age-plugin-se:
Building a lean cross-platform cryptography tool

Remko Tronçon
https://mko.re
r@mko.re
@mko.re
@remko@mas.to

Age

Simple, modern and secure file encryption tool

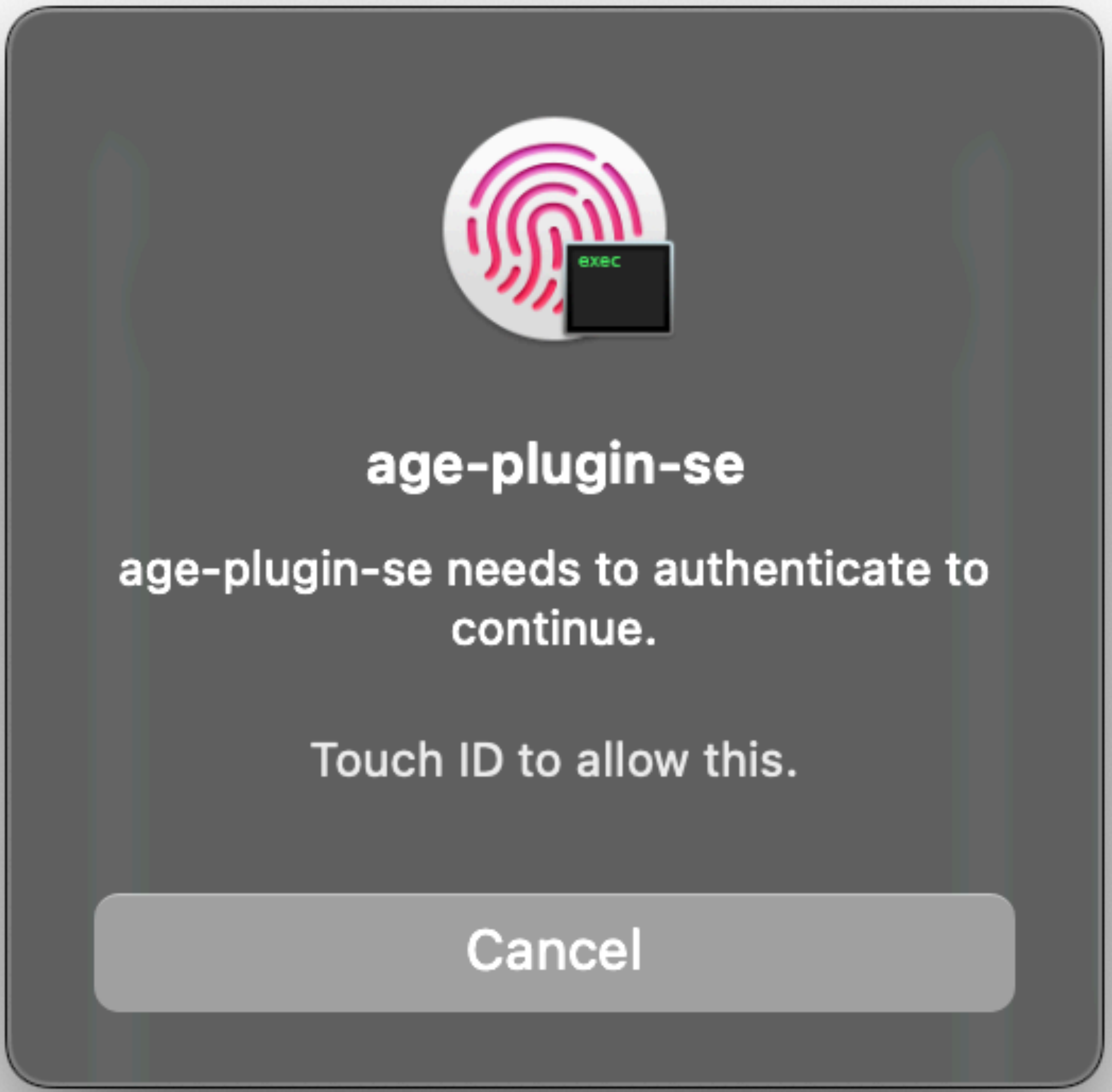
age

FILE ENCRYPTION

- Small explicit keys
- No config options
- UNIX-style composability

age-plugin-se

Use Secure Enclave to protect Age decryption keys

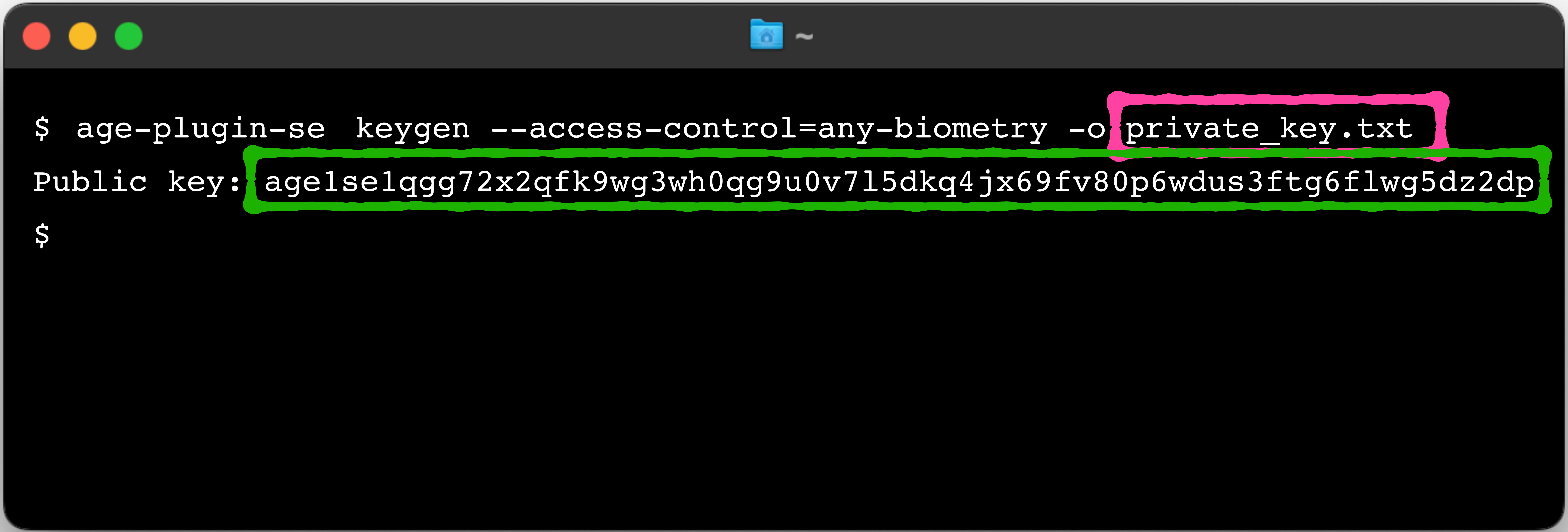

age-plugin-se

age-plugin-se needs to authenticate to continue.

