avifil32.dll  public: virtual long __stdcall CPrxAVIStream::ReadData(unsigned
c:\windows\system32\cdosys.dll    public: virtual long __stdcall CMMIStream::Read(void *,unsigned
c:\windows\system32\cdosys.dll    public: virtual long __stdcall MemIStream::Read(void *,unsigned
c:\windows\system32\cdosys.dll    public: virtual long __stdcall CMMIStream::Read(void *,unsigned
c:\windows\system32\cdosys.dll    public: virtual long __stdcall MemIStream::Read(void *,unsigned
c:\windows\system32\imapi2fs.dll  CFsiStream::Read
c:\windows\system32\imapi2fs.dll  CFsiStream::Read
<snip>
```

Known issues

Its written in Python, so it can be slow, especially if you try and search the entire system drive while processing symbols

