HDCP SUPPORT IN OPTEE

Linaro connect

Linaro Multimedia Working Group
- https://www.linaro.org/

PRODUCT PRESENTATION

MICR ADVANCED TECHNOLOGIES

SEPTEMBER 2019

NXP SECURE CONNECTIONS FOR A SMARTER WORLD

Agenda

- Quick introduction to HDCP
- Secure Video Path overview
- Current HDCP control in Linux
- Proposal to control HDCP in OPTEE
- Questions

HDCP OVERVIEW

HDCP : High bandwidth Digital Content Protection

- A digital copy protection developed by Intel™ to prevent copying of digital and audio video content. Before sending data, the source device shall check the destination device is authorized to received it.

  If so, the source device encrypts the data, only the destination device can decrypt.

  - data encryption
  - prevent non-licensed devices from receiving content

- Android and Linux NXP bsp manage HDCP at Linux Level, through libDRM. So nothing prevent a user to disable HDCP protection while secure content is under playback. It is a security holes in the Secure Video Path.

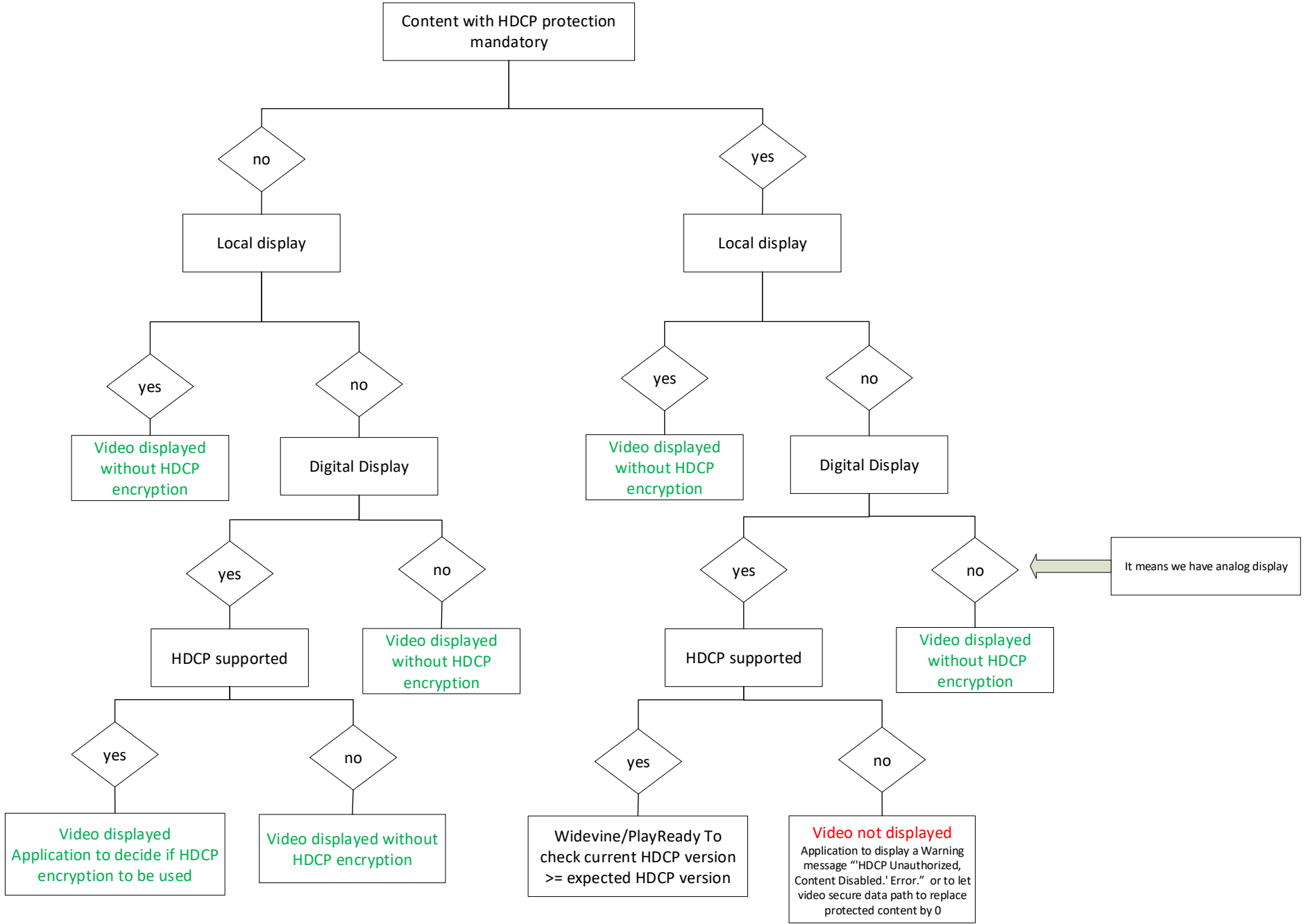
- HDCP support currently under development for wayland/Weston:

  https://gitlab.freedesktop.org/wayland/weston/merge_requests/48

- No Open Source solution exists to manage HDCP in secure mode.

