Bridgelet

Security Model & Threat Analysis

Document Version: 0.1 (MVP)
Last Updated: January 2025
Status: Early Development
Security Review Status: Pre-Audit

⚠️ **Important:** This document describes the security model for the MVP phase. A comprehensive security audit will be conducted before mainnet deployment.

1. Security Overview

Bridgelet's security model is built on three core principles:

- **Minimize Trust:** Reduce reliance on centralized components
- **Time-Bound Exposure:** Limit vulnerability windows through expiration
- **Single-Use Design:** Minimize attack surface with disposable accounts

2. Threat Model

2.1 Adversary Capabilities

We consider adversaries with the following capabilities:

- Network-level access (man-in-the-middle attacks)
- Access to claim links through phishing or social engineering
- Ability to create multiple accounts and automate attacks

- Knowledge of smart contract code and system architecture
- Computational resources for brute-force attempts

2.2 Assets to Protect

Asset	Criticality	Description
Funds in Ephemeral Accounts	CRITICAL	Payment amounts waiting to be claimed
Claim Credentials	CRITICAL	Keys/tokens that authorize fund claims
Smart Contract Logic	CRITICAL	On-chain code controlling fund movement
Recipient Personal Data	HIGH	Email addresses, phone numbers used for claims
Organization API Keys	HIGH	Credentials for accessing Bridgelet SDK
Account Metadata	MEDIUM	Transaction history and account states

3. Identified Threats

🚨 Threat T1: Claim Link Interception

Severity: CRITICAL

Description: Attacker intercepts claim link sent via email/SMS and claims funds before legitimate recipient.

Attack Vector:

- Email account compromise
- SMS interception (SIM swapping)
- Network traffic monitoring
- Phishing links that steal claim credentials

✅ Mitigation M1: Multi-Factor Claim Verification

Implementation:

- Require recipient to verify email/phone ownership during claim
- Optional: Additional verification step (OTP, security questions)
- Rate limiting on claim attempts
- Suspicious activity detection (multiple failed attempts)

MVP Status: Basic email verification implemented; enhanced MFA post-MVP

🚨 Threat T2: Smart Contract Vulnerabilities

Severity: CRITICAL

Description: Exploitable bugs in Soroban smart contracts allowing unauthorized fund access or denial of service.

Attack Vector:

- Reentrancy attacks
- Integer overflow/underflow
- Access control bypasses
- Logic errors in sweep conditions

✅ Mitigation M2: Secure Development & Auditing

Implementation:

- Follow Soroban security best practices
- Comprehensive unit and integration testing
- Formal verification for critical functions
- Third-party security audit before mainnet launch
- Bug bounty program post-launch

MVP Status: Development best practices; audit scheduled for Q2 2026

🚨 Threat T3: Private Key Compromise

Severity: HIGH

Description: Bridgelet SDK's private keys for ephemeral account creation are stolen or exposed.

Attack Vector:

- Server compromise
- Insider threat
- Insecure key storage
- Code repository leak

✅ Mitigation M3: Secure Key Management

Implementation:

- Use hardware security modules (HSM) for production keys
- Environment-based key isolation (never in code)
- Key rotation policies
- Multi-signature requirements for critical operations
- Encrypted key storage with strict access controls

MVP Status: Environment variables; HSM integration planned for production

🚨 Threat T4: Denial of Service (DoS)

Severity: MEDIUM

Description: Attacker floods system with account creation or claim requests, preventing legitimate use.

Attack Vector:

- Mass account creation requests
- Repeated claim attempts
- Resource exhaustion attacks

✅ Mitigation M4: Rate Limiting & Resource Controls

Implementation:

- API rate limiting per organization
- Account creation throttling
- Claim attempt limits per account
- CAPTCHA for suspicious activity patterns
- Cost-based deterrence (organizations pay for account creation)

MVP Status: Basic rate limiting implemented

🚨 Threat T5: Expired Account Fund Recovery Abuse

Severity: MEDIUM

Description: Attacker exploits recovery mechanism to claim expired funds meant for return to organization.