Touch ID to allow this.

Usage

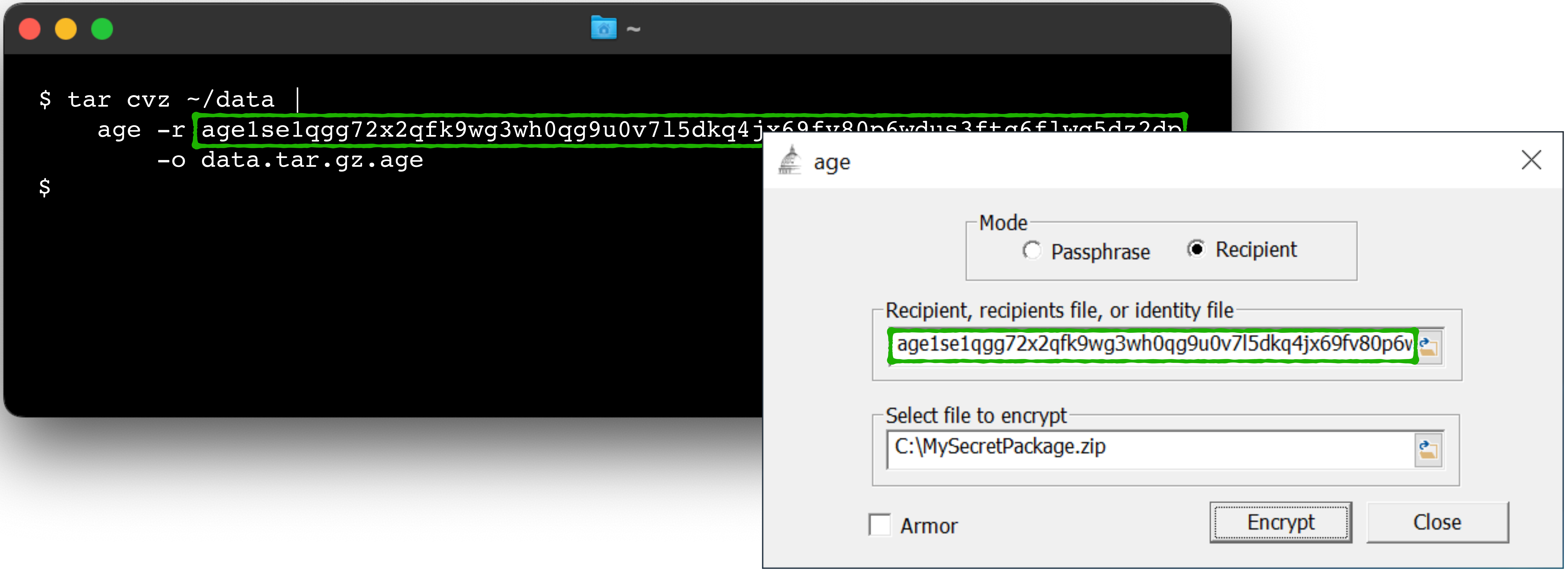
Step 1 · Create public & private key



```
$ age-plugin-se  keygen --access-control=any-biometry -o private_key.txt
Public key: age1se1qgg72x2qfk9wg3wh0qg9u0v7l5dkq4jx69fv80p6wdus3ftg6flwg5dz2dp
$
```

age-plugin-se usage

Step 2 · Encrypt files



```
$ tar cvz ~/data |
    age -r age1se1qgg72x2qfk9wg3wh0qg9u0v7l5dkq4jx69fv80p6wdus3ftq6flwg5dz2dp
        -o data.tar.gz.age
$
```