- HDCP versions:
  - HDCP 1.X: Hacked: Master key published (leak/reverse engineering)
  - HDCP 2.0: Hacked before release
  - HDCP 2.1: Hacked before release
  - HDCP 2.2: Not yet hacked
  - HDCP 2.3: Not yet hacked

HDCP control state Machine



SECURE VIDEO PATH OVERVIEW

Secure Video Path to protect video content inside the device

I.MX 8MQ Secure Video Path with Android bsp – Hong Kong Connect 2018:

- Slides:
  http://connect.linaro.org.s3.amazonaws.com/hkg18/presentations/hkg18-113.pdf

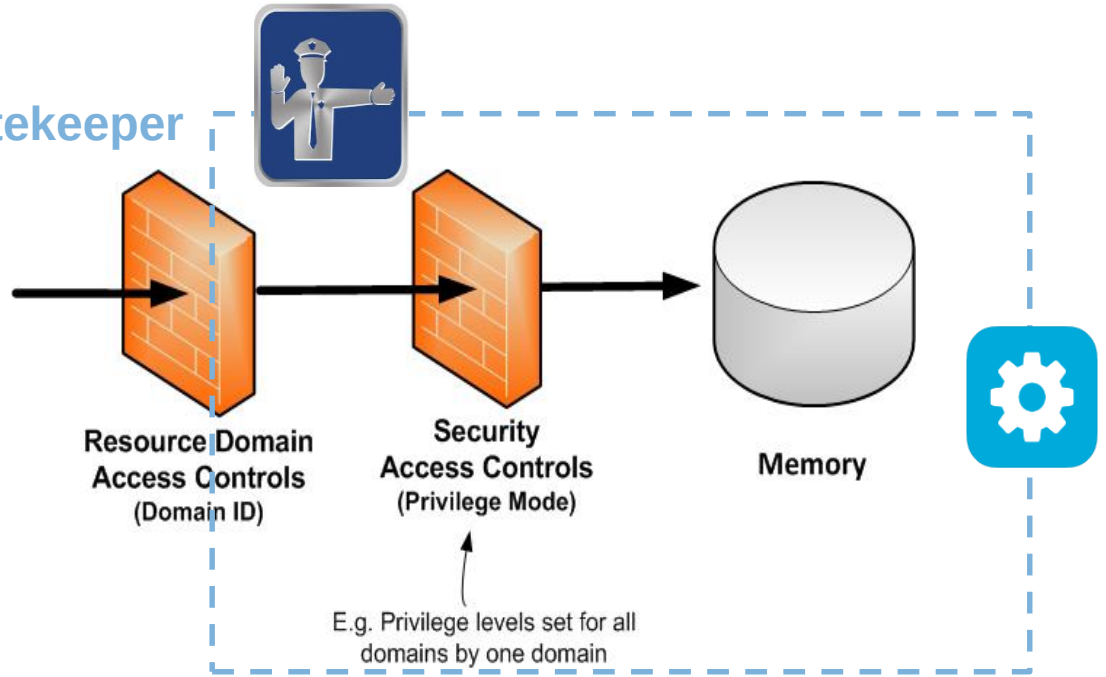
- Demos:
  https://www.youtube.com/watch?v=z27Tl5XkFJ4

I.MX 8MQ Secure Video Path with Linux bsp – San Diego Connect 2019

- Slides:
  https://static.sched.com/hosted_files/linaroconnectsandiego/d6/Linux%20DRM%20Support%20iMX8M%20-%20Architecture%20Description%20v1.2.pdf

