Contents

Compliance & Security Standards

Overview

RADIANT implements comprehensive compliance frameworks to meet enterprise security requirements across multiple regulatory standards.

Required Provider API Keys

RADIANT requires the following external AI provider API keys for deployment:

Provider	Secret Path	Purpose	Get Key
Anthropic (Claude)	`radiant/providers/anthropic`	Primary AI provider for Claude models	https://console.anthropic.com/sett

Provider	Secret Path	Purpose	Get Key
Groq	`radiant/providers/groq`	Fast LPU inference for fallback	https://console.groq.com/keys

Deployment Flow

1. **Configure Keys in Deployer** - Enter API keys in the Swift Deployer "Required Provider API Keys" section
2. **Local Storage** - Keys are stored securely in macOS Keychain
3. **AWS Upload** - During deployment, keys are uploaded to AWS Secrets Manager
4. **Lambda Access** - Lambda functions retrieve keys from Secrets Manager at runtime

Why These Providers Are Required

- **Anthropic (Claude)**: Primary provider for Claude 3.5 Sonnet, Claude Opus 4, and other Claude models via AWS Bedrock. Direct API access provides extended thinking and latest model features.
- **Groq**: Ultra-fast inference (100-200ms) on Llama and Mixtral models. Used as fallback when Bedrock is unavailable and for speed-critical applications.

SOC 2 Type II Compliance

Trust Service Criteria

Category	Controls	Implementation
Security	Access control, encryption, monitoring	Cognito, KMS, CloudWatch
Availability	Redundancy, failover, SLAs	Multi-AZ, auto-scaling
Processing Integrity	Data validation, error handling	Input validation, checksums
Confidentiality	Data classification, encryption	RLS, AES-256, TLS 1.3
Privacy	Data handling, consent	GDPR controls, retention policies

Key Controls

1. **Access Management**
    - Multi-factor authentication (MFA) required for admins
    - Role-based access control (RBAC)
    - API key rotation policies
    - Session timeout enforcement
2. **Encryption**
    - At rest: AES-256 via AWS KMS
    - In transit: TLS 1.3 minimum

    - Database: Aurora encryption enabled
    - Secrets: AWS Secrets Manager
3. **Audit Logging**
    - All API requests logged
    - Admin actions tracked
    - CloudTrail for AWS operations
    - 90-day retention minimum

HIPAA Compliance

Protected Health Information (PHI) Handling

RADIANT supports HIPAA-compliant deployments with enhanced controls:

Requirement	Implementation
Access Controls	User authentication, authorization, audit
Audit Controls	Complete activity logging, tamper-evident
Integrity Controls	Data validation, checksums, versioning
Transmission Security	TLS 1.3, encrypted channels only

HIPAA Mode Features

When HIPAA mode is enabled:

```
interface HIPAAConfig {
  enabled: boolean;
  phiDetection: boolean;         // Scan for PHI in requests
  enhancedLogging: boolean;      // Additional audit details
  dataRetentionDays: number;     // Configurable retention
  encryptionRequired: boolean;   // Force encryption
  accessReviewDays: number;      // Periodic access review
}
```

PHI Sanitization

```
// Automatic PHI detection and handling
export class PHISanitizationService {
  private patterns = [
    /\b\d{3}-\d{2}-\d{4}\b/,     // SSN
    /\b\d{9}\b/,                  // MRN
    /\b[A-Z]{2}\d{6,8}\b/,       // License numbers
    // ... additional patterns
  ];

  async sanitize(input: string): Promise<SanitizedInput> {
    // Detect and redact PHI before processing
    let sanitized = input;
    for (const pattern of this.patterns) {
      sanitized = sanitized.replace(pattern, '[REDACTED]');
    }
    return { original: input, sanitized, phiDetected: sanitized !== input };
  }
}
```

GDPR Compliance

Data Subject Rights

RADIANT implements all required GDPR data subject rights:

Right	Implementation	API Endpoint
Right to Access	Export all user data	`GET /api/gdpr/export`
Right to Rectification	Update personal data	`PATCH /api/users/{id}`
Right to Erasure	Delete all user data	`DELETE /api/gdpr/erase`
Right to Portability	Export in machine-readable format	`GET /api/gdpr/export?format=json`
Right to Object	Opt-out of processing	`POST /api/gdpr/object`
Right to Restrict	Limit processing	`POST /api/gdpr/restrict`

Data Processing

```
interface GDPRDataRequest {
  subjectId: string;          // User identifier
  requestType: 'access' | 'rectification' | 'erasure' | 'portability' | 'object' | 'restrict';
  requestedBy: string;        // Requester (user or DPO)
  verificationMethod: string; // How identity was verified
  deadline: Date;             // 30-day compliance deadline
}

export class GDPRService {
  async handleDataRequest(request: GDPRDataRequest): Promise<GDPRResponse> {
    // Log the request
    await this.auditLogger.log('gdpr_request', request);

    switch (request.requestType) {
      case 'access':
        return this.exportUserData(request.subjectId);
      case 'erasure':
        return this.eraseUserData(request.subjectId);
      case 'portability':
        return this.exportPortableData(request.subjectId);
      // ... other handlers
    }
  }

  async eraseUserData(userId: string): Promise<void> {
    // Cascade delete across all tables
    await this.db.transaction(async (tx) => {
      await tx.delete('thinktank_steps').where('session_id', 'in',
        tx.select('id').from('thinktank_sessions').where('user_id', userId));
      await tx.delete('thinktank_sessions').where('user_id', userId);
      await tx.delete('usage_records').where('user_id', userId);
      await tx.delete('api_keys').where('user_id', userId);
      await tx.delete('users').where('id', userId);
    });
  }
}
```

