Mar 31

minosgalanakis mbedtls-4.1.0 0fe989b Compare

Mbed TLS 4.1.0

Latest

Description

This release of Mbed TLS provides new features, bug fixes and minor enhancements. This release includes fixes for security issues.

Mbed TLS 4.1 is a long-term support (LTS) branch. It will be supported with bug-fixes and security fixes until at least March 2029.

Please note

- Mbed TLS 4.1.0 includes TF-PSA-Crypto 1.1.0 (as a subtree in the tarball, and as a submodule in git) and can only be built with this included version TF-PSA-Crypto.

Security Advisories

- Client impersonation while resuming a TLS 1.3 session
- Entropy on Linux can fall back to `/dev/urandom`
- PSA random generator cloning
- Compiler-induced constant-time violations
- Null pointer dereference when setting a distinguished name
- Buffer overflow in FFDH public key export
- FFDH: lack of contributory behaviour due to improper input validation

- Signature Algorithm Injection
- CCM multipart finish tag-length validation bypass
- Risk of insufficient protection of serialized session or context data leading to potential memory safety issues
- Buffer underflow in `x509_inet_pton_ipv6()`

Release Notes

API changes

- MBEDTLS_TIMING_C now requires MBEDTLS_HAVE_TIME to be enabled in the TF-PSA-Crypto configuration, unless MBEDTLS_TIMING_ALT is enabled. As a benefit, platforms where the default implementation is not supported now only need to implement MBEDTLS_PLATFORM_MS_TIME_ALT.
- When MBEDTLS_TIMING_ALT is enabled, the function mbedtls_timing_get_timer() now returns unsigned long long instead of unsigned long.

Features

- Add the function mbedtls_ssl_get_fatal_alert(), which returns the type of the last received fatal alert. This allows callers to retrieve more detailed information when mbedtls_ssl_handshake(), mbedtls_ssl_handshake_step(), or mbedtls_ssl_read() returns the generic MBEDTLS_ERR_SSL_FATAL_ALERT_MESSAGE error code.
- Function mbedtls_ssl_get_supported_group_list() is added to return the list of supported groups IDs (curves and finite fields).
- MBEDTLS_SSL_IANA_TLS_GROUPS_INFO is added to allow defining the list of mbedtls_ssl_iana_tls_group_info_t items which represent known TLS groups with corresponding informations.
  If MBEDTLS_DEBUG_C is also enabled then mbedtls_ssl_iana_tls_group_info is also available as implementation of such list.

Security

- The documentation of mbedtls_ssl_session_save(), mbedtls_ssl_session_load(), mbedtls_ssl_context_save(), and mbedtls_ssl_context_load() has been updated to clarify the responsibility of the application to preserve the confidentiality and integrity of serialized data, mitigating the risk of misuse of these APIs.

Credit to Haruto Kimura (Stella) and Eva Crystal (0xiviel) for highlighting risks associated with tampered serialized data.

- Fix a NULL pointer dereference in mbedtls_x509_string_to_names() when mbedtls_calloc() fails to allocate memory. This was caused by failing to check whether mbedtls_calloc() returned NULL. Found and reported by Haruto Kimura (Stella).
- Fix a limited buffer underflow in x509_inet_pton_ipv6(). In rare cases (e.g. on platforms with memory protection when the overread crosses page boundary) this could lead to DoS. Found and reported by Haruto Kimura (Stella). CVE-2026-25833
- Fix a bug in the TLS 1.2 client's signature algorithm check, which caused the client to accept server key exchange messages signed with a signature algorithm explicitly disallowed by the client. Found and reported by EFR-GmbH and M. Heuft of Security-Research-Consulting GmbH. CVE-2026-25834
- Fixed an issue in TLS 1.3 server handling of the second ClientHello, after sending a HelloRetryRequest message. A man-in-the-middle attacker could force a TLS 1.3 session resumption using a ticket to fall back to an unintended TLS 1.2 session resumption with an all-zero master secret. This could result in client authentication being bypassed and allow client impersonation.
  Found and reported by Jaehun Lee, Pohang University of Science and Technology (POSTECH).

Bugfix

- CMake now installs headers to `CMAKE_INSTALL_INCLUDEDIR` instead of the hard-coded `include` directory.
- Fix CMake failure on Windows because of a native directory separator. Fixes #10502.
- mbedtls_timing_get_delay() now correctly treats a timer as expired after more than 2^32 ms (about 49 days) on platforms where long is a 32-bit type. Fixes #10613.
- Support re-assembly of fragmented DTLS 1.2 ClientHello in Mbed TLS server.
- Support re-assembly of fragmented TLS 1.2 ClientHello in Mbed TLS server even if TLS 1.3 support is disabled. This removes the main limitation on support for re-assembly of fragmented handshake messages in TLS 1.2.

Changes

- Add casts to some Enums to remove compiler errors thrown by IAR 6.5. Removes Warning "mixed ENUM with other type".
- Tweak the detection of Unix-like platforms, which makes more system interfaces (timing, threading) available on Haiku, QNX and Midipix.
- Harden mbedtls_ssl_get_verify_result() against misuse. If the handshake has not yet been attempted, return -1u to indicate that the result is not available. Previously the result of verification was zero-initialized so the function would return 0 (indicating success).

Who should update

We recommend all users should update to take advantage of the bug fixes contained in this release at an appropriate point in their development lifecycle.

Note

> ❕ `mbedtls-4.1.0.tar.bz2` are our official release files. `source.tar.gz` and `source.zip` are automatically generated snapshots that GitHub is generating. They do not include external dependencies, and can't be configured

Checksum

The SHA256 hashes for the archives are:
377a09cf8eb81b5fb2707045e5522d5489d3309fed5006c9874e60558fc81d10 mbedtls-4.1.0.tar.bz2

▼ **Assets** 4

🎉 4 👀 1 5 people reacted

Mar 31

minosgalanakis mbedtls-3.6.6 `0bebf8b` Compare

Mbed TLS 3.6.6

Description

This release includes fixes for security issues.

Mbed TLS 3.6 is a long-term support (LTS) branch. It will be supported with bug-fixes and security fixes until at least March 2027.

Security Advisories

For full details, please see the following links:

- Client impersonation while resuming a TLS 1.3 session
- Entropy on Linux can fall back to `/dev/urandom`
- PSA random generator cloning
- Compiler-induced constant-time violations
- Null pointer dereference when setting a distinguished name
- Buffer overflow in FFDH public key export
- FFDH: lack of contributory behaviour due to improper input validation
- Signature Algorithm Injection
- CCM multipart finish tag-length validation bypass
- Risk of insufficient protection of serialized session or context data leading to potential memory safety issues
- Buffer underflow in `x509_inet_pton_ipv6()`

Release Notes

Features

- The automatic computation of MBEDTLS_PSA_STATIC_KEY_SLOT_BUFFER_SIZE has been improved to take into account the following key types: asymmetric keys, ciphers, AEADs, CMAC and HMAC.
- The device for reading entropy on platforms without a dedicated system call can now be configured with MBEDTLS_PLATFORM_DEV_RANDOM or mbedtls_platform_dev_random.

