ADR-014 — Hybrid Encryption for L4 Restricted Fields

ADR-014 — Hybrid Encryption for L4 Restricted Fields (PIB, JMBG, OIB, JIB, IBAN)

> **ADR Number:** ADR-014 **Title:** Use AES-256-GCM field-level encryption for JMBG and OIB; rely on disk-level encryption + application-layer access controls for PIB, JIB, and IBAN **Date:** 2026-02-25 **Author:** Petter Graff (Architect) **Status:** Proposed **Resolves:** Conflict between SECURITY-ARCHITECTURE.md (2026-02-20, "Column-level encryption: Not needed") and DATA-ENCRYPTION-POLICY.md / COMPLIANCE-FRAMEWORK.md (2026-02-23, "AES-256-GCM field-level encryption MANDATORY for all L4 fields")

---

1. Context

1.1 Situation

Bilko is a cloud accounting SaaS targeting 50K-500K SMBs across Serbia, Bosnia & Herzegovina, and Croatia. The database stores several categories of personal and financial identifiers classified as L4 Restricted:

Field	Identifier Type	Jurisdiction	Nature
JMBG (Jedinstveni maticni broj gradana)	Unique citizen number	RS, BA	**Personal** -- encodes date of birth, gender, region of birth. Equivalent to SSN. Assigned to natural persons only.

Field	Identifier Type	Jurisdiction	Nature
OIB (Osobni identifikacijski broj)	Personal identification number	HR	**Personal** -- assigned to every natural and legal person in Croatia. 11-digit number used across all government and financial interactions.
PIB (Poreski identifikacioni broj)	Tax identification number	RS	**Business** -- assigned to legal entities and entrepreneurs for tax purposes. Publicly visible on invoices, SEF portal, APR registry.
JIB (Jedinstveni identifikacioni broj)	Unique identification number	BA	**Business** -- assigned to legal entities for tax purposes. Publicly visible on UIO portal, court registry.
IBAN	International Bank Account Number	All	**Financial** -- bank account identifier. Semi-public (printed on invoices, shared for payment).

1.2 The Conflict

Two Bilko security documents contradict each other:

SECURITY-ARCHITECTURE.md (2026-02-20):

> "Column-level encryption: Not needed (disk encryption sufficient for accounting data)"

DATA-ENCRYPTION-POLICY.md and COMPLIANCE-FRAMEWORK.md (2026-02-23):

> "AES-256-GCM field-level encryption MANDATORY for all L4 fields (PIB, JMBG, OIB, JIB, IBAN)"

The earlier document (Security Architecture) was written from a pragmatic engineering perspective. The later documents (Encryption Policy, Compliance Framework) were written from a compliance/DPIA perspective and prescribed a blanket field-level encryption mandate for all L4 fields without differentiating between personal identifiers and business/financial identifiers.

1.3 Regulatory Analysis

GDPR Article 32 -- Security of Processing (Croatia)

GDPR Article 32 requires "appropriate technical and organisational measures to ensure a level of security appropriate to the risk," taking into account:

1. The state of the art
2. The cost of implementation
3. The nature, scope, context, and purposes of processing
4. The risk of varying likelihood and severity for the rights and freedoms of natural persons

Encryption is listed as one example ("inter alia as appropriate: the pseudonymisation and encryption of personal data") but is **not a blanket requirement**. The determination is risk-based.

GDPR Article 87 -- National Identification Numbers

Article 87 permits Member States to establish specific conditions for processing national identification numbers. It requires "appropriate safeguards" but does **not mandate encryption specifically**. Croatia (through AZOP) has not published binding guidance requiring field-level encryption of OIB in databases, though OIB processing is subject to purpose limitation and data minimization principles.

Serbia -- ZZPL (Sl. glasnik RS 87/2018)

Serbia's ZZPL is closely aligned with GDPR. Article 50 (equivalent to GDPR Art. 32) requires "appropriate technical, organizational and personnel measures." The Serbian Commissioner (Poverenik) has not published specific guidance mandating field-level encryption for JMBG or PIB. However, JMBG is widely recognized as highly sensitive -- it encodes biographic information (date of birth, gender, region) and its misuse enables identity fraud. The JMBG's sensitivity is acknowledged in Serbian legal practice and the Commissioner's public statements.

