

RFC-018: Zstd+Age Archive Format

Status: Implemented **Date:** January 2026 **Author:** Derrell Piper ddp@eludom.net **Implementation:** vault.scm

Abstract

This RFC specifies the Zstd+Age archive format for the Library of Cyberspace: a modern archive format combining Zstd compression with Age encryption, replacing gzip for cryptographic archives.

Motivation

The legacy cryptographic archive format uses gzip compression without encryption:

Aspect	gzip (legacy)	zstd + age
Compression speed	Slow	Fast (parallel)
Compression ratio	Good	Better
Encryption	None	Built-in (age)
Key compatibility	N/A	X25519/Ed25519 (SPKI aligned)
Streaming	Yes	Yes

Why change?

1. **Encryption at rest** – Archives contain potentially sensitive data
 2. **Faster compression** – Zstd is 3–5x faster than gzip
 3. **Better ratios** – Zstd typically achieves 10–20% better compression
 4. **Key alignment** – Age uses X25519/Ed25519, same curve family as SPKI
 5. **Parallel support** – zstd -T0 uses all cores
-

Specification

Archive Creation Pipeline

```
git archive | tar --zstd | age -r <recipients> > archive.tar.zst.age
```

Or equivalently:

```
tar --zstd -cf - <files> | age -r <recipient1> -r <recipient2> > archive.tar.zst.age
```

Archive Restoration Pipeline

```
age -d -i <identity> < archive.tar.zst.age | zstd -d | tar -xf -
```

Manifest Format

```
(sealed-archive
  (version "1.0.0")
  (format zstd-age)
  (archive "vault-1.0.0.archive.tar.zst.age")
  (compression zstd)
  (encryption age)
  (recipients ("age1ql3z7hjy54pw3hyww5ayyfg7zqgvc7w3j2elw8zmrj2kg5sfn9aqmcac8p"))
  (hash "sha512:...")
  (signature "ed25519:..."))
```

File Extensions

Extension	Contents
.archive	S-expression manifest
.tar.zst.age	Encrypted, compressed tarball

Configuration

```
; Set age recipients for encryption
(vault-config 'age-recipients
  '("age1ql3z7hjy54pw3hyww5ayyfg7zqgvc7w3j2elw8zmrj2kg5sfn9aqmcac8p"
    "age1..."))

;; Set age identity for decryption
(vault-config 'age-identity "~/config/age/identity.txt")

;; Set default archive format
(vault-config 'archive-format 'zstd-age)
```

Performance Characteristics

Compression

Zstd default (-3):

 Compression: ~400 MB/s
 Decompression: ~800 MB/s
 Ratio: ~3:1 (typical source code)

Zstd parallel (-T0):

 Compression: ~400 MB/s × cores
 Decompression: ~800 MB/s × cores

Gzip comparison:

 Compression: ~30 MB/s
 Decompression: ~200 MB/s
 Ratio: ~2.5:1

Encryption

Age X25519:

 Encrypt: ~1 GB/s (ChaCha20-Poly1305)
 Decrypt: ~1 GB/s
 Key size: 32 bytes (public), 32 bytes (private)

Age overhead:

 Header: ~200 bytes per recipient
 MAC: 16 bytes

Signature

Ed25519:

 Sign: ~10 s
 Verify: ~10 s
 Signature: 64 bytes

Total Pipeline

10 MB archive:

 Compress (zstd -T0): ~25 ms
 Encrypt (age): ~10 ms
 Hash (SHA-512): ~1 ms
 Sign (Ed25519): ~10 s
 Total: ~36 ms

vs gzip legacy:

Compress (gzip): ~300 ms
Hash (SHA-512): ~1 ms
Sign (Ed25519): ~10 s
Total: ~301 ms

Speedup: ~8x

Security Model

Encryption Layer (Age)

- **Algorithm:** X25519 + ChaCha20–Poly1305
- **Recipients:** Multiple allowed (each can decrypt)
- **Forward secrecy:** Ephemeral key per archive
- **Authenticity:** Poly1305 MAC

Signature Layer (SPKI)

- **Algorithm:** Ed25519
- **Scope:** Signs encrypted archive hash
- **Non-repudiation:** Signer cannot deny creating archive

Verification Order

1. **Verify signature** – Before decryption (fail fast on tampering)
2. **Verify hash** – Encrypted archive integrity
3. **Decrypt** – Only after signature verified
4. **Extract** – Only after successful decryption

Threat Mitigations

Threat	Mitigation
Eavesdropping	Age encryption (X25519 + ChaCha20)
Tampering	SHA-512 hash + Ed25519 signature
Replay attacks	Version + timestamp in manifest
Key compromise	Multiple recipients, key rotation
Chosen ciphertext	Poly1305 authentication

Compatibility

Age Identity Types

```
# Native age key  
age1ql3z7hjy54pw3hyww5ayyfg7zqgvc7w3j2elw8zmrj2kg5sfn9aqmcac8p  
  
# SSH key (age supports ssh-ed25519 and ssh-rsa)  
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAI...  
  
# GitHub key (via age-plugin-github or ssh)  
github:username
```

Platform Support

Platform	zstd	age
macOS	Homebrew	Homebrew
Linux	apt/yum	apt/yum or binary
FreeBSD	pkg	pkg
Windows	winget	winget or binary

