RFC-002: Cyberspace Architecture

Status: Informational **Date:** January 2026 **Author:** Derrell Piper ddp@eludom.net

---

Abstract

Cyberspace is a distributed systems architecture built on S-expressions and Scheme, designed for cryptographic security without complexity. This RFC describes the overall system design, philosophy, and components.

E Pluribus Unum

Out of many, one.

Rough consensus, cryptography,
trusted systems, running code.

---

1. Motivation

Modern distributed systems are drowning in complexity. X.509 certificates require decoder rings. Container orchestration demands armies of SREs. Security policies hide in YAML nested seventeen levels deep.

Cyberspace returns to first principles:

- **S-expressions** for everything: readable, parseable, auditable
- **Minimal TCB:** prove the crypto, evolve the rest
- **No central authority:** SPKI/SDSI namespaces over PKI hierarchies
- **Running code:** every feature traces to research, runs, and is tested

---

2. The Prime Directive

If it's in the TCB, it's in OCaml. Otherwise it's in Chicken Scheme.

```
TRUSTED COMPUTING BASE (OCaml, ~1000 lines)

   Ed25519          SHA-512          Verify
   Sign             Hash             Chain

That's it. Everything else is policy.

                          FFI (tiny surface)

EVERYTHING ELSE (Chicken Scheme, unlimited)

Vault · Audit · Replication · Names · Discovery
CLI Tools · API Server · Library · Scripts

Change it anytime. It's just policy.
```

Rationale:

- **OCaml:** Strong types, formal verification with Coq, compile-time safety
- **Chicken Scheme:** Interactive development, S-expressions everywhere, rapid evolution
- **The boundary:** Tiny, frozen, proven. The TCB does crypto and *nothing else*.

A smaller TCB means fewer bugs that can break security. We prove the crypto in Coq. Everything else can evolve freely.

---

3. Core Components

3.1 The Vault

Sealed version control with cryptographic signatures.

```
$ cyberspace vault init
$ cyberspace vault commit -m "Deploy new API"
$ cyberspace vault verify
  All signatures valid
```

Every commit is cryptographically sealed. No GPG. No separate signing step.

3.2 The Audit Trail

Tamper-evident logging with hash chains.

```
(audit-entry
  (sequence 1042)
  (timestamp "2026-01-06T15:30:00Z")
  (action (commit "Deploy v2.1"))
  (actor (public-key |...|))
  (previous-hash |...|)
  (signature |...|))
```

Append-only. Hash-chained. Signed. Export as text.

3.3 SPKI Certificates

Authorization without identity.

```
(cert
  (issuer (public-key ed25519 |base64....|))
  (subject (name alice "deploy-server"))
  (grant (tag (http-api (method POST) (path "/deploy/*"))))
  (valid (not-after "2026-12-31")))
```

Human-readable security policy. No ASN.1. No X.509.

3.4 Threshold Governance

Democracy in code.

```
$ cyberspace policy set deploy/prod --threshold 3 --signers alice,bob,carol,dave,eve
$ cyberspace execute proposal-42
  3/5 signatures verified
```

No single point of failure. No rogue admin.

3.5 Secret Sharing

Survive key loss with Shamir splitting.

```
(define shares (shamir-split master-key 5 3))
;; Distribute 5 shares. Recover with any 3.
```

3.6 Replication Layer

Federated distribution without central registry. See RFC-001.

```
(seal-publish "1.0.0" remote: "/shared/releases")
(seal-subscribe "/shared/releases" verify-key: alice-pub)
(seal-synchronize peer-remote direction: 'both)
```

3.7 The Library Directory

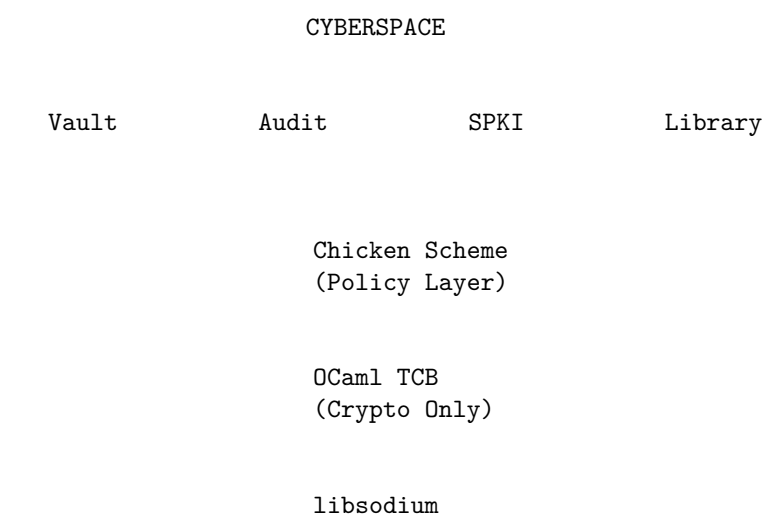
421 research papers, searchable.

```
 > papers by Lamport
 Found 15 documents
 > about SPKI
 > from 1979
```