- Applications can use the new functions psa_random_reseed() to request an immediate reseed of the PSA random generator, or psa_random_deplete() to force a reseed on the next random generator call.
- Applications can call psa_random_set_prediction_resistance() to toggle prediction resistance in the PSA random generator.

Security

- The documentation of mbedtls_ssl_session_save(), mbedtls_ssl_session_load(), mbedtls_ssl_context_save(), and mbedtls_ssl_context_load() has been updated to clarify the responsibility of the application to preserve the confidentiality and integrity of serialized data, mitigating the risk of misuse of these APIs.
  Credit to Haruto Kimura (Stella) and Eva Crystal (0xiviel) for highlighting risks associated with tampered serialized data.
- The default device for reading entropy on platforms without a dedicated system call is now /dev/random instead of /dev/urandom. This is safer on Linux in case the application runs early after the kernel boots, but may block needlessly on Linux <= 5.6. Reported by supers1ngular (BayLibre).
- Fix missing validation of the peer's key in key agreement operations using PSA_ALG_FFDH: low-order elements were not rejected as they should be. This is a problem for protocols using FFDH that expect contributory behaviour, that is, where neither party should be able to force the shared secret into a small set. Reported independently by Eva Crystal (0xiviel) and another reporter.
- Add tag length validation in mbedtls_ccm_finish() to prevent out-of-bounds reads and mitigate potential application buffer overflows where applications relied on the library to enforce tag length constraints.
  Reported by Eva Crystal (0xiviel).
- Fix a NULL pointer dereference in mbedtls_x509_string_to_names() when mbedtls_calloc() fails to allocate memory. This was caused by failing to check whether mbedtls_calloc() returned NULL. Found and reported by Haruto Kimura (Stella).
- Fix a buffer overflow in psa_export_public_key() for FFDH keys when the output buffer is too small. Found by Haruto Kimura (Stella).
- Fix a limited buffer underflow in x509_inet_pton_ipv6(). In rare cases (e.g. on platforms with memory protection when the overread crosses page

boundary) this could lead to DoS. Found and reported by Haruto Kimura (Stella). CVE-2026-25833

- If an application called psa_crypto_init() then fork() and continued to use cryptography APIs (possibly indirectly, e.g. for TLS), the random generator states were duplicated. Fix this by forcing a RNG reseed in the child process. CVE-2026-25835
- Applications running in environments where the application state is cloned (for example due to resuming a frozen system state multiple times, or due to cloning a virtual machine image) should arrange to reseed the random generator using one of the new functions psa_random_reseed() or psa_random_deplete(). CVE-2026-25835
- Fix a bug in the TLS 1.2 client's signature algorithm check, which caused the client to accept server key exchange messages signed with a signature algorithm explicitly disallowed by the client. Found and reported by EFR-GmbH and M. Heuft of Security-Research-Consulting GmbH. CVE-2026-25834
- Fixed an issue in TLS 1.3 server handling of the second ClientHello, after sending a HelloRetryRequest message. A man-in-the-middle attacker could force a TLS 1.3 session resumption using a ticket to fall back to an unintended TLS 1.2 session resumption with an all-zero master secret. This could result in client authentication being bypassed and allow client impersonation.
  Found and reported by Jaehun Lee, Pohang University of Science and Technology (POSTECH).

Bugfix

- Appease GCC 14.3's array bounds checker by inserting checks in mbedtls_xor that bail before the byte-at-a-time loop when the array size is a constant (using MBEDTLS_HAS_BUILTIN) and an exact multiple of the larger loop size.
- CMake now installs headers to `CMAKE_INSTALL_INCLUDEDIR` instead of the hard-coded `include` directory.
- Prevent loading of peristent keys whose key ID belong to the volatile range.
- Partially fix a performance regression in RSA operations introduced by a security fix in 3.6.5, by improving the performance of RSA private key operations when MBEDTLS_RSA_NO_CRT is disabled, which is the default.
- Fix compilation errors in `aesce.c` in some Visual Studio builds.
  Fixes Mbed-TLS/TF-PSA-Crypto#548.
- Fix a build failure with MinGW when the __USE_MINGW_ANSI_STDIO option is set. This was caused by the wrong format specifier being used to

print long long values (MBEDTLS_PRINTF_LONGLONG).

- Fix a bug that caused GCM tag calculations to fail, so that data was correctly encrypted but could not be authenticated. The bug was only observed with GCC 10.0 to 14.2 inclusive, when compiling with -O3, and running without AESNI or AESCE.
  Fixes Mbed-TLS/TF-PSA-Crypto#665.
- Fix a build failure with dietlibc.
- Support re-assembly of fragmented DTLS 1.2 ClientHello in Mbed TLS server.
- Support re-assembly of fragmented TLS 1.2 ClientHello in Mbed TLS server even if TLS 1.3 support is disabled. This removes the main limitation on support for re-assembly of fragmented handshake messages in TLS 1.2.
- Some functions in PK were using large buffers (around 2KB in the default configuration) on the stack, which was a problem in environments with a small stack. Those buffers are now allocated on the heap, except in configurations where ECC is the only supported key type in PK, making PK still independent of the heap in such configurations (if the ECC driver itself is not using the heap). Fixes Mbed-TLS/TF-PSA-Crypto#476.

Changes

- Add casts to some Enums to remove compiler errors thrown by IAR 6.5. Removes Warning "mixed ENUM with other type".
- Tweak the detection of Unix-like platforms, which makes more system interfaces (timing, threading) available on Haiku, QNX and Midipix.
- Harden mbedtls_ssl_get_verify_result() against misuse.
  If the handshake has not yet been attempted, return -1u to indicate that the result is not available. Previously the result of verification was zero-initialized so the function would return 0 (indicating success).

Who should update

This release includes security fixes. We strongly recommend evaluating your exposure and, if applicable, prioritizing an upgrade.

Note

> ❗ `mbedtls-3.6.6.tar.bz2` and `mbedtls-3.6.6.tar.bz2-sha256sum.txt` are our official release files. `source.tar.gz` and `source.zip` are automatically generated snapshots that GitHub is generating. They do not include external dependencies, and can't be configured

Checksum

The SHA256 hash for the archive is:
8fb65fae8dcae5840f793c0a334860a411f884cc537ea290ce1c52bb64ca007a mbedtls-3.6.6.tar.bz2

▸ **Assets** 4

🎉 2 🚀 2 3 people reacted

Oct 15, 2025

minosgalanakis mbedtls-4.0.0 ec40440 Compare

Mbed TLS 4.0.0

Description

The Mbed TLS 4.0.0 and TF-PSA-Crypto 1.0.0 releases introduce a major codebase restructuring:

- PSA Crypto functionality now resides in its own repository Mbed-TLS/TF-PSA-Crypto.
- TLS and X.509 components remain in Mbed TLS.