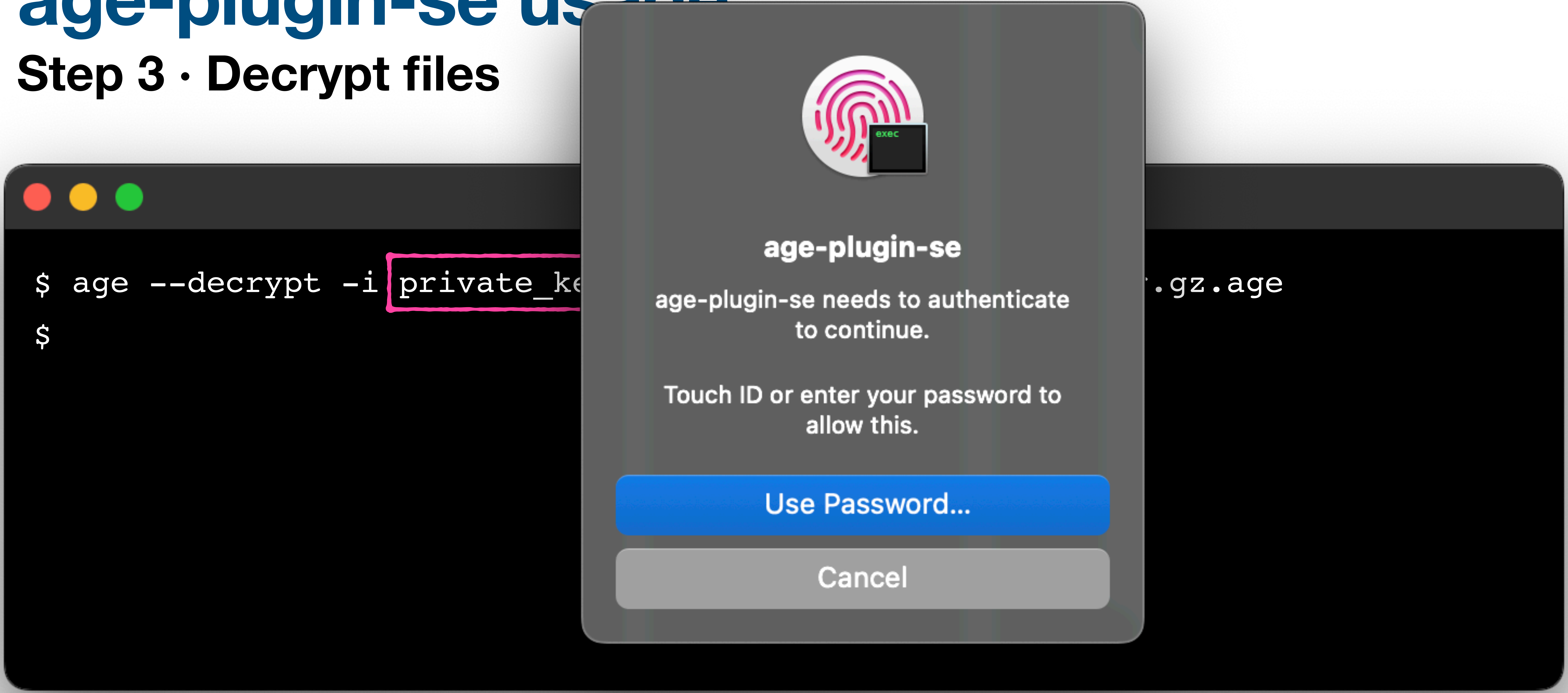
Mode: Passphrase [✓ Recipient]

Recipient, recipients file, or identity file: age1se1qgg72x2qfk9wg3wh0qg9u0v7l5dkq4jx69fv80p6v

Select file to encrypt: C:\MySecretPackage.zip

age-plugin-se usage

Step 3 · Decrypt files



```
$ age --decrypt -i private_ke                    .gz.age
$
```

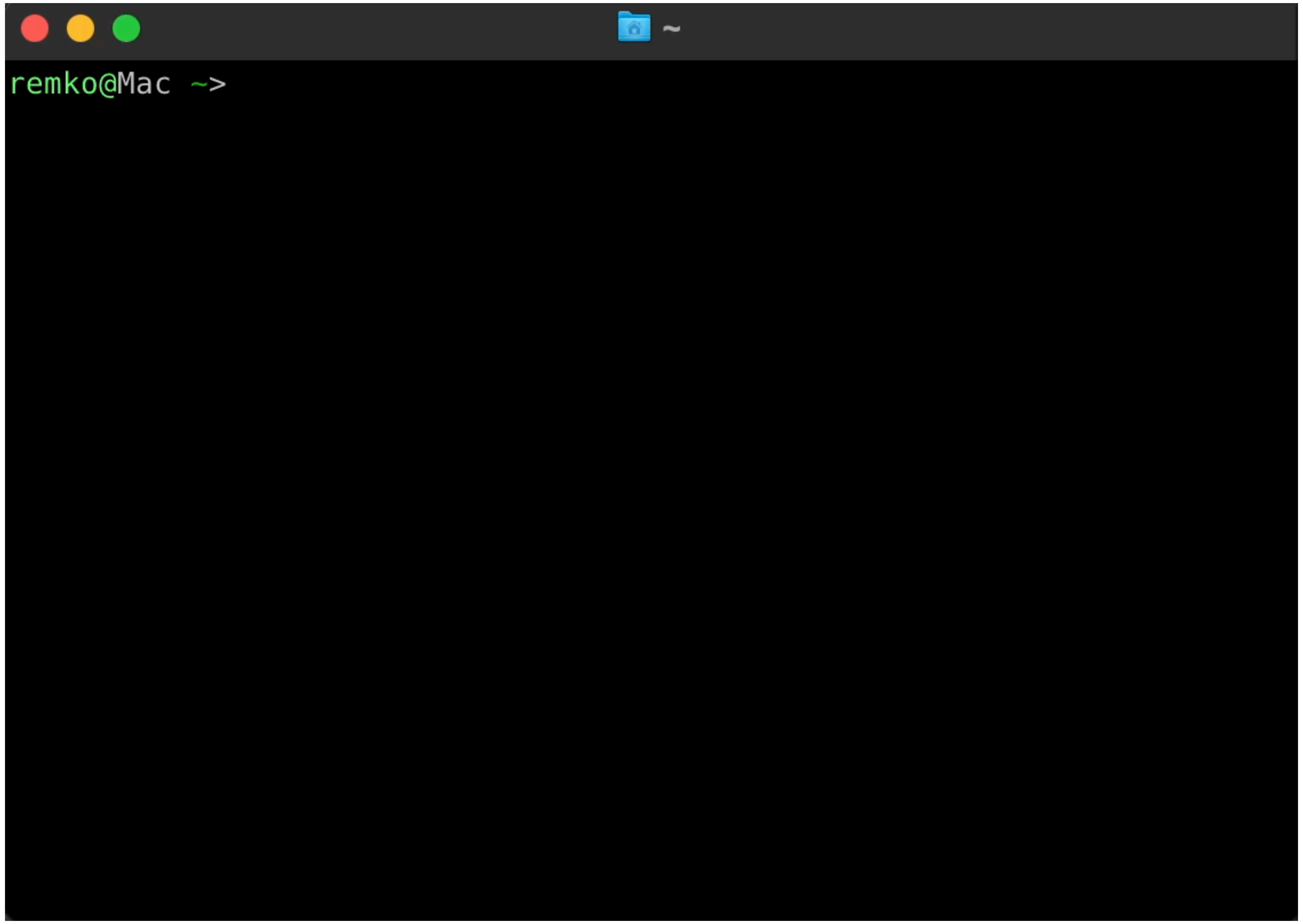
age-plugin-se

age-plugin-se needs to authenticate to continue.

