Boosting IPsec and VPN Performance in pfSense Software with IIMB

Leon Dang
Netgate
Australia
ldang@netgate.com

Abstract

OpenCrypto Framework (OCF) in FreeBSD abstracts the underlying cryptography implementation by providing a set of API functions for in-kernel data encryption, decryption and hashing. The drivers of these functions come from a variety of cryptographic providers which range from software instructions to hardware accelerators.

This paper describes the utility of an engine which replaces OCF and its integration of Intel Multi-Buffer Crypto for IPsec Library (IIMB) as the provider for AES-GCM, AES-CBC and ChaCha20-Poly1305 ciphers, offering reduced CPU overhead and significantly improved performance of pfSense supported VPNs. The solution was developed for both x86-64 and ARM-64 architectures.

Keywords—ipsec, vpn, performance

I. Introduction

The OpenBSD Cryptographic Framework was originally imported into FreeBSD to provide an API within the kernel whereby modules and programs can interface with hardware cryptographic features to achieve faster performance than possible in software. The abstraction enabled consumer software to encrypt and decrypt data via the APIs without having to manually deal with the complexities of different platforms; that is, the OCF functions call into underlying cryptographic drivers of the providers which can be a hardware device, such as Intel QuickAssist Technology (QAT), or one implemented in software. OpenBSD Cryptographic Framework evolved into what today is called the OpenCrypto Framework, and is described in FreeBSD's crypto(9) man-page.

As the need for encryption rose for end-user computing due to storage encryption, database security and online connectivity, manufacturers started to integrate technologies to accelerate cryptographic tasks. Many CPUs today come with some form of an accelerated instruction set, such as AES-NI on x86 platforms and Security extensions on ARM, which accelerate specific cipher operations for symmetric key encryption and calculating digests.

In addition, the introduction of the Streaming SIMD Extensions (SSE) on x86 platforms enabled parallel execution of vector operations on datasets in a single instruction. The original SSE register set operated on 128-bit values, and thus is able to carry out 4 x 32-bit calculations in parallel. Subsequent AVX and AVX2 instructions increased the registers to 256-bits and more recently the AVX-512 instruction set was added on both Intel and AMD processors.

While these instructions do not outperform the best hardware accelerators, they are part of the CPU and therefore readily available. They are also adequately fast (or even faster) than the bandwidth required for responsive storage and network traffic uses.

To leverage the vector instructions on their platforms, Intel developed the IPsec Multi-Buffer (IIMB) Crypto Library; a collection of ciphers used in IPsec VPNs written in assembly[1]. The library is used as a software provider for Intel's other products such as DPDK, QAT Engine and FD.io. The label "multi-buffer" means that that some ciphers, such as AES-CBC, can be parallelized by having multiple crypto jobs packed into compute registers that are handled by the CPU's vector instructions.

Intel's IPsec-MB only targets x86 platforms, specifically because of its authorship. For the ARM platform, we added AES functions from ISA-L (used for storage encryption) and ChaCha20-Poly1305 from OpenSSL.

Our intention to pursue this project of replacing OCF with a custom engine was to improve the performance of pfSense VPNs.

II. OpenCrypto Framework

Being designed to generically support varied workloads means that OCF doesn't perform batching of VPN crypto jobs on its own. It instead offloads that task onto crypto drivers and leaves it up to them to optimise the scheduling and completing requests. OCF supports both synchronous and asynchronous job handling, where it generally attempts to issue the request to the driver immediately if not busy, otherwise hands the job off to a separate thread to work on.

When an asynchronous job completes, OCF will put them in an ordered list (for OpenVPN Data Channel Offload and IPsec) and invoke the requestor's callback function. However, OCF must also support clients that do not support asynchronous handling (e.g. WireGuard) and has to complete these jobs immediately. *Figure 1 illustrates the OCF workflow.*

These two factors prevent job parallelization in OCF, which would have benefited subsequent code in the network stack and devices. As a result, even NULL cipher streams, where the payload is not modified, expose the performance weaknesses of OCF, limiting throughout to a fraction of a high-speed network interface. For example, preliminary testing showed only 8gbps throughout over a 25gbps link which could otherwise support a throughout of 22.4gbps using FreeBSD.