API Changes & Migration

The Mbed TLS 4.0.0 and TF-PSA-Crypto 1.0.0 releases include significant API changes that break backward compatibility with previous releases. Please test your integration thoroughly and follow the 4.0 migration guide and TF-PSA-Crypto's 1.0 migration guide to update your codebase to the new interfaces.

Please note

- Mbed TLS 4.0.0 includes TF-PSA-Crypto 1.0.0 (as a subtree in the tarball, and as a submodule in git) and can only be built with this included version TF-PSA-Crypto.

Security Advisories

Compared to Mbed TLS 3.6.4 (the latest LTS release that pre-dates it), Mbed TLS 4.0.0 does not directly fix any new vulnerability. However two vulnerabilities are fixed in TF-PSA-Crypto 1.0.0, which is included in Mbed TLS 4.0.0. See the release notes of TF-PSA-Crypto 1.0.0 for details about those two fixes.

Compared to Mbed TLS 3.6.0 (the latest feature release that pre-dates it), Mbed TLS 4.0.0 also fixes the following vulnerabilities, that had been fixed in Mbed TLS 3.6.4 already:

- `CTR_DRBG` prioritized over `HMAC_DRBG` as the PSA DRBG
- Stack buffer overflow in ECDSA signature conversion functions
- Limited authentication bypass in TLS 1.3 optional client authentication
- Buffer underrun in pkwrite when writing an opaque key pair
- TLS clients should generally call mbedtls_ssl_set_hostname
- Potential authentication bypass in TLS handshake
- Misleading memory management in `mbedtls_x509_string_to_names()`
- NULL pointer dereference after using `mbedtls_asn1_store_named_data()`

Release Notes

API changes

- Align the mbedtls_ssl_ticket_setup() function with the PSA Crypto API. Instead of taking a mbedtls_cipher_type_t as an argument, this function now takes 3 new arguments: a PSA algorithm, key type and key size, to specify the AEAD for ticket protection.
- The PSA and Mbed TLS error spaces are now unified. mbedtls_xxx() functions can now return PSA_ERROR_xxx values.
  There is no longer a distinction between "low-level" and "high-level" Mbed TLS error codes.
  This will not affect most applications since the error values are between -32767 and -1 as before.
- All API functions now use the PSA random generator psa_generate_random() internally. As a consequence, functions no longer take RNG parameters.

Please refer to the migration guide at :
docs/4.0-migration-guide.md.

- The list passed to mbedtls_ssl_conf_alpn_protocols() is now declared as having const elements, reflecting the fact that the library will not modify it
- Change the serial argument of the mbedtls_x509write_crt_set_serial_raw function to a const to align with the rest of the API.
- Change the signature of the runtime version information methods that took a char* as an argument to take zero arguments and return a const char* instead. This aligns us with the interface used in TF PSA Crypto 1.0.
  If you need to support linking against both Mbed TLS 3.x and 4.x, please use the build-time version macros or mbedtls_version_get_number() to determine the correct signature for mbedtls_version_get_string() and mbedtls_version_get_string_full() before calling them.
  Fixes issue #10308.
- Make the following error codes aliases of their PSA equivalents, where xxx is a module, e.g. X509 or SSL.
  MBEDTLS_ERR_xxx_BAD_INPUT_DATA -> PSA_ERROR_INVALID_ARGUMENT
  MBEDTLS_ERR_xxx_ALLOC_FAILED -> PSA_ERROR_INSUFFICIENT_MEMORY
  MBEDTLS_ERR_xxx_BUFFER_TOO_SMALL -> PSA_ERROR_BUFFER_TOO_SMALL
  MBEDTLS_ERR_PKCS7_VERIFY_FAIL -> PSA_ERROR_INVALID_SIGNATURE
- Add MBEDTLS_SSL_NULL_CIPHERSUITES configuration option. It enables TLS 1.2 ciphersuites without encryption and is disabled by default.
  This new option replaces MBEDTLS_CIPHER_NULL_CIPHER.

Default behavior changes

- The X.509 and TLS modules now always use the PSA subsystem to perform cryptographic operations, with a few exceptions documented in docs/architecture/psa-migration/psa-limitations.md. This corresponds to the behavior of Mbed TLS 3.x when MBEDTLS_USE_PSA_CRYPTO is enabled. In effect, MBEDTLS_USE_PSA_CRYPTO is now always enabled.
- psa_crypto_init() must be called before performing any cryptographic operation, including indirect requests such as parsing a key or certificate or starting a TLS handshake.
- In TLS clients, if mbedtls_ssl_set_hostname() has not been called, mbedtls_ssl_handshake() now fails with MBEDTLS_ERR_SSL_CERTIFICATE_VERIFICATION_WITHOUT_HOSTNAME if certificate-based authentication of the server is attempted.

This is because authenticating a server without knowing what name to expect is usually insecure.

Removals

- Remove support for the RSA-PSK key exchange in TLS 1.2.
- Remove deprecated mbedtls_x509write_crt_set_serial(). The function was already deprecated and superseded by mbedtls_x509write_crt_set_serial_raw().
- Remove the function mbedtls_ssl_conf_curves() which had been deprecated in favour of mbedtls_ssl_conf_groups() since Mbed TLS 3.1.
- Remove support for the DHE-PSK key exchange in TLS 1.2.
- Remove support for the DHE-RSA key exchange in TLS 1.2.
- Following the removal of DHM module (#9972 and Mbed-TLS/TF-PSA-Crypto#175) the following SSL functions are removed:
  - mbedtls_ssl_conf_dh_param_bin
  - mbedtls_ssl_conf_dh_param_ctx
  - mbedtls_ssl_conf_dhm_min_bitlen
- Remove support for the RSA key exchange in TLS 1.2.
- Remove mbedtls_low_level_strerr() and mbedtls_high_level_strerr(), since these concepts no longer exists. There is just mbedtls_strerror().
- Sample programs for the legacy crypto API have been removed.
  pkey/rsa_genkey.c
  pkey/pk_decrypt.c
  pkey/dh_genprime.c
  pkey/rsa_verify.c
  pkey/mpi_demo.c
  pkey/rsa_decrypt.c
  pkey/key_app.c
  pkey/dh_server.c
  pkey/ecdh_curve25519.c
  pkey/pk_encrypt.c
  pkey/rsa_sign.c
  pkey/key_app_writer.c
  pkey/dh_client.c
  pkey/ecdsa.c
  pkey/rsa_encrypt.c
  wince_main.c
  aes/crypt_and_hash.c

random/gen_random_ctr_drbg.c
random/gen_entropy.c
hash/md_hmac_demo.c
hash/hello.c
hash/generic_sum.c
cipher/cipher_aead_demo.c

