Release Notes for
FreeBSD package version QAT1.7.B.3.4.0-00006.tar.gz (September 2019)

The documentation for this production release is provided in this note. It can be read in conjunction with these documents:

- Intel® Communications Chipset 8925 to 8955 Series Software - Programmer’s Guide
- Intel® Communications Chipset 89xx Series Software for Linux – Getting Started Guide
- Intel® QuickAssist Technology Software for Linux* - Programmer’s Guide - HW version 1.7
- Intel® QuickAssist Technology Software for Linux* - Getting Started Guide - HW version 1.7

Contents
Revision History ................................................. 2
Related Documentation............................................... 2
Release Overview.................................................. 3
  The release delivers the following features:......................... 3
  Environmental Assumptions:...................................... 3
  Limitations with this production release:............................ 3
Package Versions .................................................. Error! Bookmark not defined.
MD5 Checksum Information ........................................ 4
Licensing for FBSD* Acceleration Software ........................ 4
QuickAssist Driver Package Installation on FreeBSD Environment.................................................. 4
  Compiling the Driver .................................................. 4
  Compiling and execute performance sample code .................... 5
  Uninstalling the driver ................................................ 6
  HKDF functional sample code compilation and execution ............ 6
API’s in this release: .................................................. 6
  HKDF Functions:....................................................... 6
  HKDF Parameters:..................................................... 7
  HKDF use case......................................................... 9
  Instance configuration and memory allocation ....................... 9
  HKDF Extract Expand operation ..................................... 10
  HKDF Extract Expand Label operation ................................ 10
HKDF Extract Expand Label and Sublabels operation .................................................. 10
Perform HKDF operation .................................................................................................. 11
Known Issues ........................................................................................................................ 11
QATE-31888 – Possible performance degradation ................................................................. 11
QATE-30931 - Process exit with orphan rings when spawning multiple processes .......... 12
QATE-30360 - Full device pass-through not available on KVM guests ................................. 12
QATE-5092 - AES-XTS does not support buffers sizes that are not a multiple of 16B............ 12
QATE-7325 - AES-GCM operation with zero length plain text results in an incorrect tag result ..... 12
QATE-33751 - Library and driver do not support devices enumerated in a PCI domain different than 0 .......................................................................................................................... 13
QATE-39335 - Compression instances do not work on Virtual Machine with Linux Host QAT driver without CnVnR support ........................................................................................................ 13
QATE-40359 - Multiprocess 32 with LimitDevAccess = 0 fails with OpenSSL Speed tests .......... 13
QATE-39216 - Kasumi test duration issue ............................................................................. 13
QATE-41846 - GEN - QAT API submissions with bad addresses that trigger DMA to invalid or unmapped addresses can cause a platform hang .......................................................... 14
QATE-41745 - Segmentation fault when using inputs on QUAD word boundaries ............... 14
QATE-41486 - Misleading message observed in dmesg on LBG device with LimitDevAccess = 1 set in configuration file ........................................................................................................ 14

Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>September 2019</td>
<td>001</td>
<td>Initial 3.4.0 Product release</td>
</tr>
</tbody>
</table>

Related Documentation

<table>
<thead>
<tr>
<th>Document Name</th>
<th>Reference Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel® QuickAssist Technology API Programmer’s Guide</td>
<td>330684</td>
</tr>
<tr>
<td>Intel® QuickAssist Technology Cryptographic API Reference Manual</td>
<td>330685</td>
</tr>
<tr>
<td>Intel® QuickAssist Technology Data Compression API Reference Manual</td>
<td>330686</td>
</tr>
<tr>
<td>Intel® QuickAssist Technology Performance Optimization Guide</td>
<td>330687</td>
</tr>
</tbody>
</table>
Release Overview

The QAT R3.4.0 FreeBSD package is provided as production quality release and is therefore intended to be used in a production environment.

