Fix a possible segfault during argument parsing in functions that accept filesystem paths.

Fixed a partially broken sanity check in the _posixsubprocess internals regarding how fds_to_pass were passed to the child. The bug had no actual impact as subprocess.py already avoided it.

Fixed tokenizer crash when processing undecodable source code with a null byte.

The hash of the key now is calculated only once in most operations in C implementation of OrderedDict.

Default implementation of __reduce__ and __reduce_ex__ now rejects builtin types with not defined __new__.

Fix parser and AST: fill lineno and col_offset of "arg" node when compiling AST from Python objects.

Avoid buffer overreads when int(), float(), compile(), exec() and eval() are passed bytes-like objects. These objects are not necessarily terminated by a null byte, but the functions assumed they were.

Fixed a crash and leaking NULL in repr() of OrderedDict that was mutated by direct calls of dict methods.

Iterating OrderedDict with keys with unstable hash now raises KeyError in C implementations as well as in Python implementation.

Fixed crash when highly nested OrderedDict structures were garbage collected.

sys.setrecursionlimit() now raises a RecursionError if the new recursion limit is too low depending at the current recursion depth. Modify also the "lower-water mark" formula to make it monotonic. This mark is used to decide when the overflowed flag of the thread state is reset.

Fix input() to prompt to the redirected stdout when sys.stdout.fileno() fails.

Prevent builtin types that are not allowed to be subclassed from being subclassed through multiple inheritance.

Fixed a number of bugs in UTF-7 decoding of misformed data.

Import trace messages emitted in verbose (-v) mode are no longer formatted twice.

On Solaris 11.3 or newer, os.urandom() now uses the getrandom() function instead of the getentropy() function. The getentropy() function is blocking to generate very good quality entropy, os.urandom() doesn't need such high-quality entropy.

The stdprinter (used as sys.stderr before the io module is imported at startup) now uses the backslashreplace error handler.

Make the line number and column offset of set/dict literals and comprehensions correspond to the opening brace.

Hide the private _Py_atomic_xxx symbols from the public Python.h header to fix a compilation error with OpenMP. PyThreadState_GET() becomes an alias to PyThreadState_Get() to avoid ABI incompatibilities.

Change three zlib functions to accept sizes that fit in Py_ssize_t, but internally cap those sizes to UINT_MAX. This resolves a regression in 3.5 where GzipFile.read() failed to read chunks larger than 2 or 4 GiB. The change affects the zlib.Decompress.decompress() max_length parameter, the zlib.decompress() bufsize parameter, and the zlib.Decompress.flush() length parameter.

Avoid incorrect errors raised by os.makedirs(exist_ok=True) when the OS gives priority to errors such as EACCES over EEXIST.

Change semantics of EventLoop.stop() in asyncio.

When we know a subprocess.Popen process has died, do not allow the send_signal(), terminate(), or kill() methods to do anything as they could potentially signal a different process.

In the Readline completer, only call getattr() once per attribute.

Fix a crash when garbage-collecting ctypes objects created by wrapping a memoryview. This was a regression made in 3.5a1. Based on patch by Eryksun.

Added "escape" to the __all__ list in the glob module.

Fixed recursive glob() with patterns starting with `**`.

Fix regression in smtplib's AUTH LOGIN support.

Fix the pydoc web server's module search function to handle exceptions from importing packages.

Got rid of circular references in regular expression parsing.

fileinput.FileInput.readline() now returns b'' instead of '' at the end if the FileInput was opened with binary mode. Patch by Ryosuke Ito.

Fixed inspect.getdoc() for inherited docstrings of properties. Original patch by John Mark Vandenberg.

Always use os.urandom as a source of randomness in uuid.uuid4.

Fixed textwrap.dedent() for the case when largest common whitespace is a substring of smallest leading whitespace. Based on patch by Robert Li.