- Remove compat-2-x.h header from mbedtls.
- The library no longer offers interfaces to look up values by OID or OID by enum values.
  The header <mbedtls/oid.h> now only defines functions to convert between binary and dotted string OID representations, and macros for OID strings that are relevant to X.509.
  The compilation option MBEDTLS_OID_C no longer exists. OID tables are included in the build automatically as needed.
- The header <mbedtls/check_config.h> no longer exists. Including it from a custom config file was no longer needed since Mbed TLS 3.0, and could lead to spurious errors. The checks that it performed are now done automatically when building the library.
- Support for secp192k1, secp192r1, secp224k1 and secp224r1 EC curves is removed from TLS.
- Remove mbedtls_pk_type_t from the public interface and replace it with mbedtls_pk_sigalg_t.
- Remove MBEDTLS_SSL_DTLS_CONNECTION_ID_COMPAT. Now only the standard version (defined in RFC 9146) of DTLS connection ID is supported.
- Remove mbedtls_ssl_conf_min_version(), mbedtls_ssl_conf_max_version(), and the associated constants MBEDTLS_SSL_MAJOR_VERSION_x and MBEDTLS_SSL_MINOR_VERSION_y. Use mbedtls_ssl_conf_min_tls_version() and mbedtls_ssl_conf_max_tls_version() with MBEDTLS_SSL_VERSION_TLS1_y instead. Note that the new names of the new constants use the TLS protocol versions, unlike the old constants whose names are based on internal encodings.
- Remove mbedtls_ssl_conf_sig_hashes(). Use mbedtls_ssl_conf_sig_algs() instead.
- Removed all public key sample programs from the programs/pkey directory.
- Removed support for TLS 1.2 static ECDH key exchanges (ECDH-ECDSA and ECDH-RSA).
- Drop support for the GNU Make and Microsoft Visual Studio build systems.

Features

- Add the function mbedtls_ssl_export_keying_material() which allows the client and server to extract additional shared symmetric keys from an SSL session, according to the TLS-Exporter specification in RFC 8446 and 5705. This requires MBEDTLS_SSL_K...

Read more

▶ **Assets** 4

🎉 9 9 people reacted

Oct 15, 2025

minosgalanakis mbedtls-3.6.5 e185d7f Compare

Mbed TLS 3.6.5

Description

This release includes fixes for security issues.

Mbed TLS 3.6 is a long-term support (LTS) branch. It will be supported with bug-fixes and security fixes until at least March 2027.

Security Advisories

The two issues fixed were timing side channels.

For full details, please see the following links:

- Padding oracle through timing of cipher error reporting
- Side channel in RSA key generation and operations (SSBleed, M-Step)

Release Notes

API changes

- When building the library as a PSA client (MBEDTLS_PSA_CRYPTO_CLIENT enabled and MBEDTLS_PSA_CRYPTO_C disabled), you need to provide the

function psa_can_do_cipher() in addition to psa_can_do_hash(). This changed was made in Mbed TLS 3.6.0 but was not announced then.

Features

- The new function mbedtls_cipher_finish_padded() is similar to mbedtls_cipher_finish(), but makes it easier to process invalid-padding conditions in constant time.

Security

- Fix a timing side channel in CBC-PKCS7 decryption that could allow an attacker who can submit chosen ciphertexts to recover some plaintexts through a timing-based padding oracle attack. Credits to Beat Heeb from Oberon microsystems AG. CVE-2025-59438
- Fix a local timing side-channel in modular inversion and GCD that was exploitable in RSA key generation and other RSA operations (see the full advisory for details), allowing a local attacker to fully recover the private key. This can be exploited on some Arm-v9 CPUs by an unprivileged attacker running code on the same core (SSBleed), or when Trustzone-M is used, by the non-secure side abusing timer interrupts (M-Step), and probably in other similar settings as well. Found and reported independently by: SSBleed: Chang Liu (Tsinghua University) and Trevor E. Carlson (National University of Singapore); M-Step: Cristiano Rodrigues (University of Minho), Marton Bognar (DistriNet, KU Leuven), Sandro Pinto (University of Minho), Jo Van Bulck (DistriNet, KU Leuven). CVE-2025-54764

Bugfix

- Fix potential CMake parallel build failure when building both the static and shared libraries.
- Fix a build error or incorrect TLS session lifetime on platforms where mbedtls_time_t is not time_t. Fixes #10236.

Changes

- The function mbedtls_mpi_gcd() now always gives a non-negative output. Previously the output was negative when B = 0 and A < 0, which was not documented, and inconsistent as all other inputs resulted in a non-negative output.

Who should update

This release includes security fixes. We strongly recommend evaluating your exposure and, if applicable, prioritizing an upgrade.

Note

> ❕ `mbedtls-3.6.5.tar.bz2` and `mbedtls-3.6.5.tar.bz2-sha256sum.txt` are our official release files. `source.tar.gz` and `source.zip` are automatically generated snapshots that GitHub is generating. They do not include external dependencies, and can't be configured

Checksum

The SHA256 hash for the archive is:
4a11f1777bb95bf4ad96721cac945a26e04bf19f57d905f241fe77ebeddf46d8 mbedtls-3.6.5.tar.bz2

▶ **Assets** 4

👍 4 4 people reacted

Jun 30, 2025

minosgalanakis mbedtls-3.6.4 c765c83 Compare

Mbed TLS 3.6.4

Description

This release includes fixes for security issues.

Mbed TLS 3.6 is a long-term support (LTS) branch. It will be supported with bug-fixes and security fixes until at least March 2027.

Who should update

We recommend all users should update to take advantage of the bug fixes contained in this release at an appropriate point in their development lifecycle.

Security Advisories

For full details, please see the following links:

- Race condition in AESNI support detection
- Heap buffer under-read when parsing PEM-encrypted material
- Unchecked return value in LMS verification allows signature bypass
- Out-of-bounds read in mbedtls_lms_import_public_key()
- Timing side-channel in block cipher decryption with PKCS#7 padding
- NULL pointer dereference after using `mbedtls_asn1_store_named_data()`
- Misleading memory management in `mbedtls_x509_string_to_names()`

Release Notes

Features

- Add the function mbedtls_ssl_export_keying_material() which allows the client and server to extract additional shared symmetric keys from an SSL session, according to the TLS-Exporter specification in RFC 8446 and 5705. This requires MBEDTLS_SSL_KEYING_MATERIAL_EXPORT to be defined in mbedtls_config.h.

