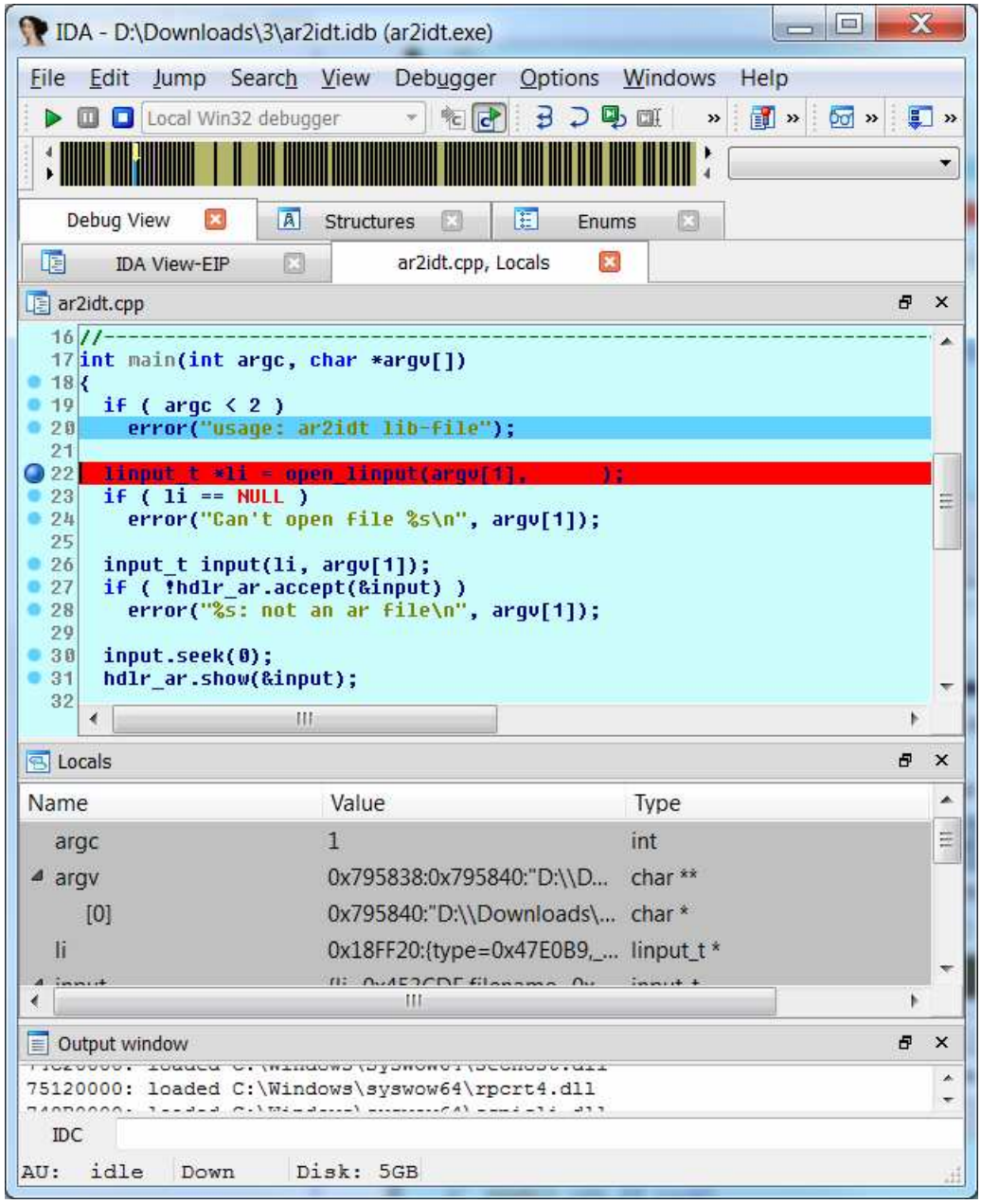
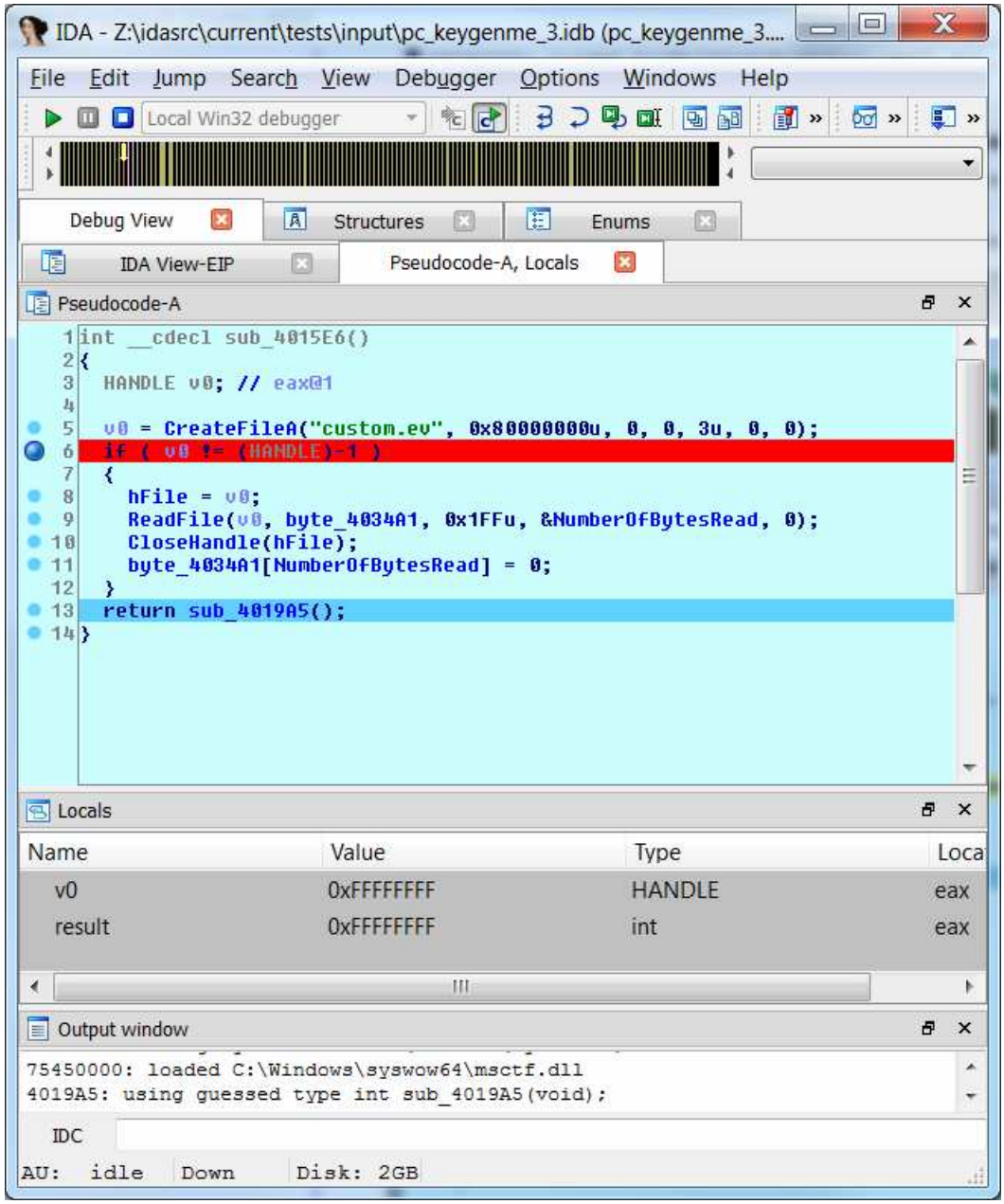
IDA: What's new in 6.3

