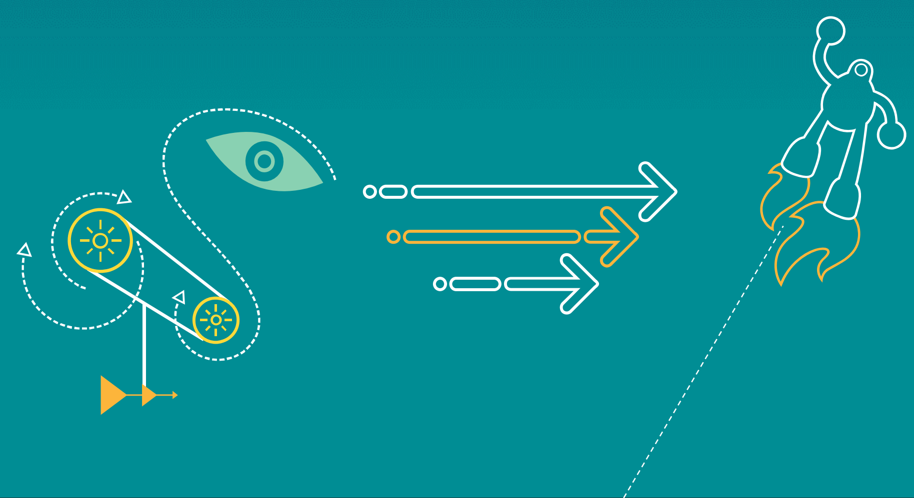
TZ XF 5.0 TrustZone Architecture Overview

QUALCOMM®

Qualcomm Technologies, Inc.

80-P9301-28 Rev. B



Confidential and Proprietary – Qualcomm Technologies, Inc.

Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

Revision History

Revision	Date	Description
A	July 2017	Initial release
B	November 2017	Updated acronyms on Slide 7, 9, 11, 43, and 44

Contents

TrustZone

Trustzone Overview

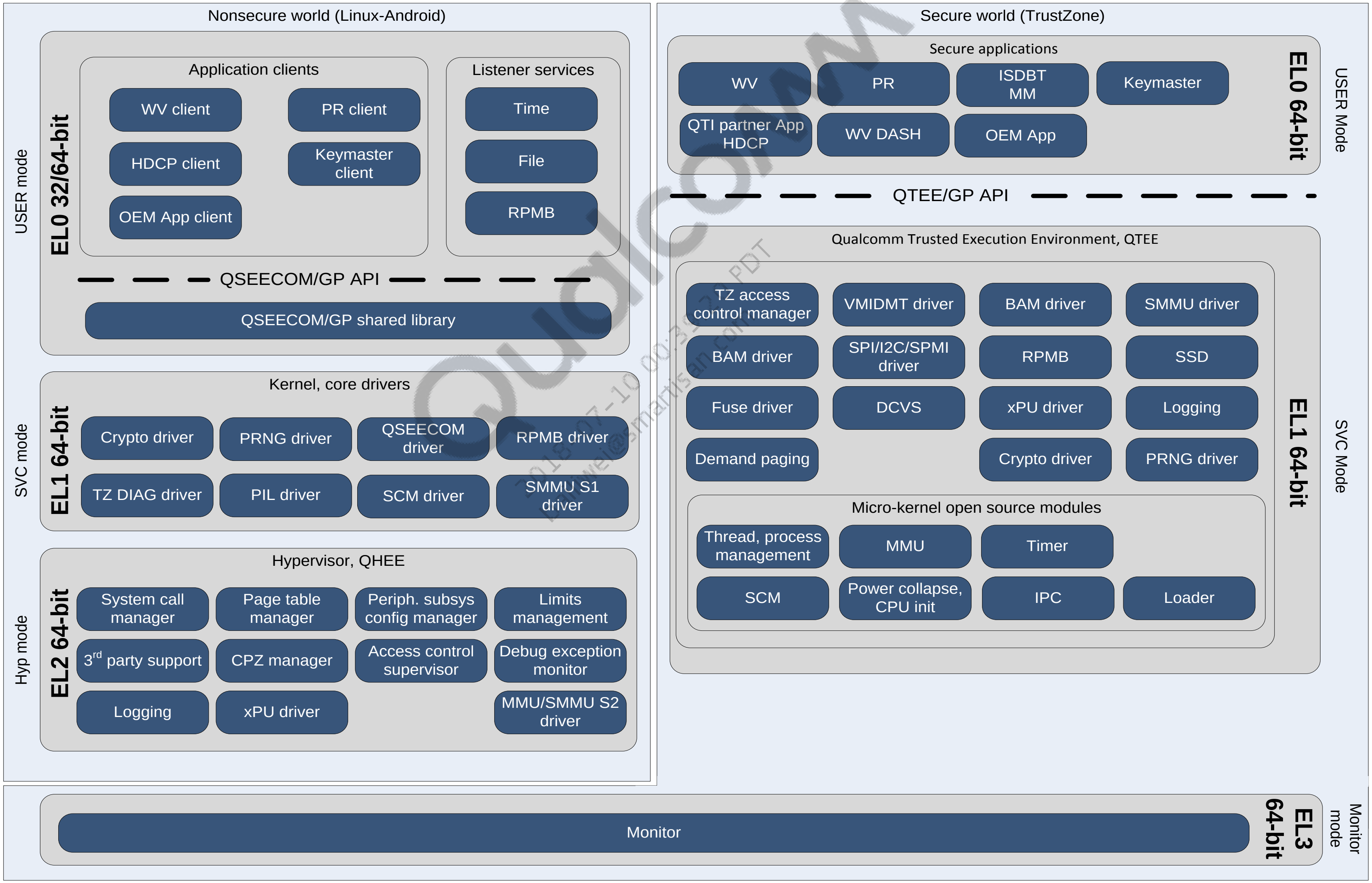
- The SDM845 chipset provides a 64-bit ARM v8.2-compliant octa-core Qualcomm® Kryo™ CPU (385) applications processor, including four customized Kryo Gold (performance optimized) and four Kryo Silver (power optimized) processors with hardware virtualization
- TrustZone (TZ) is a hardware-based security environment through a Secure mode of the ARM processor
  - High-level operating system (HLOS) runs in Nonsecure mode
  - Transition from Nonsecure to Secure mode occurs via a Secure Monitor mode
- The SDM845 TZ software is based on the Qualcomm® Trusted Execution Environment (QTEE) 5.0 architecture