Security

- Fix a buffer overread in mbedtls_lms_import_public_key() when the input is less than 3 bytes. Reported by Linh Le and Ngan Nguyen from Calif. CVE-2025-49601
- Fix a vulnerability in LMS verification through which an adversary could get an invalid signature accepted if they could cause a hash accelerator to fail. Found and reported by Linh Le and Ngan Nguyen from Calif. CVE-2025-49600
- On x86/amd64 platforms, with some compilers, when the library is compiled with support for both AESNI and software AES and AESNI is available in hardware, an adversary with fine control over which threads make progress in a multithreaded program could force software

AES to be used for some time when the program starts. This could allow the adversary to conduct timing attacks and potentially recover the key. In particular, this attacker model may be possible against an SGX enclave.
The same vulnerability affects GCM acceleration, which could allow a similarly powerful adversary to craft GCM forgeries.
CVE-2025-52496

- Fix possible use-after-free or double-free in code calling mbedtls_x509_string_to_names(). This was caused by the function calling mbedtls_asn1_free_named_data_list() on its head argument, while the documentation did no suggest it did, making it likely for callers relying on the documented behaviour to still hold pointers to memory blocks after they were free()d, resulting in high risk of use-after-free or double-free, with consequences ranging up to arbitrary code execution.
  In particular, the two sample programs x509/cert_write and x509/cert_req were affected (use-after-free if the san string contains more than one DN). Code that does not call mbedtls_string_to_names() directly is not affected.
  Found by Linh Le and Ngan Nguyen from Calif.
  CVE-2025-47917
- Fix a bug in mbedtls_asn1_store_named_data() where it would sometimes leave an item in the output list in an inconsistent state with val.p == NULL but val.len > 0. This impacts applications that call this function directly, or indirectly via mbedtls_x509_string_to_names() or one of the mbedtls_x509write_{crt,csr}*set*{subject,issuer}_name() functions. The inconsistent state of the output could then cause a NULL dereference either inside the same call to mbedtls_x509_string_to_names(), or in subsequent users of the output structure, such as mbedtls_x509_write_names(). This only affects applications that create (as opposed to consume) X.509 certificates, CSRs or CRLs, or that call mbedtls_asn1_store_named_data() directly. Found by Linh Le and Ngan Nguyen from Calif.
  CVE-2025-48965
- Fix an integer underflow that could occur when parsing malformed PEM keys, which could be used by an attacker capable of feeding encrypted PEM keys to a user. This could cause a crash or information disclosure. Found and reported by Linh Le and Ngan Nguyen from Calif.
  CVE-2025-52497
- Fix a timing side channel in the implementation of PKCS#7 padding which would allow an attacker who can request decryption of arbitrary ciphertexts to recover the plaintext through a timing oracle attack. Reported by Ka Lok Wu from Stony Brook University and Doria Tang from

The Chinese University of Hong Kong.
CVE-2025-49087

Bugfix

- Fix failures of PSA multipart or interruptible operations when the library or the application is built with a compiler where "union foo x = {0}" does not initialize non-default members of the union, such as GCC 15 and some versions of Clang 18. This affected MAC multipart operations, MAC-based key derivation operations, interruptible signature, interruptible verification, and potentially other operations when using third-party drivers. This also affected one-shot MAC operations using the built-in implementation. Fixes #9814.
- On entry to PSA driver entry points that set up a multipart operation ("xxx_setup"), the operation object is supposed to be all-bits-zero. This was sometimes not the case when an operation object is reused, or with compilers where "union foo x = {0}" does not initialize non-default members of the union. The PSA core now ensures that this guarantee is met in all cases. Fixes #9975.
- Resolved build issue with C++ projects using Mbed TLS 3.6 when compiling with the MSVC toolset v142 and earlier. Fixes mbedtls issue #7087.
- Silence spurious -Wunterminated-string-initialization warnings introduced by GCC 15. Fixes #9944.
- Fix a sloppy check in LMS public key import, which could lead to accepting keys with a different LMS or LM-OTS types on some platforms. Specifically, this could happen on platforms where enum types are smaller than 32 bits and compiler optimization is enabled. Found and reported by Linh Le and Ngan Nguyen from Calif.
- Fix a race condition on x86/amd64 platforms in AESNI support detection that could lead to using software AES in some threads at the very beginning of a multithreaded program. Reported by Solar Designer. Fixes #9840.
- Fix mbedtls_base64_decode() on inputs that did not have the correct number of trailing equal signs, or had 4*k+1 digits. They were accepted as long as they had at most two trailing equal signs. They are now rejected. Furthermore, before, on inputs with too few equal signs, the function reported the correct size in *olen when it returned MBEDTLS_ERR_BASE64_BUFFER_TOO_SMALL, but truncated the output to the last multiple of 3 bytes.

- When calling mbedtls_asn1_write_raw_buffer() with NULL, 0 as the last two arguments, undefined behaviour would be triggered, in the form of a call to memcpy(..., NULL, 0). This was harmless in practice, but could trigger complains from sanitizers or static analyzers.

Changes

- The function mbedtls_x509_string_to_names() now requires its head argument to point to NULL on entry. This makes it likely that existing risky uses of this function (see the entry in the Security section) will be detected and fixed.

Note

> ❗ `mbedtls-3.6.4.tar.bz2` and `mbedtls-3.6.4-easy-make-lib.tar.bz2` are our official release files. `source.tar.gz` and `source.zip` are automatically generated snapshot's that github is generating. They do not include external dependencies, and can't be configured. Between the `mbedtls-3.6.4*` archives, pick whichever is convenient for you, noting that `mbedtls-3.6.4.tar.bz2` has more requirements than intended to run `make lib`.

Checksum

The SHA-256 hashes for the archives are:

```
40d83e4040c3e548a61f6ac19b697b0380149abafd6923196a70b6067b543261  mbedtls-3.6.4-easy-make-lib.tar.bz2
6a7ed66b4aca38836f0eff8d8fba72992bf0c7326337608ef01de469fd8368bd  mbedtls-3.6.4-easy-make-lib.tar.xz
ec35b18a6c593cf98c3e30db8b98ff93e8940a8c4e690e66b41dfc011d678110  mbedtls-3.6.4.tar.bz2
```

Archive history:

- 2025-07-31: Replaced `mbedtls-3.6.4.tar.bz2` with a new archive that has exactly the same file content, but file timestamps have been updated to fix #10332. This archive was accidentally uploaded with xz compression, despite the `.bz2` file name.
- 2025-08-01: Restored the original `mbedtls-3.6.4.tar.bz2` and published the fix for #10332 with a new file name `mbedtls-3.6.4-easy-make-lib.tar.bz2`, and `mbedtls-3.6.4-easy-make-lib.tar.xz` for the xz version).