Bosnia & Herzegovina -- ZZLP BiH (Sl. glasnik BiH 49/2006, updated 2025)

The new ZZLP BiH (published February 28, 2025, entering force March 8, 2025) is harmonized with GDPR. AZLP BiH guidance on technical measures mentions "encryption of data" and "pseudonymization" as recommended measures. No specific mandate for field-level database encryption exists, but the general requirement for "appropriate security measures" applies.

Summary: No Law Mandates Field-Level Encryption

No regulation in any of the three jurisdictions explicitly mandates application-level field-level encryption for tax IDs or IBANs in databases. All three frameworks use GDPR-style language: "appropriate technical measures proportionate to the risk." This means the decision is a **risk-based engineering judgment**, not a binary legal requirement.

1.4 What Competitors Do

Competitor	Approach	Evidence
FreeAgent (UK accounting SaaS, 150K+ users)	AES-256 encryption at rest for all stored data. No public mention of field-level encryption for specific columns. Cyber Essentials Plus certified. AWS Ireland hosting with ISO 27001/27017/27018 datacenter compliance.	FreeAgent Security
Fiken (Norwegian accounting SaaS, 70K+ users)	No publicly available security architecture documentation. Cloud-based, Norwegian-regulated. No evidence of field-level encryption.	General inference from public documentation.
Minimax (Balkan/SEE accounting SaaS)	No publicly available security architecture documentation. Cloud-based. Standard privacy policy. No evidence of field-level encryption for tax identifiers.	Minimax Privacy

Industry pattern: Accounting SaaS competitors rely on disk-level encryption at rest (AES-256, provided by cloud hosting) combined with TLS in transit, access controls, and audit logging. None publicly document field-level encryption for tax IDs or IBANs.

1.5 Risk Assessment by Field

Field	Sensitivity if Breached	Public Availability	Risk Level
JMBG	**CRITICAL** -- enables identity fraud, encodes DOB/gender/region, irrevocable (cannot be changed)	NOT public -- protected by law, not printed on invoices	**Highest**
OIB	**HIGH** -- unique cross-system identifier, used for all gov/financial interactions, difficult to change	Semi-public -- used widely but not freely published	**High**
PIB	**MEDIUM** -- business tax ID, publicly searchable on APR (Serbian Business Registry) and SEF portal	**PUBLIC** -- printed on every invoice, searchable on gov portals	**Medium**
JIB	**MEDIUM** -- business tax ID, publicly searchable on BiH court registry and UIO portal	**PUBLIC** -- printed on every invoice, searchable on gov portals	**Medium**

Field	Sensitivity if Breached	Public Availability	Risk Level
IBAN	**MEDIUM** -- bank account number, shared routinely for payment purposes, printed on invoices	**Semi-public** -- shared with every business partner for payment	**Medium**

1.6 Technical Constraints

Prisma field-level encryption trade-offs:

Factor	Impact
Query limitations	Encrypted fields cannot be filtered, sorted, or searched with SQL operators. Only exact-match via HMAC hash column is possible.
Storage overhead	AES-256-GCM ciphertext is significantly larger than plaintext (IV + auth tag + ciphertext, base64-encoded). VARCHAR columns must be oversized.
Performance	Encrypt/decrypt on every read/write. No published benchmarks for `prisma-field-encryption`, but crypto operations add measurable latency per record. For batch operations (e.g., 1000-invoice report filtering by buyer PIB), overhead compounds.
Key management	Single `FIELD_ENCRYPTION_KEY` in Railway env var. Key rotation requires re-encrypting all rows -- a migration-level operation.
Prisma compatibility	`prisma-field-encryption` library uses Prisma client extensions (AES-256-GCM). Works but adds a dependency. Raw SQL queries (used for financial reports per ADR-011) bypass encryption middleware.
Development complexity	Developers must handle encrypted fields differently. Debugging queries on encrypted columns is harder. Test fixtures need encryption-aware setup.

---

2. Decision

We will use a hybrid approach:

Tier 1: Field-Level Encryption (AES-256-GCM) -- JMBG and OIB only

These are personal identification numbers with high breach impact and no legitimate reason for database-level querying by value:

- **JMBG** (Serbia/BiH citizen number): Encrypted at application layer before storage. HMAC-SHA256 hash stored in `jmbgHash` column for exact-match lookup when required.
- **OIB** (Croatia personal/company ID): Encrypted at application layer before storage. HMAC-SHA256 hash stored in `oibHash` column for exact-match lookup.