Note: QTEE was formerly known as Qualcomm Secure Execution Environment (QSEE).

TrustZone Overview (cont.)

- TZ software consists of TrustZone board support package (TZBSP) and QTEE components
  - TZBSP
    - Provides software support for chipset security
    - Exposes hardware abstraction layer (HAL) APIs for chipset security functions (that is, crypto, fuse block, and pseudo-random number generator (PRNG))
    - Initializes system security environment for software and hardware during bootup and wakeup from power collapse
    - Provides memory and other subsystem protection and services during runtime
  - QTEE
    - Provides security services (that is, image loading, authentication, cache management, crypto, logging, and QFPROM) to TZ secure applications
    - Provides a set of GlobalPlatform compliant APIs
- TZ software image is loaded by the boot loader (XBL) during initial device bootup process

QTEE 5.0 Security Framework Block Diagram



QTEE Features

- 36-bit physical address support
  - ARMv8 long descriptor with 36-bit PA output
  - 64-bit secure EL1 support
- ARMv8 secure monitor call (SMC) interface
  - SMC64 calls
- Obfuscation of double date rate (DDR) execution region using pseudo internal memory (pIMEM)
  - Encryption and decryption of the secure execution region in DDR
  - Authentication of read data to confirm the freshness

QTEE Features (cont.)

- Slave side access control (For NS = 0 asset protection), configure all the VMIDMTs and xPUs for slave-side access control
- Master side access control
  - Configuration of TZ-owned SMMU (ARM MMU-500) context banks
  - Run-time support for TBU configuration for secure peripheral switch use cases
- Retention flip flops (RFF)
  - RFF configuration for retention of xPUs and SMMUs over power collapse
- Guard pages for heap and stack overflow checking
- Dynamic linking of commonlib is changed in QTEE 5

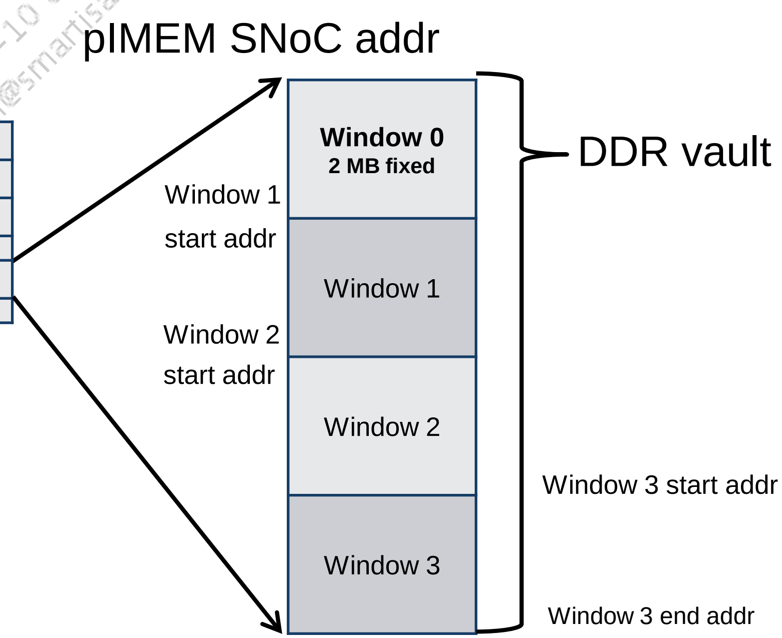
Hypervisor Features

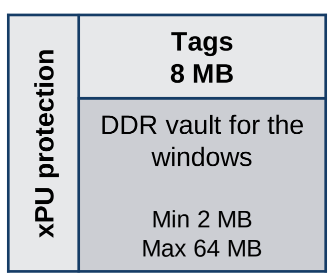
- Inclusion of stage 2 address translation, ARMv8/64-bit aligned SMMUv2
- Access control
  - Master-side access control using stage 2 MMU/SMMU
  - Remove majority of xPUs and VMIDMTs
    - xPU violations are asynchronous that makes them more difficult to debug than SMMU faults
- Content protection zone (CPZ)
  - Protect premium content using stage 2 MMU/SMMU
- Benefits
  - ARM virtualization extensions to improve access control/CPZ/security
  - SMMUv2 includes the page table formats required for heterogeneous compute on 64-bit systems
  - ARM architecture compliance: Aligning with ARM memory management