▶ **Assets** 6

Jul 4, 2025

minosgalanakis mbedtls-4.0.0… 71157fd Compare

Mbed TLS 4.0.0-beta

Pre-release

Description

The Mbed TLS 4.0.0-beta and TF-PSA-Crypto 1.0.0-beta releases introduce a major codebase restructuring:

- PSA Crypto functionality now resides in its own repository Mbed-TLS/TF-PSA-Crypto.
- TLS and X.509 components remain in Mbed TLS.

API Changes & Migration

These betas include significant API changes that break backward compatibility with previous releases. Please test your integration thoroughly and follow the 4.0-migration-guide to update your codebase to the new interfaces.

Please note

- These beta versions **must be used together** to validate your integration against the newly split interfaces.
- The 4.0 Migration Guide is currently under construction and will be continuously updated as we prepare for the full, non-beta release.

Intended audience for this beta

- Integrators: To evaluate the impact of the Mbed TLS split and API changes on your codebase.
- Early adopters: Anyone who wants to experiment with the upcoming interfaces and provide feedback before the formal release.

These betas are not production-ready. For deployments requiring stability and the latest security fixes, please continue using the LTS release Mbed TLS 3.6.4

Security Advisories

For full details, please see the following links:

- `CTR_DRBG` prioritized over `HMAC_DRBG` as the PSA DRBG
- Stack buffer overflow in ECDSA signature conversion functions
- Limited authentication bypass in TLS 1.3 optional client authentication
- Buffer underrun in pkwrite when writing an opaque key pair
- TLS clients should generally call mbedtls_ssl_set_hostname
- Potential authentication bypass in TLS handshake
- Misleading memory management in `mbedtls_x509_string_to_names()`
- NULL pointer dereference after using `mbedtls_asn1_store_named_data()`

Release Notes

API changes

- The experimental functions psa_generate_key_ext() and psa_key_derivation_output_key_ext() have been replaced by psa_generate_key_custom() and psa_key_derivation_output_key_custom(). They have almost exactly the same interface, but the variable-length data is passed in a separate parameter instead of a flexible array member. This resolves a build failure under C++ compilers that do not support flexible array members (a C99 feature not adopted by C++). Fixes #9020.
- Align the mbedtls_ssl_ticket_setup() function with the PSA Crypto API. Instead of taking a mbedtls_cipher_type_t as an argument, this function now takes 3 new arguments: a PSA algorithm, key type and key size, to specify the AEAD for ticket protection.
- The PSA and Mbed TLS error spaces are now unified. mbedtls_xxx() functions can now return PSA_ERROR_xxx values.
  There is no longer a distinction between "low-level" and "high-level" Mbed TLS error codes.
  This will not affect most applications since the error values are between -32767 and -1 as before.

- All API functions now use the PSA random generator psa_get_random() internally. As a consequence, functions no longer take RNG parameters. Please refer to the migration guide at : tf-psa-crypto/docs/4.0-migration-guide.md.

Default behavior changes

- In a PSA-client-only build (i.e. MBEDTLS_PSA_CRYPTO_CLIENT && !MBEDTLS_PSA_CRYPTO_C), do not automatically enable local crypto when the corresponding PSA mechanism is enabled, since the server provides the crypto. Fixes #9126.
- The PK, X.509, PKCS7 and TLS modules now always use the PSA subsystem to perform cryptographic operations, with a few exceptions documented in docs/architecture/psa-migration/psa-limitations.md. This corresponds to the behavior of Mbed TLS 3.x when MBEDTLS_USE_PSA_CRYPTO is enabled. In effect, MBEDTLS_USE_PSA_CRYPTO is now always enabled.
- psa_crypto_init() must be called before performing any cryptographic operation, including indirect requests such as parsing a key or certificate or starting a TLS handshake.
- The `PSA_WANT_XXX` symbols as defined in tf-psa-crypto/include/psa/crypto_config.h are now always used in the configuration of the cryptographic mechanisms exposed by the PSA API. This corresponds to the configuration behavior of Mbed TLS 3.x when MBEDTLS_PSA_CRYPTO_CONFIG is enabled. In effect, MBEDTLS_PSA_CRYPTO_CONFIG is now always enabled and the configuration option has been removed.
- In TLS clients, if mbedtls_ssl_set_hostname() has not been called, mbedtls_ssl_handshake() now fails with MBEDTLS_ERR_SSL_CERTIFICATE_VERIFICATION_WITHOUT_HOSTNAME if certificate-based authentication of the server is attempted. This is because authenticating a server without knowing what name to expect is usually insecure.

Removals

- Drop support for VIA Padlock. Removes MBEDTLS_PADLOCK_C. Fixes #5903.
- Drop support for crypto alt interface. Removes MBEDTLS_XXX_ALT options at the module and function level for crypto mechanisms only. The remaining

alt interfaces for platform, threading and timing are unchanged.
Fixes #8149.

- Remove support for the RSA-PSK key exchange in TLS 1.2.
- Remove deprecated mbedtls_x509write_crt_set_serial(). The function was already deprecated and superseeded by mbedtls_x509write_crt_set_serial_raw().
- Remove the function mbedtls_ssl_conf_curves() which had been deprecated in favour of mbedtls_ssl_conf_groups() since Mbed TLS 3.1.
- Remove support for the DHE-PSK key exchange in TLS 1.2.
- Remove support for the DHE-RSA key exchange in TLS 1.2.
- Following the removal of DHM module (#9972 and TF-PSA-Crypto#175) the following SSL functions are removed:
  - mbedtls_ssl_conf_dh_param_bin
  - mbedtls_ssl_conf_dh_param_ctx
  - mbedtls_ssl_conf_dhm_min_bitlen
- Remove support for the RSA key exchange in TLS 1.2.
- Remove mbedtls_low_level_sterr() and mbedtls_high_level_strerr(), since these concepts no longer exists. There is just mbedtls_strerror().
  * Removal of the following sample programs:
  pkey/rsa_genkey.c
  pkey/pk_decrypt.c
  pkey/dh_genprime.c
  pkey/rsa_verify.c
  pkey/mpi_demo.c
  pkey/rsa_decrypt.c
  pkey/key_app.c
  pkey/dh_server.c
  pkey/ecdh_curve25519.c
  pkey/pk_encrypt.c
  pkey/rsa_sign.c
  pkey/key_app_writer.c
  pkey/dh_client.c
  pkey/ecdsa.c
  pkey/rsa_encrypt.c
  wince_main.c
  aes/crypt_and_hash.c
  random/gen_random_ctr_drbg.c
  random/gen_entropy.c
  hash/md_hmac_demo.c

hash/hello.c
hash/generic_sum.c
cipher/cipher_aead_demo.c

- Remove compat-2-x.h header from mbedtls.
- The library no longer offers interfaces to look up values by OID or OID by enum values.
  The header <mbedtls/oid.h> now only defines functions to convert between binary and dotted string OID representations, and macros for OID strings that are relevant to X.509.
  The compilation option MBEDTLS_OID_C no longer exists. OID tables are included in the build automatically as needed.

