The Blue Book of GNX WNS

설명서 · 매뉴얼 · 참고서

Enterprise Operations Manual and Reference

문서 코드	GNX-BB-WNS-2026-04
문서 등급	Restricted / NDA 권고 / 검증 및 운영 참조용
작성 기준	theLogoofGnxcokr.txt 단일 소스 코드, WNS 공식 특허 문서, 우선심사결정서
배제 기준	기존 백서 및 청서 초안, 정합성 미흡 판정 자료, 검증되지 않은 주장
버전	v1.0 / 2026-04-29
권리자 표기	GNX Co., Ltd. / 주식회사 지엔엑스

본 문서는 라이선스 협상, 보안 검토, 운영 인수인계에 사용할 수 있도록 재구성한 공식 설명서입니다. 법률 의견, 특허 등록 확정 문서, 외부 보안 인증서는 아니며, 계약 전에는 별도 법률 검토와 독립 보안 검증을 병행해야 합니다.

목차

목차는 고정 목차입니다. 최종 계약본에서는 승인 페이지, 서명란, 문서번호 체계를 반영해 페이지 번호형 목차로 재발행할 수 있습니다.

1. 엔진 개요와 사용 원칙

1.1 Logicnoid WNS 의 작동 개념

GNX 엔진은 사용자의 입력을 장기 보관 대상으로 취급하기보다 처리 조건으로 해석하고, 그 조건을 정규화·변성·결합·판정해 실행 가능한 상태 증거로 전환하는 Logicnoid WNS 런타임이다. 일반적인 로그인 사이트나 단순 보안 저장소, 블록체인 원장과 달리 원문을 공개 표면에 남기는 것이 아니라 commit receipt, token fingerprint, resonance proof, gate state, audit signature 를 노출한다.

운영 관점에서 사용자는 세 가지 사실을 이해해야 한다. 첫째, 입력 문자열은 주소나 도메인 조회 키가 아니라 실행 제어 조건이다. 둘째, 인증과 세션은 원문 재노출이 아니라 내부 anchor 와 HMAC, PBKDF2, 쿠키 결합으로 유지된다. 셋째, 조건이 충족되지 않을 때 시스템은 실패한 척 우회하지 않고 locked-by-design 상태로 차단해야 한다.

1.2 사용자가 알아야 할 비주장

운영 매뉴얼은 엔진의 사용법을 설명하지만 절대 보안을 보장하지 않는다. Zero-Lodger 는 로그와 메모리 처리 정책의 이름이지 모든 법적 보존 의무가 사라졌다는 의미가 아니다. WNS 는 문자열 처리와 실행 제어 구조를 설명하지만 특허 등록 확정, 모든 관할권에서의 권리 유효성, 제 3 자 권리 비침해를 자동으로 보장하지 않는다.

또한 브라우저에서 local clear 버튼을 누르는 행위는 화면에 보이는 증거를 지우는 동작일 뿐, 서버 장부, 감사 기록, DB 이력까지 삭제하는 법적 삭제 요청으로 해석해서는 안 된다. 운영자는 사용자가 보는 공개 표면과 내부 운영 기록의 차이를 명확히 고지해야 한다.

운영 개념	설명	금지 해석
Commit receipt	입력 조건이 처리되어 생성된 증거	원문 복원 가능한 저장값
Fingerprint	토큰 또는 조건의 짧은 식별 표면	비밀키 또는 세션 원문
Gate state	실행 허용/차단 상태	서비스 품질 성공 보증
Zero-Lodger	마스킹과 소거 중심의 처리 원칙	모든 법적 기록 삭제
Readiness	운영 전환 가능성 점검	외부 보안 인증서

요약문

본 장은 GNX 엔진을 사용하는 기본 관점을 정리한다. 엔진은 원문 보관이 아니라 상태 증거 생성을 중심으로 작동한다. 다만 운영자는 Zero-Lodger, WNS, readiness, gate state 를 과장하지 말고 실제 기능과 법적 한계를 분리해 설명해야 한다.

2. 내부 처리 파이프라인

2.1 WNS 정규화와 파쇄

WNS 처리의 시작점은 WNSNormalizer.shred 이다. 입력이 비어 있거나 문자열이 아니면 WNSNormalizationFault 가 발생한다. 정상 입력은 NFKD 정규화와 소문자화를 거친 뒤 문자 입자로 분해된다. 한글 완성형 음절은 Unicode 공식 범위에 따라 초성, 중성, 종성으로 분리되고, 영문·숫자·허용 기호는 whitelist 를 통과한 입자로 유지된다.