pIMEM Design in SDM845

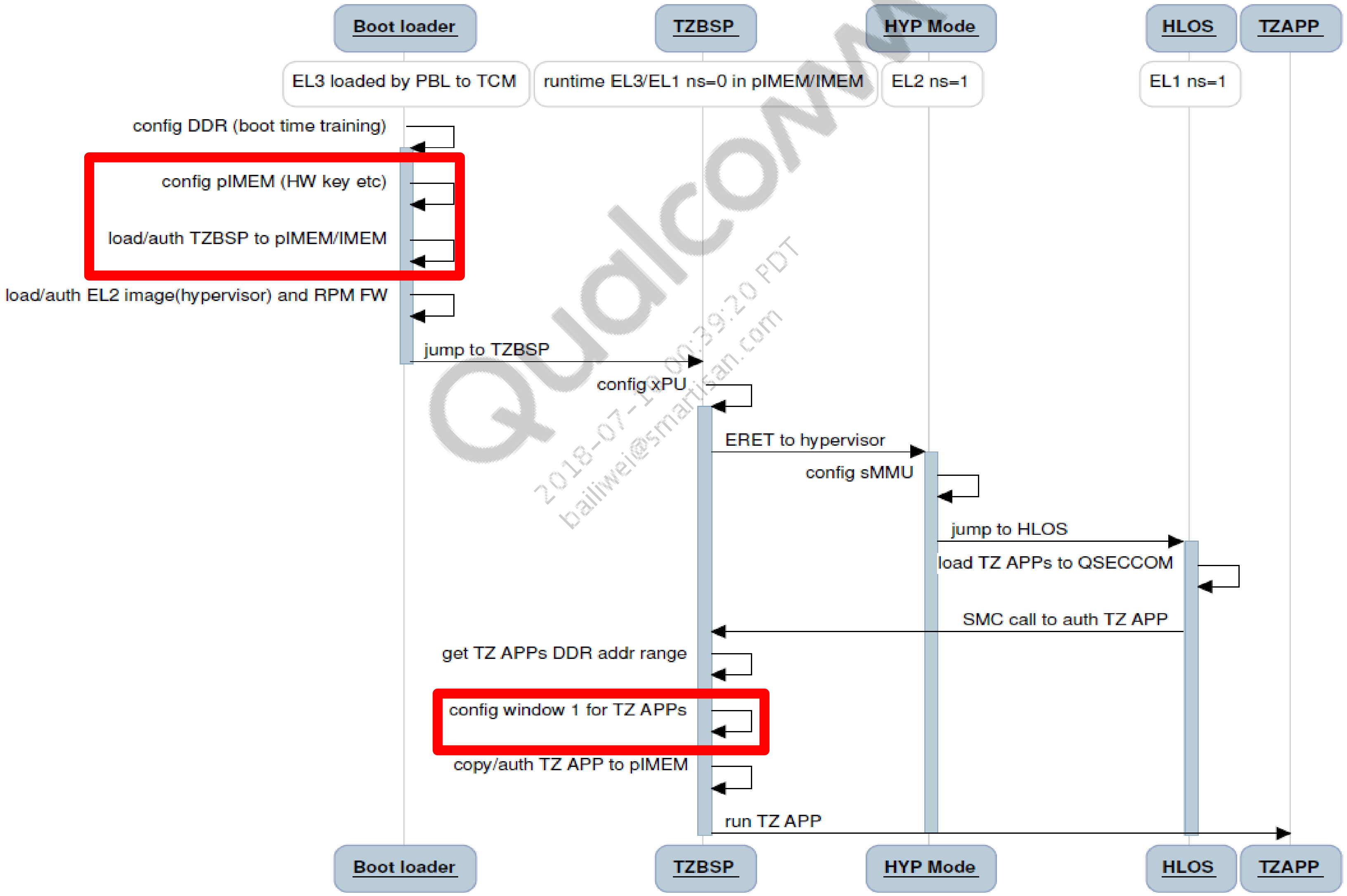
- Supports four pIMEM windows with one DDR vault with contiguous DDR space
  - Software requirement is at least two windows in the pIMEM vault
  - Fixed TZ and TZ applications DDR range in memory map
- TZ applications size ranges from 100 KB to 10 MB
- XBL initializes pIMEM and configures window 0 for TZBSP
- TZ configures window 1 for TZ applications

