

# RFC-015: Versioning and Rollback

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

---

## Abstract

This RFC specifies user-visible versioning and rollback semantics for Cyberspace vaults, providing clear mental models for version management, safe rollback operations, and recovery from mistakes.

---

## Motivation

Users need simple answers to simple questions:

- What version am I on?
- What versions exist?
- How do I go back?
- What if I make a mistake?

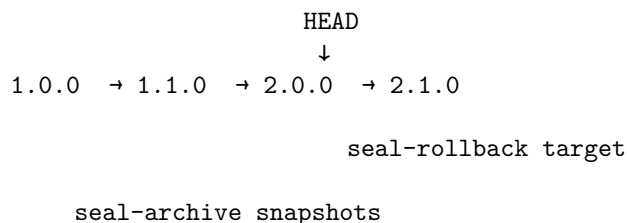
Git provides mechanisms. Cyberspace provides semantics:

```
seal-version      → "2.1.0"
seal-versions     → [1.0.0, 1.1.0, 2.0.0, 2.1.0]
seal-rollback 2.0.0 → Done. Now at 2.0.0.
seal-recover      → Here's what you can recover.
```

---

## Specification

### Version Model



**Versions** are semantic (X.Y.Z), immutable, signed.

**HEAD** is current working state.

**Rollback** creates forward commit that restores past state.

## Core Operations

**seal-version** Report current version.

```
(seal-version)
;; => "2.1.0" or "2.1.0+3" (3 commits past release)
```

Output format: - X.Y.Z - Exactly at release - X.Y.Z+N -  
N commits past release - X.Y.Z-dev - Development branch -  
unversioned - No releases yet

**seal-versions** List all versions.

```
(seal-versions)
;; => ("1.0.0" "1.1.0" "2.0.0" "2.1.0")
```

```
(seal-versions verbose: #t)
;; =>
;; 1.0.0 2026-01-01 "Initial release"
;; 1.1.0 2026-01-05 "Bug fixes"
;; 2.0.0 2026-01-10 "New governance model"
;; 2.1.0 2026-01-15 "Performance improvements"
```

**seal-rollback** Roll back to specific version.

```
(seal-rollback "2.0.0")
```

**Process:** 1. Verify version exists 2. Check for uncommitted changes (abort or stash) 3. Create rollback commit with contents of target version 4. Record in audit trail 5. Optionally create rollback tag

**Rollback commit message:**

Rollback to 2.0.0

Restored vault state to version 2.0.0.  
Previous HEAD was 2.1.0+3.

Reason: [user-provided or "Manual rollback"]

**NOT a git reset.** History preserved. Rollback is a forward operation.

**seal-diff** Compare versions.

```
(seal-diff "2.0.0" "2.1.0")  
;; Shows what changed between versions
```

```
(seal-diff "2.0.0")  
;; Shows what changed since 2.0.0
```

**seal-recover** Recovery options for mistakes.

```
(seal-recover)  
;; =>  
;; Recovery options:  
;; 1. Uncommitted changes (stashed 5 minutes ago)  
;; 2. Last commit (seal-undo)  
;; 3. Version 2.1.0 (seal-rollback "2.1.0")  
;; 4. Archive vault-2.0.0.archive (seal-restore)
```

---

## Rollback Semantics

### Forward Rollback (Default)

Before: 1.0.0 → 1.1.0 → 2.0.0 → 2.1.0 → HEAD  
                                  ↑  
                          rollback target

After: 1.0.0 → 1.1.0 → 2.0.0 → 2.1.0 → [rollback commit]

contents match

History preserved. Audit trail shows rollback.

### Hard Rollback (Dangerous)

```
(seal-rollback "2.0.0" hard: #t)
```

Actually resets to version. **Destroys history.**

Requires: – Explicit `hard: #t` flag – Confirmation prompt –  
Audit entry before destruction

### Rollback with Stash

```
(seal-rollback "2.0.0" stash: #t)
```

Stashes uncommitted changes before rollback. Can recover  
with:

```
(seal-stash-pop)
```

---

## Version Tags

### Automatic Tagging

```
(seal-release "2.1.0")  
;; Creates:  
;;   - Git tag: 2.1.0  
;;   - Signature: .vault/releases/2.1.0.sig  
;;   - Audit entry
```

### Rollback Tags

```
(seal-rollback "2.0.0" tag: "rollback-2026-01-15")  
;; Creates additional tag marking rollback point
```

### Tag Listing

```
(seal-tags)  
;; =>  
;; 1.0.0           2026-01-01  release  
;; 1.1.0           2026-01-05  release  
;; 2.0.0           2026-01-10  release  
;; 2.1.0           2026-01-15  release  
;; rollback-2026-01-15 2026-01-15  rollback
```

