MaxDeploy v1.0 – Audit and Export Structure

This document defines the audit trail logic, exportable outputs, and verification mechanisms of MaxDeploy. It ensures traceability, reproducibility, and third-party validation of all deployment actions in capsule-based environments.

# 1. Audit File Generation

Every deployment executed through MaxDeploy generates a set of cryptographically signed audit files that are traceable, verifiable, and linked to the system's Immutable Audit Trail Ledger (IATL).

Core audit outputs include:

* - meta.audit.json – structured audit file with hash, scope, operator ID
* - audit.trace.sig – digitally signed hash verification trail
* - manifest.deploy.yaml – embedded audit anchor reference

# 2. Ledger Integration

Audit files produced by MaxDeploy are registered in the IATL. These entries contain a sequential hash chain, timestamp, role signature, and context binding to the deployment capsule. The ledger is append-only and independently verifiable.

# 3. Export Mechanisms

Audit capsules can be exported using the standard capsule-compatible export protocol. External institutions (e.g., regulators, certification bodies, or independent auditors) may request export bundles containing:

* - meta.audit.json (verifiable in air-gapped environments)
* - audit.trace.sig (signed timestamped proof chain)
* - operator blueprint (to prove delegation or authority)

# 4. Reproducibility Criteria

MaxDeploy enforces reproducibility by ensuring that every verified deployment capsule can be re-executed identically on any trusted MaxOneOpen-compatible instance. This requires:

* - Reference to a fixed anchor.exec.ref
* - Lock-in of all upstream policy and manifest versions
* - Permanent hash-locked execution ID (xID)

# 5. Public Disclosure Capability

MaxDeploy supports audit transparency in OSS and institutional environments. Optional configuration allows for automatic publishing of audit trails to public registries or treaty-bound transparency portals.