Procedura 4 — Naprawa LSIG (Init Lsig Repair)

Wersja: 1.0
Status: Referencyjna
Autorzy: Antzedek & Eliasz (AJ Power)
Cel dokumentu: Przywrócić ważność podpisu **LSIG** (Logic/Latent Signature) dla istniejącego **Anchora**, gdy integralność podpisu nie przechodzi weryfikacji, przy zachowaniu immutowalności SR i pełnego śladu audytowego.

---

1. Zakres i definicje

- **LSIG:** Podpis kryptograficzny hasha `SR.pkg` (Snapshot Referencyjny).
- **SR (Snapshot Referencyjny):** Kanoniczny pakiet artefaktów. **Nie podlega zmianie** w tej procedurze.
- **Repair vs Migration:** Naprawa dotyczy **wyłącznie podpisu**; jeśli SR się zmienił lub uległ degradacji → **Migracja Anchora** (nie w tym dokumencie).
- **PT/PS:** Podpis Twórcy / Podpis Systemu (w TPM/TEE).
- **CRL/ARL:** Lista unieważnień kluczy/anchorów.

---

2. Kiedy uruchamiać (triggery)

- `VerifyEd25519(SR.hash, LSIG.sig, PS_pub) = FAIL` lub (jeśli polityka wymaga) `VerifyEd25519(SR.hash, PT.sig, PT_pub) = FAIL`.
- Utrata/rotacja kluczy `PS_priv` lub `PT_priv`.
- Przeniesienie Anchora na nowe środowisko TEE/TPM.
- Błąd w metadanych podpisu (np. format, expirations) przy zachowanym SR.

> **Warunek wstępny:** `SR.pkg` musi przechodzić test integralności (hash) względem kopii WORM/append-only.

---

3. Wymagania

- Dostęp do `SR.pkg` i `SR.hash` (z WORM).
- Dostęp do `PS_pub` / `PT_pub`; dostęp do nowych `PS_priv` / `PT_priv` (jeśli rotacja).
- Kanał bezpieczny dla podpisu offline (HSM/air-gap) oraz atest TPM/TEE.
- Repozytorium WORM/append-only na nowe artefakty (`LSIG_new.sig`, `SIG_new.json`).

---

4. Procedura krok po kroku

Krok 0 — Preflight i lockdown
1. Wejdź w **Safe Mode** (patrz Procedura 3).
2. Potwierdź integralność SR: `SR.hash' = SHA3-512(SR.pkg)` → **musi =** `SR.hash` z WORM.
3. Jeżeli ≠ → **PRZERWIJ**: to nie „repair”, tylko **migracja Anchora**.

Krok 1 — Atestacja środowiska
4. Jeżeli używasz TEE/TPM: pobierz `tpm_quote` / atest klucza `PS_priv` oraz stan pomiarów.
5. Zaloguj atest w audycie (`quote_id`, PCR, certyfikaty).

Krok 2 — Decyzja: repair czy rotacja kluczy
6. Jeśli `PS_priv/PT_priv` są ważne → przejdź do Krok 3.
7. Jeśli klucze wygasły/kompromitowane → **rotacja kluczy**: wygeneruj pary (`*_priv/_pub`), zaktualizuj metadane, zanotuj w CRL/ARL status poprzednich kluczy.

Krok 3 — Regeneracja LSIG
8. Oblicz `SR.hash` deterministycznie (ponownie).
9. Wykonaj podpis systemowy: `LSIG_new.sig = SignEd25519(SR.hash, PS_priv)` (w TEE/TPM).
10. (Jeśli polityka) Wykonaj podpis twórcy: `PT_new.sig = SignEd25519(SR.hash, PT_priv)` w HSM/offline.

Krok 4 — Metadane i łańcuch zaufania
11. Utwórz `SIG_new.json`: `ps_sig, pt_sig(optional), ps_pub_fp, pt_pub_fp, tpm_quote_id, created_at, policy_ver`.
12. Utwórz wpis w **ARL** (Anchor Record Log): nowy rekord z łańcuchem hashy (poprzedni → obecny), podpisany `PS_priv`.

Krok 5 — Zapis WORM i aktywacja
13. Zapisz `LSIG_new.sig` + `SIG_new.json` do WORM/append-only (obok starego).
14. Zaktualizuj **wskaźnik aktywnego podpisu** (konfiguracja/alias) w sposób transakcyjny (atomiczny swap).
15. Zweryfikuj aktywny podpis: `VerifyEd25519(SR.hash, LSIG_new.sig, PS_pub)` (+ `PT_new.sig` jeśli dotyczy).