A. The Choice for IPsec-MB

We chose IPsec-MB as the encryption library for in-kernel cryptography over others such as OpenSSL because:

1. it implements specific vector instructions for each CPU family, i.e. versions for SSE, AVX, AVX2, AVX-512 and VAES, which are faster than a one-fits-all library; and
2. it features job parallelization with multi-buffer for supported ciphers and multi-blocks for others.

As a comparison on a test host with AVX2, IPsec-MB is able to perform 18.1Gbps of AES-256-CBC,HMAC-SHA-256 operations on 1400 byte blocks versus OpenSSL achieving a maximum of 10.4Gbps using large 16kB blocks.

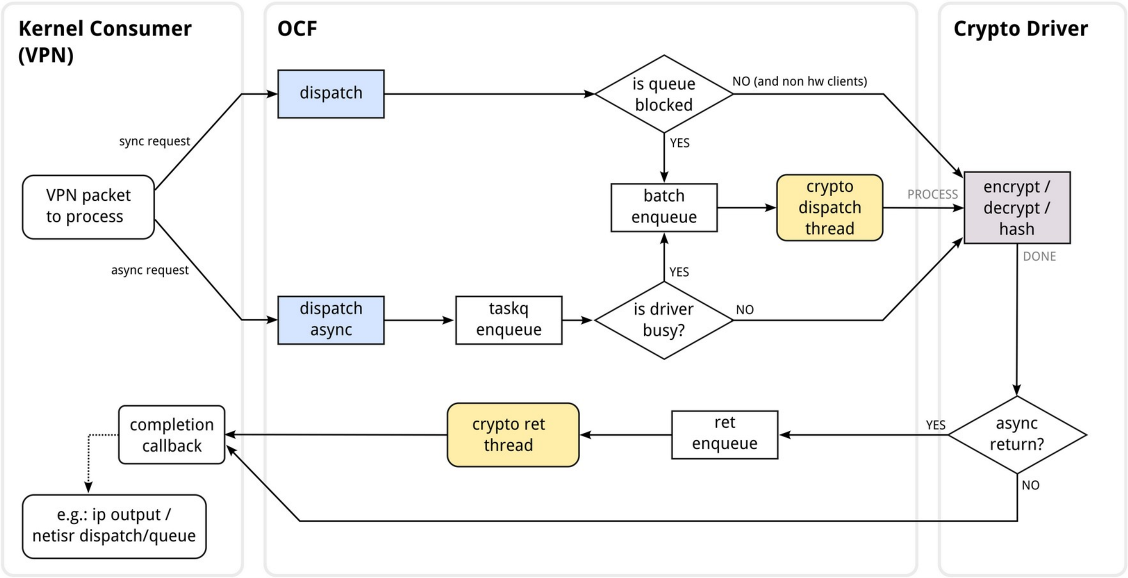


Figure 1: OpenCrypto Framework job model

B. Intel IPsec-MB Job Model

Intel IPsec-MB implements an API which allows the caller to continually feed its job manager, which in turn either puts the request into multiple vector buffers (AES-CBC) or submits it directly to CPU using multiple blocks (AEAD algorithms). For example, CPUs with only SSE are able to perform 4 concurrent AES-CBC operations per encryption cycle and increases to 8 and 16 operations for AVX and VAES respectively.

The use of IPsec-MB has the potential to achieve over 3 times the performance of serial jobs: observed in raw AES-256-CBC+HMAC-SHA-256 serial benchmarks on the test platform reached 5.4gbps whereas the multibuffer version attained 18gbps.

pfSense supports the following ciphers when the IPsec-MB kernel module is used:

- (x86-64 only) AES-CBC with 128, 192, and 256 bit keys
- AES-GCM with 128, 192 (x86-64 only) and 256 bit keys
- ChaCha20-Poly1305 using 256 bit keys

The authenticated encryption with additional data (AEAD) ciphers, AES-GCM and ChaCha20-Poly1305, use single-buffer, multi-block calculations and therefore each job completes one at a time. The technique of keeping the cryptographic code busy to avoid context switching is thus a priority for them.

