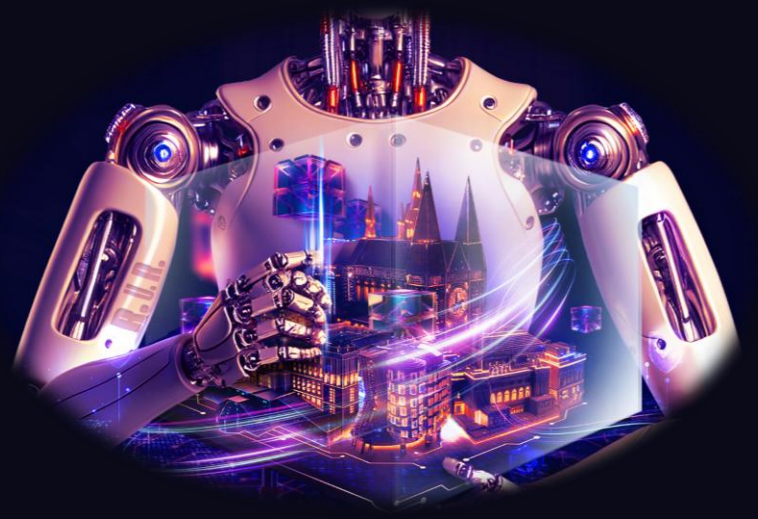

Threat Modeling & Incident Response in OpenSSL-Based Systems

Practical Lessons with STRIDE, Attack Trees & FIPS 140-3

Mehri Yahyaei -Senior Security Evaluator & Researcher
Briska Info Tech Canada Inc.
OpenSSL Conference, October 2025

Why OpenSSL Matters

Used everywhere: OpenSSL powers web servers, VPNs, email systems, mobile apps, and IoT devices.

Deep integration: A single weak spot in configuration can impact multiple services and vendors.

Operational mistakes have a big impact: Issues with certificates, cipher settings, or key rotation often create security gaps. lose entire systems.

Real-world lessons:

- Past incidents like **FREAK** and **Logjam** showed how export-grade ciphers left enabled in configs opened the door to MITM attacks.
- **Recent scans:** reveal that thousands of production servers still allow weak TLS versions or misconfigured cipher suites.
- **Cloud & IoT audits** continue to uncover apps with certificate validation disabled in production, creating silent but critical vulnerabilities.

Where Things Go Wrong

Problem	Real-world Example	Risk
Unsafe defaults	Servers in 2024 still allow weak TLS 1.2 ciphers (CBC + SHA-1).	Downgrade: MITM / weak encryption
Risky configurations	LB advertises TLS 1.3 + weak TLS 1.2, backend enforces TLS 1.3.	Clients negotiate weakest option
Flawed design assumptions	API Gateway enforces TLS, but backend microservices accept self-signed traffic.	Bypass perimeter: unauthorized access
Human errors	Cloud outage due to expired internal TLS cert in monitoring.	Service downtime + exploitable blind spot