Start address	End address	Size	Remarks
180000000	27FFFFFFF	4 GB	DDR
80000000	17FFFFFFF	4 GB	DDR
11C000000	1FFFFFFF	64 MB	pIMEM





pIMEM Initialization



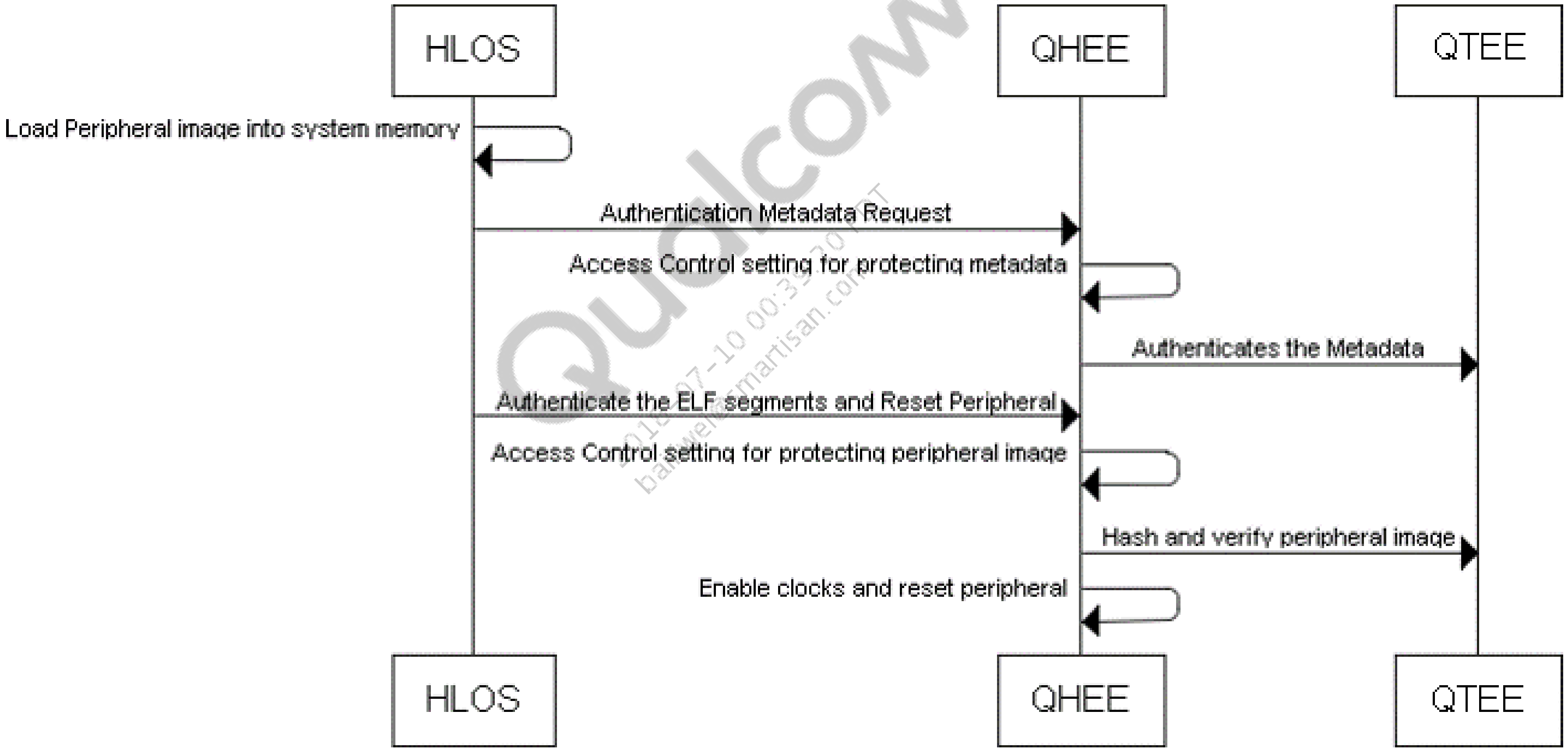
Hypervisor and QHEE Features

- SMP and pre-emptible
- No interrupt virtualization planned
  - HLOS must route S2 fault and global nonsecure interrupts to Hypervisor
- SMC trapping to hypervisor
- ARMv8 SMC interface
- Master-side access control
  - Configuration of hypervisor owned SMMU context banks
  - SMMUv2 specific updates
  - Clock Driver
- Secure PIL, SSR
- Content protection support
- Dynamic heap sharing (HLOS and SS)
  - Secure channel between aDSP and Hypervisor
- Power collapse, voting for LPM

Secure PIL

- HLOS calls into QHEE for authentication and resetting the peripheral
- QHEE applies the necessary access control and call into QTEE for authentication
- QTEE authenticates the metadata of the image in the first call, antirollback fuses are blown by QTEE if the loaded image has a higher version
- QTEE decrypts the ELF segments and hashes them to complete the authentication process
- QHEE programs the clocks and the resets vector of the subsystem
- Peripheral registers that govern reset and reset vector location are protected

Secure PIL Call Flow



Subsystem Restart

- HLOS calls into QHEE to request that a peripheral be prepared for reset
- QHEE calls the clock driver to place the subsystem in reset
- QHEE scribbles the peripheral’s image location in system memory if the secure boot fuse is blown
- QHEE does not protect the system memory for the given peripheral so that the HLOS can load the new image

Secure Watchdog

- Only secure watchdog timer (SWDT) can reset the system
  - All other WDTs generate an interrupt on bite
  - Subsystem WDT bites are handled by HLOS (subsystem restart)
  - HLOS WDT bite is handled by TZ (context dump/reset)
  - Secure WDT bite then resets the device to the first stage of reset
- SWDT has the longest bark and bite timeout in the system, maximum of 32 seconds
- Configured and accessible only by TZ
- Enabled in cold boot after TZ enables interrupts
- TZ pet secures WDT on bark