Touch ID or enter your password to allow this.

age ecosystem

Example: Password manager



```
remko@Mac ~>
```

Apple CryptoKit

SecureEnclave.P256.KeyAgreement.PrivateKey

P256.KeyAgreement.PublicKey

ChaChaPoly.SealedBox

HMAC<SHA256>.MAC

SharedSecret

Swift Crypto

Open-source replacement for CryptoKit on Linux/Windows

~~SecureEnclave.P256.KeyAgreement.PrivateKey~~

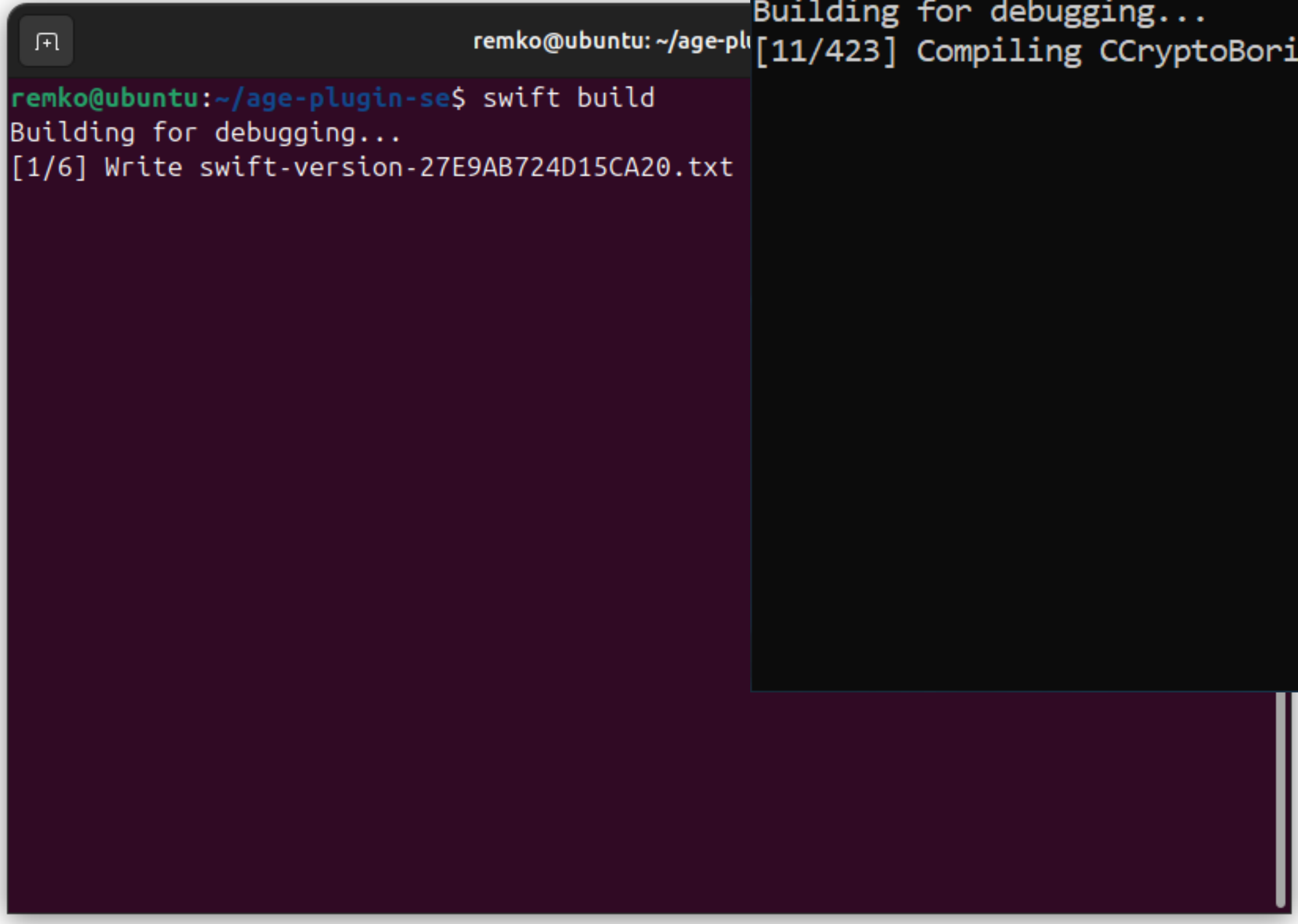
P256.KeyAgreement.PublicKey

