A new framework of cryptography virtio driver

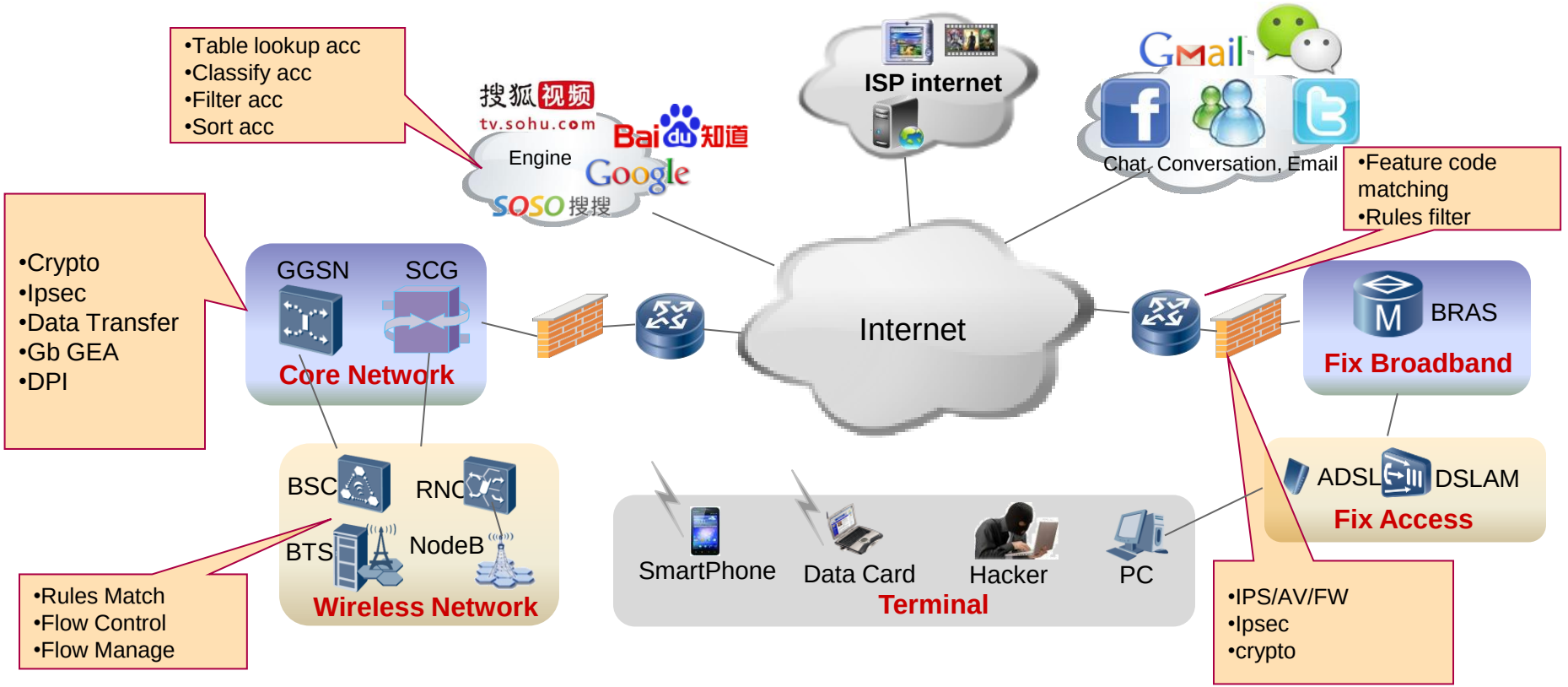
Lei Gong

arei.gonglei@huawei.com

www.huawei.com

HUAWEI TECHNOLOGIES CO., LTD.

