**Section 10: API Reference**

**🔐 CryptoLib (Core Cryptographic Engine)**

**CryptoLib(string password, CryptoLibOptions options)**

* **Description**:  
  Initializes the engine with advanced configuration (salt, behavior flags, etc.).
* **Parameters**:
  + password (string) — Session password
  + options (CryptoLibOptions) — Specifies salt, behavior flags (e.g., PBKDF2), and optional zone-specific modifiers
* **Returns**:
  + CryptoLib instance with explicit configuration

**⚙️ CryptoLibOptions**

CryptoLibOptions(byte[] salt, string zoneInfo = null!, Behaviors behavior = Behaviors.Rfc2898, int iterations = 100\_000)

**🔧 Purpose:**

Allows precise control over session entropy, behavior toggles, and password preprocessing.

**📄 Parameters:**

* **Salt (byte[])**  
  Required. Used as cryptographic entropy during PBKDF2 or SHA-based derivation.  
  Example: new byte[] { 0x1A, 0x2B, 0x3C, ... }
* **ZoneInfo (string)**Optional. Appended to the password before **CoinTable generation**. This allows zone-specific variation in encryption behavior — useful for multi-tenant deployments, user segmentation, or regional encoding requirements.
* **Behavior (Behaviors)**  
  Optional. Specifies behavior flags (e.g., Rfc2898, None). Default is Rfc2898, enabling secure PBKDF2 key expansion.
* **Iterations (int)**  
  Optional. Sets the number of PBKDF2 iterations. Higher values increase security but slow down session initialization. Default is 100\_000.

**✅ Usage Example:**

var options = new CryptoLibOptions(

salt: new byte[] { 0x1A, 0x2B, 0x3C, 0x4D, 0x5E, 0x6F, 0x70, 0x81, 0x92, 0xA3, 0xB4, 0xC5 },

zoneInfo: "US-West",

behavior: Behaviors.Rfc2898,

iterations: 150\_000

);

var crypto = new CryptoLib("my password", options);

**🔐 CryptoLib Methods**

**Encrypt((byte ID, byte TR)[] sequence, byte globalRounds, byte[] input)**

* Encrypts input using the specified transform sequence and global round count.
* Returns encrypted byte array (includes full header for self-contained decryption).

**Decrypt(byte[] encrypted)**

* Decrypts a fully formed Mango packet by extracting version, config, and transform info from its header.
* Returns original plaintext.

**EncryptBlock(byte[] input)**

* Encrypts a raw block without prepending header info.
* Assumes transform settings are cached from a prior full Encrypt() call.

**DecryptBlock(byte[] input)**

* Decrypts a raw encrypted block (headerless).
* Assumes decryption context is loaded via prior Decrypt() call.

**🔍 InputProfiler (Data Type Detection)**

**InputProfile GetInputProfile(byte[] input)**

* Classifies input data (e.g., Random, Natural, Sequence).
* Returns a fully configured InputProfile, including:
  + Sequence: Transform IDs and rounds
  + GlobalRounds: Optimized global repetition count
  + InputType: Detected classification

**Example:**

var profile = InputProfiler.GetInputProfile(input);

var encrypted = crypto.Encrypt(profile.Sequence, profile.GlobalRounds, input);

**📘 Notes & Recommendations**

* ZoneInfo can be used to introduce location- or device-bound variation for multi-tenant encryption scenarios.
* The default behavior (Rfc2898) ensures secure, high-entropy session configuration — it is highly recommended unless you are performing performance-sensitive Munge runs.