Security Documentation

Table of Contents

Security Documentation

HIPAA Compliance Overview

This POC is designed to align with HIPAA technical safeguards. All AWS services used are HIPAA-eligible and covered under the AWS Business Associate Addendum (BAA).

Security Controls Matrix

HIPAA Requirement	Control	Implementation
Access Control	Authentication	Google OAuth with staff-only access
Access Control	Authorization	Session-based access with CSRF tokens
Audit Controls	Audit Logs	Application audit logging + CloudWatch
Integrity	Data Integrity	CSRF protection, input validation
Transmission Security	Encryption in Transit	TLS 1.3 via ALB
Encryption	Encryption at Rest	EBS encryption, S3 SSE

Authentication

Google OAuth 2.0

- **Provider**: Google OAuth 2.0 / OpenID Connect
- **Flow**: Authorization Code Grant
- **Scope**: email, profile
- **Token Storage**: Server-side session (HttpOnly cookie)

Session Management

Setting	Value
Session Storage	Server-side (PHP sessions)
Cookie Flags	HttpOnly, Secure, SameSite=Lax
Session Timeout	PHP default (24 minutes idle)
CSRF Token	64-character random hex

Authorization

Access Levels

Role	Capabilities
Authenticated User	View all messages, send messages, delete own messages, view audit log
Unauthenticated	Login page only

CSRF Protection

All state-changing operations (POST requests) require a valid CSRF token: - Token generated on session start - Token validated on form submission - Token regenerated after use (optional)

Encryption

At Rest

Resource	Method	Key Management
EBS Volumes	AES-256	AWS managed
S3 Backups	SSE-S3 (AES-256)	AWS managed
Secrets	AWS Secrets Manager	AWS managed

In Transit

Connection	Protocol	Notes
Client → ALB	TLS 1.3	ACM certificate
ALB → EC2	HTTP	Internal VPC only
EC2 → S3	HTTPS	VPC endpoint
EC2 → Secrets Manager	HTTPS	VPC endpoint

Network Security

Network Isolation

- EC2 in **private subnet** (no public IP)
- Only ALB in public subnet
- NAT Gateway for outbound traffic
- VPC endpoints for AWS services

Security Groups

Principle of Least Privilege: - ALB: Only 80/443 inbound from internet - EC2: Only 80 inbound from ALB - No SSH port open (use SSM)

No SSH Access

- SSH (port 22) is **not open**
- Access via **SSM Session Manager** only
- All sessions logged to CloudWatch
- No key pairs required

Application Security

Input Validation

Input	Validation
Subject	Required, max 255 chars
Body	Required, text only
Message ID	Integer validation

Output Encoding

All user-supplied data is escaped before display:

```
htmlspecialchars($data, ENT_QUOTES, 'UTF-8')
```

Security Headers

Configured in Nginx:

```
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
X-XSS-Protection: 1; mode=block
Referrer-Policy: strict-origin-when-cross-origin
Content-Security-Policy: default-src 'self'; ...
```

SQL Injection Prevention

- **PDO** with prepared statements
- **Parameterized queries** for all database operations
- No string concatenation in SQL

Audit Logging

Events Logged

Event	Data Captured
LOGIN	User ID, timestamp, IP
LOGOUT	User ID, timestamp, IP
MESSAGE_CREATE	User ID, message ID, subject
MESSAGE_DELETE	User ID, message ID

Log Storage

Destination	Retention
SQLite (audit_log table)	Application lifetime
CloudWatch Logs	365 days

Log Format

```
{
  "timestamp": "2026-01-31T12:00:00Z",
  "user_id": 1,
  "action": "MESSAGE_CREATE",
  "details": "Created message #42: Subject here",
  "ip_address": "10.0.10.50"
}
```

Secrets Management

AWS Secrets Manager

Stored secrets: - `GOOGLE_CLIENT_ID` - `GOOGLE_CLIENT_SECRET` - `APP_SECRET` - `DB_ENCRYPTION_KEY`

Secret Retrieval

- EC2 retrieves secrets at boot via IAM role
- Secrets written to `.env` file (not in source control)

- Secrets Manager audit trail in CloudTrail

Backup Security

Backup Process

1. SQLite database backed up daily at 2 AM
2. Backup compressed with gzip
3. Uploaded to S3 with server-side encryption
4. S3 bucket policy enforces encryption

Backup Encryption

```
aws s3 cp backup.sqlite.gz s3://bucket/backups/ --sse AES256
```

Backup Retention

Age	Storage Class
0-30 days	Standard
30-365 days	Glacier
>365 days	Deleted

Vulnerability Management

Dependencies

Tool	Purpose
Composer	PHP dependency management
Docker	Container isolation

Update Process

1. Review security advisories
2. Update dependencies in `composer.json`
3. Test in development
4. Deploy via Terraform

Incident Response

Detection

- Monitor CloudWatch Logs for anomalies
- Check audit_log table for suspicious activity
- Review ALB access logs

Containment

1. Identify affected resources
2. Isolate EC2 if necessary (modify security group)
3. Preserve logs for analysis

Recovery

1. Restore from S3 backup if needed
2. Re-deploy clean infrastructure via Terraform
3. Update credentials in Secrets Manager

Security Checklist

- ▣ BAA signed with AWS
- ▣ All services HIPAA-eligible
- ▣ Encryption at rest enabled
- ▣ Encryption in transit (TLS 1.3)
- ▣ No SSH access (SSM only)
- ▣ Private subnet for compute
- ▣ Security groups configured
- ▣ Audit logging enabled
- ▣ CSRF protection
- ▣ XSS protection
- ▣ SQL injection prevention
- ▣

Secrets in AWS Secrets Manager

▣

Automated backups

▣

Backup encryption

Compliance Notes

This POC implements technical safeguards aligned with HIPAA requirements. For production use, additional controls may be required:

- Administrative safeguards (policies, training)
- Physical safeguards (AWS handles for cloud infrastructure)
- Risk assessment documentation
- Business Associate Agreements with all vendors
- Incident response procedures
- Workforce security training