Functionalities of TZ

- Cold boot
  - The boot sequence starts from a secure root of trust (APPS PBL), the boot process begins in on-chip ROM (APPS PBL)
  - Signed software images (including XBL SEC, XBL, and TZ) are loaded from flash memory into OCIMEM and authenticated prior to execution
  - APPS PBL loads, authenticates, and executes XBL SEC
  - XBL SEC running in EL3 programs xPUs, initializes pIMEM, and applies the debug policy (if present)
  - APPS PBL loads and authenticates XBL, and XBL loads and authenticates TZ
  - XBL SEC transfers execution to TZ
  - TZ programs different xPUs in the system to protect different resources
  - TZ also programs VMIDMT tables and the masters that have access to the tables

Functionalities of TZ (cont.)

- TZ sets up access control for the registers of the following hardware blocks
  - BAM
  - CPU registers
  - QGIC
  - Hardware crypto engine
  - QFPROM
    - TZ can write or blow QFPROM bits in application region
    - All other subsystems have read access, but not the write access
  - Random number generator

Functionalities of TZ (cont.)

- Warm boot
  - TZ runs within system pIMEM and APPS CPU starts in Secure mode
  - Power-collapse terminates in TZ and it programs APPS CPU to start from TZ reset vector
- TZ performs the following initializations during warm boot:
  - CPU core system initialization, which configures CPU to a known state
  - Initializes CPU security
  - Initializes L2 cache if it is power collapsed
  - Switches to the Nonsecure mode

Functionalities of TZ (cont.)

- Secure peripheral image loader (PIL)
  - TZ Secure PIL authenticates different subsystem images and configures related xPUs to protect the subsystem memory regions
  - During initialization, the PIL authentication service in TZBSP protects the registers responsible for taking the peripheral processors out of reset
  - HLOS loads the ELF file for subsystem image at a 4-byte aligned location in DDR RAM
  - HLOS requests the PIL authentication service in TZBSP to securely authenticate the images
    - HLOS cannot authenticate the images as it can potentially be tempered
  - TZBSP protects the memory areas used by the subsystem images with the MPU
  - TZBSP initializes a processor's registers and necessary clocks
  - Once an image is validated, the PIL authentication service resets the respective subsystem core so it can boot

Functionalities of TZ (cont.)

- Power collapse
  - Some xPUs are power collapsible and they are mostly related to multimedia functionalities
- Crypto BAM
  - TZ configures every BAM pipe crypto to prevent HLOS from using OEM hardware key and Qualcomm Technologies, Inc. (QTI) hardware key
- Secure debug
  - JTAG debug functionality can be disabled by OEM using QFPROM; debug functionality can be re-enabled in TZ using software OVERRIDE registers
  - When debug is enabled in a device, hardware keys are replaced by dummy keys; TZ software thus uses the dummy keys when debug is re-enabled
  - Watchdog debug

BSP Drivers in TZ

- QFPROM
  - QFPROM fuse blow framework is part of TZ
  - OEMs can use the following APIs to read and write QFPROM from their TZ application
    - qsee_fuse_read()
    - qsee_fuse_write()
- Crypto engine
  - Two general-purpose crypto engines (one each for modem and TZ)
  - Supported operations:
    - AES 128/192/256 with the following modes: ECB, CBC, CTR, GCM, CCM, CTS, and XTS
    - DES 128/192/256
    - 3DES (Triple DES)
    - SHA1, SHA256/384/512
    - HMAC SHA1/SHA2
    - ECDH and ECDSA
    - RSA 1K/2K/4K

BSP Drivers in TZ (cont.)

- Random number generator (RNG)
  - Can only be configured by TZ
  - Both TZ and HLOS can access the generated random numbers
  - Generates up to 2048 bytes of random data per function call
- SPI driver
  - Can be used to interface with external devices (For example, fingerprint sensor)
  - Configured by TZ and is enabled by default
- I^2C driver
  - Securely interface with external devices over I^2C
  - Used for secure touch feature