The lru_cache() wrapper objects now can be copied and pickled (by returning the original object unchanged).

typing: Don't crash on Union[str, Pattern].

asyncio: Raise error from drain() when socket is closed.

Cleaned up and fixed minor bugs in C implementation of OrderedDict.

Improved Unicode support in SMTPHandler through better use of the email package. Thanks to user simon04 for the patch.

Remove mentions of the formatter module being removed in Python 3.6.

Fixed a bug in C implementation of OrderedDict.move_to_end() that caused segmentation fault or hang in iterating after moving several items to the start of ordered dict.

zipfile now works in threads disabled builds.

smtpd's SMTPChannel now correctly raises a ValueError if both decode_data and enable_SMTPUTF8 are set to true.

distutils raises OSError instead of DistutilsPlatformError when MSVC is not installed.

Fixed protocol for the STACK_GLOBAL opcode in pickletools.opcodes.

Updates asyncio datagram create method allowing reuseport and reuseaddr socket options to be set prior to binding the socket. Mirroring the existing asyncio create_server method the reuseaddr option for datagram sockets defaults to True if the O/S is 'posix' (except if the platform is Cygwin). Patch by Chris Laws.

Add asyncio.run_coroutine_threadsafe(). This lets you submit a coroutine to a loop from another thread, returning a concurrent.futures.Future. By Vincent Michel.

Fix CGIRequestHandler to split the query from the URL at the first question mark (?) rather than the last. Patch from Xiang Zhang.

Prevent CGIRequestHandler from collapsing slashes in the query part of the URL as if it were a path. Patch from Xiang Zhang.

C implementation of functools.lru_cache() now calculates key's hash only once.

Constructor and update method of weakref.WeakValueDictionary now accept the self and the dict keyword arguments.

Constructor of collections.UserDict now accepts the self keyword argument.

Fixed comparison of traceback.FrameSummary.

Added support for BINBYTES8 opcode in Python implementation of unpickler. Highest 32 bits of 64-bit size for BINUNICODE8 and BINBYTES8 opcodes no longer silently ignored on 32-bit platforms in C implementation.

Fix string.Formatter problem with auto-numbering and nested format_specs. Patch by Anthon van der Neut.

Rewrite the guts of asyncio.Queue and asyncio.Semaphore to be more understandable and correct.

Failed readline.set_completer_delims() no longer left the module in inconsistent state.

Default implementation of tzinfo.fromutc() was returning wrong results in some cases.

Allow the ssl module to be built with older versions of LibreSSL.

Prevent overflow in _Unpickler_Read.

The XML encoding declaration written by Element Tree now respects the letter case given by the user. This restores the ability to write encoding names in uppercase like "UTF-8", which worked in Python 2.

Make deque_clear() safer by emptying the deque before clearing. This helps avoid possible reentrancy issues.

platform module now reads Windows version from kernel32.dll to avoid compatibility shims.

Fix datetime.strftime() failure when errno was already set to EINVAL.

Fix rounding in fromtimestamp() and utcfromtimestamp() methods of datetime.datetime: microseconds are now rounded to nearest with ties going to nearest even integer (ROUND_HALF_EVEN), instead of being rounding towards minus infinity (ROUND_FLOOR). It's important that these methods use the same rounding mode than datetime.timedelta to keep the property: (datetime(1970,1,1) + timedelta(seconds=t)) == datetime.utcfromtimestamp(t). It also the rounding mode used by round(float) for example.

Fix datetime.datetime.now() and datetime.datetime.utcnow() on Windows to support date after year 2038. It was a regression introduced in Python 3.5.0.

Omitted internal frames in traceback functions print_stack(), format_stack(), and extract_stack() called without arguments.

Fix a regression of Python 3.5.0 in os.waitpid() on Windows.

socket.socket.getaddrinfo() now calls PyUnicode_AsEncodedString() instead of calling the encode() method of the host, to handle correctly custom string with an encode() method which doesn't return a byte string. The encoder of the IDNA codec is now called directly instead of calling the encode() method of the string.