Highlights

- Experimental source-level debugging



Current implementation requires PDB files with source line number info (Windows-only).
Other debugging formats to be added in the future.
Source file breakpoints are possible.
Also implemented in the decompiler - you can now step through the decompiled text.



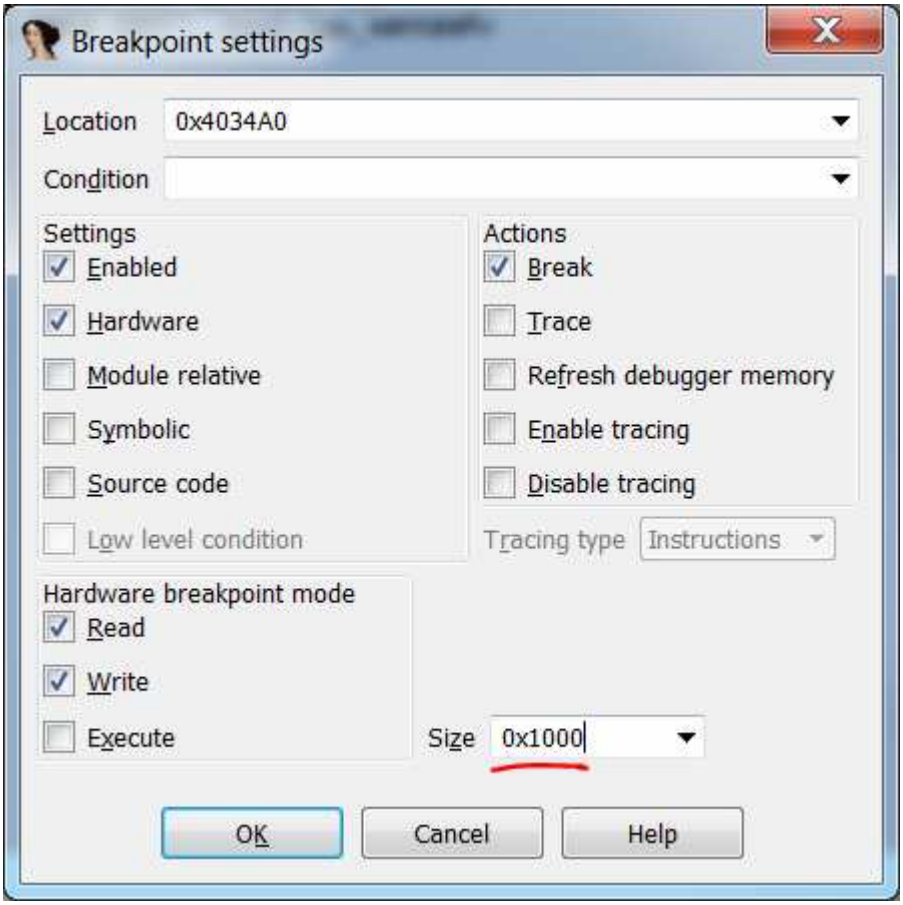
Local and global variables are displayed.
Hooks for providing lines info and source files are available in the SDK, but API can change in future.