분해된 입자는 WNSAntiCollisionMatrix 에서 원래 위치 index 와 문자 코드 기반의 entropy suffix 를 받는다. 그 결과 서브 토큰은 "입자#엔트로피" 형태가 되고, 이후 정렬되어 직렬화된다. 위치 정보가 결합되기 때문에 단순 정렬로 인한 anagram 충돌 가능성이 감소하며, 동일 입력은 동일한 파쇄 결과를 만들도록 결정론성을 유지한다. 파쇄 결과는 SecureMemoryManager 가 관리하는 Buffer 로 반환되고, 사용 후 3-pass 방식으로 소거해야 한다.

2.2 MCSS, Phantom Token, Resonance

MCSS 는 ID 와 비밀번호를 각각 WNS 로 파쇄한 뒤 master key 로 HMAC-SHA512 를 수행한 1 차 변성값이다. 코드에서는 ID 와 비밀번호 사이에 GNX_MCSS_CHAIN_LOCK 구분자를 넣어 chain extension 위험을 줄이는 방향으로 구현되어 있다. MCSS 는 원문을 대체하는 계산 재료이며, 사용 후 파쇄 버퍼는 즉시 소거된다.

Phantom Token 은 세션마다 달라지는 동적 변이값이다. high-resolution time, free memory, random bytes, client entropy 가 결합되고 proof key 로 HMAC-SHA384 가 생성된다. Resonance 는 MCSS 와 Phantom Token 을 PBKDF2-HMAC-SHA512 로 결합하는 단계다. 기본 반복 횟수는 150,000 이고 Phantom Token 마지막 바이트를 이용해 추가 가중치를 부여한다. 운영자는 이 연산이 CPU 비용을 가지므로 로그인 폭주, rate limit, worker capacity 를 함께 설계해야 한다.

단계	입력	출력	운영 주의
WNS shred	raw id, raw pwd, target identity	secure buffer 또는 hash 재료	입력 검증 및 소거 필수
MCSS	shredded id, shredded pwd, master key	512-bit hex material	원문 대체값으로 취급
Phantom Token	time, memory, random, client entropy	세션 동적 token	재사용 금지 및 fingerprint 만 노출
ZKV Anchor	MCSS, deterministic salt	DB 저장 anchor	등록/인증 절차 일치 필요
Bident Session	session payload, HMAC signature	HttpOnly cookie	raw token JSON 반환 금지

요약문

본 장은 엔진의 계산 경로를 설명한다. WNS 는 입력을 구조화하고, MCSS 는 원문 자격 증명을 변성하며, Phantom Token 과 PBKDF2 Resonance 는 동적 세션 증거를 만든다. 운영자는 계산 비용, 소거 시점, 원문 비노출 원칙을 동시에 관리해야 한다.

3. API 와 상태 전이

3.1 등록 및 인증

등록 API 는 /api/v1/users/register 이다. 요청 body 에는 id, pwd, advancedIdentity 가 필요하다. 등록 파이프라인은 WNS 파쇄, MCSS 생성, deterministic ZKV anchor 생성, PostgreSQL ACID transaction, 감사 이벤트 기록으로 구성된다. 성공 시 identifier 와 receipt 중심의 표면을 반환하고 plaintext_retained=false, plaintext_label_retained=false 같은 원칙을 명시한다.

인증 API 는 /api/v1/auth/resonance 이다. 요청 body 에는 id, pwd, clientEntropy 가 포함될 수 있다. 엔진은 Phantom Token 을 생성하고 등록과 동일한 방식으로 ZKV anchor 를 재계산한 뒤, DB 에서 가져온 anchor 와 constant-time 으로 비교한다. 성공하면 __Host-gnx_bident 쿠키를 HttpOnly, Secure, SameSite=strict, path=/, 1 시간 만료로 발급한다. JSON 응답에는 raw token 이 아니라 receipt, session ledger, token fingerprint, mutation invariants 가 포함된다.