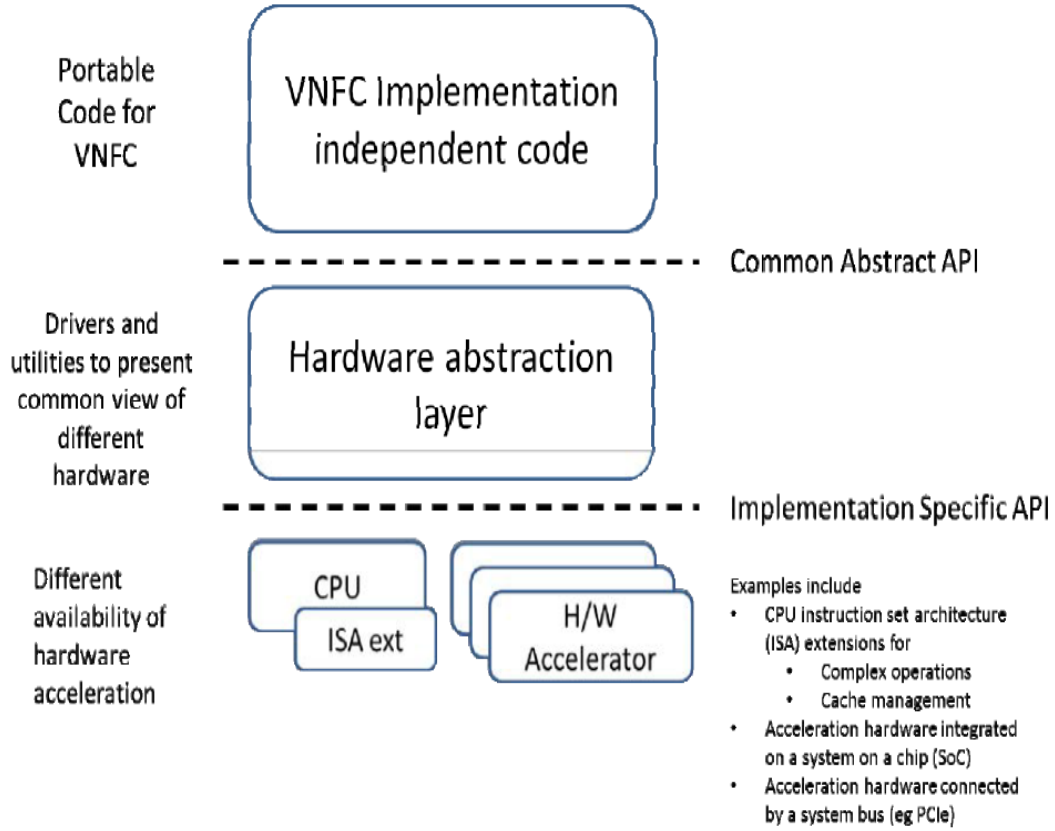
Scenarios of Hardware Acceleration



Acceleration Technologies

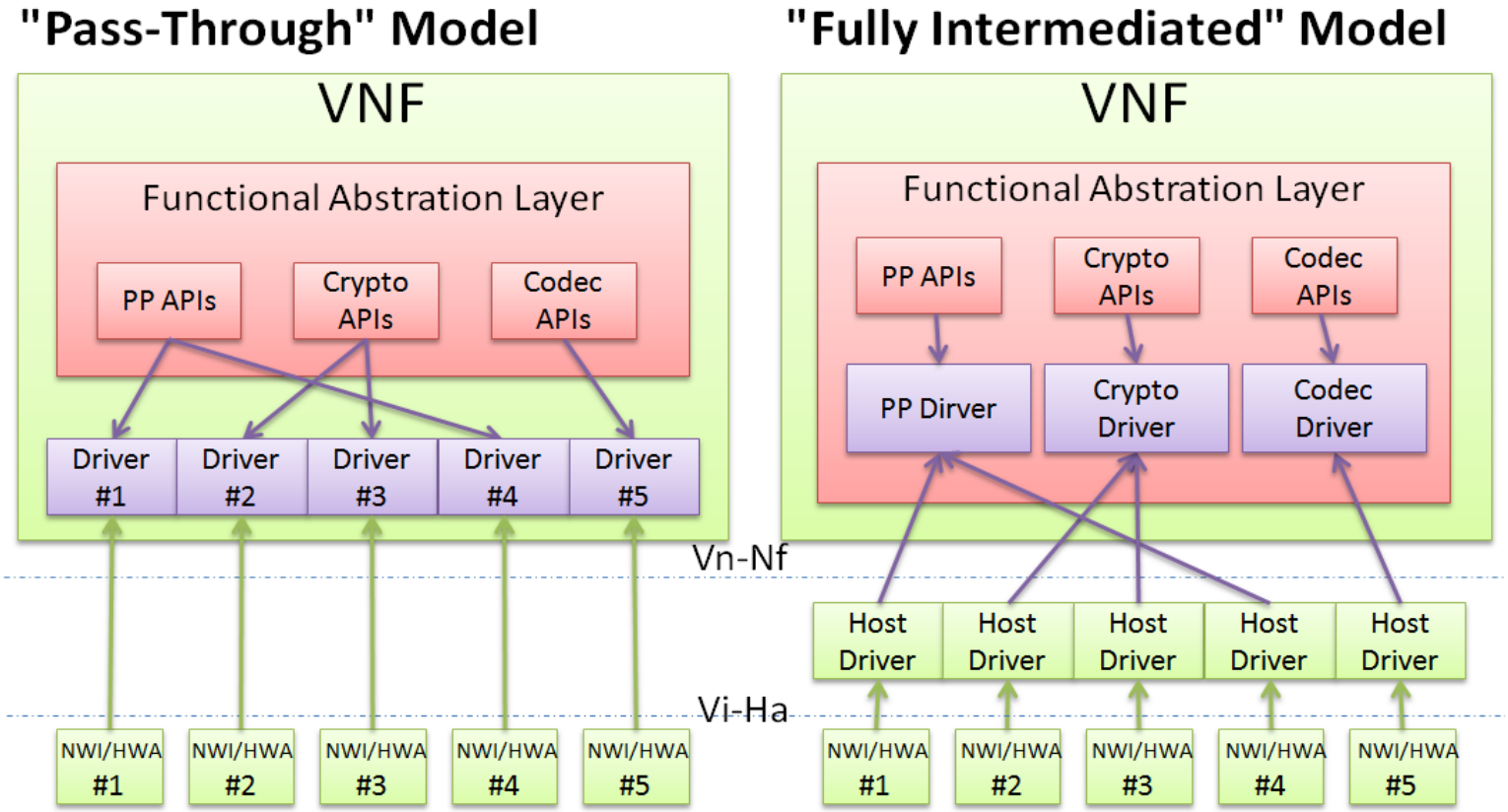
- On Chip HW based acceleration - e.g. AES, CRC, Cryptography, Transcoding.
- Compute Intensive acceleration - e.g. Heterogeneous Computing/GPU.
- Compute Acceleration Pool.
- Network Intensive function acceleration - e.g. NP, FPGA, CPU based support for data plane workload
- Acceleration and data traffic optimization - e.g. NAT, ACL, DPI.
- Storage Acceleration - e.g. Storage Clusters.
- NIC based acceleration - e.g. SR-IOV, vSwitch Bypass, Network Intensive processing.

Abstract model of Accelerators (NFV)



Note: the original figure forwarded from ETSI GS NFV-INF 003 V1.1.1 (2014-12)

