Security Policy

Supported Versions

We provide security updates for the following versions of JOJI Oi:

Version	Supported
1.0.x	:white_check_mark:
< 1.0	:x:

Security Features

Data Protection

Input Sanitization

- All user inputs are sanitized to prevent injection attacks
- HTML/XML tags are stripped from content
- Null bytes and control characters are removed
- Input length limits are enforced

File System Security

- Path traversal protection (no `../` sequences allowed)
- Restricted file extensions (`.json` , `.md` , `.txt` , `.log` only)
- Secure file permissions (640 for data files, 600 for WAL)
- Atomic file operations to prevent corruption

Memory Protection

- Content size limits (10KB per segment)
- Memory usage monitoring and limits
- Automatic cleanup of old data
- Secure memory allocation patterns

Access Control

File Locking

- Exclusive file locks prevent concurrent access issues
- Lock timeout protection against deadlocks
- Automatic lock cleanup on process termination
- Cross-platform locking implementation

Process Isolation

- Non-root user execution in containers
- Minimal system privileges required
- Isolated data directories
- Secure temporary file handling

Network Security

Metrics Endpoint

- Metrics server runs on configurable port (default: 8000)
- No authentication by default (should be secured in production)
- Consider using reverse proxy with authentication
- Restrict network access to trusted sources

Health Checks

- Health endpoint provides system status
- No sensitive information exposed
- Rate limiting recommended for production
- Monitor for unusual access patterns

Cryptographic Considerations

Data at Rest

- No built-in encryption (implement at filesystem level)
- Secure file permissions as first line of defense
- Consider using encrypted storage volumes
- Regular backup encryption recommended

Data in Transit

- No built-in TLS (use reverse proxy)
- Metrics endpoint should be secured
- Consider VPN for remote monitoring
- Encrypt backup transfers

Security Best Practices

Deployment Security

Production Environment

```
# Create dedicated user
sudo useradd -r -s /bin/false jojiai

# Set secure permissions
sudo chown -R jojiai:jojiai /opt/jojiai
sudo chmod 750 /opt/jojiai
sudo chmod 640 /opt/jojiai/data/*.json
sudo chmod 600 /opt/jojiai/data/*.wal

# Restrict network access
sudo ufw allow from 10.0.0.0/8 to any port 8000
sudo ufw allow from 10.0.0.0/8 to any port 8001
```

Container Security

```
# Use non-root user
RUN groupadd -r jojiai && useradd -r -g jojiai jojiai
USER jojiai

# Minimal base image
FROM python:3.11-slim

# Read-only root filesystem
docker run --read-only --tmpfs /tmp jojiai:latest

# Drop capabilities
docker run --cap-drop=ALL jojiai:latest
```

Configuration Security

Environment Variables

```
# Secure configuration
export JOJIAI_MEMORY_PATH="/secure/path/memory"
export JOJIAI_BACKUP_PATH="/secure/path/backups"
export JOJIAI_LOG_LEVEL="INFO"  # Avoid DEBUG in production
export JOJIAI_METRICS_PORT="8000"
```

File Permissions

```
# Data directory
chmod 750 /path/to/memory
chown jojiai:jojiai /path/to/memory

# Configuration files
chmod 640 /path/to/config.json
chown jojiai:jojiai /path/to/config.json

# Log files
chmod 640 /path/to/logs/*.log
chown jojiai:jojiai /path/to/logs/
```

Monitoring Security

Log Security

- Avoid logging sensitive information
- Secure log file permissions (640)
- Regular log rotation and cleanup
- Monitor for security-related events

Metrics Security

- Secure metrics endpoint access
- Monitor for unusual patterns
- Set up alerting for security events
- Regular security metric reviews

Backup Security

Backup Encryption

```
# Encrypt backups
tar -czf - /path/to/backups | gpg --cipher-algo AES256 --compress-algo 1 --symmetric --
output backup.tar.gz.gpg

# Decrypt backups
gpg --decrypt backup.tar.gz.gpg | tar -xzf -
```

Secure Storage

- Store backups in secure locations
- Use encrypted storage volumes
- Implement access controls
- Regular backup integrity checks

Vulnerability Management

Security Updates

We regularly review and update dependencies for security vulnerabilities:

```
# Check for security issues
safety check

# Update dependencies
pip install --upgrade jojiai

# Verify installation
memoryctl validate
```

Security Scanning

Static Analysis

```
# Security linting
bandit -r src/

# Dependency scanning
safety check --json

# Code quality
flake8 src/ --select=E9,F63,F7,F82
```

Runtime Security

```
# Monitor for security events
grep -i "security\|error\|warning" /path/to/logs/*.log

# Check file permissions
find /path/to/memory -type f -exec ls -la {} \;

# Monitor process activity
ps aux | grep jojiai
```

Incident Response

Security Incident Handling

Immediate Response

1. **Isolate** affected systems
2. **Assess** the scope of impact
3. **Contain** the security breach
4. **Document** all actions taken
5. **Notify** relevant stakeholders

Investigation Steps

1. **Preserve** evidence and logs
2. **Analyze** attack vectors
3. **Identify** compromised data
4. **Determine** root cause
5. **Implement** fixes

Recovery Process

1. **Patch** vulnerabilities
2. **Restore** from clean backups
3. **Update** security measures
4. **Monitor** for reoccurrence
5. **Review** and improve processes

Reporting Security Issues

Contact Information

- **Email**: security@jojiai.com
- **PGP Key**: Available on request
- **Response Time**: 24-48 hours

What to Include

1. **Description** of the vulnerability
2. **Steps** to reproduce
3. **Potential impact** assessment
4. **Suggested** mitigation
5. **Your contact** information

Responsible Disclosure

- Allow 90 days for fix development
- Coordinate disclosure timeline
- Provide credit for discovery
- Follow coordinated vulnerability disclosure

Security Checklist

Pre-Deployment

- [ ] Review security configuration
- [ ] Set appropriate file permissions
- [ ] Configure secure network access
- [ ] Enable security monitoring
- [ ] Test backup and recovery procedures

Regular Maintenance

- [ ] Update dependencies monthly
- [ ] Review security logs weekly
- [ ] Rotate credentials quarterly
- [ ] Conduct security assessments annually
- [ ] Update incident response procedures

Monitoring

- [ ] Failed authentication attempts
- [ ] Unusual file access patterns
- [ ] High error rates
- [ ] Memory usage spikes
- [ ] Network connection anomalies

Compliance

Data Protection

- Input validation and sanitization
- Secure data storage practices
- Access control mechanisms
- Audit logging capabilities
- Data retention policies

Industry Standards

- Follow OWASP security guidelines
- Implement defense in depth
- Regular security assessments
- Vulnerability management program
- Incident response procedures

Additional Resources

Security Tools

- Bandit (https://bandit.readthedocs.io/) - Python security linter
- Safety (https://pyup.io/safety/) - Dependency vulnerability scanner
- OWASP ZAP (https://owasp.org/www-project-zap/) - Web application security scanner

Security Guidelines

- OWASP Top 10 (https://owasp.org/www-project-top-ten/)
- NIST Cybersecurity Framework (https://www.nist.gov/cyberframework)
- Python Security Best Practices (https://python.org/dev/security/)

Training Resources

- Secure Coding Practices (https://owasp.org/www-project-secure-coding-practices-quick-reference-guide/)
- Python Security Training (https://realpython.com/python-security/)
- Container Security (https://kubernetes.io/docs/concepts/security/)

For questions about this security policy, please contact security@jojiai.com.