Secure Storage

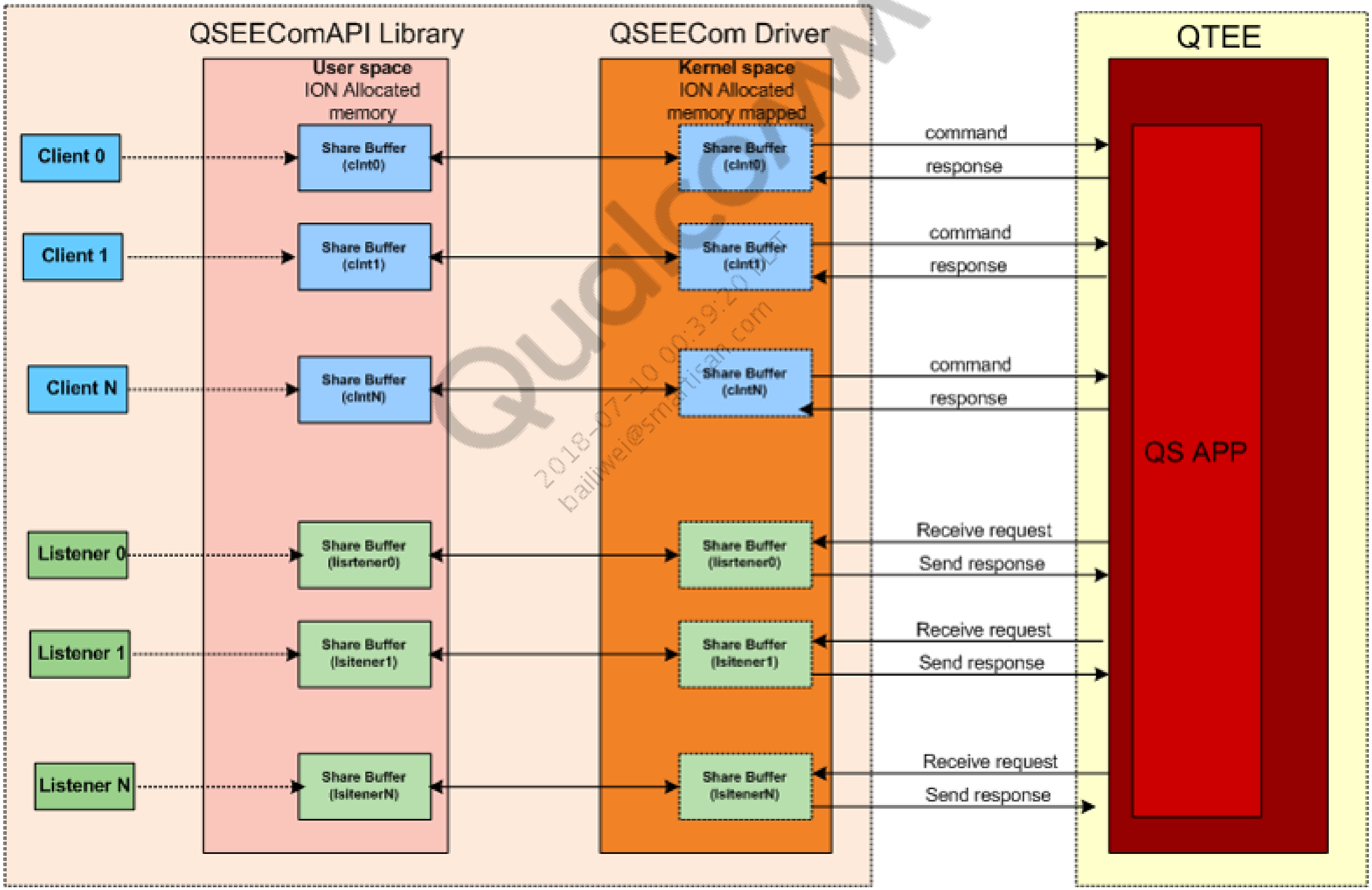
- The secure file system (SFS) provides encrypted storage of sensitive data in the file system
  - Encrypted with hardware key, it can only be decrypted in TrustZone
- Insecure unless debug is disabled and secure boot is enabled
- SFS supports antirollback protection via replay protection memory block (RPMB)
- RPMB partition
  - Accesses controlled block in UFS memory
  - Manages data through authenticated and replay-protected method
  - Used by SFS and HLOS full disk encryption
  - Requires one-time key provisioning
    - Occurs automatically when secure boot is enabled
    - Can disable automatic RPMB provisioning with DevConfig OEM_disable_rpmb_autoprovisioning property
    - Reprovisioning is allowed when DevConfig OEM_allow_rpmb_key_provision property is set

QTEE

Components Overview

- HLOS Nonsecure world
  - User space
    - Client application
    - Listener service: Listens for any requests coming from QSAPP
    - QSEEComAPI library: Interface to the QSEECom kernel driver
  - Kernel space
    - QSEECom driver
    - SCM Driver – Interface to QTEE
- Secure world
  - QTEE
  - QSAPP/TZAPP – Secure application

Components Overview (cont.)



QSEECom Client

- HLOS application to initiate all requests to a TZ application such as APPS The client:
  - Invokes QSEECom_start_app() to issue a request to load the secure application
  - Retrieves the handle to the QSEECom driver
  - Sends the commands or the requests to the TZ application by using the retrieved handle when the secure application is loaded
- Sample client code
  - Location: vendor/qcom/proprietary/securemsm/sampleclient/qseecom_sample_client.c
  - Provides support functionalities to load and unload applications, measures crypto performance, RSA, and stress test

QSEECom Listener

- HLOS service module that services requests originates from QTEE
- QSEECom daemon starts the listeners that are used to interact with QSEECom
  - Invoke the QSEECom_register_listener() call to register the QTEE listeners with QSEECom
  - Upon successful registration, the QSEECom driver stores the listener ID in a listener service queue
  - With the QSEECom driver registered, each listener service must call QSEECom_receive_req() to start the listener service
- One thread from each service is blocked after QSEECOM_IOCTL_RCV_REQ is called
- The thread is signaled when QSEECom receives a command that contains a particular listener service ID
- Daemon code
  - Location: vendor/qcom/proprietary/securemsm/daemon/qseecomd.c
- Test listener code
  - Location: vendor/qcom/proprietary/securemsm/securitytest/qseecom_security_test.c