Data Plane Acceleration Proposal

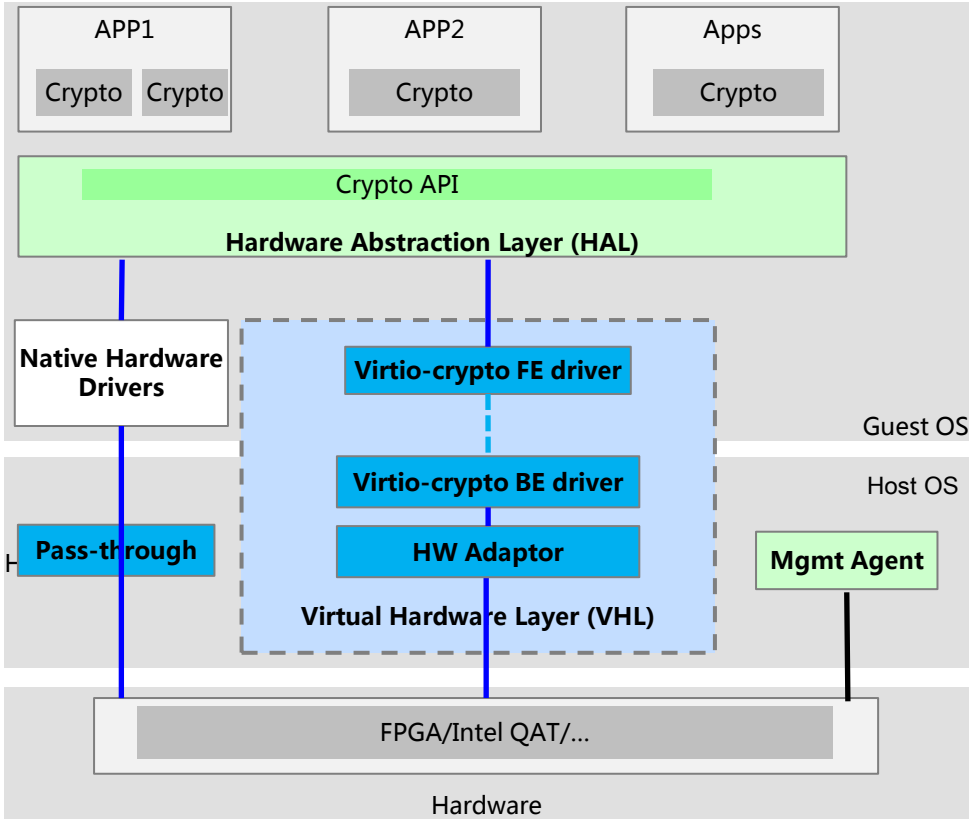


https://wiki.opnfv.org/dpacc/dpacc_project_proposal

Why Virtio-crypto?

- Programmability
- Portability
- Scalability
- Hardware agnostic

Virtualization of Cryptography Accelerator



1. **HAL**

   Provide acceleration APIs and runtimes

2. **VHL**

   Provide virtual accelerators:

   1) virito-crypto FE driver

   2) virtio-crypto BE driver

   3) HW Adaptor : support different crypto accelerators

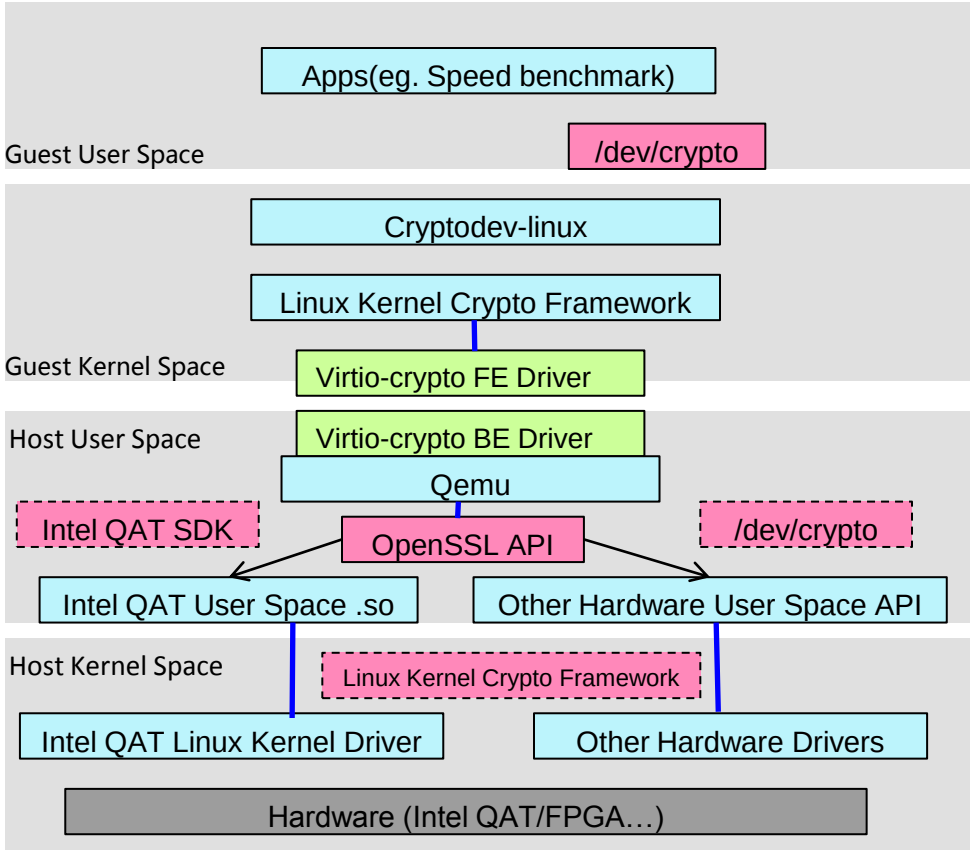
3. **Pass-through**

   Accelerator pass-through

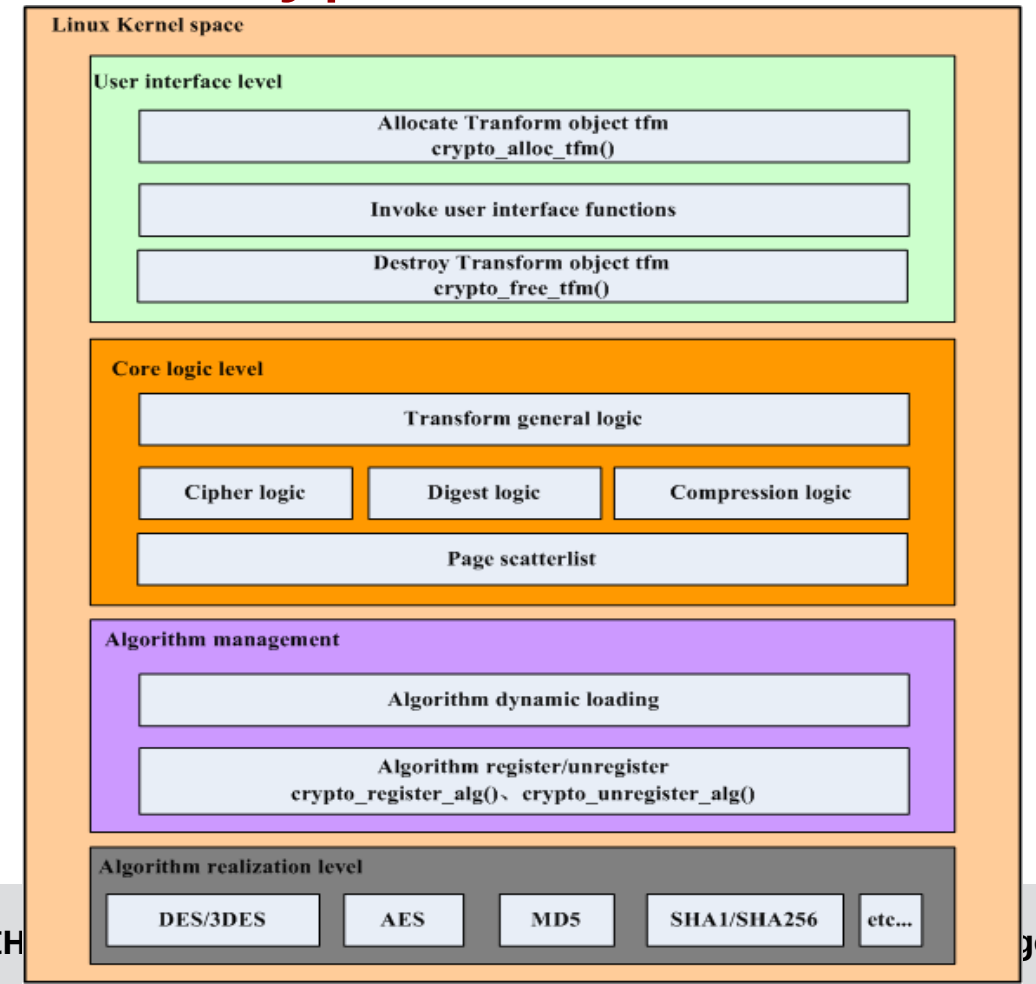
4. **Mgmt Agent**

   Accelerator management

Flow of Virtio-crypto Prototype



Linux Kernel Crypto Framework



The kernel Crypto API

- A cryptography framework in the Linux kernel
- Can do Cipher, Hash, Compress, RNG,. . .
- Used by:
  - ✓Network stack: IPsec, . . .
  - ✓Device Mapper: dm-crypt, RAID, . . .
  - ✓Userland Accessing:
    - ✓AF_ALG
    - ✓Cryptodev
- Maillist: linux-crypto@vger.kernel.org