- Trace Replayer

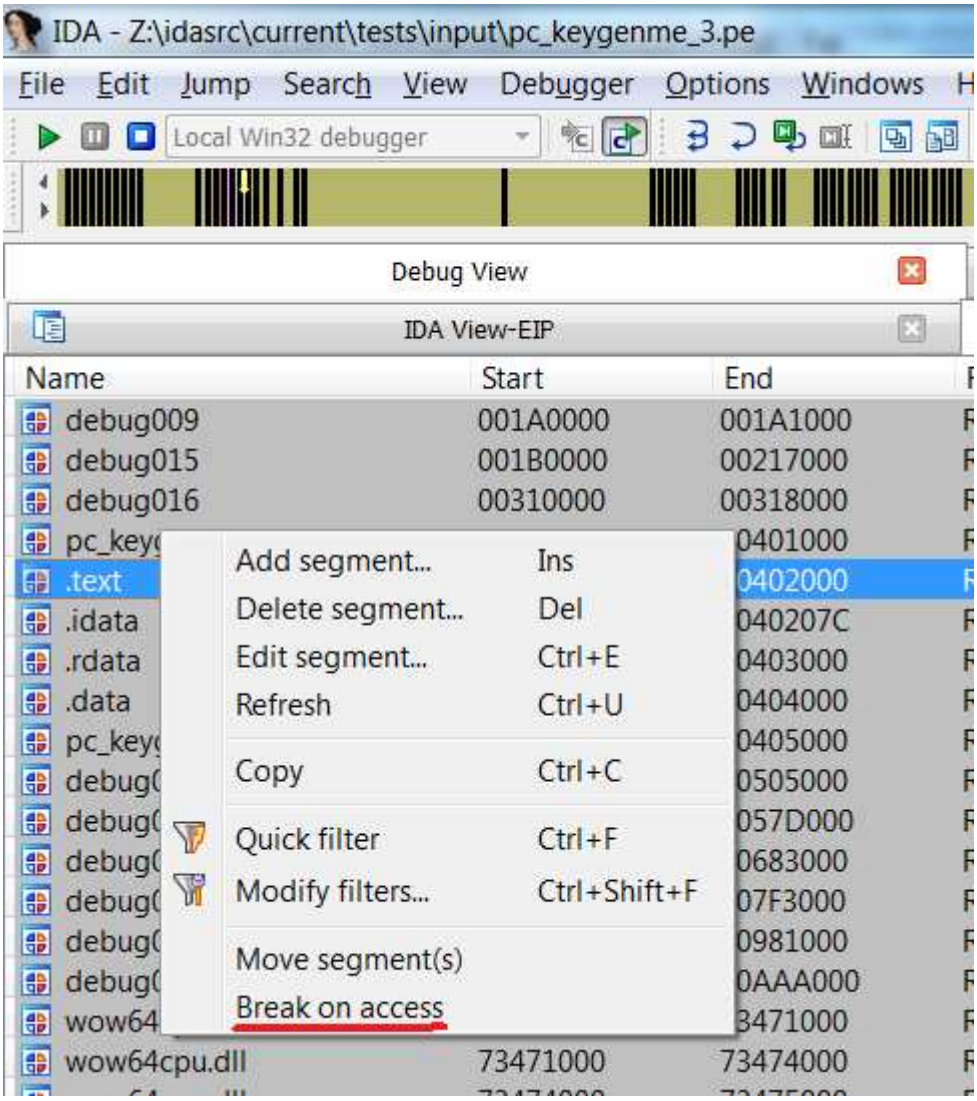
  Record execution traces, save, load and compare them.
  Tracing can be enabled and disabled in breakpoint actions.
  Replay the recording, step forward and backwards.
  Show executed blocks and functions in IDA's graph view and proximity view.
  See our blog for more details.

- Page-level breakpoints

  Arbitrarily-sized memory breakpoints implemented using page permissions.



Can break on writes, reads and execution.



Currently available in the Win32, Bochs, and WinDbg debugger backends.