Consent Management

```
interface ConsentRecord {
  userId: string;
  consentType: 'marketing' | 'analytics' | 'ai_training' | 'data_sharing';
  granted: boolean;
  grantedAt: Date;
  ipAddress: string;
  userAgent: string;
  version: string;  // Consent policy version
}

// All processing requires valid consent
async function checkConsent(userId: string, purpose: string): Promise<boolean> {
  const consent = await db.query(
    'SELECT granted FROM consent_records WHERE user_id = $1 AND consent_type = $2',
    [userId, purpose]
  );
  return consent?.granted === true;
}
```

Data Retention

Data Type	Retention Period	Basis
User accounts	Until deletion requested	Contract
Session data	90 days	Legitimate interest
Audit logs	7 years	Legal requirement

Data Type	Retention Period	Basis
Usage analytics	2 years	Legitimate interest
AI training data	Until consent withdrawn	Consent

ISO 27001 Compliance

Information Security Management System (ISMS)

RADIANT's infrastructure aligns with ISO 27001:2022 requirements:

Annex A Controls

A.5 Organizational Controls

Control	Description	Implementation
A.5.1	Policies for information security	Documented security policies
A.5.2	Information security roles	Defined RACI matrix
A.5.3	Segregation of duties	Role-based access, dual approval
A.5.7	Threat intelligence	AWS GuardDuty, threat feeds
A.5.15	Access control	Cognito + IAM + RLS
A.5.23	Information security for cloud	AWS Well-Architected
A.5.29	Information security during disruption	DR procedures

A.6 People Controls

Control	Description	Implementation
A.6.1	Screening	Background checks for admins
A.6.3	Information security awareness	Training programs
A.6.5	Responsibilities after termination	Access revocation procedures

A.7 Physical Controls

Control	Description	Implementation
A.7.1	Physical security perimeters	AWS data center security
A.7.4	Physical security monitoring	AWS compliance certifications

A.8 Technological Controls

Control	Description	Implementation
A.8.1	User endpoint devices	MDM for admin devices
A.8.2	Privileged access rights	IAM policies, MFA required
A.8.3	Information access restriction	RLS, tenant isolation
A.8.4	Access to source code	GitHub branch protection
A.8.5	Secure authentication	Cognito, JWT, API keys
A.8.7	Protection against malware	WAF, input validation
A.8.9	Configuration management	CDK, Infrastructure as Code
A.8.10	Information deletion	GDPR erasure, retention policies
A.8.11	Data masking	PHI sanitization, PII redaction
A.8.12	Data leakage prevention	DLP policies, egress controls
A.8.15	Logging	CloudWatch, audit trails
A.8.16	Monitoring activities	CloudWatch alarms, dashboards
A.8.20	Networks security	VPC, security groups, NACLs
A.8.22	Segregation of networks	Private subnets, VPC endpoints
A.8.24	Use of cryptography	KMS, TLS 1.3, AES-256
A.8.25	Secure development lifecycle	Code review, security scanning
A.8.28	Secure coding	OWASP guidelines, linting

Risk Assessment Matrix

Risk Category	Likelihood	Impact	Controls
Data breach	Low	Critical	Encryption, RLS, monitoring
Service outage	Medium	High	Multi-AZ, auto-scaling, DR
Unauthorized access	Low	Critical	MFA, RBAC, audit logging
Insider threat	Low	High	Segregation, dual approval
Supply chain attack	Low	High	Dependency scanning, SBOMs

Security Architecture

Defense in Depth

```
                INTERNET




              AWS WAF
   Rate limiting, SQL injection, XSS protection





           CloudFront / ALB
         TLS 1.3 termination
```

```
              API Gateway
       JWT validation, API key auth



                VPC (Private)

                 Lambda Functions
           Input validation, RLS context




             Aurora PostgreSQL
        Row-Level Security, Encryption
```

Two-Person Approval

Sensitive operations require dual admin approval:

```
interface ApprovalRequest {
  id: string;
  requesterId: string;
  actionType: 'delete_tenant' | 'modify_billing' | 'grant_super_admin' |
              'bulk_export' | 'disable_security';
  resourceId: string;
  payload: Record<string, unknown>;
  status: 'pending' | 'approved' | 'rejected' | 'expired';
  requiredApprovals: number;  // Usually 2
  approvals: Approval[];
  expiresAt: Date;
}

// Cannot approve own requests
async function approveRequest(requestId: string, approverId: string) {
  const request = await getRequest(requestId);

  if (request.requesterId === approverId) {
    throw new Error('Cannot approve own request');
  }

  if (request.approvals.some(a => a.approverId === approverId)) {
    throw new Error('Already approved');
  }
  // Add approval
  request.approvals.push({ approverId, approvedAt: new Date() });

  // Execute if threshold met
  if (request.approvals.length >= request.requiredApprovals) {
    await executeApprovedAction(request);
  }
}
```

Audit Logging

Log Structure

```
interface AuditLog {
  id: string;
  timestamp: Date;
  tenantId: string;
  userId: string;
  adminId?: string;
  action: string;
  resourceType: string;
  resourceId: string;
  ipAddress: string;
  userAgent: string;
  requestId: string;
  oldValue?: Record<string, unknown>;
  newValue?: Record<string, unknown>;
  result: 'success' | 'failure';
  errorMessage?: string;
}
```

Logged Actions

- All authentication events (login, logout, MFA)
- All API requests with parameters
- All data modifications (create, update, delete)
- All admin actions
- All security events (failed auth, rate limits)
- All GDPR requests
- All compliance-related operations

Log Retention

Log Type	Retention	Storage
API Access	90 days	CloudWatch
Security Events	1 year	S3 + Glacier
Audit Trail	7 years	S3 + Glacier

Log Type	Retention	Storage
GDPR Requests	7 years	Aurora