III. Porting IPsec-MB

The released source code for IPsec-MB is provided as a library for userspace programs to link against. Using it in the kernel by creating the iimb.ko module for FreeBSD required completing two main steps:

1. porting Intel IPsec-MB code to be kernel-ready and built by FreeBSD-native tools; and
2. implementing the IIMB engine which takes over OCF for the supported ciphers; all without modifying the consuming VPN code.

IPsec-MB only supports assembling with NASM, which isn't part of FreeBSD's regular toolchain. One of the goals of creating iimb.ko was to avoid incompatibilities with the kernel and therefore a requirement for building it was to not use any external tools. To achieve this, the first task was converting IPsec-MB's assembly code to be GNU (gas) and Clang compatible. This involved transpiling the NASM-oriented assembly to the required syntax via an intermediate script which takes the NASM preprocessor output and performs the necessary conversions, in essence "`nasm -E input.asm | nasm2gas > output.s`". The resulting assembly code retains Intel syntax, but any surrounding definitions are gas and clang styles. Verifying the conversion accuracy of the 300+ files was achieved by passing all IPsec-MB test vectors, across the different SIMD modes from SSE to AVX-512.

The remaining C code in IPsec-MB was modified to be kernel compatible by changing any userspace APIs into those available in the kernel. These include memory allocation and entering floating point unit context to access the CPU's vector registers.

For ARM 64-bit, the iimb.ko's job management code was derived from the ARM reference implementation of IPsec-MB[2], which was primarily developed to support 3G and 5G telecommunication protocols and therefore lacks both AES and ChaCha-Poly ciphers. ARM-iimb.ko only supports AES-GCM and ChaCha20-Poly1305 which were incorporated from the Intel Intelligent Storage Acceleration Library (ISA-L) and OpenSSL library respecitvely.

A. Taking Control from OCF

An OCF session is one which is associated with the consumer's requested cryptographic algorithm and keyset. The lifecycle of a session can be summarized as:

1. a consumer calls `crypto_newsession()` to create a new session for a particular cipher and/or hash

algorithm. OCF will in turn call the underlying crypto driver to initialize state.

2. for each payload to be encrypted or decrypted, the consumer calls either `crypto_dispatch()` or `crypto_dispatch_async()` and OCF submits the job to the associated driver

3. when the session is no longer needed the consumer calls `crypto_freesession()` to release resources, usually done when a connection terminates or a new security key has been exchanged with a VPN peer.

To avoid any code changes to consumers of FreeBSD's OCF, a crypto engine interface was added to enable IIMB (the engine implemented in iimb.ko) to register itself when its module was loaded. The function callbacks of a crypto engine is referenced from the following data structure:

```
struct crypto_engine {
   /*
    * probe engine to see if it supports the
    * session's crypto parameters
    */
   crypto_probe_fn           ce_probe;

   /*
    * create the session and initialize any
    * internal data structures
    */
   crypto_newsession_fn     ce_newsession;

   /* release the session resources */
   crypto_freesession_fn    ce_freesession;

   /*
    * synchronous dispatch, invoked by
    * crypto_dispatch
    */
   crypto_dispatch_fn       ce_dispatch;

   /*
    * assynchronous dispatch, invoked by
    * crypto_dispatch_async
    */
   crypto_dispatch_async_fn ce_dispatch_async;
};
```

With this change, at new session creation OCF will instead probe iimb.ko to determine if it supports the requested cipher. If it does, IIMB will initialize the session without interacting with any of OCF's regular crypto drivers. For the lifetime of the session, OCF will call into IIMB's dispatch function to handle the consumer's cryptographic request.