Correctly compute stack usage of the BUILD_MAP opcode.

Comparing call_args to a long sequence now correctly returns a boolean result instead of raising an exception. Patch by A Kaptur.

Make sure that HTMLParser.feed() returns all the data, even when convert_charrefs is True.

shutil.make_archive() with the "zip" format now adds entries for directories (including empty directories) in ZIP file.

Fixed a crash caused by setting non-string key of expat parser. Based on patch by John Leitch.

Exit pdb if file has syntax error, instead of trapping user in an infinite loop. Patch by Xavier de Gaye.

Fix a race condition at Python startup if the file descriptor of stdin (0), stdout (1) or stderr (2) is closed while Python is creating sys.stdin, sys.stdout and sys.stderr objects. These attributes are now set to None if the creation of the object failed, instead of raising an OSError exception. Initial patch written by Marco Paolini.

Fix error handling and a race condition (related to garbage collection) in collections.OrderedDict constructor.

Fixed setting binary mode in Python implementation of FileIO on Windows and Cygwin. Patch from Akira Li.

Fix (another) memory leak in SSLSocket.getpeercer().

Disable the vulnerable SSLv3 protocol by default when creating ssl.SSLContext.

Fix memory leak in SSLSocket.getpeercert().

Sockets returned from accept() shouldn't appear to be nonblocking.

When threading.Event is reinitialized, the underlying condition should use a regular lock rather than a recursive lock.

Fix regression in unittest.expectedFailure on subclasses. Patch from Berker Peksag.

cgi.FieldStorage.read_multi() now ignores the Content-Length header in part headers. Patch written by Peter Landry and reviewed by Pierre Quentel.

Fix overrun error in deque.index(). Found by John Leitch and Bryce Darling.

Fix docstring in http.server.test. Patch from Chiu-Hsiang Hsu.

Improve message in configparser.InterpolationMissingOptionError. Patch from Łukasz Langa.

Honour TestCase.longMessage correctly in assertRegex. Patch from Ilia Kurenkov.

Fixed functools.singledispatch on classes with falsy metaclasses. Patch by Ethan Furman.

asyncio: ensure_future() now accepts awaitable objects.

Stop the debugger engine (normally in a user process) before closing the debugger window (running in the IDLE process). This prevents the RuntimeErrors that were being caught and ignored.

Prevent IDLE from hanging when a) closing the shell while the debugger is active (15347); b) closing the debugger with the [X] button (15348); and c) activating the debugger when already active (24455). The patch by Mark Roseman does this by making two changes. 1. Suspend and resume the gui.interaction method with the tcl vwait mechanism intended for this purpose (instead of root.mainloop & .quit). 2. In gui.run, allow any existing interaction to terminate first.

Change 'The program' to 'Your program' in an IDLE 'kill program?' message to make it clearer that the program referred to is the currently running user program, not IDLE itself.

Improve the appearance of the IDLE editor window status bar. Patch by Mark Roseman.

Change the handling of new built-in text color themes to better address the compatibility problem introduced by the addition of IDLE Dark. Consistently use the revised idleConf.CurrentTheme everywhere in idlelib.

Extension configuration is now a tab in the IDLE Preferences dialog rather than a separate dialog. The former tabs are now a sorted list. Patch by Mark Roseman.

Re-activate the config dialog help button with some content about the other buttons and the new IDLE Dark theme.

IDLE now has an 'IDLE Dark' built-in text color theme. It is more or less IDLE Classic inverted, with a cobalt blue background. Strings, comments, keywords, ... are still green, red, orange, ... To use it with IDLEs released before November 2015, hit the 'Save as New Custom Theme' button and enter a new name, such as 'Custom Dark'. The custom theme will work with any IDLE release, and can be modified.

README.txt is now an idlelib index for IDLE developers and curious users. The previous user content is now in the IDLE doc chapter. 'IDLE' now means 'Integrated Development and Learning Environment'.

