User Guide

MANDIANT pdbxtract

Version 1.0

Overview

pdbxtract allows a user to explore symbolic type information as extracted from Microsoft PDB files. This tool has applications primarily for reverse engineering of Windows-based applications and for exploring the internals of Windows kernel components.

A programming database (PDB) file is a binary file containing program debug information in a Microsoft-proprietary format. This file is produced by the compiler/linker when a program is built. The information it contains is used by debuggers to debug a program and can greatly assist a developer in debugging program issues by resolving function pointers to symbolic names, for example. Perhaps the most useful and richest source of debugging information contained in PDBs is type data, which holds detailed information about data structures, constants, and other named symbols. While this information is primarily used to debug program components, it also can be used to gain insight into how core operating system components work by observing both the format of the data structure and how the structure is used.

pdbxtract is not a pure PDB parser – it only extracts type information using Microsoft's Debug Interface Access (DIA) COM. There are various DIA alternatives available, such as Volatility's open source pdbparse [1] and the PDB utility that comes with the Undocumented Windows 2000 Secrets book [2]. It is important to note that pdbxtract does not parse or capture the wealth of other information available in a PDB, including: functions, debug streams, modules, publics, globals, files, section information, injected sources, source files, OEM specific types, compilands, and others.

Supported Platforms and Pre-Requisites

pdbxtract runs on all Windows platforms that have .NET 2.0 or greater installed. The tool requires the following components which are distributed with the installation package:

- System.Data.SQLite.dll - SQLite managed code interface
- Interop.Dia2lib.dll - DIA COM interface wrapper for managed code; generated by Visual Studio from msdia100.dll
- msdia100.dll - the DIA COM interface, a redistributable Microsoft DLL provided in the visual studio SDK; pdbxtract registers the com interface by executing rundll32.exe on this DLL.

The accompanying utility, pdbfetch, requires .NET 2.0 or greater and either the x86 or x64 version of Microsoft Debugging Tools to be installed on the operating system.

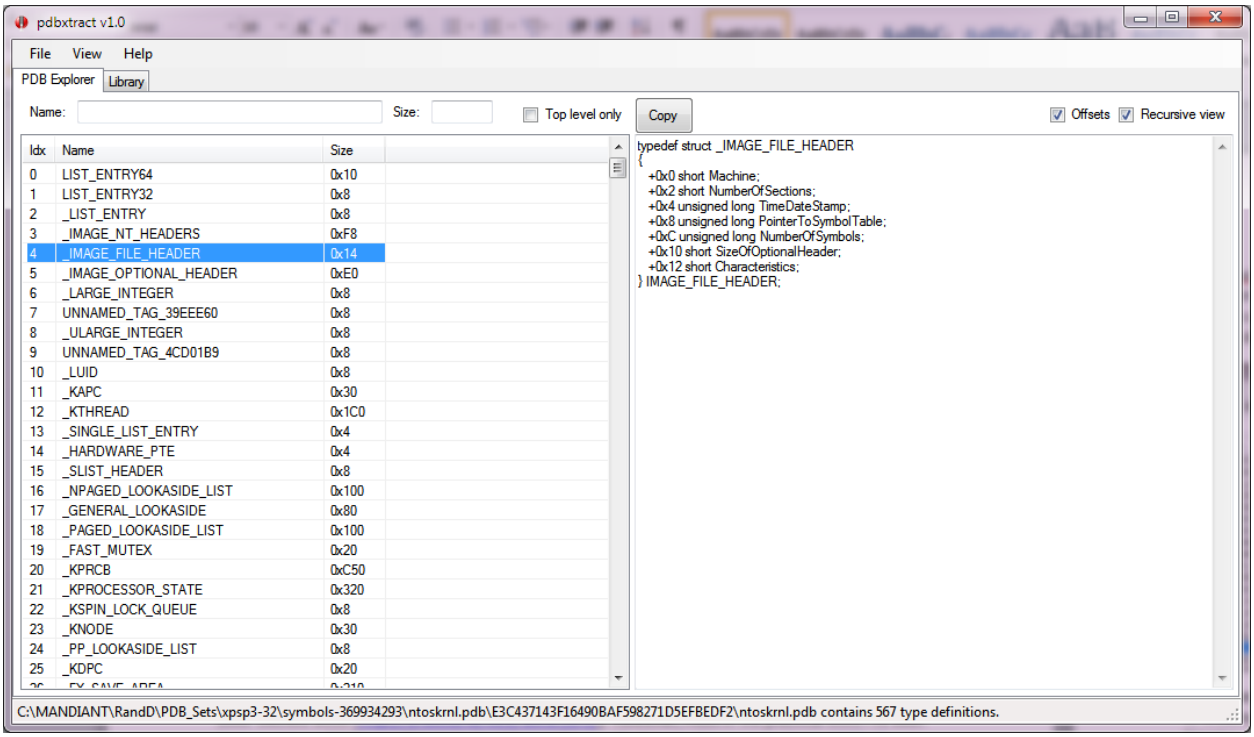
pdbxtract intrinsically supports all PDB formats supported by the msdia100.dll provided in the Visual Studio 2010 SDK.

