TA-152-R0

Authored and Designed by Riyan Dahiya

Cipher Specifications

Contents

1. About

TA-152-R0 is a custom symmetric encryption scheme, with an experimental scope. It operates using a streaming substitution cipher with a continuously evolving permutation state matrix.

The design focusses on:

1. Per-byte state evolution.
2. Deterministic reversibility.
3. Minimal runtime dependencies.
4. Explicit, inspectable behaviour.
5. Simple, byte-oriented operation without reliance on block primitives.

THE ALGORITHM IS CURRENTLY EXPERIMENTAL, AND IS NOT SUITABLE FOR PRODUCTION CRYPTOGRAPHY.

2. Design Characteristics

2.a. Overview

TA-152-R1 is a stateful cipher, with an evolving 256-byte permutation. The state matrix covers all possible byte permutations (0x00 – 0xFF). It uses a 128-bit key, processed one byte per round, and has a block size of 1 byte.

Currently, this scheme only provides a degree of confidentiality, and lacks an authentication or integrity mechanism.

2.b. Key

Keys are 16 bytes in length, are consumed cyclically, and influence permutation evolution on a per-byte basis.

Key reuse is allowed, but discouraged.

Key Format: Raw Binary (16 Bytes)

Weak Keys: Palindromic, and uniform keys are potentially weak. (weak keys have not been formally identified)

2.c. Internal State

The algorithm maintains the following internal state during operation:

i. **base_mx[256]**
Current substitution permutation.

ii. inverse_mx[256]
Inverse substitution permutation, maintained incrementally to allow efficient decryption.

iii. keypos
Index for key byte (mod 16).

The complete state evolves with every byte processed.

2.d. Permutation Evolution

For each byte processed:

1. A round's chunk size is derived from the key value.
   key = 0 or 1 → chunk_size = 2
   otherwise, chunk_size = key

2. The permutation array is partitioned into contiguous chunks of chunk_size.

3. Each chunk is reversed in place.

4. If any tail bytes remain, they are also reversed as a contiguous chunk.

The permutation update is involutive only when mirrored exactly during decryption. Hence, strict ordering symmetry between encryption and decryption is required.

3. Encryption

For each plaintext byte P:

1. Update permutation using the current key byte.

2. Substitute:
   C = base_mx[P]

3. Output C.

4. Advance key index modulo 16.

The ciphertext feedback causes each ciphertext byte to depend on all previous ciphertext bytes, increasing diffusion across the stream.

4. Decryption

For each ciphertext byte C:

1. Update permutation using the current key byte.

2. Substitute:
   P = inverse_mx[C]

3. Output P.

4. Advance key index modulo 16.

Due to ciphertext feedback, decryption must proceed sequentially from the beginning of the stream.

5. File Extension

The encryption function appends a .t152e file extension to the encrypted file. This file extension is purely cosmetic, and is implemented for easier visibility of encrypted data. The presence or absence of this extension does not affect the parsing or decryption of a ciphertext file.

6. Notes and Limitations

The scheme provides deterministic encryption; identical plaintext encrypted under the same key will always produce identical ciphertext. No integrity or authenticity is guaranteed by this scheme. It is potentially vulnerable to known-plaintext analysis under key reuse, and is intended strictly as a pedagogical and experimental construction.