Internally, IIMB launches a number of worker threads which varies depending on the number of host CPU cores: 1 thread for less than 4 cores, 2 threads for less than 8 cores, and 4 threads for those with more cores. Each crypto session is bound to a single IIMB thread, which keeps job completions sequential. A larger number of threads does not necessarily correlate with better throughput as there are overhead barriers present in the kernel which hinder any theoretical performance gains, such as context switching between different workloads and processes.

Jobs are batched in IIMB with up to 256 items in each set. The decision to limit the number of threads and bulk job tasks is to balance the requirements of IIMB while allowing other kernel code to run at the same time, thereby never locking up the host from progressing.

IV. Additional Experimentation of WireGuard

The in-kernel if_wg WireGuard implementation launches two sets of taskqueues: one for encrypt and the other for decrypt. Each set of taskqueues launches the same number of threads as the number of CPU units available. Rather than relying on OCF to perform and manage asynchronous jobs, WireGuard submits the packet to its worker threads in a pseudo round-robin order which then make synchronous crypto requests to OCF. In testing, we observed that the round-robin mechanism has a problem where the selection of the CPU for a job is not atomic and therefore results in jobs running on the same CPU repeatedly which goes against the intentions of the WireGuard design to spread the load evenly.

In addition to overriding OCF as the crypto engine, we made two more changes to WireGuard to identify performance differences when utilising IIMB. These were:

- adding AES-256-GCM encryption as the transport cipher to understand the overhead of ChaCha20-Poly1305; and
- adding WireGuard asynchronous dispatch instead of using its own taskqueues.

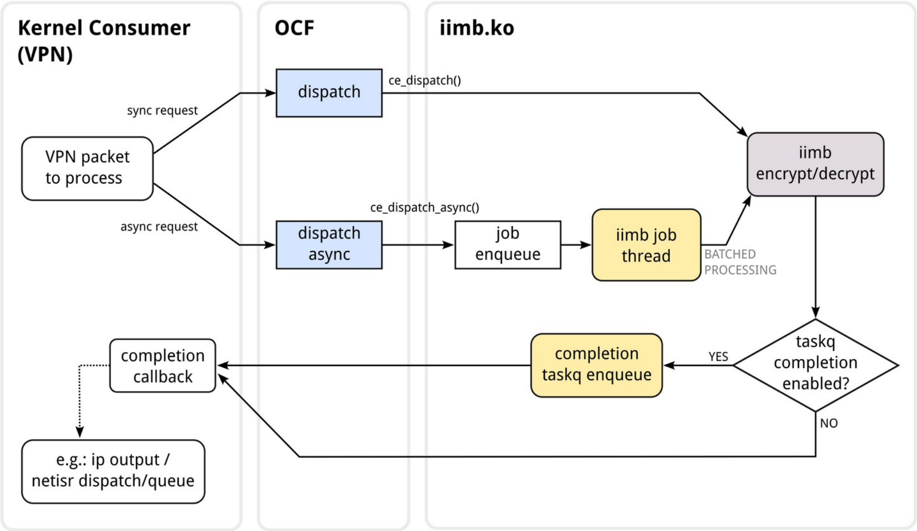


Figure 2: IIMB engine job model

The latter required moving parts of WireGuard's packet processing code responsible for sending mbufs to the input and output functions of the network stack into an OCF-style job completion callback function .

We observed in both cases that performance improved over the released version of WireGuard on FreeBSD; the results are described in the Test Results section following.

V. Test Environment

Performance testing was carried out across different infrastructure setups of VPN peers, using FreeBSD-CURRENT available between November and December 2022. These included both virtual machines and Netgate hardware products.

iperf3 single TCP stream (one client to one server) was used to carry out the benchmarks.