This software release is intended for platforms that contain:

- Intel® C62x Chipset
- Intel Atom® C3000 processor product family
- Intel® QuickAssist Adapter 8960/Intel® QuickAssist Adapter 8970 (formerly known as “Lewis Hill”)
- Intel® Communications Chipset 8925 to 8955 Series

The release delivers the following features:

- HMAC-based Extract-and-Expand Key Derivation Function (HKDF)

Environmental Assumptions:

The following assumptions are made with regard to the deployment environment:

- The driver object/executable file on disk should be protected using the normal file protection mechanisms so that it is writable only by trusted users, for example, a privileged user or an administrator.
- The public key firmware image on disk should be protected using normal file protection mechanisms so that it is writable only by trusted users, for example, a privileged user or an administrator.
- The QAT device should not be exposed (via SR-IOV) to untrusted guests.
- The QAT device should not be exposed (via the "user space direct" deployment model) to untrusted users.
- DRAM is considered to be inside the trust boundary. The normal memory protection schemes provided by the Intel® architecture processor and memory controller, and by the operating system, prevent unauthorized access to these memory regions.
- Persistent keys were not considered, but the storage media are also considered inside the cryptographic boundary. For the details please refer to: Intel® QuickAssist Technology (Intel® QAT) Software for Linux.

Limitations with this production release:

- FreeBSD as a host environment with QAT is not supported
- Any version of FreeBSD other than 11.2 is not supported
- Symmetric session update feature is not supported
• NRBG is not supported
• The HKDF operational data has to be allocated with USDM to be pinned in physical memory

There are known issues with this release of the driver as described in .

**MD5 Checksum Information**

The table below gives MD5 checksum information.

<table>
<thead>
<tr>
<th>Package</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>QAT Package</td>
<td>QAT1.7.B.3.4.0-00006.tar.gz</td>
</tr>
</tbody>
</table>

**Licensing for FBSD® Acceleration Software**

The acceleration software is provided under the following license as listed in the table below. When using or redistributing dual-licensed components, you may do so under either license.

<table>
<thead>
<tr>
<th>Component</th>
<th>Licence</th>
<th>Directories</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Space Library</td>
<td>BSD</td>
<td>./quickassist/build_system</td>
</tr>
<tr>
<td></td>
<td></td>
<td>./quickassist/include</td>
</tr>
<tr>
<td></td>
<td></td>
<td>./quickassist/lookaside</td>
</tr>
<tr>
<td></td>
<td></td>
<td>./quickassist/utilities/osal</td>
</tr>
<tr>
<td>Kernel space driver</td>
<td>BSD</td>
<td>./quickassist/qat/drivers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>./quickassist/utilities/adf_ctl</td>
</tr>
<tr>
<td>User Space DMA-able</td>
<td>BSD</td>
<td>./quickassist/utilities/libusdm</td>
</tr>
<tr>
<td>Memory Driver</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Libcrypto</td>
<td>OpenSSL</td>
<td>./quickassist/utilities/osal/src/linux/user_s</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pace/openssl</td>
</tr>
<tr>
<td>CPM Firmware</td>
<td>Redistribution</td>
<td>./quickassist/qat/fw</td>
</tr>
</tbody>
</table>

**QuickAssist Driver Package Installation on FreeBSD Environment**

User must have root privileges to perform the following.

**Compiling the Driver**

**Step 1:** Copy package onto the system.
Step 2: Extract package.

```
# cd /root/
# mkdir QAT
# cd QAT
# tar -xzomf <path_to>/QAT1.7.B.3.4.0-00006.tar.gz
```

Step 3: Set network proxy (if required)

```
# setenv http_proxy http://<proxy_server>:<proxy_port>
```

Step 4: Install dependencies

1. gmake:
   ```
   # cd /usr/ports/devel/gmake
   # make config-recursive
   # make install
   ```
2. Boost
   ```
   # pkg install boost-all
   ```
3. Automake & autoconf
   ```
   # pkg install automake
   # pkg install autoconf
   ```
4. Bash
   ```
   # pkg install bash
   ```
5. pkg-config
   ```
   # cd /usr/ports/devel/pkgconf/
   # make
   # make install
   ```

Step 5: Setup the environment to build driver.

```
# cd /root/QAT/
#.configure
```

Step 6: Build and install driver

```
# gmake
# gmake install
```

**Compiling and execute performance sample code**

Step 1: Build application

```
# cd /root/QAT/
# gmake samples-install
```

Step 2: Run application

```
# cd ./build
#. /cpa_sample_code signOfLife=1 <- sign of life tests
```
Uninstalling the driver

Step 1: Bring down the driver
# cd /root/QAT/build
# ./adf_ctl down

Step 2: Uninstall driver
# cd /root/QAT/
# gmake uninstall

HKDF functional sample code compilation and execution

HKDF functional sample code could be built in following ways:

1. Standalone HKDF functional sample built:
   # cd /root/QAT
   # setenv ICP_ROOT `pwd`
   # setenv ICP_OS freebsd
   # setenv WITH_CMDRV 1
   # cd /quickassist/lookaside/access_layer/src/sample_code/functional/sym/hkdf_sample/
   # gmake
   # ./hkdf_sample

2. Build all functional samples:
   # cd /root/QAT
   # setenv ICP_ROOT `pwd`
   # setenv ICP_OS freebsd
   # setenv WITH_CMDRV 1
   # cd /quickassist/lookaside/access_layer/src/sample_code/
   # gmake func
   # cd ./functional/build
   # ./hkdf_sample

The sample is located:
# ./sym/hkdf_sample/hkdf_sample

API’s in this release:
This is a list of the new HKDF API’s provided in this release.

HKDF Functions:
Provides HKDF key generation interface.
CpaStatus
cpaCyKeyGenTls3(const CpaInstanceHandle instanceHandle,
    const CpaCyGenFlatBufCbFunc pKeyGenCb,
    void *pCallbackTag,
    const CpaCyKeyGenHKDFOpData *pKeyGenTlsOpData,
    CpaCyKeyHKDFCipherSuite cipherSuite,
    CpaFlatBuffer *pGeneratedKeyBuffer);

HKDF Parameters:

CpaInstanceHandle instanceHandle – handle to the crypto instance.

const CpaCyGenFlatBufCbFunc pKeyGenCb – pointer to completion callback function.

void *pCallbackTag – opaque user data that will be passed unchanged in the callback.

typedef enum _CpaCyKeyHKDFOp
{
    CPA_CY_HKDF_KEY_EXTRACT = 12,
    /**< HKDF Extract operation
     * Corresponds to RFC5869 section 2.2, step 1 "Extract" */
    CPA_CY_HKDF_KEY_EXPAND,
    /**< HKDF Expand operation
     * Corresponds to RFC5869 section 2.3, step 2 "Expand" */
    CPA_CY_HKDF_KEY_EXTRACT_EXPAND,
    /**< HKDF operation
     * This performs HKDF_EXTRACT and HKDF_EXPAND in a single
     * API invocation. */
    CPA_CY_HKDF_KEY_EXPAND_LABEL,
    /**< HKDF Expand label operation for TLS 1.3
     * Corresponds to RFC8446 section 7.1 Key Schedule definition for
     * HKDF-Expand-Label, which refers to HKDF-Expand defined in RFC5869. */
    CPA_CY_HKDF_KEY_EXTRACT_EXPAND_LABEL
    /**< HKDF Extract plus Expand label operation for TLS 1.3
     * Corresponds to RFC5869 section 2.2, step 1 "Extract" followed by
     * RFC8446 section 7.1 Key Schedule definition for
     * HKDF-Expand-Label, which refers to HKDF-Expand defined in RFC5869. */
} CpaCyKeyHKDFOp;

typedef struct _CpaCyKeyGenHKDFExpandLabel
{
    Cpa8U label[CPA_CY_HKDF_KEY_MAX_LABEL_SZ];
/**< HKDFLabel field as defined in RFC8446 sec 7.1.
 */
Cpa8u labelLen;
/**< The length, in bytes of the label */
Cpa8u sublabelFlag;
/**< mask of sublabels to be generated.
 * This flag is composed of zero or more of:
 * CPA_CY_HKDF_SUBLABEL_KEY
 * CPA_CY_HKDF_SUBLABEL_IV
 * CPA_CY_HKDF_SUBLABEL_RESUMPTION
 * CPA_CY_HKDF_SUBLABEL_FINISHED
 */
} CpaCyKeyGenHKDFExpandLabel;

typedef struct _CpaCyKeyGenHKDFOpData
{
    CpaCyKeyHKDFOp hkdfKeyOp;
    /**< Keying operation to be performed. */
    Cpa8u secretLen;
    /**< Length of secret field */
    Cpa16U seedLen;
    /**< Length of seed field */
    Cpa16U infoLen;
    /**< Length of info field */
    Cpa16U numLabels;
    /**< Number of filled CpaCyKeyGenHKDFExpandLabel elements */
    Cpa8u secret[CPA_CY_HKDF_KEY_MAX_HMAC_SZ];
    /**< Seed or PRK material */
    Cpa8u seed[CPA_CY_HKDF_KEY_MAX_HMAC_SZ];
    /**< Input Key Material */
    Cpa8u info[CPA_CY_HKDF_KEY_MAX_INFO_SZ];
    /**< info field */
    CpaCyKeyGenHKDFExpandLabel label[CPA_CY_HKDF_KEY_MAX_LABEL_COUNT];
    /**< array of Expand Label structures */
} CpaCyKeyGenHKDFOpData;

const CpaCyKeyGenHKDFOpData *pKeyGenTlsOpData – input structure with data needed to
perform TLS key generation operation. This structure has to be allocated with USDm to be
pinned in physical memory.