Implementation: Use `prisma-field-encryption` Prisma client extension with `/// @encryption:encrypt` annotation on JMBG and OIB fields in schema.prisma. Add `/// @encryption:hash(jmbg)` for searchable hash columns.

Rationale: JMBG and OIB are irrevocable personal identifiers. A database breach exposing plaintext JMBG enables identity fraud with no mitigation path for the victim (JMBG cannot be changed). The risk justifies the query limitations and performance overhead because:

- These fields are rarely queried in bulk (lookup is by known value, not range/partial match)
- These fields are displayed to users infrequently (only on contact detail views, not list views)
- The DPIA (section 3) already documents AES-256-GCM for these fields -- implementing it fulfills the documented risk mitigation

Tier 2: Disk-Level Encryption + Application Controls -- PIB, JIB, IBAN

These are business identifiers or semi-public financial data where the risk profile does not justify the significant query/performance trade-offs of field-level encryption:

- **PIB** (Serbia tax ID): Publicly searchable on APR. Printed on every invoice. Encrypted at disk level (Railway AES-256). Protected by org-scoping middleware, RBAC, audit trail, and TLS in transit.
- **JIB** (BiH tax ID): Publicly searchable on court registry. Printed on every invoice. Same controls as PIB.
- **IBAN**: Shared with every business partner for payment. Printed on invoices. Masked in list responses (show last 4 digits only). Same disk-level + application-layer controls.

Rationale: Encrypting PIB at the field level when it is publicly available on Serbian government portals (APR, efaktura.mfin.gov.rs) provides negligible additional security benefit while significantly degrading query performance. The same applies to JIB. IBAN is routinely shared for payment and is masked in API list responses. For all three, the defense-in-depth stack (disk encryption + TLS + org-scoping + RBAC + audit trail + API masking) provides security proportionate to the risk, consistent with GDPR Article 32's risk-based approach.

Implementation Summary

Field	Encryption Level	Search Support	Display Masking
JMBG	AES-256-GCM field-level	HMAC-SHA256 hash for exact match	Show only last 3 digits in UI
OIB	AES-256-GCM field-level	HMAC-SHA256 hash for exact match	Show only last 3 digits in UI
PIB	Disk-level (Railway AES-256)	Full SQL query support	Full value shown (public data)
JIB	Disk-level (Railway AES-256)	Full SQL query support	Full value shown (public data)
IBAN	Disk-level (Railway AES-256)	Full SQL query support	Masked in list views (last 4 only), full in detail view

Key Management

- `FIELD_ENCRYPTION_KEY`: 32-byte hex string stored in Railway secrets. Used for JMBG and OIB encryption only.
- `FIELD_HMAC_KEY`: Separate 32-byte hex string stored in Railway secrets. Used for deterministic hash columns.
- Key rotation: Annual. Rotation requires a migration script to re-encrypt all JMBG/OIB rows -- scoped to Contact table only (manageable volume at MVP scale).
- Per Key Management Policy (POL-SEC-KM-001), keys are never committed to source code and are provisioned only through Railway environment secrets or Vaultwarden.

---

3. Alternatives Considered

Option A: No field-level encryption for any L4 field (original Security Architecture position)

Pros:

- Zero query limitations -- PIB, JMBG, OIB, JIB, IBAN all fully searchable/sortable
- Zero performance overhead from application-layer crypto
- Simpler development -- no encryption middleware, no hash columns, no key management
- Consistent with what competitors do (FreeAgent, Fiken, Minimax)
- Disk-level encryption (Railway AES-256) protects against physical theft and disk-level breaches

Cons:

- A database breach (e.g., SQL injection bypassing Prisma, Railway compromise, backup leak) exposes all JMBG/OIB in plaintext
- JMBG exposure is irrevocable -- victims cannot change their JMBG
- Contradicts the DPIA (section 3) which documents field-level encryption as a committed mitigation
- Weaker posture for regulatory audits and GDPR accountability demonstrations
- Negligent if a breach occurs and DPA asks why field-level encryption was not implemented despite being documented in the DPIA

Why not chosen: The risk of JMBG/OIB exposure in a breach is too high to leave unmitigated. Removing a documented DPIA control without justification creates regulatory liability.