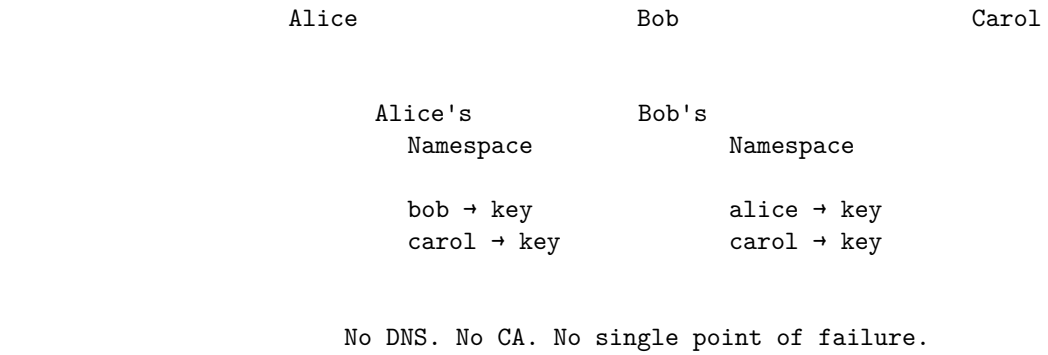
---

4. Architecture

4.1 Layer Diagram



4.2 No Central Authority



SPKI/SDSI gives you: - **Local namespaces:** "bob" means what *you* say it means - **Authorization without identity:** Grant permissions to keys, not people - **Delegation chains:** Alice → Bob → Carol, each step verified

---

5. Research Foundations

Every feature traces to a foundational paper:

Feature	Paper	Author	Year
Vault signatures	"New Directions in Cryptography"	Diffie & Hellman	1976
Audit trails	"How to Time-Stamp a Digital Document"	Haber & Stornetta	1991
Merkle proofs	"A Digital Signature Based on a Conventional Encryption Function"	Merkle	1987

Feature	Paper	Author	Year
Threshold sigs	“How to Share a Secret”	Shamir	1979
Logical clocks	“Time, Clocks, and the Ordering of Events”	Lamport	1978
Capabilities	“Protection”	Lampson	1971
SPKI certs	“SPKI Certificate Theory”	Ellison et al.	1999

421 papers in the library. Not just referenced—*studied, implemented, running*.

---

6. Implementation Status

Lamport OTP	Merkle Trees	Capabilities
ChaCha20	Poly1305	Lamport Signatures
SPKI Certs	Vault	Audit Trails
Replication	Threshold Sigs	Shamir Sharing
Library Directory		

Each traces to original research. Each runs. Each is tested.

---

7. Security Considerations

7.1 TCB Minimization

The attack surface is limited to ~1000 lines of OCaml calling libsodium. This code is:

- Formally specified
- Proven in Coq
- Frozen (rarely changes)

7.2 No Single Point of Failure

- **No CA**: SPKI namespaces are local

- **No central server:** Federation, not empire
- **No single key:** Threshold signatures, Shamir sharing

7.3 Auditability

- All security policy is human-readable S-expressions
- All history is hash-chained and signed
- All audit trails are exportable text

---

8. Getting Started

```
git clone git@github.com:ddp/cyberspace.git
cd cyberspace/spki/scheme
./spki-keygen alice
./seal init --key alice.private
./seal commit -m "Hello, Cyberspace"
```

---

9. Future Work

- **ChaCha20-Poly1305 AEAD** - Authenticated encryption
- **Lamport Logical Clocks** - Distributed ordering
- **TLA+ Specifications** - Model-checked protocols
- **Coq Extraction** - Verified OCaml from proofs
- **Federation Protocol** - Cross-instance sync
- **Byzantine Paxos** - Fault-tolerant consensus

---

10. References

1. Diffie, W., & Hellman, M. (1976). New directions in cryptography.
2. Haber, S., & Stornetta, W. S. (1991). How to time-stamp a digital document.
3. Merkle, R. C. (1987). A digital signature based on a conventional encryption function.
4. Shamir, A. (1979). How to share a secret.
5. Lamport, L. (1978). Time, clocks, and the ordering of events in a distributed system.
6. Lampson, B. W. (1971). Protection.
7. Ellison, C., et al. (1999). SPKI certificate theory. RFC 2693.

Changelog

- **2026-01-06** – Initial specification

"E Pluribus Unum"

github.com/ddp/cyberspace

Built on 50 years of cryptographic research. Standing on the shoulders of giants.