ChaChaPoly.SealedBox

HMAC<SHA256>.MAC

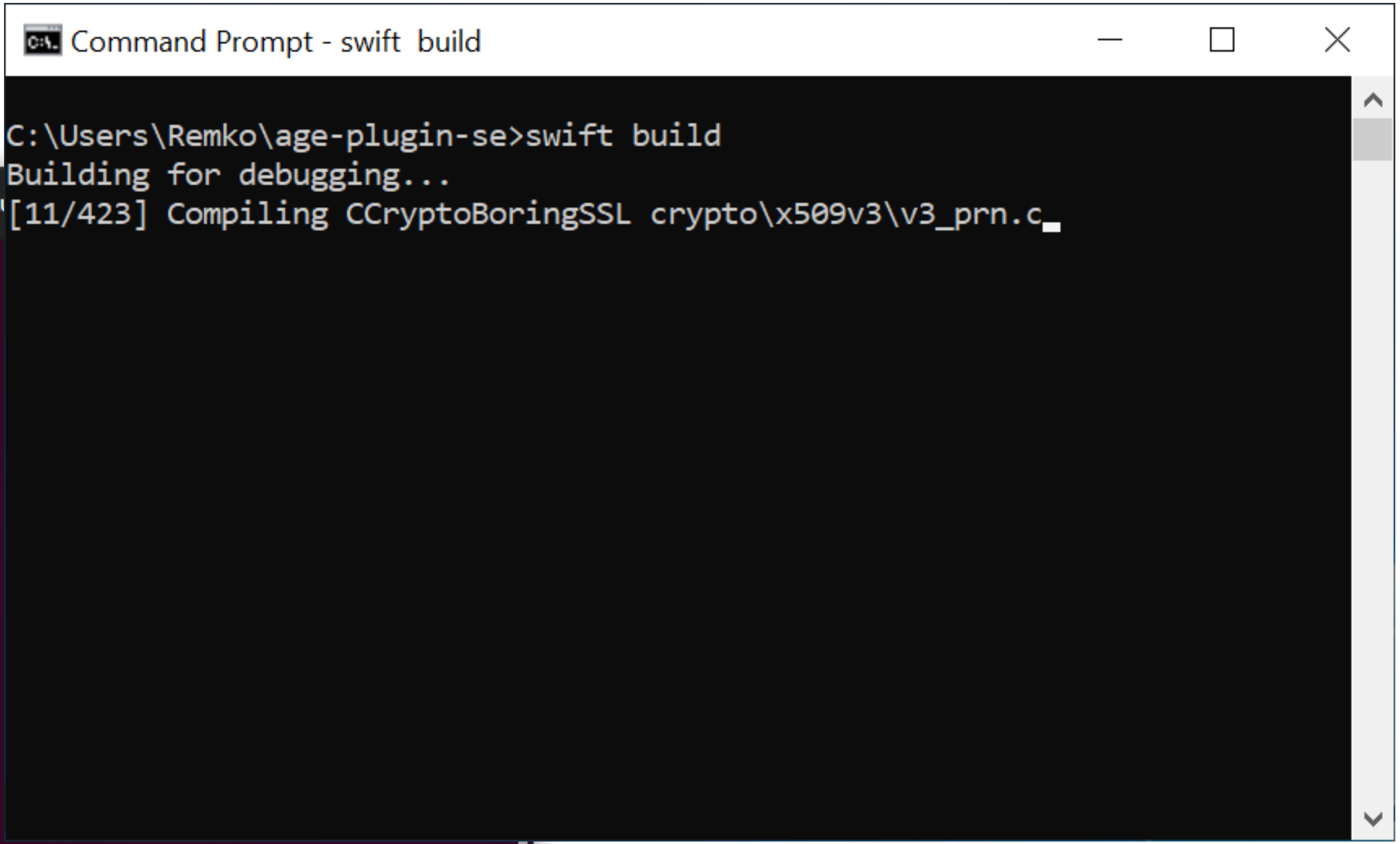
SharedSecret

Build



remko@ubuntu: ~/age-pl

```
remko@ubuntu:~/age-plugin-se$ swift build
Building for debugging...
[1/6] Write swift-version-27E9AB724D15CA20.txt
```



Command Prompt - swift build

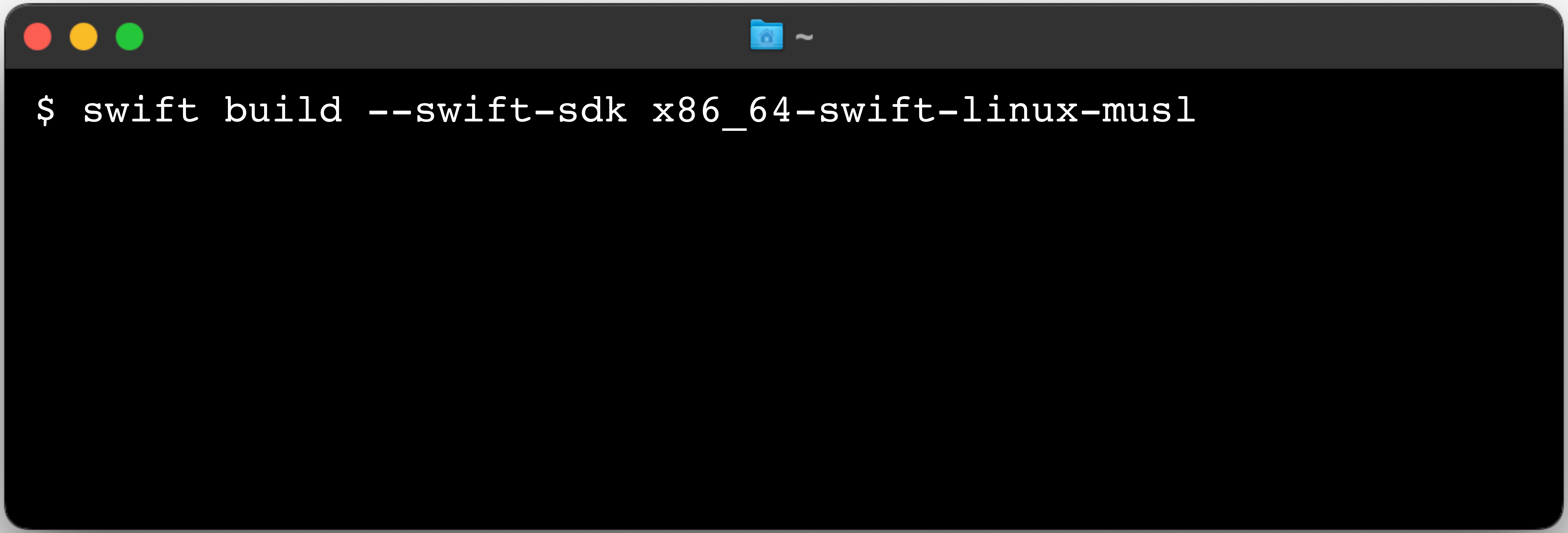
```
C:\Users\Remko\age-plugin-se>swift build
Building for debugging...
[11/423] Compiling CCryptoBoringSSL crypto\x509v3\v3_prn.c
```