AF_ALG introduction

* Supports CIPHER, HASH
* Socket-based interface

+ In-kernel code for years
+ Inherently asynchronous
- OpenSSL has out-of-tree engine for AF ALG
- GnuTLS does not have support for AF ALG
- Not many examples
- Higher latency

Cryptodev introduction

* Supports CIPHER, HASH, AEAD
* Uses character device interface

+ Compatible with OpenBSD /dev/crypto
+ API compatible, not OpenBSD code
+ OpenSSL has engine for cryptodev
+ GnuTLS has support for cryptodev
+ Has nice examples
+ Lower latency
- Out of kernel tree code (for years)
- Adds arbitrary IOCTLs

Cryptodev howto

Cryptodev usage pattern:

a) int cfd = open("/dev/crypto");
b) Fill in common struct cryptodev ctx
c) Fill in struct crypt op
d) Pass struct crypt op into kernel via ioctl()
e) Retrieve results
f) close(cfd);

Virtio-crypto BE driver

* Emulate virtio-crypto devices in Qemu:

  Command line: -device virtio-crypto-pci,id=crypto0

* Support different backend drivers:

  OpenSSL, Cryptodev, Intel QAT SDK

* Support multiple virtio devices for each VM
* Fit Virtio-1.0 spec
* Cooperate with the virtio-crypto driver in guest

Virtio-crypto device

```
# lspci –v
[skip]
00:05.0 Unclassified device [00ff]: Red Hat, Inc Device 103f
        Subsystem: Red Hat, Inc Device ffff
        Flags: bus master, fast devsel, latency 0, IRQ 34
        I/O ports at c000 [size=512]
        Memory at febd3000 (32-bit, non-prefetchable) [size=4K]
        Capabilities: [40] MSI-X: Enable+ Count=2 Masked-
        Kernel driver in use: virtio-pci
        Kernel modules: virtio_pci
```

Virtio-crypto FE driver

* As a hardware crypto device
* Support different algorithms:

  Cipher, Hash, AEAD
* Support multiple virtio devices for each VM
* Fit Virtio-1.0 spec

Virtio-crypto module

```
# modinfo virtio-crypto
filename:       virtio-crypto.ko
author:         Gonglei <arei.gonglei@huawei.com>
license:        GPL
description:    Virtio crypto device driver
srcversion:     B5B95C74287DAE3AB7C134D
alias:          virtio:d0000FFFFv*
depends:        virtio_ring,virtio
vermagic:       3.0.76-0.11 SMP mod_unload modversions
parm:           virtio_crypto_verbosity:0: normal, 1: verbose, 2:
debug (int)
```

Register algorithms

```
static struct crypto_alg virtio_crypto_algs[] = { {
        .cra_name = "cbc(aes)",
        .cra_driver_name = "virtio_crypto_aes_cbc",
        .cra_priority = 4001,
        .cra_flags = CRYPTO_ALG_TYPE_ABLKCIPHER | CRYPTO_ALG_ASYNC,
        .cra_blocksize = AES_BLOCK_SIZE,
        .cra_ctxsize  = sizeof(struct virtio_crypto_ablkcipher_ctx),
        .cra_alignmask = 0,
        .cra_module = THIS_MODULE,
        .cra_type = &crypto_ablkcipher_type,
        .cra_init = virtio_crypto_ablkcipher_init,
        .cra_exit = virtio_crypto_ablkcipher_exit,
        .cra_u = {
          .ablkcipher = {
                                .setkey = virtio_crypto_ablkcipher_setkey,
                                .decrypt = virtio_crypto_ablkcipher_decrypt,
                                .encrypt = virtio_crypto_ablkcipher_encrypt,
                                .min_keysize = AES_MIN_KEY_SIZE,
                                .max_keysize = AES_MAX_KEY_SIZE,
                                .ivsize = AES_BLOCK_SIZE,
                        },
        },
},…
```