---

## Recovery Scenarios

### Scenario 1: Bad Commit

```
;; Oops, committed broken code  
(seal-undo) ; Undo last commit  
;; or  
(seal-rollback "2.1.0") ; Back to last release
```

### Scenario 2: Bad Release

```
;; Released 2.2.0 but it's broken  
(seal-rollback "2.1.0") ; Back to good version  
(seal-release "2.2.1") ; Patch release
```

### Scenario 3: Lost Changes

```
;; Accidentally discarded work
(seal-recover)                ; See options
(seal-stash-pop)              ; If stashed
;; or
(seal-reflog)                 ; Find lost commits
```

### Scenario 4: Corrupted Vault

```
;; Something went very wrong
(seal-restore "vault-2.0.0.archive" verify-key: "alice.pub")
```

---

## Audit Integration

Every version operation recorded:

```
(audit-entry
  (action
    (verb seal-rollback)
    (object "2.0.0")
    (parameters
      (from "2.1.0+3")
      (method forward)
      (reason "Production incident"))))
(seal ...))
```

Audit trail answers: – Who rolled back? – When? – From what version? – Why? (if reason provided)

---

## CLI Interface

*# Current version*

```
$ seal version
2.1.0+3
```

*# List versions*

```
$ seal versions
1.0.0  2026-01-01  Initial release
1.1.0  2026-01-05  Bug fixes
2.0.0  2026-01-10  New governance model
2.1.0  2026-01-15  Performance improvements
```

```

# Rollback
$ seal rollback 2.0.0
Rolling back to 2.0.0...
Rollback complete. Now at 2.0.0.

# Rollback with reason
$ seal rollback 2.0.0 --reason "Production incident #123"

# Compare versions
$ seal diff 2.0.0 2.1.0
M config.scm
A new-feature.scm
D deprecated.scm

# Recovery options
$ seal recover
Recovery options:
  1. seal undo           - Undo last commit
  2. seal rollback 2.1.0 - Return to last release
  3. seal stash pop      - Recover stashed changes

```

---

## Safety Mechanisms

### Uncommitted Changes

```

(seal-rollback "2.0.0")
;; Error: Uncommitted changes detected.
;; Use --stash to save changes, or --force to discard.

```

### Protected Versions

```

(vault-config 'protected-versions '("1.0.0" "2.0.0"))

(seal-rollback "0.9.0")
;; Warning: Rolling back past protected version 1.0.0
;; This may break compatibility. Continue? [y/N]

```

### Rollback Limits

```

(vault-config 'max-rollback-distance 10)

(seal-rollback "0.1.0")
;; Error: Rollback distance (20 versions) exceeds limit (10).
;; Use --force to override.

```

---

## Implementation

### seal-rollback Implementation

```
(define (seal-rollback version #!key hard stash force reason tag)
  "Roll back vault to specified version"

  ;; Validate version exists
  (unless (tag-exists? version)
    (error "Version not found" version))

  ;; Check for uncommitted changes
  (when (uncommitted-changes?)
    (cond
      (stash (seal-stash))
      (force (void))
      (else (error "Uncommitted changes. Use stash: or force:"))))

  ;; Record current state
  (let ((current (seal-version)))

    ;; Perform rollback
    (if hard
      (hard-rollback! version)
      (forward-rollback! version reason))

    ;; Create tag if requested
    (when tag
      (run-command "git" "tag" tag))

    ;; Audit
    (audit-append
      actor: (get-vault-principal (vault-config 'signing-key))
      action: (list 'seal-rollback version)
      motivation: (or reason (sprintf "Rollback from ~a to ~a" current version))))

    (print " Rolled back to " version)))

(define (forward-rollback! version reason)
  "Create rollback commit (preserves history)"
  (run-command "git" "checkout" version "--" ".")
  (run-command "git" "commit" "-m"
    (sprintf "Rollback to ~a\n\n~a" version (or reason ""))))
```

```
(define (hard-rollback! version)
  "Reset to version (destroys history)"
  (print "WARNING: Hard rollback destroys history")
  (run-command "git" "reset" "--hard" version))
```

---

## Security Considerations

### Rollback Attacks

Attacker forces rollback to vulnerable version.

**Mitigation:** – Threshold signatures for rollback (RFC-007)  
– Protected version list – Audit trail detection

### History Manipulation

Hard rollback can hide evidence.

**Mitigation:** – Prefer forward rollback – Audit before hard  
rollback – Federation replication preserves history

---

## References

1. Semantic Versioning 2.0.0 (semver.org)
  2. Git Reset, Checkout, and Revert (git-scm.com)
  3. RFC-003: Cryptographic Audit Trail
  4. RFC-006: Vault System Architecture
- 

## Changelog

- 2026-01-06 – Initial specification
- 

**Implementation Status:** Proposed **Operations:** seal-version,  
seal-versions, seal-rollback, seal-diff, seal-recover  
**Rollback Model:** Forward commit (history-preserving)