- User interface

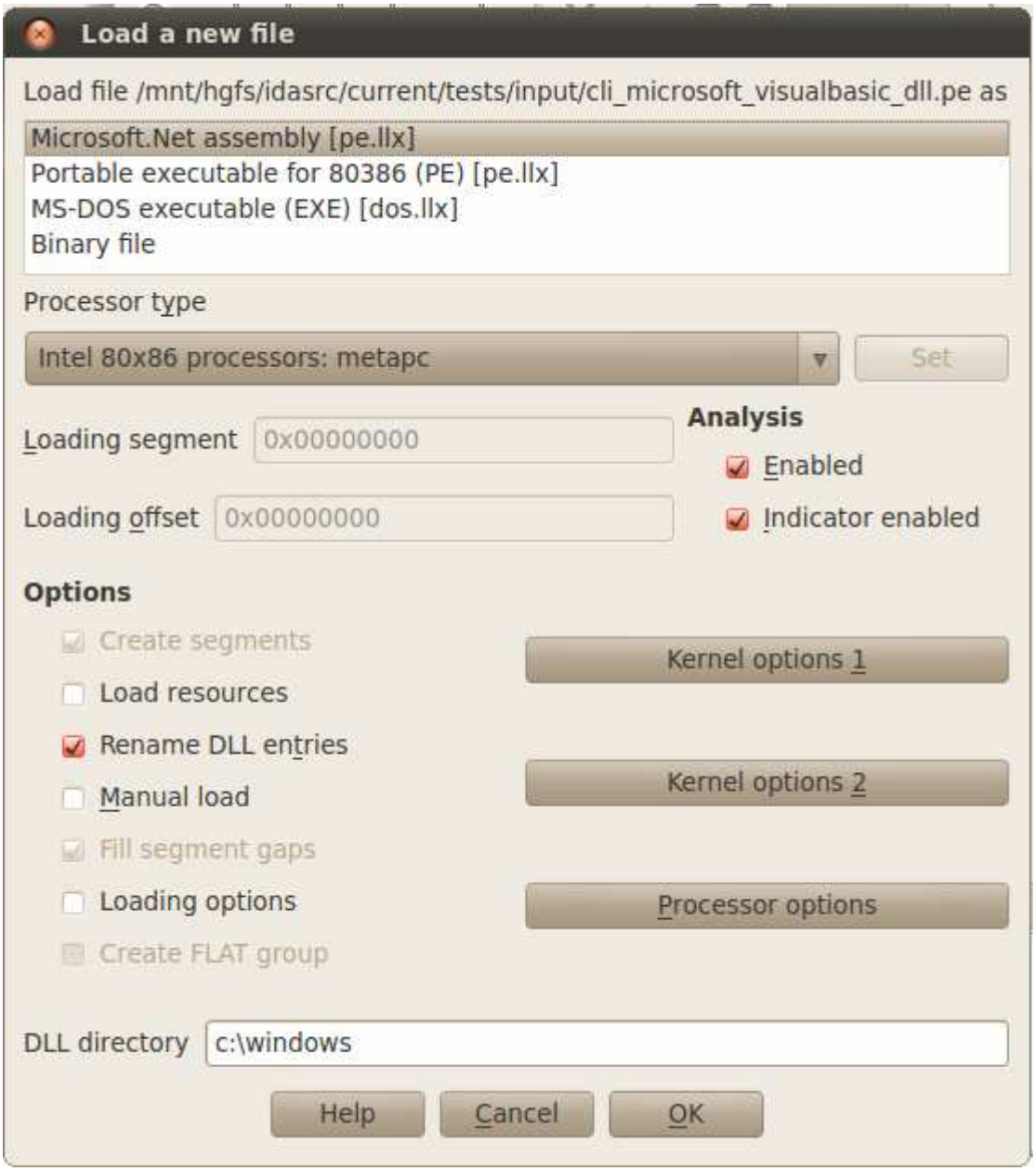
  Updated Qt libraries to version 4.8.1
  Full-screen mode. Default hotkey is F11 on Windows and Linux and Cmd-Shift-F on OS X.
  Full support for Numpad keys in shortcuts.

- FLIRT signatures

  Many improvements in file parsers and sigmake.
  Support for 64-byte long patterns (increased from 32 bytes).
  Improved resolving and reporting of collisions in sigmake.
  Visual C++ signatures regenerated from scratch; you should see a lot less "unknown_libname" in the listings.
  Improvements in pelf parser:

  - added option for generating one pattern per function instead of per code section
  - ELF64 support
  - record and honor Thumb bit for ARM files

- .NET file loader



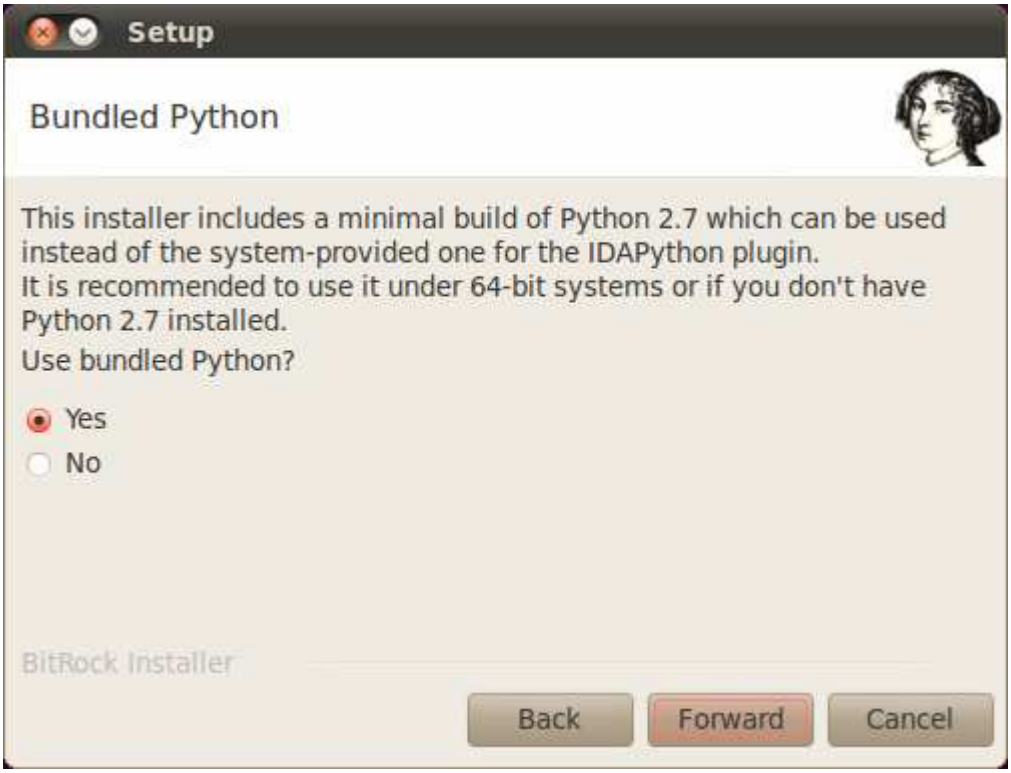
Works in Linux, OS X.
Implemented loading of .NET files from scratch, without relying on .NET or Mono libraries.
Fixed several bugs in the process.