The virtual setup (illustrated in Figure 3) is built on the following platform:

- Hardware:
    - AMD Ryzen 5 5600 6-core/12-thread, 64GB DDR4-3600. Up to AVX2 capable.
    - Mellanox ConnectX-5EN 25Gbps PCIe-3, featuring an embedded switch capable of delivering 50Gbps traffic between the virtual functions.
- Host OS: Ubuntu 22.04 LTS, Kernel 6.03 running KVM and QEMU
- 4 FreeBSD-CURRENT VMs, each with:
    - 4 guest-CPUs, 4GB RAM
    - 2x ConnectX SR-IOV virtual functions in passthrough

Native hardware tests were performed on the following systems running pfSense FreeBSD:

- ARM A53 based Netgate SG-1100 (1GB) and SG-2100 (2GB). These feature Marvell's SafeXcel cryptographic functions in addition to the CPU's AES extensions.
- Intel based Netgate SG-4100 and SG-8200. Both have Intel's QuickAssist technology in addition to AES-NI and SSE3 features.

VPNs configured for testing include:

- OpenVPN 2.6 with DCO (if_ovpn) kernel module
- IPsec-ESP (netipsec)
- WireGuard (dev/wg)

A. *Raw Cryptography Performance*

We measured the per-core cryptographic performance of each system to determine the maximum values that could be achieved if data was sent to it continuously.

Using a combination of `"openssl speed -evp"` and IPsec-MB test units, we observed the numbers below. Note that QuickAssist Technology was disabled on the SG-8200 to obtain CPU-only performance.

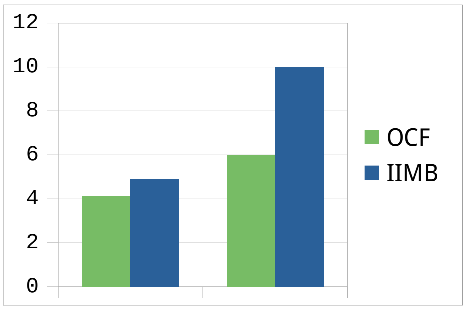
Cipher	Ryzen 5 5600	SG-1100 ARM A53	SG-8200 Intel Atom C3758
AES-128-GCM	40.0Gbps	3.9Gbps	7.7Gbps
AES-256-GCM	41.7Gbps	3.5Gbps	6.4Gbps
AES-256-CBC-HMAC-SHA256	12.0Gbps	3.5Gbps	3.5Gbps
ChaCha20-Poly1305	23.4Gbps	1.5Gbps	3.0Gbps

VI. Test Results

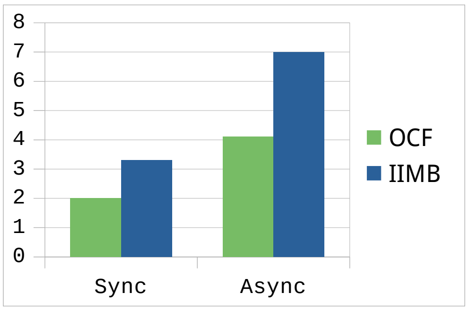
A. *Ryzen-5 5600 Virtual Test Setup*

These are the results from the virtual test setup.

- OpenVPN DCO (AES-256-GCM):
    - OCF: sync: 4.1Gbps async: 6.0Gbps
    - IIMB: sync: 4.9Gbps async: 10Gbps



- OpenVPN DCO (ChaCha20-Poly1305):
    - OCF: sync: 2.0Gbps async: 4.10Gbps
    - IIMB: sync: 3.3Gbps async: 7.0Gbps



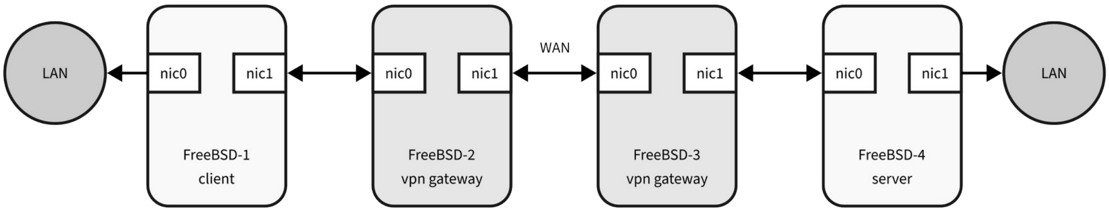
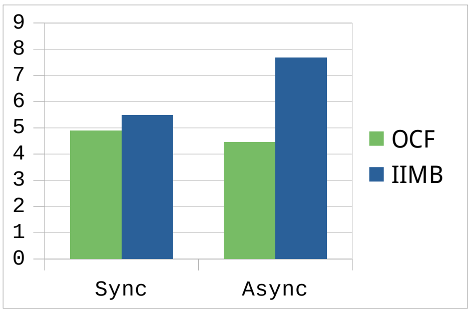


Figure 3: Testbed layout. FBSD-2 and FBSD-3 are VPN gateways. FBSD-1 is the iperf3 client and FBSD-4 the server.