3.2 세션, 디스플레이 락, 터널

세션 기반 실행은 SecurityMatrices.requireBidentSession 을 통과해야 한다. 이 middleware 는 Authorization Bearer 또는 __Host-gnx_bident 쿠키에서 토큰을 추출하고 tunnelingGateway 의 검증을 거친다. 토큰이 없거나 무결성이 맞지 않으면 401 로 종료된다. 따라서 운영자는 프록시, TLS termination, cookie domain/path, SameSite 정책이 이 전제와 충돌하지 않도록 구성해야 한다.

디스플레이 락 보고 API 는 /api/v1/display/lock 이다. 요청에는 targetIdentity, clientEntropy, nonce, hardwareSignature 가 필요하다. HardwareAttestationManager 는 nonce timestamp, Redis replay window, HMAC signature 를 검증한다. 세션 터

널 개방 API 는 /api/v1/session/establish 이며, targetIdentity 에 대한 display ready 신호가 제한 시간 안에 도착해야 tunnel ticket 과 signaling endpoint 를 발행한다. 실패 시 정상적인 실패가 아니라 의도된 차단 상태인 INTERDEPENDENT_GATE_LOCKED_BY_DESIGN 으로 취급한다.

Route	목적	보호 장치	대표 응답
/api/v1/users/register	WNS/ZKV 등록	ACID transaction, anchor, audit	IDENTITY_MAPPING_COMPLETE 또는 commit receipt
/api/v1/auth/resonance	인증과 세션 결합	ZKV, timingSafeEqual, HttpOnly cookie	BIDENT_RESONANCE_BOUND
/api/v1/display/lock	단말 디스플레이 준비 보고	nonce, Redis replay, HMAC attestation	AIPHONE_LOCK_CONFIRMED
/api/v1/session/establish	상호 의존 터널 개방	Bident session, display lock wait	INTERDEPENDENT_LOCK_SECURED
/api/v1/tunnel/:tunnelId	터널 상태 조회	Bident session	tunnel record 또는 404
/health/live, /health/ready	생존과 준비성 점검	Postgres, Redis, KMS, migration	ok=true/false

요약문

본 장은 사용자와 운영자가 호출하는 주요 API 를 정리한다. 등록과 인증은 원문이 아니라 anchor 와 receipt 중심으로 처리되고, 세션 이후의 실행은 Bident session 과 display lock 을 모두 통과해야 한다. locked-by-design 응답은 장애가 아니라 통제 정책의 정상 작동으로 해석해야 한다.

4. 운영 환경과 배포

4.1 필수 구성 요소

엔진은 Node.js Express 애플리케이션이며 PostgreSQL, Redis, SMTP, 환경변수 기반 secret, 선택적 KMS round-trip 에 의존한다. 운영 환경에서는 GNX_PG_URL, GNX_REDIS_URL, GNX_MASTER_KEY_B64 또는 GNX_MASTER_SECRET 계열, GNX_PROOF_HMAC_KEY_B64 또는 GNX_ATTESTATION_SECRET 계열, GNX_COOKIE_SIGN_KEY_B64 또는 GNX_COOKIE_SECRET 계열, GNX_AUDIT_HMAC_KEY_B64, GNX_SMTP_HOST, GNX_SMTP_USER, GNX_HTTP_ADDR, GNX_ENVELOPE_KEK_HEX 같은 값을 관리해야 한다.

키 값은 코드 저장소나 이미지에 포함해서는 안 되며 secret manager, KMS, CI/CD protected variable, instance role 기반 주입 방식으로 관리하는 것이 바람직하다. Redis 는 일반 cache 가 아니라 nonce replay window, rate limit, Pub/Sub signal, tunnel state TTL 의 핵심 의존성이다. PostgreSQL 은 anchor, sessions, audit, security events, interlocks, tunnels, immune events 등 장기 증적을 관리한다.

4.2 기동 및 준비성 점검

기동 시 DatabaseIntegrityGuard.ensureSchema 는 GNX_SCHEMA_PACK 에 포함된 DDL 을 트랜잭션으로 적용하고 migration checksum 을 기록한다. 이후 Redis manager, event bus, anomaly matrix, health matrix, route binding 이 완료되어야 HTTP server 가 listen 한다. 운영자는 기동 로그에서 schema secured, Redis active, Lua script loaded, readiness gate 통과 여부를 확인해야 한다.