Attack Vector:

- Front-running recovery transactions

- Exploiting race conditions in expiration logic

✅ Mitigation M5: Secure Recovery Process

Implementation:

- Hard-coded recovery address in smart contract (organization wallet)
- Time-locked recovery (only after expiration + grace period)
- Immutable recovery logic
- Transparent on-chain verification

MVP Status: Implemented in smart contract

4. Security Architecture

4.1 Defense in Depth

Multiple security layers protect against threats:

1. **Network Layer:** TLS/HTTPS for all communications
2. **Application Layer:** Input validation, authentication, authorization
3. **Smart Contract Layer:** On-chain access controls and validations
4. **Data Layer:** Encryption at rest and in transit

4.2 Trust Boundaries

```
┌─────────────────────────────────────────────┐
│  Untrusted Zone (Public Internet)           │
│  - Recipients                                │
│  - Claim links                               │
└─────────────────┬───────────────────────────┘
                  │ TLS/HTTPS
┌─────────────────▼───────────────────────────┐
│  Semi-Trusted Zone (Bridgelet SDK)          │
│  - API authentication                        │
│  - Rate limiting                              │
│  - Claim verification                         │
└─────────────────┬─────────────────────────────┘
                  │ Authenticated calls
┌─────────────────▼─────────────────────────────┐
│  Trusted Zone (Smart Contracts)               │
│  - Immutable on-chain logic                   │
│  - Cryptographic guarantees                   │
│  - No human override possible                 │
└───────────────────────────────────────────────┘
```

5. Security Best Practices

5.1 For Organizations Using Bridgelet

- ✅ Secure claim link delivery (use HTTPS, secure email providers)
- ✅ Educate recipients about phishing risks
- ✅ Monitor for unusual claiming patterns
- ✅ Set appropriate expiration windows (not too long)
- ✅ Rotate API keys regularly
- ✅ Use production keys only in production environments

5.2 For Recipients

- ✅ Verify sender before clicking claim links
- ✅ Check URL authenticity (official Bridgelet domain)
- ✅ Use secure wallet providers
- ✅ Never share claim codes with anyone
- ✅ Claim funds promptly (don't wait until expiration)

5.3 For Bridgelet Developers

- ✅ Follow secure coding standards (OWASP)
- ✅ Regular dependency updates and vulnerability scans
- ✅ Code review for all security-sensitive changes
- ✅ Principle of least privilege for all components
- ✅ Comprehensive logging and monitoring

6. Incident Response

6.1 Security Incident Classifications

Level	Examples	Response Time
P0 - Critical	Active exploit, mass fund theft	Immediate (< 1 hour)

Level	Examples	Response Time
P1 - High	Vulnerability discovered, potential exploit	< 4 hours
P2 - Medium	Service degradation, isolated incidents	< 24 hours
P3 - Low	Minor issues, low-risk vulnerabilities	< 1 week

6.2 Response Procedures

1. **Detection:** Automated monitoring alerts team
2. **Assessment:** Determine severity and impact
3. **Containment:** Limit damage (pause system if needed)
4. **Eradication:** Fix vulnerability, deploy patch
5. **Recovery:** Resume normal operations
6. **Post-Mortem:** Document lessons learned

7. Audit & Compliance

7.1 Pre-Launch Requirements

- ✅ Smart contract security audit by reputable firm
- ✅ Penetration testing of SDK and APIs
- ✅ Code review by independent security experts
- ✅ Testnet deployment and public bug bounty

7.2 Ongoing Security

- Regular security assessments (quarterly)
- Continuous monitoring and alerting
- Vulnerability disclosure program
- Security patches within 48 hours of discovery

8. Future Enhancements

Post-MVP Security Roadmap

- **Q2 2026:** Hardware wallet support for organizations
- **Q2 2026:** Multi-signature claim requirements (optional)
- **Q3 2026:** Formal verification of critical contract functions
- **Q3 2026:** Decentralized claim verification (zero-knowledge proofs)
- **Q4 2026:** Insurance integration for large deployments

Bridgelet Security Model v0.1

This document will be updated as threats evolve and mitigations improve.

Report security issues: security@bridgelet.org