Features

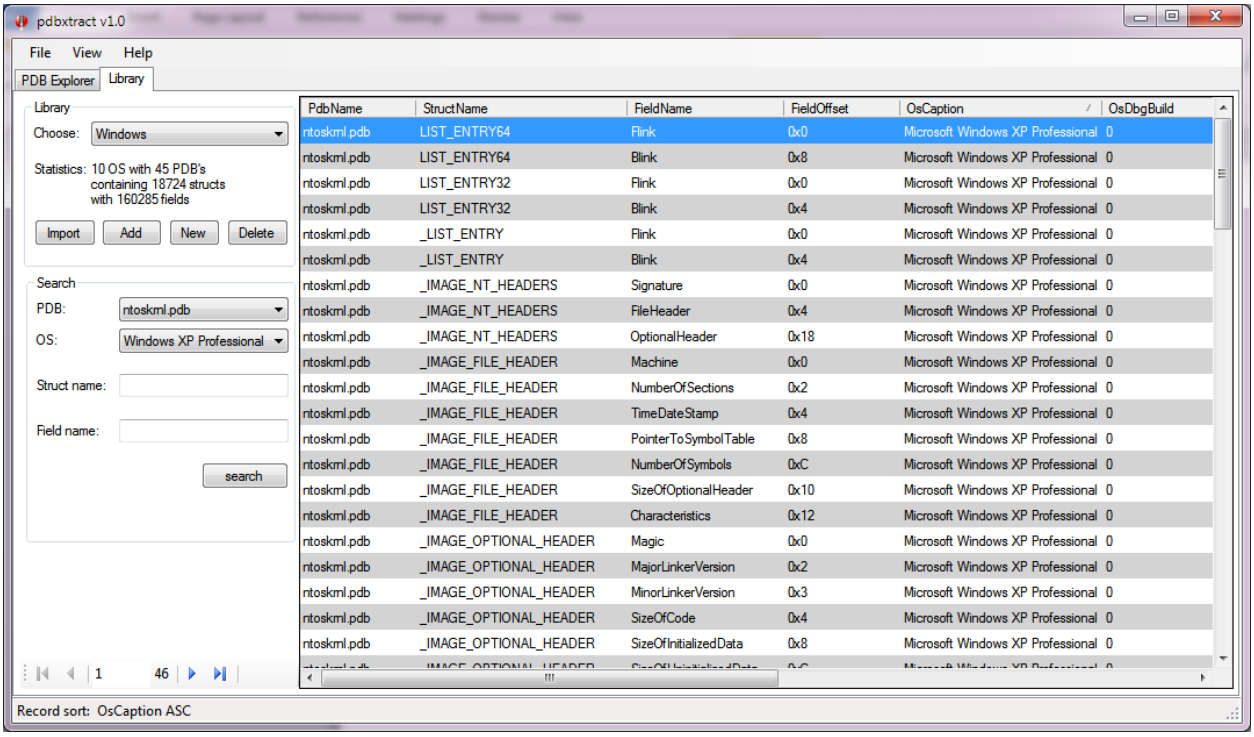
- Parses user defined types (UDT's) from Microsoft PDB formats
- Search UDT's by type name or size
- Filter UDT's by top-level only
- C-style struct generated for struct UDT's
- Field offsets
- Recursive view for struct UDT's
- Copy-and-paste to clipboard
- Export information to XML
- Export an IDA header file for structs
- Create a searchable SQLite database of type information from multiple PDB sources across multiple Windows operating systems
- Library view: right-click and send to PDB Explorer
- Accompanying utility, pdbfetch, allows a user to easily import new type definitions for other executables and libraries

Interface Overview

pdbxtract has two main features: exploring a single PDB (PDB Explorer) and searching a library of PDBs for one or more operating systems. PDB Explorer opens the PDB, parses type information using DIA, and displays a list of all structs, unions and enums. A C-style struct (with offsets) definition will be displayed in the text area in the right pane, as shown below for the type IMAGE_FILE_HEADER.



The library tab, shown below, allows the user to create and search a SQLite library of PDB type information. pdbxtract includes a default library which contains type information for several important operating systems binaries: variants of the Windows kernel (ntoskrnl.exe, ntkrpamp, ntkrnlpa, ntkrnlmp); the NDIS driver (ndis.sys); the GUI driver (win32k.sys); and variants of the Hardware Abstraction Layer driver (hal.dll, halaacpi.dll, halmacpi.dll).



The library included with pdbxtract covers several of Microsoft's major operating system releases, but it is easy to add more symbols. pdbxtract includes a utility, called pdbfetch, that simply runs Microsoft's symchk utility to download the symbols for the file names supplied in the text file provided as a command-line switch. Note that this utility does require internet access and generates the same network traffic that symchk generates. Pdbfetch creates a "PDB set" which consists of the directory structure with containing PDBs as created by symchk plus a manifest.xml file which summarizes the OS platform information. To use a PDB set in pdbxtract, switch to the library tab and click "new" to create a new library from the PDB set or "add" to add them to an existing library. Once a PDB set is added to a library, it can be deleted, as the resulting sqlite pdbx database that's created will contain the type information.

Exporting Type Information to an IDA header file

Pdbxtract supports an advanced feature that allows a user to export type information to an IDA header file, which can be imported into the IDA Pro Disassembler. The IDA header file simply contains C struct definitions for the required UDT's. To generate an IDA header file for a particular type, navigate to the PDB Explorer tab. Load UDT information into the tab by selecting "File->Load PDB" or "File->Import-

>XML". Select the UDT that you wish to export and navigate to "File->Export->IDA .h file". The resulting file will contain the C struct for the selected UDT and any subordinate UDT's.

Appendix A – References

[1] Volatility pdbparse - http://code.google.com/p/pdbparse/

[2] Undocumented Windows 2000 Secrets - http://undocumented.rawol.com/