Krok 6 — Wyjście z Safe Mode i raport
16. Wyłącz Safe Mode.
17. Wygeneruj **raport audytowy** (PDF/JSON) z hashami, atestami, CRL/ARL, podpisami i timestampem.

---

5. Polityki bezpieczeństwa

- **SR jest nietykalny**: jakakolwiek modyfikacja SR → migracja, nigdy „naprawa LSIG".
- **Dual-sign (zalecane):** aktywacja Anchora wymaga zgodności podpisów `PS` i `PT` (model „człowiek + maszyna").
- **HSM/air-gap dla PT:** podpis twórcy wykonywany poza systemem operacyjnym hosta.
- **CRL/ARL:** prowadź listę unieważnień kluczy i rewizji anchorów; publikuj fingerprinty, statusy i daty.

---

6. Model zagrożeń (skrót)

- **Fałszywy LSIG:** atakujący podmienia podpis → wykrywane przez weryfikację `PS_pub/PT_pub`.
- **Kradzież klucza:** kompromitacja `PS_priv/PT_priv` → rotacja kluczy, wpis do CRL, podpis nowymi kluczami.

- **Atak na TEE/TPM:** nieprawidłowy `tpm_quote`/PCR → odrzucenie podpisu systemowego, tryb ograniczony.
- **Replay starych podpisów:** ARL/łańcuch hashy + timestampy zapobiegają regresji.
- **Desynchronizacja repozytoriów:** WORM + transakcyjny wskaźnik aktywnego podpisu.

---

7. Telemetria i audyt

- Zdarzenia: `LSIG_REPAIR_START`, `TPM_ATTEST_OK/FAIL`, `KEY_ROTATE`, `LSIG_REPAIR_APPLY`, `ACTIVE_SIG_SWAP_OK`, `LSIG_REPAIR_DONE`.
- Pola: `ts, actor, sr_hash, ps_pub_fp, pt_pub_fp, quote_id, crl_entry, arl_entry, new_sig_hash, prev_chain_hash`.

---

8. Testy akceptacyjne (AT)

- **AT-INTEGRITY:** `SR.hash` z WORM = hash z odczytu.
- **AT-SIG:** nowe podpisy PS/PT przechodzą weryfikację.
- **AT-SWAP:** aktywny wskaźnik podpisu przełączony atomowo.
- **AT-AUDIT:** raport zawiera komplet pól i łańcuch hashy ARL.

---

9. Pseudokod referencyjny

```
function lsig_repair():
  enter_safe_mode()
  if SHA3_512(read_WORM("SR.pkg")) != read_WORM("SR.hash"):
     log("SR_MISMATCH")
     return require_anchor_migration()

  quote = tpm_attest()
  log("TPM_ATTEST", quote)

  SR_hash = read_WORM("SR.hash")
  LSIG_new = Ed25519_Sign(SR_hash, PS_priv)
  PT_new   = option(SignEd25519(SR_hash, PT_priv))
  SIG_new  = build_sig_json(LSIG_new, PT_new, quote)

  write_WORM(LSIG_new, SIG_new)
  atomic_swap_active_sig(LSIG_new)

  assert VerifyEd25519(SR_hash, LSIG_new, PS_pub)
  if require_dual: assert VerifyEd25519(SR_hash, PT_new, PT_pub)

  log("LSIG_REPAIR_DONE")
  exit_safe_mode()
  return OK
```

---

10. Przykładowe komendy (CLI, Linux)

```
# Hash SR
sha3sum SR.pkg > SR.hash

# Podpis systemowy (Ed25519 w TEE/TPM zależnie od integracji)
openssh-keygen -Y sign -f PS_priv -n file SR.hash > LSIG_new.sig

# Podpis twórcy (offline/HSM)
openssh-keygen -Y sign -f PT_priv -n file SR.hash > PT_new.sig

# Weryfikacja
openssh-keygen -Y verify -f PS_pub -n file -s LSIG_new.sig < SR.hash
```

---

11. Uwagi kryptograficzne

- **Algorytmy:** domyślnie Ed25519 + SHA3-512; dopuszczalne alternatywy (ECDSA/P-256) oraz tor **post-quantum** (np. Dilithium) – wtedy wersjonuj `algos` w `META.json`.
- **Rotacja:** rotacja = nowy rekord ARL; **nie** nadpisuj starych artefaktów.
- **FIPS/Compliance:** jeżeli wymagane, użyj bibliotek z certyfikatami (FIPS 140-3).

---

12. Noty końcowe

- „Naprawa LSIG” **nigdy** nie modyfikuje `SR.pkg`.
- Każda próba „naprawy” przy niespójnym SR musi zostać odrzucona i skierowana do **migracji Anchora**.
- Raport z procedury stanowi część ciągłej zgodności z Boskim Ładem (kanonem) systemu.