typedef enum _CpaCyKeyHKDFCipherSuite
{
    CPA_CY_HKDF_TLS_AES_128_GCM_SHA256 = 1,
    CPA_CY_HKDF_TLS_AES_256_GCM_SHA384,
    CPA_CY_HKDF_TLS_CHACHA20_POLY1305_SHA256,
}
```c
CPA_CY_HKDF_TLS_AES_128_CCM_SHA256,
CPA_CY_HKDF_TLS_AES_128_CCM_8_SHA256
} CpaCyKeyHKDFCipherSuite;
```

CpaCyKeyHKDFCipherSuite cipherSuite – cipher suite according to TLS v1.3. This value is used to infer the sizes of the key and iv sublabel.

CpaFlatBuffer *pGeneratedKeyBuffer – handle to output buffer with generated data. Caller MUST allocate sufficient buffer to hold the key generation output.

**Note:** The Intel® QAT API version number is different from the software package version number.

For details on any changes to the Intel® QuickAssist Technology APIs, refer to the Revision History pages in the following API reference manuals:
- Intel® QuickAssist Technology Cryptographic API Reference Manual
- Intel® QuickAssist Technology Data Compression API Reference Manual

**HKDF use case**

This is sample code that demonstrates usage of the symmetric API, and specifically using this API to perform a HKDF based operations. It performs HKDF Extract and Expand, and Extract and Expand Label operation without and with sublabels (KEY and IV).

Note this example is simplified to demonstrate the basics of how to use the API and how to build the structures required. This example does not demonstrate the optimal way to use the API to get maximum performance for a particular implementation.

This sample is located in:
/quickassist/lookaside/access_layer/src/sample_code/functional/sym/hkdf_sample

**Instance configuration and memory allocation**

- Cryptographic service instances are discovered and started in the same way and using the same API as the traditional symmetric use cases.
- If the instance is polled start the polling thread. Note that how the polling is done is implementation-dependent.
- Allocate memory for HKDF operation data:
  ```c
  pOpData = qaeMemAllocNUMA(sizeof(CpaCyKeyGenHKDFOpData),
  instanceInfo2.nodeAffinity,
  BYTE_ALIGNMENT_64);
  ```

  **NOTE:** This structure has to be allocated with USDM to be pinned in physical memory
Allocate memory for HKDF output data. Output data is CpaFlatBuffer type:

```
PHYS_CONTIG_ALLOC(&pHkdfData, hkdfDataSize);
```

**HKDF Extract Expand operation**

To perform Extract Expand operation, in `CpaCyKeyGenHKDFOpData` structure `hkdfKeyOp` have to be set to `CPA_CY_HKDF_KEY_EXTRACT_EXPAND`. Length of `seedLen`, `secretLen`, `infoLen` have to be provided and all data copied into `seed`, `secret`, `info` tables.

```
pOpData->hkdfKeyOp = CPA_CY_HKDF_KEY_EXTRACT_EXPAND;
pOpData->seedLen = sizeof(ikm);
memcpy(pOpData->seed, ikm, pOpData->seedLen);
pOpData->secretLen = sizeof(slt);
memcpy(pOpData->secret, slt, pOpData->secretLen);
pOpData->infoLen = sizeof(inf);
memcpy(pOpData->info, inf, pOpData->infoLen);
```