Building for Alpine Linux

Glibc 🤜 🤛 Musl

https://mko.re/blog/swift-alpine-packaging/

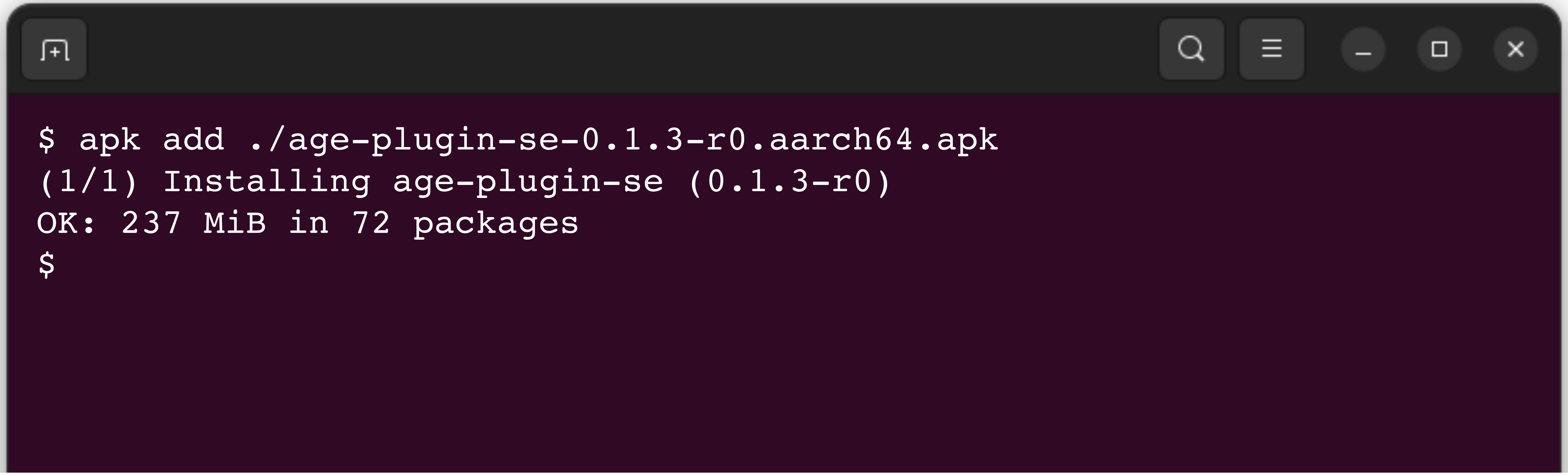
Cross-compiling on macOS



```
$ swift build --swift-sdk x86_64-swift-linux-musl
```

Packaging for Alpine Linux

Cross-platform packaging script



```
$ apk add ./age-plugin-se-0.1.3-r0.aarch64.apk
(1/1) Installing age-plugin-se (0.1.3-r0)
OK: 237 MiB in 72 packages
$
```

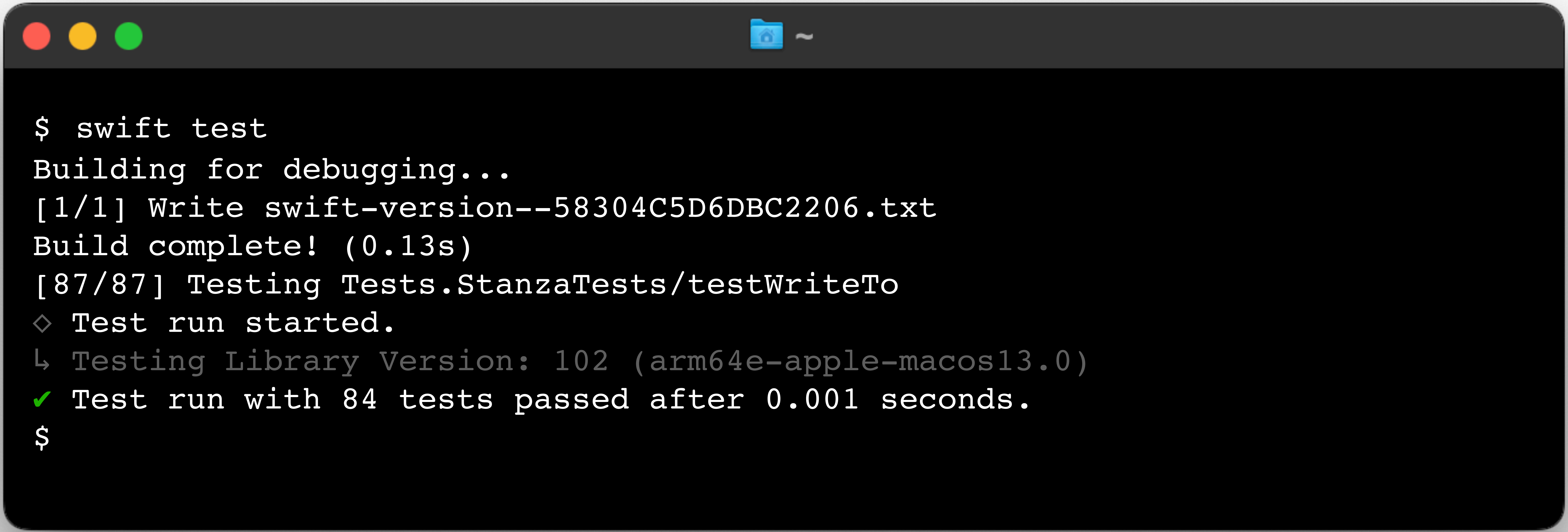
Cross-compiled binary size

Dynamic		252 KiB
Dynamic	🐧	1.2 MiB
Dynamic + static Swift stdlib	🐧	43.4 MiB
Dynamic + static Swift stdlib + dylibs	🐧	48.1 MiB
Static	🐧	100.4 MiB