- Processor modules

  new: M16C from Renesas (formerly Hitachi).
  new: unSP from SunPlus.
  new: TMS320C1 from Texas Instruments (contributed by Jeremy Cooper).
  new: Philips XA51 (contributed by Petr Novak).
  MIPS: Toshiba TX19a extensions and MIPS-MT, MIPS-3D, smartMIPS instructions.
  PPC: support for paired single (Gekko) and VMX128 (Xbox360 Xenon) instructions.
  PPC: added support for chip-specific SPRs, DPRs and memory-mapped registers.

- IDAPython

  Switched to Python 2.7 on Windows and Linux.
  Bundle prebuilt Python with Linux installer and offer to use it on x64 distros: this should resolve most of the IDAPython issues under that OS.



More APIs wrapped.
Added missing IDC functions to idc.py

Changelist

Processor Modules

- + 8051: added register definitions for 8032 variants
- + ARM: added recognition of R7 as the frame pointer in the thumb mode
- + AVR: added I/O port definitions for ATtiny2313 and ATtiny2313a (courtesy of Marcel Kilgus)
- + AVR: print immediate operands as unsigned by default (except for subi/sbci)
- + C166: added Tasking assembler style; added C166-specific SEG/@seg and SOF/@sof operators
- + C166: allow user to skip automatic creation of 64K chunks for binary code
- + CR16: added registers for CR16MCS9
- + H8: Added register definitions for H8S/2215R
- + I960: print memory-mapped register names in lda instructions
- + I960: relax memb operands decoding (apparently some assemblers do not produce completely correct instructions)
- + M16C: new processor module: Renesas (formerly Hitachi) M16C. Support for M16C/60, M16C/20 and M16C/Tiny models.
- + MIPS: added MIPS-MT, MIPS-3D, smartMIPS extensions
- + MIPS: added support for Toshiba TX19A instructions
- + PC: added support for "int 29h" (__fastfail call on win8)
- + PC: handle __alloca_probe_16 and __alloca_probe_8
- + PC: improved analysis of function frames that reuse ebp as a temporary register despite setting it up as a frame pointer
- + PC: improved analysis of function prologs
- + PC: improved recognition of import function thunks
- + PC: improved recognition of some jump tables generated by Mingw compiler
- + PC: recognize function prologs with inlined SEH setup (push offset __except_handler3) and parse SEH tables for them
- + PC: renamed some fields of the CPPEH_RECORD structure to match official names (e.g. "disabled" -> "TryLevel")
- + PC: decode RDRAND instruction
- + PC: improve recognition of SEH4 and GS/EH cookie set up in prologs
- + PPC: added support for device-specific SPRs, DPRs and memory-mapped registers; added definitions for mpc5xx
- + PPC: added support for paired single (Gekko) and VMX128 (Xbox360 Xenon) instructions
- + SuperH: handle switch patterns

- + TMS320C1: new processor module: Texas Instruments TMS320C1x series (contributed by Jeremy Cooper)
- + UNSP: new processor module: SunPlus unSP
- + V850: create stack variables in instructions like "movea N, sp, rX"
- + XA51: Philips XA51 (contributed by Petr Novak)

File Formats

- + CLI: the loader for .NET files is now available in Linux and OS X
- + COFF: added support for ARM COFF modules in AR files produced by Microsoft VC
- + COFF: support TMS320C3x files
- + ELF: mark TLS-specific relocations in x64 .o files
- + ELF: PPC: add support for R_PPC_DTPMOD32, R_PPC_DTPREL32 relocations
- + ELF: support for 4 new ARM relocs (TLS offsets (GOT & non-GOT), thumb32 MOVT, thumb32 MOVW)
- + ELF: X64: properly handle R_X86_64_GOTPCREL
- + EPOC: added support for BYTEPAIR code compression
- + MACHO: added support for ARMv7-specific object relocations (ARM_RELOC_HALF, ARM_RELOC_HALF_SECTDIFF)
- + MACHO: format and comment Mach-O headers
- + MACHO: handle LC_FUNCTION_STARTS load command and create functions for the addresses in the list
- + MACHO: warn the user if the file being loaded is encrypted
- + PDB: improved detection of data versus code symbols
- + PDB: improved handling of unnamed types
- + PDB: improved PDB loading on Linux/OS X to make the results close to those of Windows
- + PDB: support remote fetching of PDB symbols under Linux/OS X for PE drivers (.sys files)
- + PDB: print detailed info about PDB matching attempts with -z10000
- + PE: all sections with the executable flag set are loaded by default regardless of their name
- + PE: handle self-modifying relocation blocks
- + PE: if the PE header was loaded into database, format and comment its fields
- + PE: PECPU_ARMI files sometimes use Thumb-2 instructions, so set the ARM architecture accordingly
- + PE: speed up loading of files with large number of exports

Kernel

- + improved propagation of argument type info
- + avoid repeatedly calling simplex analysis by postponing the stack analysis until the final pass completely analyzes the function