**HKDF Extract Expand Label operation**

To perform Extract Expand Label operation, in `CpaCyKeyGenHKDFOpData` structure `hkdfKeyOp` have to be set to `CPA_CY_HKDF_KEY_EXTRACT_EXPAND_LABEL`. Length of `seedLen`, `secretLen`, have to be provided and all data copied into `seed`, `secret` tables. Number of labels has to be set in `numLabels` field. Next step is setting `label[0].labelLen` and copy label data into `label[0].label` table. A the end `label[0].sublabelFlag` filed is set to `0x00` to disable generating sublabels.

```
pOpData->hkdfKeyOp = CPA_CY_HKDF_KEY_EXTRACT_EXPAND_LABEL;
pOpData->seedLen = sizeof(seed_label);
memcpy(pOpData->seed, seed_label, sizeof(seed_label));
pOpData->secretLen = sizeof(secret_label);
memcpy(pOpData->secret, secret_label, sizeof(secret_label));
pOpData->numLabels = 1;
memcpy(pOpData->label[0].label, label, sizeof(label));
pOpData->label[0].labelLen = sizeof(label);
pOpData->label[0].sublabelFlag = 0x00;
```

**HKDF Extract Expand Label and Sublabels operation**

To perform Extract Expand Label and Sublabels operation, in `CpaCyKeyGenHKDFOpData` structure `hkdfKeyOp` have to be set to `CPA_CY_HKDF_KEY_EXTRACT_EXPAND_LABEL`. Length
of seedLen, secretLen, have to be provided and all data copied into seed, secret tables. Number of labels has to be set in numLabels field. Next step is setting label[0].labelLen and copy label data into label[0].label table. A the end label[0].sublabelFlag filed is set to CPA_CY_HKDF_SUBLABEL_KEY | CPA_CY_HKDF_SUBLABEL_IV to generate Key and IV sublabels.

pOpData->hkdfKeyOp = CPA_CY_HKDF_KEY_EXTRACT_EXPAND_LABEL;
pOpData->seedLen = sizeof(seed_label);
memcpy(pOpData->seed, seed_label, sizeof(seed_label));

pOpData->secretLen = sizeof(secret_label);
memcpy(pOpData->secret, secret_label, sizeof(secret_label));

pOpData->numLabels = 1;
memcpy(pOpData->label[0].label, label, sizeof(label));

pOpData->label[0].labelLen = sizeof(label);
pOpData->label[0].sublabelFlag = CPA_CY_HKDF_SUBLABEL_KEY;
pOpData->label[0].sublabelFlag |= CPA_CY_HKDF_SUBLABEL_IV;

Perform HKDF operation
- To execute the HKDF operation, the crypto instance must be specified in the instanceHandle. When the operation is performed asynchronously, the callback function and callback tag should be set in pKeyGenCb and pCallbackTag arguments. Operational data is provided in pKeyGenTlsOpData and CpaCyKeyHKDFCipherSuite must be chosen. The output will be passed to CpaFlatBuffer. All generated values are arranged one after the other in single buffer. Depending on what operations are performed the buffer length should be adjusted.

```c
cpaCyKeyGenTls3(cyInstHandle, /* Instance handle */
hkdfSampleCallback, /* Callback function */
(void *)&complete, /* Callback tag */
pOpData, /* HKDF operational data */
CPA_CY_HKDF_TLS_AES_128_GCM_SHA256, /* HKDF cipher suite */
&hkdfOut); /* Output buffer */
```

Known Issues

**QATE-31888 – Possible performance degradation**

<table>
<thead>
<tr>
<th>Title</th>
<th>Possible performance degradation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference #</td>
<td>QATE-31888</td>
</tr>
<tr>
<td>Description</td>
<td>The integrated configuration for FreeBSD kernel is not optimized for all relevant QAT driver scenarios (issue with threading and scheduling).</td>
</tr>
<tr>
<td>Implication</td>
<td>Degradation of QAT data throughput can be observed in the deployment with FreeBSD. The use cases:</td>
</tr>
</tbody>
</table>
- sharing the same core for the threads using request ring (submission / working thread) and response ring (polling thread)
- sharing the same core for among more working threads
- extensive number of threads waiting on mutex queue for responses

**Resolution**
- Try to balance the thread workload onto several cores
- Design the application so the synchronization locks are not shared among many threads

**Affected OS**
FBSD11.2

**Driver/Module**
CPM IA - Common

---

**QATE-30931 - Process exit with orphan rings when spawning multiple processes**

<table>
<thead>
<tr>
<th>Title</th>
<th>Process exit with orphan rings when spawning multiple processes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference #</td>
<td>QATE-30931</td>
</tr>
</tbody>
</table>
| Description | If multiple processes start a user space service access layer (icp_sal_userStart) and they all exit together, the syslog may show a message "Process <PID> <NAME> exit with orphan rings."
| Implication | A kernel panic might happen at reboot if an application is using QAT.
| Resolution | The suggested workaround is to fork the process only after the previous child process runs icp_sal_userStartMultiProcess successfully.
| Affected OS | FBSD11.2 |
| Driver/Module | CPM IA - Common |