Option B: AES-256-GCM field-level encryption for ALL L4 fields (Encryption Policy position) -- Selected with modification

Pros:

- Maximum defense-in-depth -- all L4 data encrypted at field level
- Simplest compliance narrative ("all sensitive data is encrypted at rest at the field level")
- Matches the DPIA and Encryption Policy as written

Cons:

- PIB and JIB are publicly available data -- encrypting them provides marginal security benefit
- IBAN is shared routinely -- encrypting it adds complexity for minimal breach-impact reduction
- Encrypting PIB/JIB breaks SQL `WHERE` clause filtering for invoice lookups by buyer tax ID -- a common query pattern for SEF e-invoice reconciliation and accounting workflows
- Encrypting IBAN breaks bank reconciliation queries
- Performance overhead on every invoice read/write for fields that are public or semi-public
- Over-engineering -- GDPR Art. 32 requires measures "appropriate to the risk," not maximum possible measures
- No competitor in the accounting SaaS space implements field-level encryption for business tax IDs

Why not chosen in full: Encrypting publicly available identifiers (PIB, JIB) at the field level fails the GDPR Art. 32 proportionality test. The cost (query limitations, performance, complexity) is not justified by the risk reduction (minimal, since the data is publicly accessible). However, the principle of encrypting personal identifiers (JMBG, OIB) is correct and adopted.

Option C: Hybrid approach -- Encrypt JMBG and OIB only (Selected)

Pros:

- Proportionate to risk: highest-sensitivity fields get strongest protection
- Preserves full query capability for business tax IDs (PIB, JIB) and IBAN -- critical for accounting workflows
- Manageable scope: only Contact table fields encrypted, not Organization or BankAccount
- HMAC hash columns enable exact-match lookup for the encrypted fields
- Consistent with GDPR Art. 32 risk-based approach
- Defensible in regulatory audit: "we assessed risk per field and applied measures proportionate to each"
- Limited performance impact: JMBG/OIB are read infrequently (contact detail views only)

Cons:

- More nuanced compliance narrative than "everything is encrypted"
- Two tiers of L4 data treatment requires clear documentation
- IBAN remains in plaintext at database level (mitigated by disk encryption + masking)

Risk accepted: A database breach would expose PIB, JIB, and IBAN in plaintext within the encrypted disk. This is accepted because: (a) PIB and JIB are publicly available, (b) IBAN is routinely shared and masked in API responses, (c) disk-level encryption protects against physical/infrastructure breaches, (d) org-scoping prevents cross-tenant access at application level.

Option D: PostgreSQL pgcrypto column-level encryption (database-side)

Pros:

- Encryption happens at the database layer -- transparent to application code
- Works with raw SQL queries (unlike Prisma middleware)

Cons:

- Requires `pgp_sym_encrypt()` / `pgp_sym_decrypt()` in every query -- significantly increases SQL complexity
- Key must be passed in every query or stored in a PostgreSQL session variable -- key management is harder
- Railway managed PostgreSQL may not support pgcrypto extension
- Prisma has no native support for pgcrypto -- requires `$queryRaw` for all encrypted field operations, negating the benefits of the Prisma ORM (ADR-011)
- Breaks Prisma schema-as-code principle

Why not chosen: Incompatible with Prisma ORM and Railway managed PostgreSQL constraints. Application-layer encryption via `prisma-field-encryption` is a better fit for the existing stack.

---

4. Consequences

4.1 Positive Consequences

- JMBG and OIB are protected against database breaches -- even full DB dump yields only ciphertext
- HMAC hash columns enable exact-match lookup without exposing plaintext
- PIB, JIB, IBAN remain fully queryable -- no impact on accounting workflows, SEF reconciliation, or bank reconciliation
- Risk-proportionate approach is defensible under GDPR Art. 32 and ZZPL Art. 50
- Aligns with DPIA documented mitigations (AES-256-GCM for personal identifiers)
- Key management scope is limited (one encryption key, one HMAC key, scoped to two field types)