SCM Driver

- Secure channel manager driver maintains a communication channel between HLOS, TZBSP, and QTEE
- Uses ARM SMC instruction to switch from kernel Nonsecure Supervisor mode to TZ Secure Supervisor mode
- Key API

```
int scm_call(u32 svc_id, u32 cmd_id, const void *cmd_buf, size_t cmd_len,
            void *resp_buf, size_t resp_len)
```

- Key data structures

```
struct scm_command {
u32   len;
u32   buf_offset;
u32   resp_hdr_offset;
u32   id;
u32   buf[0];
};
```

- Code – kernel/msm-4.9/drivers/soc/qcom/scm.c

QSEEComAPI Library

- Provides an abstracted API for the use of IOCTL
- Allows for memory management to be handled under the hood
- Summary of the API:

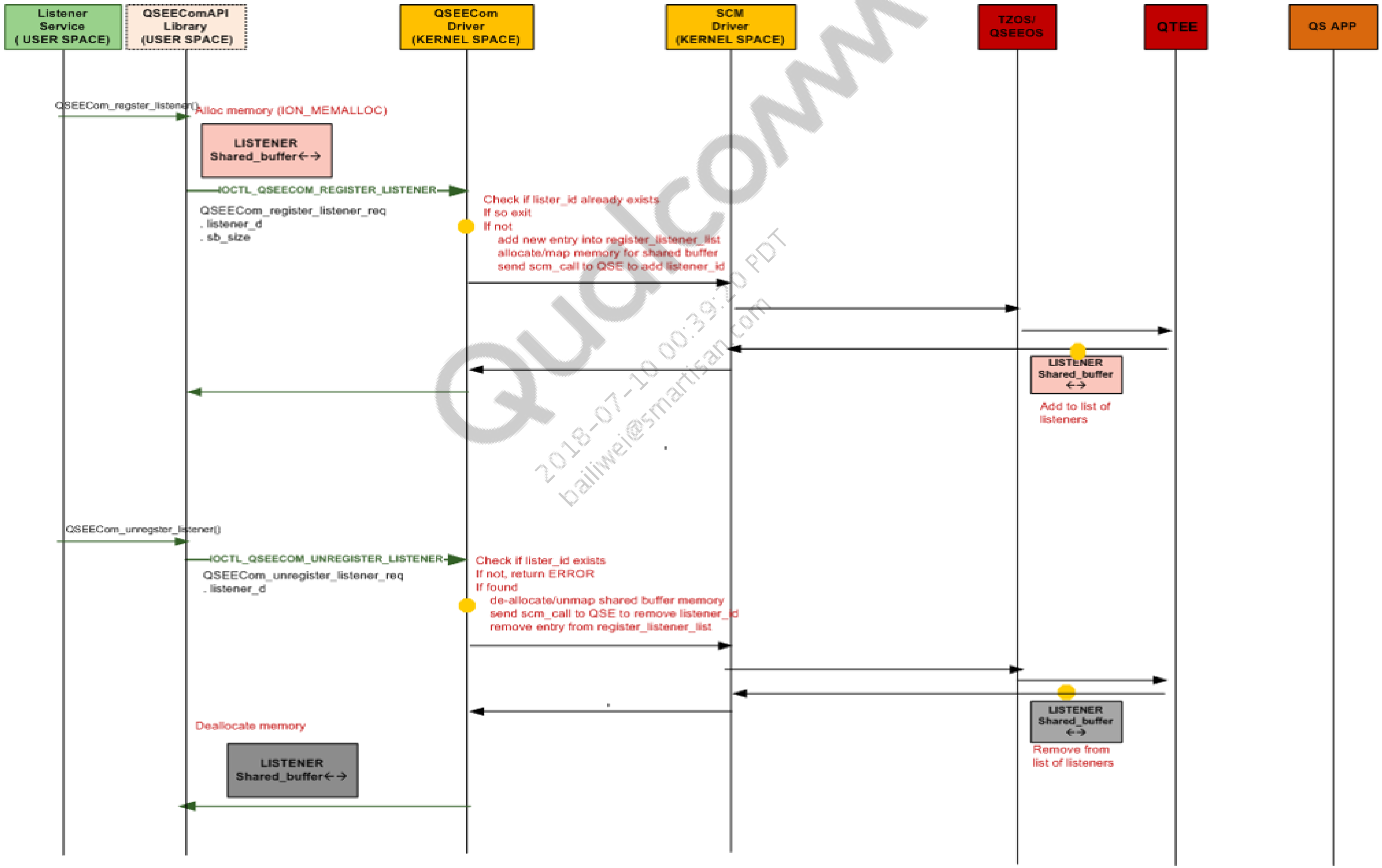
```
int QSEECom_start_app (struct QSEECom_handle *handle, const char *fname,
uint32_t sb_size);
int QSEECom_shutdown_app (struct QSEECom_handle *handle);
int QSEECom_send_cmd (struct QSEECom_handle *handle, void *send_buf,
                       uint32_t sbuf_len, void *rcv_buf, uint32_t rbuf_len);

int QSEECom_register_listener (struct QSEECom_handle *handle,
                       uint32_t lstnr_id, uint32_t length, uint32_t flags);
int QSEECom_unregister_listener (struct QSEECom_handle *);
int QSEECom_receive_req (struct QSEECom_handle *handle, void * buf, uint32_t
len);
int QSEECom_send_resp (struct QSEECom_handle *handle, void *send_buf, uint32_t
len);
```