/health/live 는 프로세스 생존만 확인하고 /health/ready 는 PostgreSQL, Redis, KMS, migration table 을 확인한다. load balancer 와 deployment controller 는 ready 를 기준으로 트래픽 전환을 결정해야 한다.

운영 구성	필수성	확인 방법
PostgreSQL	ZKV anchor, sessions, audit, events, schema	/health/ready postgres probe
Redis	replay defense, Pub/Sub, rate limit, TTL state	/health/ready redis probe

KMS/KEK	wrapped secret round-trip, readiness proof	GNX_ENVELOPE_KEK_HEX 및 kms probe
SMTP	Zero-Lodger mail spooler	SMTP host/user/pass/from 확인
TLS/Proxy	Secure cookie, Host cookie, no-store	reverse proxy 와 cookie policy 점검
SIEM Sink	JSON log, audit event ingestion	stdout/stderr collector 와 redaction sample 확인

요약문

본 장은 엔진 배포에 필요한 환경을 정리한다. GNX 는 단일 Node.js 코드로 보이지만 실제 운영 안정성은 PostgreSQL, Redis, KMS, SMTP, TLS, SIEM 의 조합에 달려 있다. 트래픽 전환은 반드시 /health/ready 기준으로 결정해야 한다.

5. 관측, 감사, 장애 대응

5.1 로그와 감사 이벤트

ZeroLodgerLogger 는 JSON 구조로 로그를 출력하며 severity, host, pid, message, context 를 포함한다. phone, email, IP, JWT, uuid 등은 마스킹 대상이고 password, secret, token, msisdn, phone, id, auth, key, hash 같은 key 이름은 즉시 차단 대상으로 처리된다. 감사 이벤트는 action, subject hash, timestamp, outcome 을 결합해 HMAC signature 를 생성하고 AUDIT severity 로 배출한다.

운영자는 로그가 민감 정보를 완전히 제거한다고 가정하지 말고 정기적으로 샘플 검사를 수행해야 한다. 특히 신규 라우트, 신규 context 객체, 예외 stack, SMTP task, 외부 라이브러리 오류는 마스킹 규칙 밖의 데이터를 남길 수 있다. SIEM 연동 시 원문 payload 를 보관하지 않는 ingest pipeline 을 구성하고, 보존 기간과 접근 권한을 별도로 제한한다.

5.2 장애·보안 이벤트 운영

장애 유형은 기동 실패, secret 누락, PostgreSQL 오류, Redis 오류, KMS round-trip 실패, rate limit 과탐, attestation replay, display lock timeout, tunnel ticket 검증 실패, schema migration 실패로 나눌 수 있다. 원칙은 fail-closed 이다. 인증·세션·터널·readiness 에서 불확실성이 발생하면 허용하지 않고 차단해야 한다.

보안 이벤트가 발생하면 먼저 사용자에게 상세 내부 사유를 반환하지 않는다. 클라이언트에는 401, 403, locked-by-design, generic error 를 반환하고, 상세 원인은 sanitized log 와 audit signature 로 내부 조사한다. 운영자는 incident ID, 시간, affected route, subject hash, Redis/DB 상태, key version, deployment version, 조치 내역을 기록해야 한다. 중대한 secret 노출 가능성이 있으면 키 회전, 세션 폐기, Redis nonce namespace 초기화, DB 접근 로그 검토를 즉시 수행한다.

이벤트	운영 조치	고객 표면
AUTH 실패 증가	ZKV timing, credential stuffing, rate limit 확인	RESONANCE_FAILED 또는 401
Display lock timeout	단말 attestation, Pub/Sub, clock skew 확인	LOCKED_BY_DESIGN
Redis 장애	replay/rate/tunnel 의존성 점검, failover	ready=false 또는 500/403
PostgreSQL 장애	anchor/session/audit 저장 경로 점검	ready=false 및 인증 중단
KMS invalid	KEK 형식, 권한, 회전 상태 점검	ready=false
로그 마스킹 의심	샘플 수집 중단, SIEM 접근 제한, rule 보정	고객에게 내부 상세 미노출

요약문

본 장은 관측과 사고 대응 방식을 설명한다. 로그는 JSON 과 감사 서명을 중심으로 남기되 민감 정보가 남지 않는지 지속 검증해야 한다. 장애 시에는 원인을 숨기는 것이 아니라 고객 표면과 내부 조사 표면을 분리하고, 불확실하면 허용하지 않는 fail-closed 원칙을 유지한다.