Threat Modeling
Why & How

🔍 Anticipate attacks before they happen

🌳 Use STRIDE & Attack Trees

👥 Collaborative & Repeatable Process

🛡️ Prioritize mitigations by likelihood & impact

STRIDE Applied to OpenSSL

S: Spoofing
Weak cert/hostname validation → MITM risk

E: Elevation of Privilege
Crypto API misuse, poor key handling

OpenSSL Threat Modeling

T: Tampering
Unsafe builds or tampered binaries

D: Denial of Service
Handshake floods / renegotiation abuse

R: Repudiation
Missing TLS handshake/error logs

I: Info Disclosure
Weak TLS 1.2 ciphers or chain issues

STRIDE in Action, OpenSSL

STRIDE	OpenSSL Case Example	What to Monitor	Quick Mitigation
S — Spoofing	CVE-2022-0778: cert parsing bug → potential fake cert acceptance	Unexpected issuer/chain changes; hostname-verification disabled in builds	Enforce strict hostname verification; curated trust store; pin critical endpoints
T — Tampering	Supply-chain: modified OpenSSL packages (trojanized builds)	Checksum/signature mismatches; unsigned artifacts in CI	Signed artifacts, SBOM & reproducible builds; verify checksums in CI
R — Repudiation	Weak/no TLS negotiation logging → actions hard to prove	Sparse TLS logs; missing handshake params/cert fingerprints	Enable detailed TLS logs; centralize & make append-only; timestamp key events
I — Info Disclosure	Heartbleed (CVE-2014-0160): memory leak of secrets	Vuln scans; anomalous memory/heap errors; unusual key reuse	Patch immediately; rotate keys/certs; reduce attack surface & monitor
D — DoS	CVE-2022-3602: X.509 email addr overflow → crash; handshake floods	Handshake spikes; CPU exhaustion during TLS setup; renegotiation rates	Rate-limit handshakes; restrict renegotiation; upstream DDoS protection
E — EoP	API misuse: nonce/IV reuse, insecure wrappers exposing keys	Secrets scanner hits; permissive FS perms on keys; code patterns	Use HSM/KMS; least privilege; lint crypto API usage; rotate on suspicion

Attack Trees
Example: Compromise TLS confidentiality

Goal: Compromise TLS confidentiality

Path A: Downgrade attack
- Force protocol downgrade
- Negotiate weak TLS cipher
- Capture traffic → decrypt

Path B: Bypass validation (MITM)
- Obtain/forge a certificate
- Present forged cert to client
- Intercept & modify traffic

Path C: Key theft
- Access backup/dev machine
- Exfiltrate private key
- Use the key to impersonate/decrypt

Path	Likelihood	Impact	Risk
A: Downgrade	3	4	12
B: MITM	2	5	10
C: Key theft	2	5	10

Quick Wins:
- Disable weak ciphers
- Enforce strict cert validation
- Restrict private key access

Case Study: Weak Cipher Configuration

Symptom:	Diagnosis:	Remediation:
Handshake failures after update	Cipher mismatch between LB & backend	Unify policy, disable legacy ciphers, staged rollout

Weak or inconsistent cipher policies don’t just weaken security; they can also break availability.

Consistency + Strong Ciphers = Secure and Reliable TLS

Incident Response for Crypto Systems

Detection
- TLS handshake anomalies, renegotiation abuse
- Unexpected certificate chain changes
- Spike in TLS errors or downgrade attempts

Containment
- Isolate affected endpoints / block IPs
- Disable TLS fallback mechanisms
- Route traffic through hardened gateways

Eradication
- Staged key/cert rotation with rollback
- Rebuild compromised binaries/containers
- Disable weak/deprecated ciphers

Verification
- Automated TLS handshake test suites
- Cipher scans across infrastructure
- Audit logs for compliance & non-repudiation

FIPS 140-3
Operational Considerations

🔒 Validated Crypto Modules

- Use only FIPS-validated modules (track certificate #)
- Avoid mixing non-validated code paths
- Monitor vendor validation status & updates

🔁 Key Lifecycle Discipline

- Approved DRBG for keygen; documented entropy sources
- Secure storage (HSM/KMS), least-privilege access
- Rotation schedule and secure destruction procedures

🕘 Patching & Rollout Constraints

- Patches may affect validation status—plan validated builds
- Staging + rollback plans; change control gates
- Vendor-coordinated timelines for critical fixes

📓 Documentation & Evidence

- Entropy, PRNG behavior, health tests documented
- Key management SOPs & tamper-evident logs
- Audit-ready artifacts for assessments

🧭 Operational Planning

- Dedicated staging envs for crypto changes
- Separation of duties & approvals (RACI)
- Compliance liaison for FIPS/NIST alignment

Practical Checklists & Automation

🔍 Nightly CI handshake tests using openssl s_client + alerting

🔑 Cipher matrix compatibility checks (LB ↔ backend ↔ clients)

📜 Automated certificate chain watchers and expiry alerts

🚀 Small scripts for quick staging checks and canary rolls

Takeaways & Next Steps

Threat model early
+ automate checks
+ rehearse IR

🧩 Resilient OpenSSL Systems

Use FIPS/NIST
+ document key lifecycle

Share artifacts &
playbooks across teams

Future Outlook: What’s Next in OpenSSL Security

🚀 **TLS 1.3 Adoption**
- Faster handshakes
- New downgrade attack gaps

👨‍🔬 **Post-Quantum Crypto**
- NIST PQC rollouts
- Transition roadmaps needed

🛡️ **Supply Chain Security**
- Tampered builds & deps
- Stronger verification pipelines

Questions? / Contact

Thank you for your attention! I'm happy to discuss specifics after the talk.

Email: mehri@briskatech.ca

LinkedIn: https://linkedin.com/mehri-yahyaei