4.2 Negative Consequences

- JMBG and OIB fields cannot be filtered, sorted, or partially matched in SQL -- only exact-match via hash -- *Mitigation: These fields are rarely queried in bulk; exact-match via HMAC covers the primary use case (lookup by known value)*
- `prisma-field-encryption` library is a third-party dependency (47ng/prisma-field-encryption, MIT license) -- *Mitigation: Library is well-maintained, 600+ GitHub stars, uses Node.js native `crypto` module internally; can be replaced with custom Prisma extension if abandoned*
- Key rotation requires re-encryption migration -- *Mitigation: Scoped to Contact table rows only; at MVP scale (<10K contacts), migration completes in seconds*
- Raw SQL queries (used for financial reports per ADR-011) bypass Prisma encryption middleware -- *Mitigation: JMBG/OIB are never included in financial report queries; they are contact metadata, not transaction data*

4.3 Technical Debt Created

- DATA-ENCRYPTION-POLICY.md and COMPLIANCE-FRAMEWORK.md must be updated to reflect the tiered L4 approach (JMBG/OIB = field-level, PIB/JIB/IBAN = disk-level + controls) instead of blanket "all L4 = field-level" -- *Plan: Update immediately after ADR approval*
- SECURITY-ARCHITECTURE.md line "Column-level encryption: Not needed" must be updated to reflect the hybrid decision -- *Plan: Update immediately after ADR approval*
- DPIA data inventory table should differentiate encryption tier per field -- *Plan: Update in next DPIA revision*

4.4 Schema Changes Required

```
model Contact {
  // ... existing fields ...

  // Tier 1: Field-level encrypted (AES-256-GCM)
  jmbg       String?  @db.Text  /// @encryption:encrypt  -- Serbian/BiH citizen number
  jmbgHash   String?  @map("jmbg_hash") @db.VarChar(64)  /// @encryption:hash(jmbg)
  oib        String?  @db.Text  /// @encryption:encrypt  -- Croatian personal ID
  oibHash    String?  @map("oib_hash") @db.VarChar(64)   /// @encryption:hash(oib)

  // Tier 2: Disk-level encryption only (full query support)
  // registrationNumber (PIB/JIB) -- already exists, no change
  // vatNumber -- already exists, no change
}

model BankAccount {
  // iban field -- already exists as @db.VarChar(50), no encryption change
  // Protected by: disk encryption + org-scoping + RBAC + API masking
}
```

4.5 Environment Variables Required

```
# Generate with: openssl rand -hex 32
FIELD_ENCRYPTION_KEY=<64-char-hex-string>  # AES-256 key for JMBG/OIB encryption
FIELD_HMAC_KEY=<64-char-hex-string>         # HMAC-SHA256 key for hash columns
```

---

5. Decision Rationale Summary

The core insight driving this decision is that **not all L4 data carries equal breach risk**. The original Security Architecture was too permissive (no field-level encryption at all). The Encryption Policy was too aggressive (field-level encryption for publicly available business IDs). This ADR resolves the conflict with a risk-proportionate hybrid:

1. **JMBG/OIB**: Irrevocable personal identifiers with high breach impact. Field-level encryption is justified and proportionate.
2. **PIB/JIB**: Publicly available business identifiers. Field-level encryption adds cost without meaningful risk reduction.

3. **IBAN**: Routinely shared financial identifier. API masking + disk encryption is proportionate.

This approach satisfies GDPR Article 32's "appropriate to the risk" standard, avoids the EDPB-documented pitfall of "security theatre" (implementing controls that do not meaningfully reduce risk), and preserves the query performance essential for accounting workflows.

References

- GDPR Article 32 -- Security of Processing
- GDPR Article 87 -- Processing of National Identification Numbers
- Serbia ZZPL (Sl. glasnik RS 87/2018)
- BiH ZZLP -- AZLP Security Measures Guidance
- Croatia AZOP -- Data Protection Overview
- prisma-field-encryption Library
- FreeAgent Security
- ICO Encryption Guidance
- JMBG Structure and Sensitivity
- Bilko DPIA (docs/security/DPIA.md)
- Bilko Data Encryption Policy (docs/security/DATA-ENCRYPTION-POLICY.md)
- Bilko Compliance Framework (docs/security/COMPLIANCE-FRAMEWORK.md)
- Bilko Key Management Policy (docs/security/KEY-MANAGEMENT-POLICY.md)

Approval

Role	Name	Date	Signature
Author	Petter Graff	2026-02-25	
Tech Lead			
DPO			
CTO / Architect	Alem		

Revision #3
Created 2026-02-24 23:51:16 UTC by John
Updated 2026-05-31 20:04:25 UTC by John