Compiler-induced constant-time violations (CVE-2025-66442)

Title	Compiler-induced constant-time violations
CVE	CVE-2025-66442
Date	31 March 2026
Affects	All versions of Mbed TLS and TF-PSA-Crypto
Not affected	None
Impact	Timing-based side channel leakage in RSA and CBC/ECB decryption
Severity	LOW
Credits	Jing Liu, Zhiyuan Zhang, Lucía Martínez Gavier, Gilles Barthe and Marcel Böhme (Max

Vulnerability

Summary: Modern versions of Clang introduce timing side channels into padding validation code on some platforms when LLVM's select-optimize feature is enabled.

The precise timing of certain operations involving secret data can reveal the nature of the data. This is especially true for decryption of formats involving padding such as RSA-PKCS#1v1.6, RSA-OAEP, and symmetric ciphers using the modes ECB or CBC with padding.

An adversary who can submit chosen ciphertexts and observe their decryption precisely may be able to mount a padding oracle attack: if the adversary can measure the exact timing with which the victim rejects invalid padding, they may obtain some information. By successively tweaking a target ciphertext and learning how the padding is invalid, the adversary may be able to decrypt this target ciphertext.

TF-PSA-Crypto and Mbed TLS include some specially-written code that attempts to report invalid padding in constant time and constant flow, so that the way in which padding is invalid does not leak through the timing of the decryption operation or through shared system state such as caches and branch predictors.

Some compiler optimizations can defeat this constant-time protection, if the compiler recognizes the functional behavior of the code and emits binary code that is functionally equivalent, but not constant-time.

Researchers from MPI-SP have discovered that modern versions of Clang perform an optimization that defeats constant-time protection when given the LLVM select-optimize feature is enabled (e.g. `clang -O -mllvm --disable-select-optimize=false`). This optimization causes a branch to be introduced during an optimization phase.

On some platforms, the library has assembly language constant-time primitives, which are not affected by this optimization. This is the case for all 32-bit and 64-bit arm and x86 processors, provided that `MBEDTLS_HAVE_ASM` is enabled in the library configuration and Clang or some other GCC-like compiler is used.

Impact

If TF-PSA-Crypto or Mbed TLS is built with Clang 18 with the LLVM select-optimize feature enabled, compiling for 64-bit RISC-V, the following features are known to be vulnerable to a timing oracle attack:

- RSA PKCS#1v1.5 decryption;
- One-and-zeros unpadding (`MBEDTLS_PADDING_ONE_AND_ZEROS`) (not available in TF-PSA-Crypto).

The attack may allow an adversary who can submit ciphertexts and can make precise timing measurements of the decryption process to recover ciphertexts, but not the key.

Note that it is likely that other compiler versions and other processor architectures are affected. Due to the very high variability of compiler optimizations, we cannot provide more details at this time.

Affected versions

All current versions of TF-PSA-Crypto and Mbed TLS are affected.

Work-around

Clang with default optimization sets (`-O3` or less, `-Os`, `-Oz`) is not known to be affected. Other compilers are not known to be affected.

Arm and x86 architectures (both 32-bit and 64-bit) are not affected if `MBEDTLS_HAVE_ASM` is enabled in the library configuration.

In general, we recommend that users do not enable advanced compiler optimization options, especially ones such as select-optimize which can introduce branches.

If possible, we recommend that applications do not use encryption mode that rely on variable-length padding: use an AEAD mode rather than CBC, and use RSA OAEP rather than PKCS#1v1.5 encryption.

Resolution

Affected users should rebuild TF-PSA-Crypto or Mbed TLS with the LLVM option select-optimize disabled, for example by using only default optimization flags such as `-O2` or `-Os`.

Fix commits

No fix is available at this time.