# Module 07 – Add-on Sandbox Enforcement & Execution Containment

Module ID: ADDON-SBX-007

Version: 4.1 (Revised CTO Edition)

Layer: Add-on Policy Layer

Status: RELEASE

Dependencies: Module 00, Module 06

## 0. Purpose & Enforcement Point

This module defines the sandbox enforcement and execution containment logic for all add-on components in MaxOneOpen v4.1. It guarantees that no add-on process can escape declared runtime boundaries, modify system state, or initiate external communications beyond scope. All executions are capsule-bound, identity-isolated, and manifest-limited.

## 1. Sandbox Invocation, Lifecycle & Enforcement Layer

Each add-on execution is instantiated via the Add-on Sandbox Controller (ASC), which:  
- validates the Manifest Capsule (Module 06);  
- spawns a hardened, time-limited execution sandbox (SBX\_ID);  
- injects runtime rules from Module 05;  
- commits all interaction points to capsule trace.  
No execution may bypass or modify its assigned sandbox instance.

## 2. Containment Rules, IO Boundaries & Memory Control

Sandbox execution is constrained by:  
- strict memory caps and garbage isolation;  
- no filesystem persistence or global state access;  
- IO limited to manifest-declared interfaces only;  
- all outbound traffic routed via Module 09 (Twin Messaging Relay).  
Any violation triggers immediate sandbox termination and capsule flag.

## 3. Trigger Points, Timeout Control & Violation Response

Each sandbox is monitored for:  
- timeout breaches (TTL expiration);  
- unauthorized system calls (outside rule scope);  
- unclassified network or memory access patterns.  
Violation generates a `Sandbox Violation Capsule (SVC)` and triggers rollback logic via Module 13 (LedgerSync).

## 4. Capsule Format, Execution Trace & Forensic Anchors

Each sandbox session creates:  
- `Sandbox Execution Capsule (SEC)` – `{ sbx\_id, manifest\_id, exec\_hash, duration, result }`  
- `Violation Capsule (SVC)` – `{ capsule\_id, reason, triggering\_event, hash\_anchor, timestamp }`  
- `Containment Policy Record (CPR)` – declares enforced limits per session.  
All outputs are traceable via Module 14 (Audit) and linked to the originating identity.

## 5. Intermodular Enforcement Map & Control Hooks

This module integrates with:  
- Module 01 (Execution Gate) to approve sandboxed launch  
- Module 05 (ConfigBinding) for rule injection  
- Module 06 (Manifest Engine) for scope resolution  
- Module 09 (Twin Messaging) for relay-based IO access  
- Module 13 (LedgerSync) for rollback and state commitment  
- Module 14 (Audit Capsule) for session trace export

## CTO Validation Matrix

Module 07 (CTO Edition) guarantees the following verifiable conditions:  
- Every add-on runs in an isolated, rule-bound sandbox: YES  
- Sandbox boundaries are enforced at memory, IO, and time level: YES  
- Violations produce audit capsules and auto-revoke triggers: YES  
- No add-on can persist state or bypass runtime controls: YES  
- All executions are identity-linked, manifest-scoped, and traceable: YES