- IPsec-ESP (AES-128-GCM)
  - OCF: sync: 4.88Gbps async: 4.46Gbps
  - IIMB: sync: 5.48Gbps async: 7.68Gbps



- IPsec-ESP (async AES-256-CBC,HMAC-SHA256)
  - OCF: 4.2Gbps
  - IIMB: 5.3Gbps
- WireGuard (sync only)
  - OCF: 4.4Gbps
  - IIMB: 6.0Gbps
- Modified WireGuard with asynchronous crypto using IIMB
  - ChaCha20-Poly1305: 7.0Gbps
  - AES-256-GCM: 10Gbps

Note:

- Unmodified WireGuard uses all CPUs with low idle time (<15%), whereas IPsec and OpenVPN use 3 CPUs for network and encryption workloads (20% idle) and 60% idle.
- Linux native WireGuard attained 7.5Gbps on the same virtual setup.
- Intermittent peak throughput for OpenVPN DCO AES-256-GCM was observed to be 12Gbps, but was not regularly reproduced as it slowed back to 10Gbps.

B. *Netgate SG-1100 - ARM-64*

The SG-1100 was connected to a faster VPN peer to measure its immediate performance. Values are in Megabits per second (Mbps).

VPN	OCF-sync	OCF-async	IIMB-sync	IIMB-async
OpenVPN AES-256-GCM	387	376	460	523
OpenVPN ChaCha20-Poly1305	320	266	451	440
IPsec AES-128-GCM	350	276	450	385
WireGuard (ChaCha20-Poly1305)	330	-	400	-

C. *Netgate SG-8200 – Intel Atom*

A pair of SG-8200s were connected using their primary 10Gbps SFP network port. The iperf3 systems were connected to their secondary 10Gbps port.

The speeds shown below are in Gigabits per second (Gbps).

VPN	OCF-async	IIMB-async	QAT
OpenVPN AES-256-GCM	1.50	2.50	3.18
IPsec AES-128-GCM	1.55	1.64	1.70
WireGuard (ChaCha20-Poly1305)	1.50	2.05	-

* QAT: QuickAssist Technology enabled

VII. Future Work

Additional work and ideas on the next phase of IIMB include:

- Interoperability with QAT and hardware cryptographic devices.
- Adding AES-CBC support to ARM-64.
- Improving netipsec performance.

VIII. Conclusions

We embarked on this development to improve the performance of the VPNs available in pfSense, not only for faster data transfers, but to also assist with connection scaling by increasing crypto throughput and at the same time with reduced CPU consumption.

The IPsec-MB library has shown its viability as the leading CPU-only cryptographic provider on x86-64 based systems. By incorporating it into our IIMB crypto engine, we have been able to achieve significant gains to the various VPNs available in FreeBSD.

The advantage of the IIMB engine for kernel cryptographic workloads extends to ARM-64 when ARM SIMD extensions are available. The improved performance comes not only from the instruction set, but by IIMB densely completing queued jobs.

IX. Acknowledgements

This piece of work would not have been possible without Netgate's sponsorship and support. Thank you to:

- Netgate engineers for reviewing and testing IIMB.
- Intel IPsec-MB developers for their responsiveness.

X. References

1. https://github.com/intel/intel-ipsec-mb Intel(R) Multi-Buffer Crypto for IPsec Library
2. https://gitlab.arm.com/arm-reference-solutions/ipsec-mb ARM reference implementation of IPsec-MB