FLIRT, TIL & IDS

- + FLIRT: for new version ARM signatures, set the T segreg (Thumb/ARM mode) according to the matched lib function
- + FLIRT: many improvements in file parsers and sigmake; better resolving of collisions
- + FLIRT: pelf: supply "-f" to create one pattern per function, instead of one pattern per text section.
- + FLIRT: pelf: support 64-bit ELF files
- + FLIRT: pelf: support for R_ARM_XPC25 & R_ARM_THM_XPC22 relocation types.
- + FLIRT: pmacho: support for fat Mach-O archives with AR subfiles in them.
- + FLIRT: sigmake: accept 64-bytes patterns .pat files
- + FLIRT: sigmake: "-r" switch to ignore references to other functions when creating patterns
- + FLIRT: support for 64-bytes signatures in IDA
- + FLIRT: when pattern matching succeeds but xref matching fails, notify the user about functions that were candidates for a certain piece of code.
- + IDS: IDA now can load .idt files from .zip archives
- + vc32rtf.sig: better signature; more leaves, less collisions.
- + updated vcseh.sig; added patterns for _EH_prolog/epilog functions
- + loadint: added comments for I/O ports commonly used in BIOS code: 2E-2F,4E-4F,70-77,92,B2-B3,EB

Scripts & SDK

- + IDAPython: added a configuration option (USE_LOCAL_PYTHON) to python.cfg to enable using a local library with Python modules (under IDADIR/python)
- + IDAPython: added missing IDC functions to idc.py
- + IDAPython: switched precompiled plugin on Windows and Linux to use Python 2.7

- + IDAPython: UI_Hooks class automatically unhooks itself when IDA quits, avoiding a crash otherwise
- + IDAPython: wrap more functions from nalt.hpp
- + IDC: added GetMemberId()
- + SDK: added 'changed_stkpnts' IDB event
- + SDK: added choose3() function to invoke the chooser that benefits from additional callbacks
- + SDK: added create_ea_viewer() and improved jumpto() with an additional argument
- + SDK: added DBG_FLAG_FAKE_MEMORY for debuggers without process memory
- + SDK: added for_all_bpts() function to iterate over breakpoints
- + SDK: added functions for the new tracing functionality
- + SDK: added get_name_of_named_type()
- + SDK: added hexview sample plugin
- + SDK: added processor_t::adjust_libfunc_ea
- + SDK: added qunlink() to remove a file
- + SDK: enabled the 'deprecated function' warning and marked the deprecated sdk functions so that the compiler complains about them
- + SDK: get_loader_name_from_dll(), get_loader_name() retain the file extension for scripted loaders
- + SDK: improved randomness in qtmpnam()
- + SDK: now it is possible to create an explicit code cross-reference to the next instruction (it will not get converted to a flow xref)
- + SDK: QueueSet, replacement for QueueMark, allowing for user-friendly messages.
- + SDK: removed FORM_MDI and added a warning that the next version of IDA won't support plugins with native windows

Installer

- + installer: all debug servers are now collected in the "dbgsrv" subdirectory of IDA
- + installer: Linux: bundle a Python 2.7 install with IDA, and offer to use it by default under Linux x64
- + installer: on OS X, add symlinks to IDA binaries directory and debug servers to the install directory

User Interface

- + UI: qt: added full screen mode. The default hotkey is F11 on Windows and Linux and Cmd-Shift-F on OS X.
- + UI: qt: it is now possible to configure the caret blinking interval
- + UI: qt: Numpad keys are treated correctly and don't conflict with normal keys
- + UI: qt: possibility to specify a hotkey for a chooser action
- + UI: for the "Don't display this message again" checkbox, add a comment if it applies only to current session or database (i.e. it's not global)
- + UI: switched to Qt 4.8.1
- + UI: replaced crash handler with Google Breakpad on Linux
- + UI: setting IDA_NOEH=1 disables IDA's crash handler on Linux/OS X (previously worked only on Windows)
- + UI: added "Break on access" to the segments popup menu if the currently selected debugger supports page breakpoints
- + UI: added Edit, Operand types, Set operand type command
- + UI: Do not show the 'copying huge amounts of data, continue?' dialog unless copying takes more than several seconds
- + UI: don't show edit/delete menu items in choosers when nothing is selected
- + UI: print xrefs to structures and members in the structures list (similar to xrefs in disassembly view)

Debugger

- + BOCHS: added support for Bochs 2.5.x
- + BOCHS: warn if detected version is greater than expected
- + BOCHS: PE TLS callbacks with wrong calling convention could mess up the stack and cause a weird exception in bochsys.dll
- + debugger: added support for arbitrary-sized memory breakpoints (implemented using page permissions). First implementation available for Win32 and Linux.
- + debugger: added "warn", "log" and "silent" options for reaction to exceptions
- + debugger: debug traces can now be saved, loaded and compared
- + debugger: experimental source-level debugging feature. Currently available only on Windows and requires PDB files with line number info.
- + debugger: input/output redirection is now specified as part of the argument string, not the input file name
- + debugger: OS X: disable ASLR on Lion; explicitly specify the desired bitness of the debugged process

- + debugger: OS X: support for debugging on Lion (handle relocatable dyld)
- + debugger: support loading of COFF debug info from PE files (used by Cygwin/MinGW compiler)
- + debugger: unlink, rename, mkdir functions are available in low level breakpoint conditions
- + debugger: Win32: when attaching, show full executable paths in the list and also label 32/64-bit processes if running on a 64-bit OS
- + debugger: WinCE: initial support for WinCE 6.0 debugging
- + debugger: WinCE: new debugger module and server for debugging WinCE devices over TCP/IP; now it's possible to debug WinCE devices from Linux (since ActiveSync is not required)