Fallback

If zstd or age unavailable, fall back to cryptographic format:

```
(if (and (command-exists? "zstd") (command-exists? "age"))  
  (seal-archive version format: 'zstd-age)  
  (seal-archive version format: 'cryptographic))
```

Implementation

seal-archive-zstd-age

```
(define (seal-archive-zstd-age version output)  
  "Create zstd-compressed, age-encrypted archive with SPKI signature"  
  
  (let ((recipients (vault-config 'age-recipients))  
        (signing-key (vault-config 'signing-key))  
        (encrypted-file (sprintf "~a.tar.zst.age" output)))  
  
    ;; Require recipients  
    (unless (pair? recipients)
```

```

(error "No age recipients configured"))

;; Build recipient args: -r key1 -r key2 ...
(let ((recipient-args
      (string-join (map (lambda (r) (sprintf "-r ~a" r)) recipients) " ")))

;; Create archive: git archive / zstd / age
(let ((temp-tar (sprintf "/tmp/vault-~a.tar" version)))
  (run-command "git" "archive" "--prefix=vault/"
               (sprintf "--output=~/a" temp-tar) version)
  (system (sprintf "zstd -T0 -c ~/a | age ~/a > ~/a"
                  temp-tar recipient-args encrypted-file))
  (delete-file temp-tar))

;; Sign the encrypted archive
(when signing-key
  (let* ((archive-bytes (read-file-bytes encrypted-file))
         (archive-hash (sha512-hash archive-bytes))
         (signature (ed25519-sign signing-key archive-hash)))

;; Write manifest
(with-output-to-file output
  (lambda ()
    (write `(sealed-archive
              (version ,version)
              (format zstd-age)
              (archive ,encrypted-file)
              (compression zstd)
              (encryption age)
              (recipients ,recipients)
              (hash ,(blob->hex archive-hash))
              (signature ,(blob->hex signature)))))))))))

seal-restore-zstd-age

(define (seal-restore-zstd-age manifest verify-key target-dir identity)
  "Restore zstd+age encrypted archive"

  (let ((version (cadr (assq 'version (cdr manifest))))
        (archive-file (cadr (assq 'archive (cdr manifest))))
        (hash-hex (cadr (assq 'hash (cdr manifest)))))
        (sig-hex (cadr (assq 'signature (cdr manifest)))))
        (id-file (or identity (vault-config 'age-identity)))))

  ;; Require identity

```

```

(unless id-file
  (error "No age identity configured"))

<;; Verify signature first (before decryption)
(when verify-key
  (let* ((archive-bytes (read-file-bytes archive-file))
         (archive-hash (sha512-hash archive-bytes))
         (expected-hash (hex->blob hash-hex))
         (signature (hex->blob sig-hex))
         (pubkey (read-key-from-file verify-key)))

  (unless (equal? archive-hash expected-hash)
    (error "Archive hash mismatch"))

  (unless (ed25519-verify pubkey archive-hash signature)
    (error "Signature verification failed")))

<;; Decrypt and extract: age -d / zstd -d / tar -x
(system (sprintf "age -d -i ~a < ~a | zstd -d | tar -xf - -C ~a"
                  id-file archive-file target-dir)))

```

Usage Examples

Create Encrypted Archive

```

<;; Configure recipients
(vault-config 'age-recipients
  '("age1ql3z7hjy54pw3hyww5ayyfg7zqgvc7w3j2elw8zmrj2kg5sfn9aqmcac8p"))

<;; Create archive
(seal-archive "1.0.0" format: 'zstd-age)
<;; => vault-1.0.0.archive
<;; => vault-1.0.0.archive.tar.zst.age

```

Restore Encrypted Archive

```

<;; Configure identity
(vault-config 'age-identity "~/.config/age/identity.txt")

<;; Restore with verification
(seal-restore "vault-1.0.0.archive"
  verify-key: "publisher.public"
  target: "./restored")

```

Command Line Equivalents

```
# Create
git archive --prefix=vault/ 1.0.0 | \
zstd -T0 | \
age -r age1ql3z7hjy54pw3hyww5ayyfg7zqgvc7w3j2elw8zmrj2kg5sfn9aqmcac8p \
> vault-1.0.0.tar.zst.age

# Restore
age -d -i ~/.config/age/identity.txt < vault-1.0.0.tar.zst.age | \
zstd -d | \
tar -xf -
```

Migration

From Cryptographic to Zstd-Age

```
; Re-archive existing releases with new format
(for-each
  (lambda (version)
    (seal-archive version format: 'zstd-age))
  (get-local-releases))
```

Backward Compatibility

Both formats can coexist. The manifest format field determines restoration method: - (format cryptographic) → seal-restore-cryptographic – (format zstd-age) → seal-restore-zstd-age

References

1. Zstd – Facebook’s compression algorithm
 2. Age – Modern encryption tool
 3. RFC-006 – Vault System Architecture
 4. RFC-004 – SPKI Authorization
-

Changelog

- 2026-01-06 – Initial specification
-

Implementation Status: Complete **Compression:** Zstd (parallel) **Encryption:** Age (X25519 + ChaCha20-Poly1305) **Signature:** Ed25519 (SPKI)