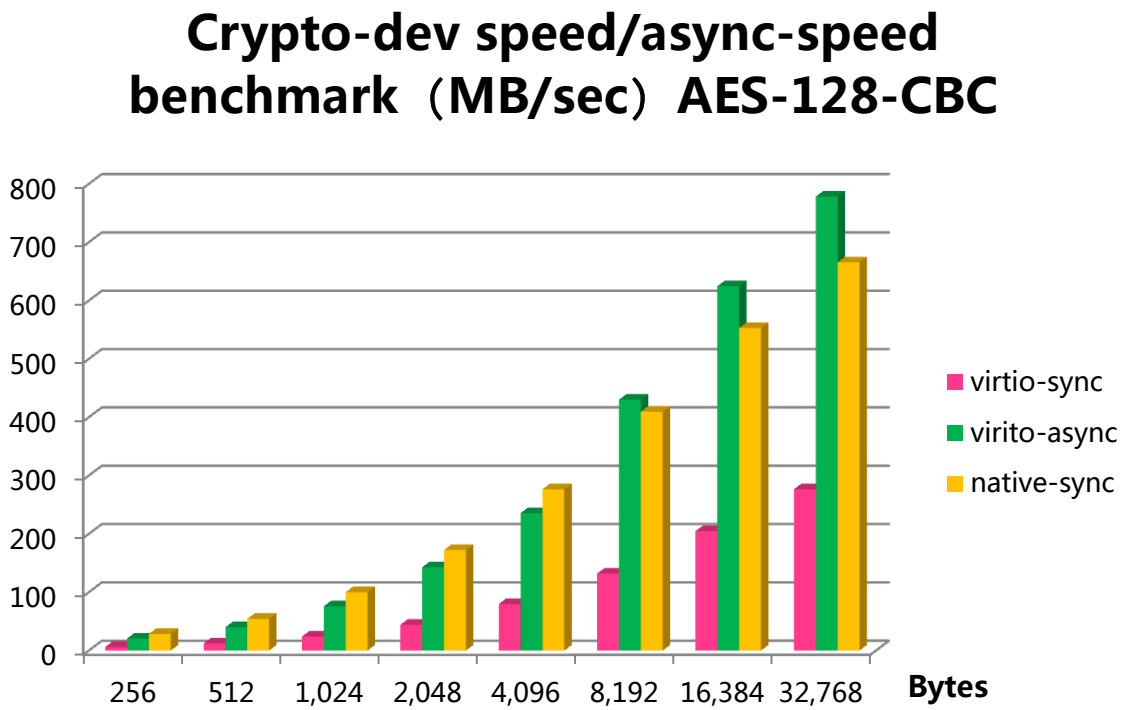
Virtio-crypto synchronous running

```
#cryptodev-linux-1.7 # ./tests/cipher -
requested cipher CRYPTO_AES_CBC, got cbc(aes) with driver virtio_crypto_aes_cbc
AES Test passed
requested cipher CRYPTO_AES_CBC, got cbc(aes) with driver virtio_crypto_aes_cbc
requested cipher CRYPTO_AES_CBC, got cbc(aes) with driver virtio_crypto_aes_cbc
Test passed
```

Virtio-crypto asynchronous running

```
#cryptodev-linux-1.7 # ./tests/async_cipher -
cryp1 written out
cryp2 written out
cryp1 + cryp2 successfully read
result 1 passed
result 2 passed
AES Test passed
running test_crypto
test_crypto: got the session
test_crypto: data encrypted
test_crypto: session finished
test_crypto: got new session
test_crypto: data encrypted
Test passed
```

Performance



- **Hardware**
  1) Intel(R) Xeon(R) CPU E5-2620 v3 @ 2.40GHz
  2) Intel QAT Coleto Creek PCIe DH895xCC SKU2
- **Software**
  Guest: Suse11.3 with 8 GB memory, 8vcpu
  Host: KVM 3.12, QEMU 2.4-rc3

TODO:

1. Performance optimization:
   virtio-crypto-dataplane, batch processing, etc.
2. Other crypto algorithms support
3. Virtio-crypto upstream:
   virtio-crypto spec, virtio-crypto code…

Q/A

Thank you!