Bugfixes

- BUGFIX: 'produce exe' command was inviting the user to overwrite the current idb file
- BUGFIX: .pdata section of PE files for ARMI architecture was not parsed correctly
- BUGFIX: added a workaround for integer overlow in 'operator new []' if compiled with GCC
- BUGFIX: AF2_STKARG option was ignored by the analysis engine
- BUGFIX: an attempt to create a huge segment that can not be created could corrupt the database in some cases
- BUGFIX: ARM: more correct frame setup in Thumb mode (local variables were lumped together with saved registers)
- BUGFIX: automatic database snapshots were not working if no snapshots existed before
- BUGFIX: C166: I/O registers with addresses above 64K were not handled
- BUGFIX: C166: memory accesses to I/O registers did not use symbolic names if their address was not present in database
- BUGFIX: C166: some instructions that used SFR encodings to access GPRs were decoded incorrectly
- BUGFIX: C166: some invalid DSP instructions were accepted by the disassembler
- BUGFIX: C166: the C166v2 instructions ENWDT and SBRK were not decoded
- BUGFIX: calling get_member_name() with a NULL buffer would crash IDA
- BUGFIX: CLI: array dimensions display was wrong
- BUGFIX: clicking 'Cancel' while uploading a file was not working
- BUGFIX: CR16: register pair operands were printed in wrong order
- BUGFIX: CR16: some CR16B instructions were not decoded
- BUGFIX: creating an enum for a processor with 32-bit wide bytes would lead to interr
- BUGFIX: DBG: CodeView NB11 debug information embedded in PE files was not handled properly
- BUGFIX: DbgByte() and similar functions could not be used in bpt conditions if the debugger backend was WinDbg
- BUGFIX: debugger could crash if user requested to terminate the process but the process was already dying (occurs very rarely)
- BUGFIX: debugger: in WinDbg kernel mode, sometimes it was impossible to continue after stopping at a breakpoint
- BUGFIX: debugger: system properties were not available for the applications launched by IDA's remote debugger server
- BUGFIX: debugger: the "Analyze module" command could put IDA into infinite loop in some cases
- BUGFIX: do not allow handling debug events (i.e., calling GetDebuggerEvent) from a breakpoint condition
- BUGFIX: EBP value reported by the windbg module was not always correct (e.g. at the function entry)
- BUGFIX: ELF: handle files with bogus sh_info values for REL sections (produced by some versions of GNU gold linker)
- BUGFIX: ELF: RELA relocs should ignore the original value and use just the addend
- BUGFIX: ELF: some files from LynxOS could not be loaded
- BUGFIX: ELF: some MIPS relocations were handled incorrectly
- BUGFIX: empty strings in collapsed structures were printed incorrectly
- BUGFIX: for collapsed items IDA was not considering the collapsed line as the most important line; breakpoints were displayed on a wrong line for such items
- BUGFIX: forms: pressing Enter on a readonly combobox would crash IDA
- BUGFIX: GDB: after continuing from a signal IDA kept sending the signal when continuing from next events
- BUGFIX: GDB: debugging of big-endian ARM targets did not work correctly
- BUGFIX: GDB: fixes for multi-thread debugging (resolves issue with VMWare 8.x multi-processor VMs)
- BUGFIX: GDB: floating-point registers were displayed as integer ones
- BUGFIX: H8: addresses of @aa:8 and @aa:16 operands were truncated on output
- BUGFIX: IDA complained on first saving of database if CREATE_BACKUPS was set to YES
- BUGFIX: IDA could crash if a function iterator was still alive at the exit time
- BUGFIX: IDA could crash trying to save desktop if the connection to the remote debugger server was lost
- BUGFIX: IDA could crash when refreshing an empty process list
- BUGFIX: IDA could crash when starting debugging with Bochs
- BUGFIX: IDA could interr when clicking inside text part of hex view in edit mode
- BUGFIX: IDA was refusing to load relocatable ELF files with non-zero section bases
- BUGFIX: IDA would crash if CleanupAppcall() was called while no Appcall was in progress
- BUGFIX: IDAPython: Functions() could miss some functions if the specified range was starting with a function tail chunk