@mko.re @remko@mas.to

Testing

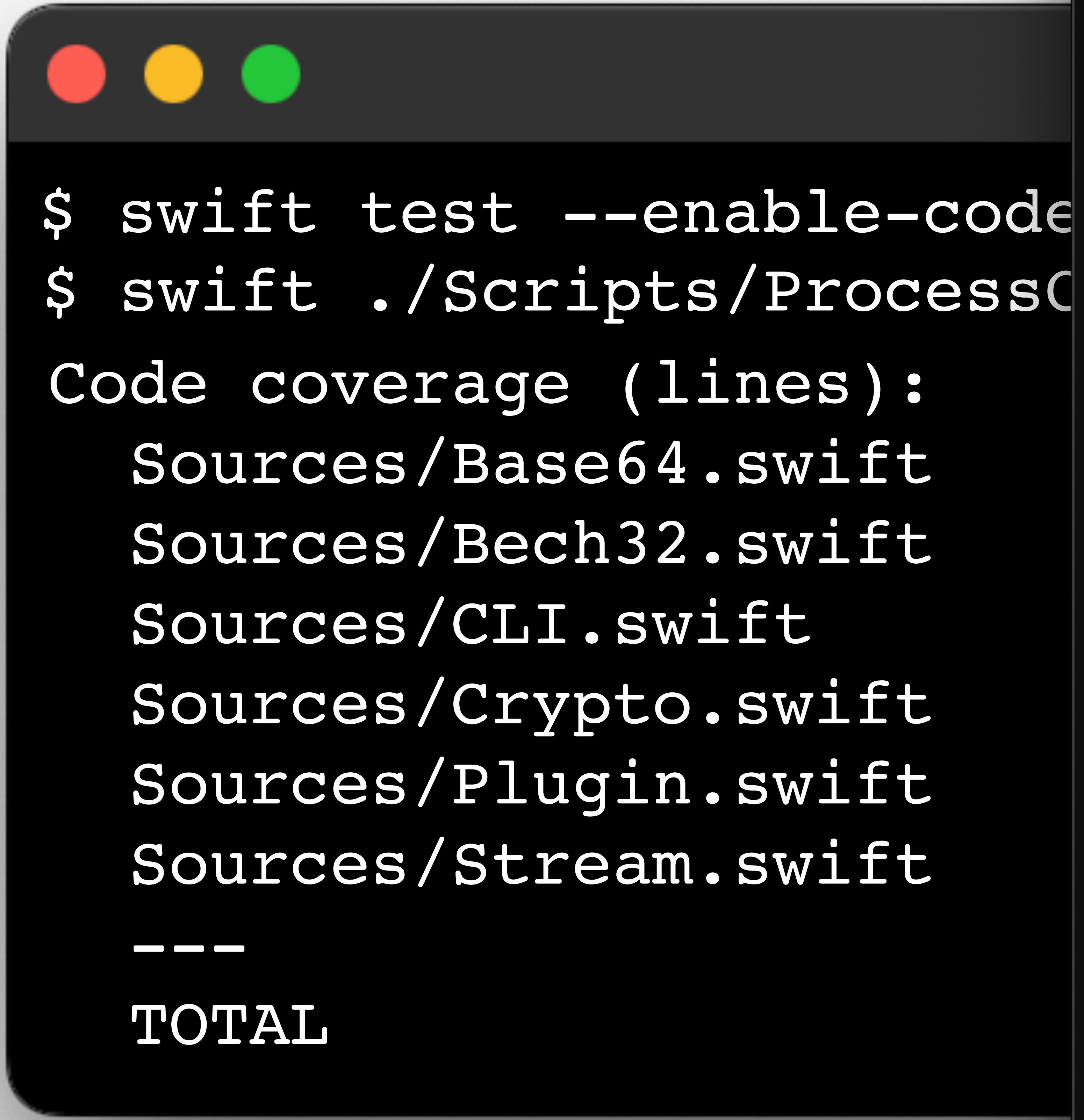
Unit tests



```
$ swift test
Building for debugging...
[1/1] Write swift-version--58304C5D6DBC2206.txt
Build complete! (0.13s)
[87/87] Testing Tests.StanzaTests/testWriteTo
◇ Test run started.
↳ Testing Library Version: 102 (arm64e-apple-macos13.0)
✔ Test run with 84 tests passed after 0.001 seconds.
$
```