6. 관리자 참고서

6.1 데이터베이스 객체

GNX_SCHEMA_PACK 은 운영 DB 에 필요한 표준 객체를 정의한다. gnx_schema_migrations 는 적용 이력과 checksum 을 보관한다. gnx_users_zkv 는 user_id_hash, zkv_anchor, advanced_identity, entropy_salt_hash, is_active 를 저장한다. gnx_audit_log 와 gnx_security_events 는 감사와 보안 이벤트를 보관하고, gnx_sessions 는 session_digest 와 만료 상태를 관리한다.

AIPhone 및 터널 계층에서는 gnx_display_interlocks 와 gnx_tunnels 가 target identity hash, challenge nonce hash, attestation hash, tunnel ticket hash, state, timestamp 를 기록한다. 추가로 resistance vaults, somatic cells, synapses, immune events 계열은 확장된 증거·변성·위험·면역 이벤트 저장소로 설계되어 있다. 운영자는 각 테이블의 원문 저장 여부, retention, 백업 암호화, 접근권한을 검토해야 한다.

6.2 운영 규칙과 금지 사항

운영 규칙은 단순하다. 첫째, 원문 ID, 비밀번호, token, secret 을 로그·응답·스크린샷·이메일에 남기지 않는다. 둘째, ready=false 상태에서 트래픽을 받지 않는다. 셋째, Redis 를 단순 캐시처럼 무단 flush 하지 않는다. 넷째, KMS/KEK 와 master secret 을 동일한 권한 경계에 두지 않는다. 다섯째, display lock 실패를 우회하는 임시 백도어 route 를 만들지 않는다.

관리자가 해서는 안 되는 일도 명확하다. DB 에서 zkv_anchor 를 복사해 외부 채팅이나 문서에 붙여넣지 않는다. 장애 분석을 위해 req.body 전체를 dump 하지 않는다. 보안 검증 편의를 이유로 Secure cookie 나 HttpOnly 를 끄지 않는다. rate limit 이 불편하다는 이유로 AnomalyDefenseMatrix 를 장기간 비활성화하지 않는다. 폐기된 백서나 청서의 문구를 새 계약서에 되살리지 않는다.

구분	권장	금지
비밀값	Secret manager/KMS 로 주입	소스, 이미지, 문서에 하드코딩
로그	sanitized JSON 과 audit signature	req.body dump, raw token 기록
쿠키	HttpOnly, Secure, SameSite=strict	디버깅을 위한 보안 속성 해제
DB	least privilege, encrypted backup	anchor 외부 반출
Redis	namespace 관리, failover, TTL 확인	무단 flush, replay window 무력화
문서	본 청서와 백서 기준으로 갱신	폐기 문서 재사용

요약문

본 장은 관리자가 실제 운영 중 지켜야 할 규칙을 제시한다. GNX 엔진의 보안성은 코드만으로 결정되지 않고 운영자의 비밀값 관리, 로그 습관, readiness 준수, Redis/DB 접근 통제, 문서 통제에 의해 유지된다.

부록. 근거 자료와 사용 제한

본 문서의 기술 서술은 업로드된 진성 엔진 소스 코드(theLogoofGnxcokr.txt)와 WNS 특허 명세서(Kipo WNS(1).pdf), WNS 우선심사결정서(WNS_우선심사결정서.pdf)를 기준으로 새로 작성하였다. 사용자가 폐기 지시한 기존 백서 및 청서 초안은 구조, 문장, 논리, 표현, 주장 모두에서 참고하지 않았다.

우선심사결정서는 해당 출원이 우선심사 대상으로 인정되었다는 절차적 사실을 보여주며, 특허 등록 또는 침해 비침해 판단을 확정하지 않는다. 라이선스 계약 문서에는 등록 여부, 권리범위, 실시권 범위, 소스 공개 범위, 검증 환경의 책임 분장을 별도 조항으로 명시해야 한다.

요약문

본 부록은 문서의 근거와 한계를 명시한다. 본 문서는 공식 검토용 설명서이지만 외부 인증서, 법률 의견서, 특허 등록 확정 통지서를 대체하지 않는다.