- BUGFIX: IDAPython: op_t.is_reg() was broken
- BUGFIX: IDAPython: scripts residing in directories with specific names next to the IDB could be executed automatically during IDA startup
- BUGFIX: idaw/idal would display "internal error" while trying to show the commandline usage topic (-?,-h switch)
- BUGFIX: IDC: #include "absolute_path" was not accepted by ida
- BUGFIX: IDC: GetManyBytes() would interr if called while win32 debugger was active
- BUGFIX: IDC: proper exception messages were not displayed in some cases (e.g. for breakpoint conditions)
- BUGFIX: IDC: negation of floating point values was impossible
- BUGFIX: if some TILs could not be loaded, the local TIL would not be loaded either
- BUGFIX: in proximity view, some edges between functions may not be added if a function B references function A but function A was already visited before.
- BUGFIX: instant debugger for OS X was not working
- BUGFIX: it was impossible to save a temporary database using the menu command
- BUGFIX: MACHO: fix some ObjC metadata parsing issues
- BUGFIX: MACHO: relocations of type X86_64_RELOC_BRANCH were not correctly applied in final linked files
- BUGFIX: MIPS: jalrc instruction was incorrectly marked as not returning
- BUGFIX: MSP430: jc and jnc instructions were swapped
- BUGFIX: PC: an interr could happen if code changed during debugging
- BUGFIX: PC: instructions like 'pop [esp+N]' use the updated value of esp; IDA was not aware of that
- BUGFIX: PC: it was impossible to assemble 'jmp short' in the presence of non-trivial segment selectors
- BUGFIX: PDB: dbgeng.dll was freed too early in some cases
- BUGFIX: PDB: fix "Parse error near: GUID" messages when loading PDBs during debugging
- BUGFIX: PDB: recursive self-referencing type definitions in PDB files could result in interrs
- BUGFIX: PDB: some structures involving unnamed unions could not be imported into IDB
- BUGFIX: qsem_wait() could return too early on linux (because of EINTR)
- BUGFIX: qt: "Script file..." Menu option was always defaulting to the IDC directory on Linux/OS X
- BUGFIX: qt: changing the color of a graph node with shadows disabled would crash IDA
- BUGFIX: qt: enabling accessibility on OSX could cause IDA to crash deep inside Qt
- BUGFIX: qt: hotkeys set in idagui.cfg for switching between graph, flat and proximity views were ignored under some circumstances
- BUGFIX: qt: in case of a wrong input in a form field the control didn't get focus
- BUGFIX: qt: in IDA 6.2 Shift + double click was not selecting the current identifier
- BUGFIX: qt: it was npt possible to cancel adding children/parents of selected nodes in proximity view
- BUGFIX: qt: it was not possible to enter expressions in the structure offset dialog
- BUGFIX: qt: message boxes could show up on the wrong screen in a multi-screen environment
- BUGFIX: qt: not specifying the initial directory in askfile was resulting in a wrong one
- BUGFIX: qt: proximity view code for handling shortcuts "+" and "-" was handling also the cases were Ctrl, Alt or Shift keys were pressed
- BUGFIX: qt: setting the selection of multiple rows in the chooser was not behaving correctly and was also slow
- BUGFIX: qt: the arrows in disasm views opened by the user were not correctly resized
- BUGFIX: qt: the default shortcut context for local actions was wrong
- BUGFIX: qt: the hex view wasn't saving its configuration
- BUGFIX: qt: the native file dialog on OSX doesn't allow shortcuts such as copy and paste because of a bug in Qt, use the Qt file dialog instead
- BUGFIX: qt: the waitdialog wasn't refreshing the label without a wasBreak call
- BUGFIX: SDK: del_segm() was ignoring SEGMOD_SILENT; also pass on the silent flags when deleting or adding additional segments in add_segm_ex
- BUGFIX: SDK: description of parameters for the 'b' form specifier (combobox) was incorrect
- BUGFIX: SDK: qsem_create() could fail on OS X with ENAMETOOLONG; now we use MD5 of the name instead
- BUGFIX: SDK: validate_name() could overwrite its input buffer by one byte
- BUGFIX: SuperH: wrong cross-references could be created for @(<delta>,gbr) operands if delta was greater than 0x7F
- BUGFIX: the screen was not always refreshed after changing an item color from a script
- BUGFIX: the screen was not always refreshed after renaming a location from a script
- BUGFIX: there was no error dialog box if the user entered erroneous declaration while inserting a new local type (however, detailed error messages were still printed in the output window)
- BUGFIX: TIL: the time_t type was incorrectly defined as 64-bit in "mssdk" and related type libraries
- BUGFIX: TMS320C3x: 16-bit immediate operands could not be converted to enums
- BUGFIX: TMS320C3x: it was not possible to use custom offsets for operands with displacement
- BUGFIX: TMS320C3x: register renaming did not work properly for operands with complex addressing modes
- BUGFIX: Tricore: floating-point data items were not printed as such
- BUGFIX: TXT: file timestamps were wrong in the text UI's file browser on Windows
- BUGFIX: UI: accidentally pressing A in the struct view would spoil the current struct field
- BUGFIX: UI: expanding collapsed segments did not always work

- BUGFIX: UI: choosers that display contents from the database (e.g. instructions with comments) could be using wrong encoding
- BUGFIX: UI: context menu was always shown at the mouse position, even if triggered from keyboard
- BUGFIX: UI: copying strings with custom encoding (e.g. UTF-16LE) would copy incorrect data to clipboard
- BUGFIX: UI: crash in hexview if user specified unsigned represention for floating values using keyboard shortcuts
- BUGFIX: UI: IDA could lock up when calling up the "Structure Offsets" dialog
- BUGFIX: UI: instruction comments could disappear in the find all occurrences retrieved list
- BUGFIX: UI: it wasn't possible to effectively change the hotkey for proximity view
- BUGFIX: UI: numeric keypad keys were not working in hex view's edit mode
- BUGFIX: UI: plugin comments would not show up in the status bar
- BUGFIX: UI: setting the default debugger did not work
- BUGFIX: UI: Shift+Home, Shift+End were working incorrectly in choosers
- BUGFIX: UI: some actions would print unnecessary "Command <...> failed" in the Output window when cancelled by the user
- BUGFIX: UI: status bar in choosers was not refreshed after some navigation events
- BUGFIX: UI: the structure offsets dialog could be displayed even without selection
- BUGFIX: UI: too many bookmarks could make the context menu unusable
- BUGFIX: UI: ui_saved event was happening too early, before the database was fully saved
- BUGFIX: using "Create EXE file" was incorrectly trying to load a DLL if the file was loaded with a scripted loader. Now a proper message is displayed (saving files with scripted loaders is not supported)
- BUGFIX: when mapping a local type to another, the corresponding IDB structure or enum was not being deleted
- BUGFIX: windmp: the check for 64-bit data in the dump file was not working properly
- BUGFIX: wrong input values in the 'load binary file' dialog were silently preventing the user from closing the dialog and continuing; added a warning message