- Released as a library built from QSEEComAPI.c

QSEECom Driver Design

- QTEE communicator driver is a kernel module to communicate between the kernel and QTEE
- Process IOCTL passed in from QSEEComAPI layer
- Driver APIs
  - qseecom_open()
    - Returns a handle to QSEECom device (QSEECom_dev_handle)
  - qseecom_release()
    - Releases a handle to QSEECom device (QSEECom_dev_handle)
  - qseecom_ioctl()
    - Processes various IOCTLs related to TZ app, listener, command, and request
- Code
  - Location: /kernel/drivers/misc/qseecom.c

IOCTL – REGISTER_LISTENER, UNREGISTER_LISTENER



TZ Configuration

- TZ exposes configuration options through XML files, such as ssg/securemsm/trustzone/qsee/mink/oem/config/sdm845/oem_config.xml file
- Configuration options include enablement of RPMB autoprovisioning, secure watchdog bark, and bite times, and RPMB enablement for TZ application SFS
- The XML configuration files are built into devcfg.mbn

TZ Debugging

TZ Counters

- Power collapse counters are
  - Increased when a core enters and exits power collapse
  - Stored in pIMEM
- In IMEM dump, the counters are found at TZBSP_SHARED_IMEM_DIAG_ADDR
- Counters for CPU core 0 are found in addresses listed in table
- Counters for remaining CPU cores are listed in the same order and format in pIMEM after CPU core 0

Counter address in IMEM	Counter definition
TZBSP_DIAG_BASE + 0xa4	CPU0 warm boot entry counter
TZBSP_DIAG_BASE + 0xa8	CPU0 warm boot exit counter
TZBSP_DIAG_BASE + 0xac	CPU0 power collapse termination entry counter
TZBSP_DIAG_BASE + 0xb0	CPU0 power collapse termination exit counter
TZBSP_DIAG_BASE + 0xb4	CPU0 jump address to HLOS
TZBSP_DIAG_BASE + 0xb8	CPU0 jump address instruction

TZ Counters FAQs

Q. Is a core stuck in warm boot?

A.

- For core 0, if warm boot entry counter is equal to warm boot exit counter, then it enters and exits TZ the same number of times.
    - If there is a mismatch, the core is still in TZ.
- For remaining cores, if warm boot entry counter is equal to warm boot exit counter + 1, then it enters and exits TZ the same number of times.
    - First power-up is considered a warm boot, and there is no exit.
    - If warm boot entry counter is equal to warm boot exit counter + 2, it is still in TZ.

Q. Is a core in power collapse?

A.

- For core 0, if warm boot entry counter matches the number of power collapses, then the core is online.
    - If warm boot entry counter is less than the number of power collapses, core 0 is power collapsed.
- For remaining cores, if warm boot entry counter is equal to number of power collapses + 1, the core is online.
    - If warm boot entry counter is equal to number of power collapses, the core is power collapsed..

Watchdog Reset

- A processor is unable to pet its watchdog:

  The watchdog expires and bites, and notifies the HLOS
- The HLOS cannot pet its watchdog:

  The HLOS watchdog expires and bites, and notifies TrustZone
- TrustZone cannot pet its watchdog:

  The AP's secure watchdog expires and bites, and causes a chip reset

TZBSP Diag Area in IMEM

- 4 KB region of IMEM allocated to diagnostic area
  - xPU allows read access to HLOS to entire diag region
- 2 KB of diag used for logging ring buffer
- VMID names structure
- Boot debug structure
  - Counts for entry or exit
  - Last warm boot exit location
  - Last warm boot exit instruction
- Reset debug structure
  - Number of resets
  - Last reset reason
- Interrupt debug structure
  - Number of interrupts seen on each CPU
  - Interrupt name, number, type

TZBSP Logs and Ring Buffer

- TZBSP outputs logs to two places
  - JTAG terminal
  - TZBSP ring buffer
- Read the ring buffer in Android using ADB

```
>adb shell
>mount –t debugfs none /d
>cat /d/tzdbg/log
>cat /d/tzdbg/qsee_log
```

- More files are available at: /d/tzdbg for other debugging

References

Title	Number
Qualcomm Technologies, Inc.	
SDM845 Security Overview	80-P9301-27
Replay Protected Memory Block (RPMB) Security on QSEE	80-P8327-1

Acronym or term	Definition
CRC	Cyclic redundancy check
CPZ	Content protection zone
DCC	Data capture and compare
DDR	Double date rate
HAL	Hardware abstraction layer
HLOS	High-level operating system
PIL	Peripheral image loader
pIMEM	Pseudo internal memory
RNG	Random number generator
RPMB	Replay protection memory block

References (cont.)

Acronym or term	Definition
SFS	Secure file system
SMC	Secure monitor call
SNoC	System network on chip
SWDT	Secure watchdog timer
TZ	TrustZone
TZBSP	TrustZone board support package
QTEE	Qualcomm Trusted Execution Environment (Formerly known as Qualcomm Secure Execution Environment (QSEE))

Questions?

https://createpoint.qti.qualcomm.com