Features

- When the new compilation option MBEDTLS_PSA_KEY_STORE_DYNAMIC is enabled, the number of volatile PSA keys is virtually unlimited, at the expense of increased code size. This option is off by default, but enabled in the default mbedtls_config.h. Fixes #9216.
- Add a new psa_key_agreement() PSA API to perform key agreement and return an identifier for the newly created key.
- Added new configuration option MBEDTLS_PSA_STATIC_KEY_SLOTS, which uses static storage for keys, enabling malloc-less use of key slots.
  The size of each buffer is given by the option MBEDTLS_PSA_STATIC_KEY_SLOT_BUFFER_SIZE. By default it accommodates the largest PSA key enabled in the build.
- Add an interruptible version of key agreement to the PSA interface.
  See psa_key_agreement_iop_setup() and related functions.
- Add an interruptible version of generate key to the PSA interface.
  See psa_generate_key_iop_setup() and related functions.
- Add the function mbedtls_ssl_export_keying_material() which allows the client and server to extract additional shared symmetric keys from an SSL session, according to the TLS-Exporter specification in RFC 8446 and 5705.
  This requires MBEDTLS_SSL_KEYING_MATERIAL_EXPORT to be defined in mbedtls_config.h.

Security

- Unlike previously documented, enabling MBEDTLS_PSA_HMAC_DRBG_MD_TYPE does not cause the PSA subsystem to use HMAC_DRBG: it uses HMAC_DRBG only when MBEDTLS_PSA_CRYPTO_EXTERNAL_RNG and MBEDTLS_CTR_DRBG_C are

disabled.
C...

Read more

▶ **Assets** 4

May 13, 2025

yanesca v3.6.3.1 6fb5120 Compare

Mbed TLS 3.6.3.1

Description

This release is fully identical to Mbed TLS 3.6.3 in content, but without the `.gitmodules` file, which was left in by mistake and was causing difficulties for users who are getting Mbed TLS through git.

Mbed TLS 3.6 is a long-term support (LTS) branch. It will be supported with bug-fixes and security fixes until at least March 2027.

Note

> ❗ `mbedtls-3.6.3.1.tar.bz2` and `mbedtls-3.6.3.1.tar.bz2-sha256sum.txt` are our official release files. `source.tar.gz` and `source.zip` are automatically generated snapshot's that github is generating. They do not include external dependencies, and can't be configured

Checksum

The SHA256 hash for the archive is:
243ed496d5f88a5b3791021be2800aac821b9a4cc16e7134aa413c58b4c20e0c mbedtls-3.6.3.1.tar.bz2

▶ **Assets** 4

😄 2 ❤️ 6 7 people reacted

Mar 24, 2025

minosgalanakis mbedtls-3.6.3 22098d4 Compare

Mbed TLS 3.6.3

Description

This release of Mbed TLS provides the fix for a tls compatibility issue of handling fragmented handshake messages. This release includes fixes for security issues.

Mbed TLS 3.6 is a long-term support (LTS) branch. It will be supported with bug-fixes and security fixes until at least March 2027.

Security Advisories

For full details, please see the following links:

- Potential authentication bypass in TLS handshake
- TLS clients should generally call mbedtls_ssl_set_hostname

Release Notes

Default behavior changes

- In TLS clients, if mbedtls_ssl_set_hostname() has not been called, mbedtls_ssl_handshake() now fails with MBEDTLS_ERR_SSL_CERTIFICATE_VERIFICATION_WITHOUT_HOSTNAME if certificate-based authentication of the server is attempted. This is because authenticating a server without knowing what name to expect is usually insecure. To restore the old behavior, either call mbedtls_ssl_set_hostname() with NULL as the hostname, or enable the new compile-time option MBEDTLS_SSL_CLI_ALLOW_WEAK_CERTIFICATE_VERIFICATION_WITHOUT_HOSTNAME.

Features

- Added new configuration option MBEDTLS_PSA_STATIC_KEY_SLOTS, which uses static storage for keys, enabling malloc-less use of key slots.
  The size of each buffer is given by the option MBEDTLS_PSA_STATIC_KEY_SLOT_BUFFER_SIZE. By default it accommodates the largest PSA key enabled in the build.
- MD module can now perform PSA dispatching also when `MBEDTLS_PSA_CRYPTO_CLIENT && !MBEDTLS_PSA_CRYPTO_C`, even though this configuration is not officially supported. This requires that a PSA Crypto provider library which:
  - supports the required `PSA_WANT_ALG_xxx` and
  - implements `psa_can_do_hash()` on the client interface is linked against Mbed TLS and that `psa_crypto_init()` is called before performing any PSA call.

Security

- Note that TLS clients should generally call mbedtls_ssl_set_hostname() if they use certificate authentication (i.e. not pre-shared keys).
  Otherwise, in many scenarios, the server could be impersonated.
  The library will now prevent the handshake and return MBEDTLS_ERR_SSL_CERTIFICATE_VERIFICATION_WITHOUT_HOSTNAME if mbedtls_ssl_set_hostname() has not been called.
  Reported by Daniel Stenberg.
  CVE-2025-27809
- Zeroize a temporary heap buffer used in psa_key_derivation_output_key() when deriving an ECC key pair.
- Zeroize temporary heap buffers used in PSA operations.
- Fix a vulnerability in the TLS 1.2 handshake. If memory allocation failed or there was a cryptographic hardware failure when calculating the Finished message, it could be calculated incorrectly. This would break the security guarantees of the TLS handshake.
  CVE-2025-27810

Bugfix

- When MBEDTLS_SSL_TLS1_3_COMPATIBILITY_MODE is disabled, work with peers that have middlebox compatibility enabled, as long as no problematic middlebox is in the way. Fixes #9551.

- Fix invalid JSON schemas for driver descriptions used by generate_driver_wrappers.py.
- Use 'mbedtls_net_close' instead of 'close' in 'mbedtls_net_bind' and 'mbedtls_net_connect' to prevent possible double close fd problems. Fixes #9711.
- Fix undefined behavior in some cases when mbedtls_psa_raw_to_der() or mbedtls_psa_der_to_raw() is called with bits=0.
- Fix compilation on MS-DOS DJGPP. Fixes #9813.
- Fix missing constraints on the AES-NI inline assembly which is used on GCC-like compilers when building AES for generic x86_64 targets. This may have resulted in incorrect code with some compilers, depending on optimizations. Fixes #9819.
- Support re-assembly of fragmented handshake messages in TLS (both 1.2 and 1.3). The lack of support was causing handshake failures with some servers, especially with TLS 1.3 in practice. There are a few limitations, notably a fragmented ClientHello is only supported when TLS 1.3 support is enabled. See the documentation of mbedtls_ssl_handshake() for details.
- Fix definition of MBEDTLS_PRINTF_SIZET to prevent runtime crashes that occurred whenever SSL debugging was enabled on a copy of Mbed TLS built with Visual Studio 2013 or MinGW.
  Fixes #10017.
- Remove Everest Visual Studio 2010 compatibility headers, which could shadow standard CRT headers inttypes.h and stdbool.h with incomplete implementatios if placed on the include path, eg. when building Mbed TLS with the .sln file shipped with the project.
- Fix issue where psa_key_derivation_input_integer() is not detecting bad state after an operation has been aborted.