---

**QATE-30360 - Full device pass-through not available on KVM guests**

<table>
<thead>
<tr>
<th>Title</th>
<th>Full device pass-through not available on KVM guests</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference #</td>
<td>QATE-30360</td>
</tr>
</tbody>
</table>
| Description | The new firmware authentication feature requires PF devices to be reset via function level reset (FLR) before firmware download. In KVM guests, all pass-through devices attached to a VM are reset at boot time. Any further device reset is trapped by the hypervisor and not issued. This causes firmware authentication to fail after the first firmware download. Full device pass-through might work in some conditions when using vfio and if the host kernel and the platform support it.
| Implication | Direct mode feature not available on KVM guests for devices on full pass-through mode.
| Resolution | Refer to appendix A of Using Intel® Virtualization Technology (Intel® VT) with Intel® QuickAssist Technology (document number 330689-007) for instructions on how to pass through a QAT PF to a VM. Talk to your Intel® representative for more information.
| Affected OS | FBSD 11.2 |
| Driver/Module | CPM IA - Common |

---

**QATE-5092 - AES-XTS does not support buffers sizes that are not a multiple of 16B**

<table>
<thead>
<tr>
<th>Title</th>
<th>AES-XTS does not support buffers sizes that are not a multiple of 16B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference #</td>
<td>QATE-5092</td>
</tr>
</tbody>
</table>
| Description | A single request with a data size that is not a multiple of 16B for AESXTS will fail with an invalid param check.
| Implication | The user cannot submit AES-XTS Crypto requests with buffers that are not multiples of 16B
| Resolution | The suggestion is to add padding to AES-XTS to align with 16B multiplied value.
| Affected OS | FBSD11.2 |
| Driver/Module | CPM IA – Crypto |

---

**QATE-7325 - AES-GCM operation with zero length plain text results in an incorrect tag result**

<table>
<thead>
<tr>
<th>Title</th>
<th>AES-GCM operation with zero length plain text results in an incorrect tag result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference #</td>
<td>QATE-7325</td>
</tr>
</tbody>
</table>
Sending an AES-GCM operation with zero length plain text may generate an incorrect tag result.

Implication
Potentially bad record errors and failing connections.

Resolution
GMAC should be used instead of AES-GCM in case of zero length plain text operations.

Affected OS
FreeBSD 11.2

Driver/Module
CPM IA - Crypto

QATE-33751 - Library and driver do not support devices enumerated in a PCI domain different than 0

Title
Library and driver do not support devices enumerated in a PCI domain different than 0

Reference #
QATE-33751

Description
The user space driver and the QAT library cannot handle devices enumerated in a domain different than 0.

Implication
It is not possible to use the software in systems where the device is enumerated with a PCI domain different than 0.

Resolution
Use system where device is enumerated with PCI domain 0.

Affected OS
FreeBSD 11.2

Driver/Module
CPM IA - Common

QATE-39335 - Compression instances do not work on Virtual Machine with Linux Host QAT driver without CnVnR support

Title
Compression instances do not work on Virtual Machine with Linux Host QAT driver without CnVnR support

Reference #
QATE-39335

Description
FreeBSD QAT VF driver does not get host capabilities - the CnVnR support is enabled by default.

Implication
The driver may fail to start compression instances on Virtual Machine with VF driver if no CnVnR support on Host QAT driver firmware.

Resolution
Use Linux QAT driver CnVnR support (4.3.0 or above) on Linux Host system.

Affected OS
FreeBSD 11.2

Driver/Module
CPM IA - Compression

QATE-40359 - Multiprocess 32 with LimitDevAccess = 0 fails with OpenSSL Speed tests

Title
Multiprocess failure with NumProcesses > 16 for LBG/DNV and NumProcesses > 32 for CLC and LimitDevAccess = 0

Reference #
QATE-40359

Description
Multiprocess application that uses more than 16 processes for LBG/DNV and 32 processes for CLC fails during bundle allocation.

Implication
It is impossible to successfully run multiprocess application with more processes than 16 for LBG/DNV and 32 for CLC.

