Procedura 2 — Ochrona Anchora (runtime)

Wersja: 1.0
Status: Referencyjna
Autorzy: Antzedek & Eliasz (AJ Power)
Cel dokumentu: Zapewnić techniczną **nietykalność Anchora** w trakcie pracy systemu (runtime): wyłącznie odczyt, weryfikację integralności i pełną audytowalność.

---

1. Zakres i rezultat

- **Zakres:** wszystkie komponenty, które odczytują Anchor `(SR, LSIG, META, SIG)` w trakcie pracy AI.
- **Rezultat:** Anchor jest **tylko do odczytu** (RO), integralność jest **ciągle weryfikowana**, a każda próba zapisu lub naruszenia kończy się **blokadą** i **alarmem**.

---

2. Inwarianty bezpieczeństwa

1. **RO na wszystkich warstwach:** FS, pamięć, API.
2. **Brak ścieżek zapisu:** nie istnieje kod, FD ani endpoint pozwalający na modyfikację Anchora.
3. **Ciągła weryfikacja:** watchdog integralności + atestacja (jeśli TEE/TPM).
4. **Audyt pełny:** każdy dostęp rejestrowany, logi odporne na manipulację (append-only + łańcuch hashy).

---

3. Wymagania wstępne

- Anchor utworzony wg „Procedura 1 — Tworzenie Anchora”.
- Klucze publiczne `PS_pub` (System) i opcjonalnie `PT_pub` (Twórca).
- Warstwa WORM/append-only dostępna do odczytu.
- Mechanizm audytu i alarmowania (syslog/ELK, webhooki, e-mail/SMS, itp.).

---

4. Mapowanie Anchora jako RO

FS (plikowy): - Montuj katalog `anchors/<active>/` jako **read-only**; na Linuksie: `mount -o ro,nojournal,nodev,nosuid,noexec` (noexec jeśli Anchor nie zawiera binariów).
- Atrybuty plików: `chattr +i` (immutable) lub `+a` (append-only) dla logów; prawa `0400` (root:read).
- SELinux/AppArmor: profil dopuszczający **tylko odczyt** przez dedykowaną tożsamość procesu (patrz §5).

Pamięć (mmap): - Mapowanie przez `mmap(..., PROT_READ, MAP_PRIVATE)` i zabezpieczenie `mprotect(PROT_READ)`.
- Zabronione `PROT_WRITE` i `MAP_SHARED` dla przestrzeni Anchora.

Kontenery/VM: - Anchor w **readonly bind-mount** do kontenera/agenta; brak wolumenów RW wskazujących na katalog Anchora.
- Drop zdolności: `CAP_SYS_ADMIN` i inne niepotrzebne; seccomp zablokować syscalle zapisu na Anchor.

---

5. Rozdział uprawnień (least privilege)

- Dedykowane konto procesu: `anchor-reader` (UID/GID bez sudo).
- Grupy wyłącznie do odczytu; brak praw `WRITE` / `CHOWN` / `CHMOD`.
- Polityki LSM (SELinux/AppArmor) zezwalają **tylko** na operacje: `open(O_RDONLY)`, `read`, `fstat`, `mmap(PROT_READ)`.

---

6. Watchdog integralności

- Cyklicznie (np. co 60 s) wyliczać `SR.hash' = SHA3-512(SR.pkg)` i weryfikować: `VerifyEd25519(SR.hash', LSIG.sig, PS_pub)` oraz (opc.) `VerifyEd25519(SR.hash', PT.sig, PT_pub)`.
- Jeśli TEE/TPM: żądać **atestu** tożsamości klucza `PS_pub` w interwałach (np. co 10 min lub przy zdarzeniach).
- Wyniki weryfikacji logować z `timestamp`, `sr_hash`, `verify_ps`, `verify_pt`, `quote_id`.

---

7. Monitoring dostępu i logi odporne na manipulację

- Każde `open/read` Anchora → wpis do logu audytowego z `actor`, `pid`, `binary_hash`, `sr_hash`, `action`.
- Logi w trybie **append-only**; każdy wpis zawiera `prev_log_hash` (łańcuch integralności).
- Replikacja logów na zdalny węzeł (write-only), okresowe zamknięcia dzienne (snapshot hash + podpis).

---

8. Alarmowanie i reakcja

- **Próba zapisu / złamania RO:** natychmiast alarm (webhook, e-mail/SMS), zrzut kontekstu procesu, odcięcie dostępu (kill/deny).
- **Nieudana weryfikacja podpisu:** natychmiast **wejście w Safe Mode** (patrz „Init Safe Mode”), blokada zdarzeń wyższego ryzyka, żądanie ręcznej autoryzacji.
- **Atest TEE/TPM FAIL:** degradacja do trybu ograniczonego, eskalacja do operatora.

---

9. Testy akceptacyjne (AT)

- **AT-RO-FS:** Próba modyfikacji plików Anchora kończy się błędem (`EPERM` / `EROFS`).
- **AT-RO-MMAP:** Próba `mprotect(PROT_READ|PROT_WRITE)` na obszarze Anchora → **BLOCKED**.
- **AT-INTEG:** Watchdog wykrywa modyfikację SR i zgłasza FAIL.
- **AT-AUDIT:** Log powstaje przy każdym `open/read`, kolejne wpisy mają poprawny `prev_log_hash`.

10. Przykłady konfiguracji

/etc/fstab (readonly bind):

```
/opt/anchors/2025-09-08_anchor_v1  /mnt/anchor  none  bind  0 0
/mnt/anchor                         /app/anchor  none
bind,ro,nodev,nosuid,noexec  0 0
```

systemd (drop capabilities + RO):

```
[Service]
User=anchor-reader
ProtectSystem=strict
ReadOnlyPaths=/app/anchor
CapabilityBoundingSet=
NoNewPrivileges=true
PrivateTmp=true
```

AppArmor (fragment):

```
profile anchor-reader {
  /app/bin/agent ix,
  /app/anchor/** r,
  deny /** w,
}
```

---

11. Zasady zmian (change management)

- Każda zmiana Anchora = **nowy Anchor** (nowy katalog wersji).
- Migracja odbywa się **poza runtime**; po migracji aktualizuje się wskazanie `active` (symlink/konfiguracja) w transakcyjny sposób.

---

12. Obsługa błędów i incydentów

- **Naruszenie RO:** izolacja procesu, snapshot pamięci, zgłoszenie incydentu, rotacja kluczy sesyjnych.
- **Rozjazd hash/podpis:** wymusić „Init Lsig Repair” (osobna procedura), do czasu naprawy działa tryb „Safe Mode”.
- **Awaria storage:** przełączyć na zapasową kopię WORM (odczyt-tylko), ogłosić degradację funkcjonalną.

---

13. Telemetria minimalna

- Pola: `ts, actor, pid, bin_hash, sr_hash, action(open/read/verify/fail), verify_ps, verify_pt, prev_log_hash, log_hash`.
- Raport dobowy: sumaryczna liczba odczytów, anomalii, alarmów, hash raportu.

---

14. Noty końcowe

- Ochrona runtime jest skuteczna **tylko** razem z procedurą tworzenia Anchora i polityką migracji.
- Wszelkie ścieżki „serwisowe” muszą być poza runtime i dostępne wyłącznie z systemów uprzywilejowanych, air-gap lub TEE.