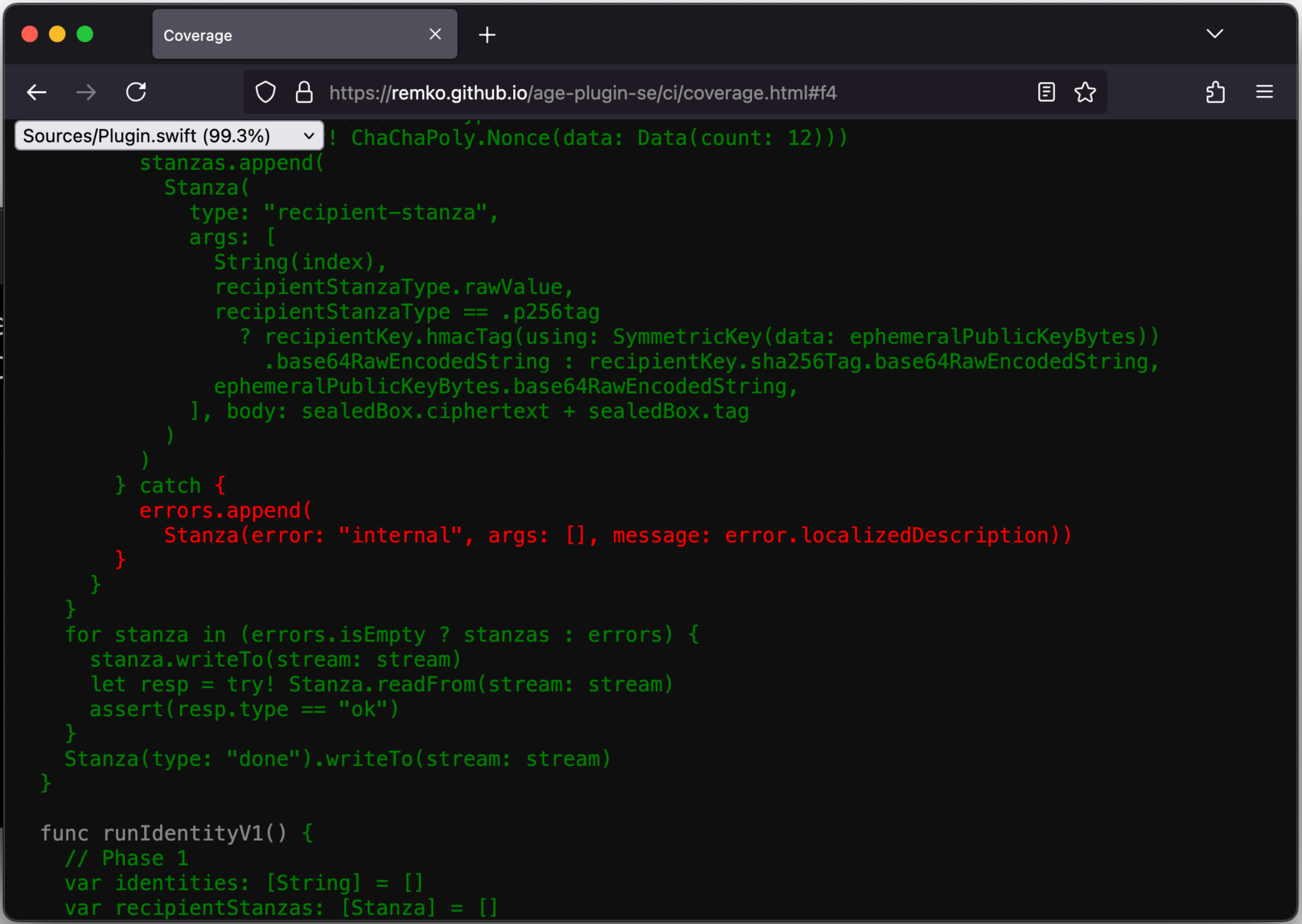
Testing

Coverage



```
$ swift test --enable-code
$ swift ./Scripts/ProcessC
Code coverage (lines):
   Sources/Base64.swift
   Sources/Bech32.swift
   Sources/CLI.swift
   Sources/Crypto.swift
   Sources/Plugin.swift
   Sources/Stream.swift
   ---
   TOTAL
```

Coverage 85%



Coverage
https://remko.github.io/age-plugin-se/ci/coverage.html#f4
Sources/Plugin.swift (99.3%)

```
                       ! ChaChaPoly.Nonce(data: Data(count: 12)))
        stanzas.append(
          Stanza(
            type: "recipient-stanza",
            args: [
              String(index),
              recipientStanzaType.rawValue,
              recipientStanzaType == .p256tag
                ? recipientKey.hmacTag(using: SymmetricKey(data: ephemeralPublicKeyBytes))
                  .base64RawEncodedString : recipientKey.sha256Tag.base64RawEncodedString,
              ephemeralPublicKeyBytes.base64RawEncodedString,
            ], body: sealedBox.ciphertext + sealedBox.tag
          )
        )
      } catch {
        errors.append(
          Stanza(error: "internal", args: [], message: error.localizedDescription))
      }
    }
  }
  for stanza in (errors.isEmpty ? stanzas : errors) {
    stanza.writeTo(stream: stream)
    let resp = try! Stanza.readFrom(stream: stream)
    assert(resp.type == "ok")
  }
  Stanza(type: "done").writeTo(stream: stream)
}

func runIdentityV1() {
  // Phase 1
  var identities: [String] = []
  var recipientStanzas: [Stanza] = []
```

Testing

`swift test` + nvim-coverage + vim-test

```
3 S/Plugin.swift  T/PluginTests.swift
class Plugin {
  init(crypto: Crypto, stream: Stream) {
    self.crypto = crypto
    self.stream = stream
  }

  func generateKey(accessControl: KeyAccessControl, recipientType: RecipientType, now: Date) throws
    -> (String, String)
  {
    if !crypto.isSecureEnclaveAvailable {
      throw Error.seUnsupported
    }
    #if !os(Linux) && !os(Windows)
      let createdAt = now.ISO8601Format()
      var accessControlFlags: SecAccessControlCreateFlags = [.privateKeyUsage]
      if accessControl == .anyBiometry || accessControl == .anyBiometryAndPasscode {
        accessControlFlags.insert(.biometryAny)
      }
      if accessControl == .currentBiometry || accessControl == .currentBiometryAndPasscode {
        accessControlFlags.insert(.biometryCurrentSet)
      }
      if accessControl == .passcode || accessControl == .anyBiometryAndPasscode
        || accessControl == .currentBiometryAndPasscode
      {
        accessControlFlags.insert(.devicePasscode)
      }
      if accessControl == .anyBiometryOrPasscode {
        accessControlFlags.insert(.userPresence)
      }
      var error: Unmanaged<CFError>?
      guard
        let secAccessControl = SecAccessControlCreateWithFlags(
          kCFAllocatorDefault, kSecAttrAccessibleWhenUnlockedThisDeviceOnly,
          accessControlFlags,
          &error)
      else {
        throw error!.takeRetainedValue() as Swift.Error
      }
    #else
      // FIXME: ISO8601Format currently not supported on Linux:
      //   https://github.com/apple/swift-corelibs-foundation/issues/4618
      // This code is only reached in unit tests on Linux anyway
      let createdAt = "1997-02-02T02:26:51Z"
      let secAccessControl = SecAccessControl()
NORMAL  main                                        utf-8  swift  6%  30:7
```

Links

- **age-plugin-se:** https://github.com/remko/age-plugin-se
- **age:** http://age-encryption.org
- **Swift Crypto:** https://github.com/apple/swift-crypto
- **nvim-coverage:** https://github.com/andythigpen/nvim-coverage
- **vim-test:** https://github.com/vim-test/vim-test

Blog posts

- https://mko.re/blog/age-plugin-se/
- https://mko.re/blog/swift-alpine-packaging/
- https://mko.re/blog/swiftpm-coverage/