Refining FreeBSD's Kernel Crypto Framework

John Baldwin

BSDCan

June 2022

Overview

- What is the kernel crypto framework?
- Brief History
- Changes in FreeBSD 13
- Future Work

What is OCF?

- OCF (formerly OpenBSD Crypto Framework, now Open Crypto Framework) is an interface around crypto drivers
- Supports symmetric operations (encryption, authentication)
- Aimed at supporting in-kernel uses (IPsec, KTLS, GELI)
- Does not include “library” APIs for direct software crypto
  - Software ”drivers” sometimes use these APIs

OCF History

- OCF originally ported from OpenBSD to FreeBSD 5.0 by Sam Leffler
- Initial version supported IPsec and userland `/dev/crypto` interface
- Supported asymmetric operations useful for public-key via `/dev/crypto`
- Contemporary ciphers: DES-CBC, MD5, SHA-1, Encrypt-then-Authenticate (EtA)

OCF History

- AES-GCM added as an Authenticated Encryption with Associated Data (AEAD) algorithm in 11.0 by John-Mark Gurney
- OpenSSL's engine using `/dev/crypto` rewritten in 1.1.0 dropping asymmetric support

Motivation

- Worked on porting two crypto drivers from Linux to FreeBSD in 12.0-CURRENT
- Netflix's KTLS used a home-grown crypto layer due to OCF performing poorly

State of OCF in 11

- Sessions described by a linked-list of structures
- Operations described by a linked-list of descriptors
- Drivers would typically walk the linked-lists once to determine cipher vs auth before performing other checks

State of OCF in 11

- AEAD integration was workable, but not always intuitive
  - Separate algorithms for cipher and auth
  - Shared key specified twice for each
  - Auth descriptor described AAD, auth was implicit for cipher descriptor (did not match how EtA worked)
  - MAC/tag verified in driver (EtA verified in consumer)

State of OCF in 11

- IV handling was a kind of tri-state to support random IV's generated by hardware
  - But no drivers in the tree did this
- Crypto session handles shared between drivers and consumers were integer IDs
  - Required drivers to map integer to private pointer for each operation
  - Drivers either used an O(n) loop that did not scale or table with a lock

What To Do?

- From a driver author's perspective, OCF was clunky and obtuse
- OCF seemed overly flexible
  - Arbitrary linked lists of operations
- Modern hardware and use cases (IPsec and TLS) were more constrained
  - Combinations of only two operations (AEAD, MtE, EtA)
  - Fairly consistent layout of buffers (AAD, IV, payload, MAC)

Goal: Make OCF Easier to Work With

- As a driver author, I just wanted something less painful
  - Replace linked lists with flat structures
  - Abstractions around crypto buffers
- Wanted some different flexibility for KTLS
  - Separate input/output buffers
  - Separate AAD buffers
- Was not aiming for performance
  - But reducing complexity might help

One fix in FreeBSD 12

- Crypto session integer ids replaced with opaque `crypto_session_t` type by Conrad Meyer
- Object is a pointer to an allocated structure that contains a pointer to a driver-private structure allocated by the framework
  - `crypto_get_driver_session`
- Replaces locked O(n) lookup in operations with lock-less O(1)

Session Parameters

- New structure describing session parameters: `struct crypto_session_params`
  - Explicit mode (Cipher, Digest, EtA, AEAD, Compression)
  - Algorithms and parameters (e.g. key and MAC lengths)
  - Flags for optional features
  - Session keys if not using per-op keys

Session Probe

- New device driver hook: `cryptodev_probesession`
- Previously, drivers claimed support for algorithms and OCF assumed combinations like EtA were supported
  - Driver would have to fail session creation if unsupported
- Driver's probe hook can check all session parameters
- Probe hook returns a bidding value like device probe routines
  - Differentiate accelerated software (e.g. AES-NI) vs co-processor

Request Structure

- Linked-list of descriptors replaced with new inline members
  - Start and length of AAD, payload, MAC
  - Separate IV buffer
  - Per-request keys
- Crypto buffer fields moved into separate structure (still stored inline)

IV / Nonce

- Tri-state removed by moving random generation out of drivers and into OCF itself
  - IV exists either in separate buffer or inline from driver's perspective
- `crypto_read_iv` helper eliminated copy/pasted code in almost all drivers

Crypto Buffers

- New type: `struct crypto_buffer`
- Supported flat buffers, iovecs, and mbufs initially
- Later extended with array of pages by Alan Somers
- Cursor objects permit iterating over virtual address ranges in drivers and software backends
- New bus_dma method for drivers

Semantic Changes

- AEAD algorithms now have a single algorithm and key
- Drivers validate EtA MAC during decrypt operations like AEAD

Testing Drivers

- Testing coverage for drivers was uneven
  - NIST KAT tests for AES-GCM and AES-CBC added by John-Mark Gurney in 11.0
- Useful to have reproducible, simple KAT tests for all algorithms
- New tool: `cryptocheck`
  - Generates a random buffer, key, and IV/nonce
  - Uses OpenSSL as gold standard
  - Can test various AAD and buffer sizes

Later Extensions

- Added as new flags in session parameters
  - Drivers can opt-in
  - Support in software fallback required
- Separate input and output buffers
- Separate AAD buffer
- IPsec ESN added by Semihalf

New Ciphers Added

- AES-CCM (AEAD) for ZFS
- ChaCha20-Poly1305 and XChaCha20-Poly1305 (both AEAD) for TLS and WireGuard

Things Removed

- Asymmetric cryptography (Gone in 14)
  - Modern OpenSSL doesn't use it
  - Undocumented
  - No software fallback, limited driver support
- Various older ciphers deprecated by industry (Gone in 13)
  - DES, 3DES, Blowfish, MD5 HMAC

Is It Better?

- Less duplicated code in drivers
- Less “busy” work in drivers (don’t have to intuit mode from linked-list)
- KTLS now uses OCF instead of custom framework
- Other developers have added extensions
  - Unmapped I/O for GELI (Alan Somers)
  - IPsec ESN (Semihalf)

Future Work

- Scheduling of async requests (Mark Johnston has done some work here, but semantics still a bit odd)
- Move compression out of OCF
- Split consumer sessions (IPsec) from driver sessions

Q & A

- Thanks to Chelsio Communications and Netflix for sponsoring much of my work
- Questions?