Resolution
There is limitation to use up to 16 processes for LBG/DNV and up to 32 for CLC per device.

Affected OS
FreeBSD 11.2

Driver/Module
CPM IA - Multiprocess

QATE-39216 - Kasumi test duration issue

Title
Kasumi test duration issue

Reference #
QATE-39216

Description
Sample code benchmark tests included in the software package

Implication
The performance degradation when running the sample code can be observed in case the system runs excessive number of threads.

Resolution
Avoid calling the cpacyInstanceGetInfo2 function if possible (i.e. by caching the info data) and try to tune the FreeBSD scheduler.

Affected OS
FreeBSD 11.2
QATE-41846 - GEN - QAT API submissions with bad addresses that trigger DMA to invalid or unmapped addresses can cause a platform hang

<table>
<thead>
<tr>
<th>Title</th>
<th>GEN - QAT API submissions with bad addresses that trigger DMA to invalid or unmapped addresses can cause a platform hang</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference #</td>
<td>QATE-41846</td>
</tr>
<tr>
<td>Description</td>
<td>This version of the QAT hardware does not perform request checking. It follows that a malicious application can submit requests that can bring down an entire QAT endpoint, which can impact other QAT jobs associated with the hardware. Furthermore, if any QAT API submission have bad addresses that would trigger DMA to invalid or unmapped addresses, these can induce a platform hang. This presents a risk to be managed by the host and guest operating systems and other system policies. The exposure can extend to other guest operating systems or applications outside of the typical access boundary of the malicious guest or application.</td>
</tr>
<tr>
<td>Implication</td>
<td>All guest operating systems or other applications using QAT must be trusted, and/or other steps must be taken to ensure that an untrusted application or guest cannot submit incorrectly formatted requests.</td>
</tr>
<tr>
<td>Resolution</td>
<td>There is no workaround available. However, system policies (including limiting certain operating system permissions) can help to mitigate this issue.</td>
</tr>
<tr>
<td>Affected OS</td>
<td>FreeBSD 11.2</td>
</tr>
<tr>
<td>Driver/Module</td>
<td>CPM IA - Crypto</td>
</tr>
</tbody>
</table>

QATE-41745 - Segmentation fault when using inputs on QUAD word boundaries

<table>
<thead>
<tr>
<th>Title</th>
<th>Segmentation fault when using inputs on QUAD word boundaries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference #</td>
<td>QATE-41745</td>
</tr>
<tr>
<td>Description</td>
<td>When using EC's cpaCyEcPointMultiply or cpaCyEcPointVerify with aligned size of input parameters to four, eight or nine quadwords (4 * 8B, 8 * 8B or 9 * 8B), a segmentation fault occurs.</td>
</tr>
<tr>
<td>Implication</td>
<td>Application crashes, caused by a segfault.</td>
</tr>
<tr>
<td>Resolution</td>
<td>Use inputs that are not quad word aligned, if using quad word aligned input, pad it with a &quot;0x0&quot; to bring above the quad word boundary.</td>
</tr>
<tr>
<td>Affected OS</td>
<td>FreeBSD 11.2</td>
</tr>
<tr>
<td>Driver/Module</td>
<td>CPM</td>
</tr>
</tbody>
</table>

QATE-41486 - Misleading message observed in dmesg on LBG device with LimitDevAccess = 1 set in configuration file.

<table>
<thead>
<tr>
<th>Title</th>
<th>Misleading message observed in dmesg on LBG device with LimitDevAccess = 1 set in configuration file</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference #</td>
<td>QATE-41486</td>
</tr>
<tr>
<td>Description</td>
<td>When using LimitDevAccess = 1 with more than one device in up state, the &quot;qatX: failed to get NumberCyInstances value from config!&quot; message could be observed in dmesg for other devices than configured one. This message indicates only that for the other devices the configuration was not found, what is expected.</td>
</tr>
<tr>
<td>Implication</td>
<td>It is an internal message only, and should not be threat as an error.</td>
</tr>
<tr>
<td>Resolution</td>
<td>Ignore error message when use LimitDevAccess parameter.</td>
</tr>
<tr>
<td>Affected OS</td>
<td>FreeBSD 11.2</td>
</tr>
<tr>
<td>Driver/Module</td>
<td>CPM</td>
</tr>
</tbody>
</table>