Changes

- Improve performance of PSA key generation with ECC keys: it no longer computes the public key (which was immediately discarded). Fixes #9732.

Who should update

We recommend all users should update to take advantage of the bug fixes contained in this release at an appropriate point in their development lifecycle.

Note

> ❗ `mbedtls-3.6.3.tar.bz2` are our official release files. `source.tar.gz` and `source.zip` are automatically generated snapshot's that github is generating. They do not include external dependencies, and can't be configured

Checksum

The SHA256 hashes for the archives are:
64cd73842cdc05e101172f7b437c65e7312e476206e1dbfd644433d11bc56327 mbedtls-3.6.3.tar.bz2

▸ **Assets** 4

❤️ 5 🚀 1 6 people reacted

Mar 24, 2025

minosgalanakis mbedtls-2.28.10 2fc8413 Compare

Mbed TLS 2.28.10

Description

This release of Mbed TLS provides bug fixes and minor enhancements. This release includes fixes for security issues.

Mbed TLS 2.28.10 is the last release of the 2.28 LTS and won't receive bug fixes or security fixes anymore.
Users are advised to upgrade to a maintained version.

Security Advisories

For full details, please see the following links:

- Potential authentication bypass in TLS handshake
- TLS clients should generally call mbedtls_ssl_set_hostname

Release Notes

Default behavior changes

- In TLS clients, if mbedtls_ssl_set_hostname() has not been called, mbedtls_ssl_handshake() now fails with MBEDTLS_ERR_SSL_CERTIFICATE_VERIFICATION_WITHOUT_HOSTNAME if certificate-based authentication of the server is attempted.
  This is because authenticating a server without knowing what name to expect is usually insecure. To restore the old behavior, either call mbedtls_ssl_set_hostname() with NULL as the hostname, or enable the new compile-time option MBEDTLS_SSL_CLI_ALLOW_WEAK_CERTIFICATE_VERIFICATION_WITHOUT_HOSTNAME.
  The content of ssl->hostname after mbedtls_ssl_set_hostname(ssl, NULL) has changed, see the documentation of the hostname field in the mbedtls_ssl_context struct type for details.

Security

- Note that TLS clients should generally call mbedtls_ssl_set_hostname() if they use certificate authentication (i.e. not pre-shared keys).
  Otherwise, in many scenarios, the server could be impersonated.
  The library will now prevent the handshake and return MBEDTLS_ERR_SSL_CERTIFICATE_VERIFICATION_WITHOUT_HOSTNAME if mbedtls_ssl_set_hostname() has not been called.
  CVE-2025-27809
- Zeroize temporary heap buffers used in PSA operations.
- Fix a vulnerability in the TLS 1.2 handshake. If memory allocation failed or there was a cryptographic hardware failure when calculating the Finished message, it could be calculated incorrectly. This would break the security guarantees of the TLS handshake.
  CVE-2025-27810

Bugfix

- Use 'mbedtls_net_close' instead of 'close' in 'mbedtls_net_bind' and 'mbedtls_net_connect' to prevent possible double close fd problems. Fixes #9711.
- Fix compilation on MS-DOS DJGPP. Fixes #9813.

- Fix missing constraints on the AES-NI inline assembly which is used on GCC-like compilers when building AES for generic x86_64 targets. This may have resulted in incorrect code with some compilers, depending on optimizations. Fixes #9819.
- Fix issue where psa_key_derivation_input_integer() is not detecting bad state after an operation has been aborted.
- Fix definition of MBEDTLS_PRINTF_SIZET to prevent runtime crashes that occurred whenever SSL debugging was enabled on a copy of Mbed TLS built with Visual Studio 2013 or MinGW.
  Fixes #10017.
- Remove Everest Visual Studio 2010 compatibility headers, which could shadow standard CRT headers inttypes.h and stdbool.h with incomplete implementatios if placed on the include path, eg. when building Mbed TLS with the .sln file shipped with the project.

Who should update

We recommend all users should update to take advantage of the bug fixes contained in this release at an appropriate point in their development lifecycle.

Note

> ❗ `mbedtls-2.28.10.tar.bz2` are our official release files. `source.tar.gz` and `source.zip` are automatically generated snapshot's that github is generating. They do not include external dependencies, and can't be configured

Checksum

The SHA256 hashes for the archives are:
19e5b81fdac0fe22009b9e2bdcd52d7dcafbf62bc67fc59cf0a76b5b5540d149 mbedtls-2.28.10.tar.bz2

▶ **Assets** 4

Oct 15, 2024

davidhorstmann-arm mbedtls-3.6.2 107ea89 Compare

Mbed TLS 3.6.2

Description

This release of Mbed TLS provides the fix for a security vulnerability.

Mbed TLS 3.6 is a long-term support (LTS) branch. It will be supported with bug-fixes and security fixes until at least March 2027.

Security Advisories

For full details, please see the following links:

- Buffer underrun in pkwrite when writing an opaque key pair

> ❗ **Release notes are truncated in GitHub's releases page**: Please refer to the 3.6.2 release page.

Release Notes

Security

- Fix a buffer underrun in mbedtls_pk_write_key_der() when
  called on an opaque key, MBEDTLS_USE_PSA_CRYPTO is enabled,
  and the output buffer is smaller than the actual output.
  Fix a related buffer underrun in mbedtls_pk_write_key_pem()
  when called on an opaque RSA key, MBEDTLS_USE_PSA_CRYPTO is enabled
  and MBEDTLS_MPI_MAX_SIZE is smaller than needed for a 4096-bit RSA key.
  CVE-2024-49195

Who should update

We recommend all users should update to take advantage of the bug fixes contained in this release at an appropriate point in their development lifecycle.

Note

> ❕ `mbedtls-3.6.2.tar.bz2` is our official release file. `source.tar.gz` and `source.zip` are automatically generated snapshots that github generates. They do not include external dependencies, and can't be configured

Checksum

The SHA256 hash for the archive is:

8b54fb9bcf4d5a7078028e0520acddefb7900b3e66fec7f7175bb5b7d85ccdca mbedtls-3.6.2.tar.bz2

▶ **Assets** 4

👍 1 🚀 4 👀 1 6 people reacted