Users can now set breakpoint colors in Settings -> Custom Highlighting. Original patch by Mark Roseman.

Inactive selection background now matches active selection background, as configured by users, on all systems. Found items are now always highlighted on Windows. Initial patch by Mark Roseman.

Idle: make calltip and completion boxes appear on Macs affected by a tk regression. Initial patch by Mark Roseman.

Idle ScrolledList context menus (used in debugger) now work on Mac Aqua. Patch by Mark Roseman.

Make right-click for context menu work on Mac Aqua. Patch by Mark Roseman.

Associate tkinter messageboxes with a specific widget. For Mac OSX, make them a 'sheet'. Patch by Mark Roseman.

Enhance the initial html viewer now used for Idle Help. Properly indent fixed-pitch text (patch by Mark Roseman). Give code snippet a very Sphinx-like light blueish-gray background. Re-use initial width and height set by users for shell and editor. When the Table of Contents (TOC) menu is used, put the section header at the top of the screen.

Condense and rewrite Idle doc section on text colors.

Explain some differences between IDLE and console Python.

Explain need for *print* when running file from Idle editor.

Doc: augment Idle feature list and no-subprocess section.

Update doc for Idle command line options. Some were missing and notes were not correct.

Most of idlelib is private and subject to change. Use idleib.idle.* to start Idle. See idlelib.__init__.__doc__.

Idle: add synchronization comments for future maintainers.

Replace help.txt with help.html for Idle doc display. The new idlelib/help.html is rstripped Doc/build/html/library/idle.html. It looks better than help.txt and will better document Idle as released. The tkinter html viewer that works for this file was written by Mark Roseman. The now unused EditorWindow.HelpDialog class and helt.txt file are deprecated.

Deprecate unused idlelib.idlever with possible removal in 3.6.

Remove extraneous code (which also create 2 & 3 conflicts).

Add remaining doc links to source code for Python-coded modules. Patch by Yoni Lavi.

Rewrite Comparisons section in the Expressions chapter of the language reference. Some of the details of comparing mixed types were incorrect or ambiguous. NotImplemented is only relevant at a lower level than the Expressions chapter. Added details of comparing range() objects, and default behaviour and consistency suggestions for user-defined classes. Patch from Andy Maier.

Clarify the default size argument of stack_size() in the "threading" and "_thread" modules. Patch from Mattip.

Overhaul tempfile docs. Note deprecated status of mktemp. Patch from Zbigniew Jędrzejewski-Szmek.

Update the types of some PyTypeObject fields. Patch by Joseph Weston.

Fix unittest discovery examples. Patch from Pam McA'Nulty.

Added tests for OrderedDict subclasses.

Make test_compileall not fail when an entry on sys.path cannot be written to (commonly seen in administrative installs on Windows).

Prevents assert dialogs appearing in the test suite.

`PCbuild\rt.bat` now accepts an unlimited number of arguments to pass along to regrtest.py. Previously there was a limit of 9.

Add LLVM support for PGO builds and use the test suite to generate the profile data. Initial patch by Alecsandru Patrascu of Intel.

Windows MSIs now have unique display names.

It is now possible to build Python on Windows without errors when external libraries are not available.

Updates shortcuts to start Python in installation directory.

Changes default all-users install directory to match per-user directory.

Improves installer error messages for unsupported platforms.

Display correct directory in installer when using non-default settings.

Disables use of SSE2 instructions in Windows 32-bit build

Adds logging to installer for case where launcher is not selected on upgrade.

Windows uninstallation should not remove launcher if other versions remain

py.exe launcher is missing icons

Windows installer does not precompile for -O or -OO.

Makes Back button in installer go back to upgrade page when upgrading.

Increases font size of the installer.

Clarifies that the non-web installer will download some components.

Restores requestedExecutionLevel to manifest to disable UAC virtualization.

Removed very outdated PC/example_nt/ directory.

Fix output of python-config --extension-suffix.