Secure Data Path on i.MX 8M
RDC: Resource Domain Controller

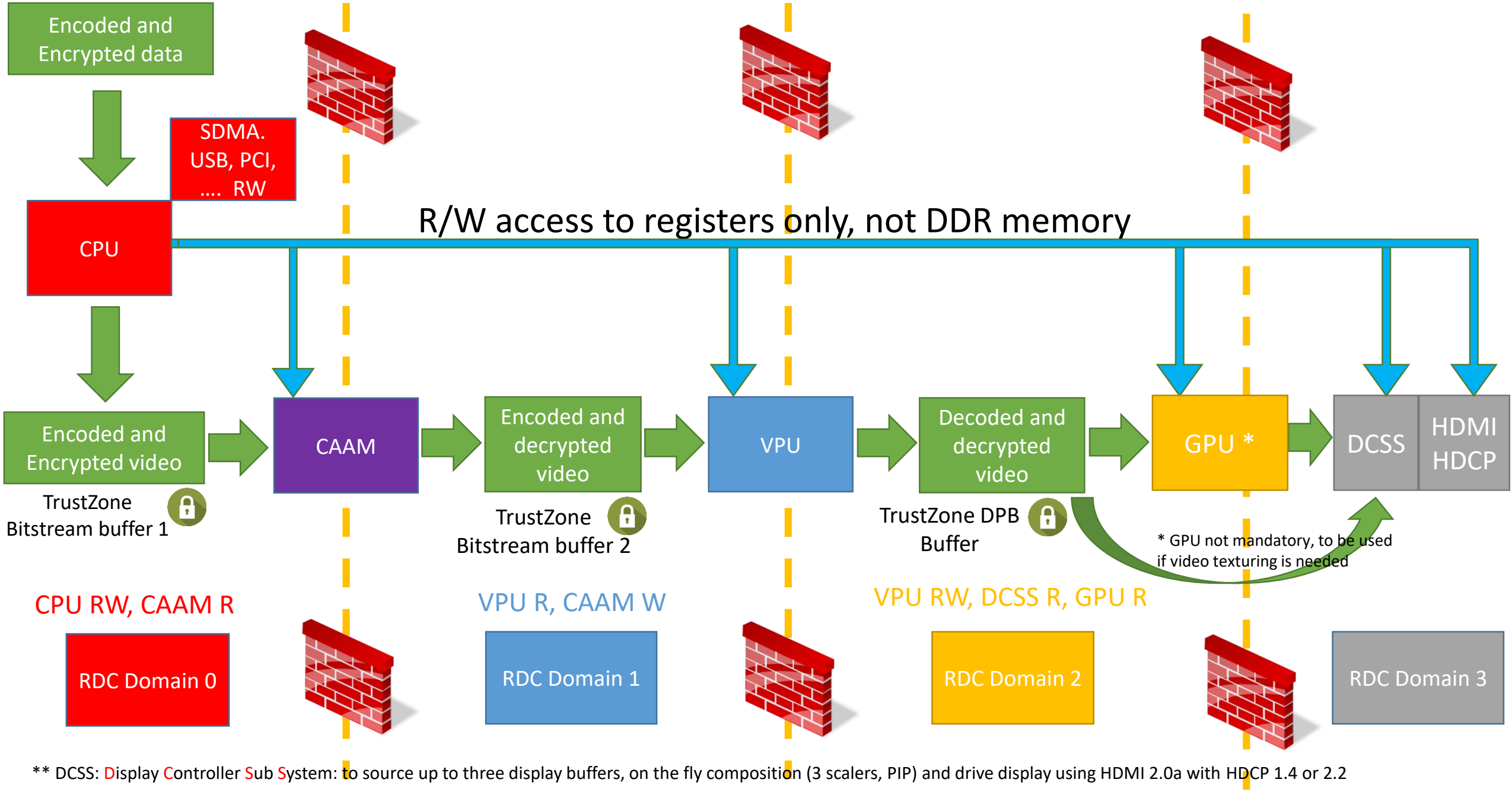
RDC the SDP gatekeeper



Use a configuration set at boot (within ATF)
Registers locked till next reset/reboot

- Assignment of cores and bus masters to a resource domain (4 domains, 27 bus masters)
- Peripherals and memory regions assigned right accesses based on domain IDs (118 Peripherals, 52 memory regions)
- Memory read/write access controls for each resource domain and region (up to 8 regions per domains)

Secure Video Path on i.MX 8MQ



** DCSS: Display Controller Sub System: to source up to three display buffers, on the fly composition (3 scalers, PIP) and drive display using HDMI 2.0a with HDCP 1.4 or 2.2

HDCP MANAGEMENT BY LINUX

HDCP managed by the Rich Execution Environment (Android, Linux, RDK …)

HDCP very high level requirements:

- User Application shall be able to disable/enable HDCP content protection, allowing users to record clear content, or to encrypt content according to the security policy of the content to prevent illegal copying of digital content.

- User Application shall be able to notify user when HDCP versions is not compliant with content to be displayed.

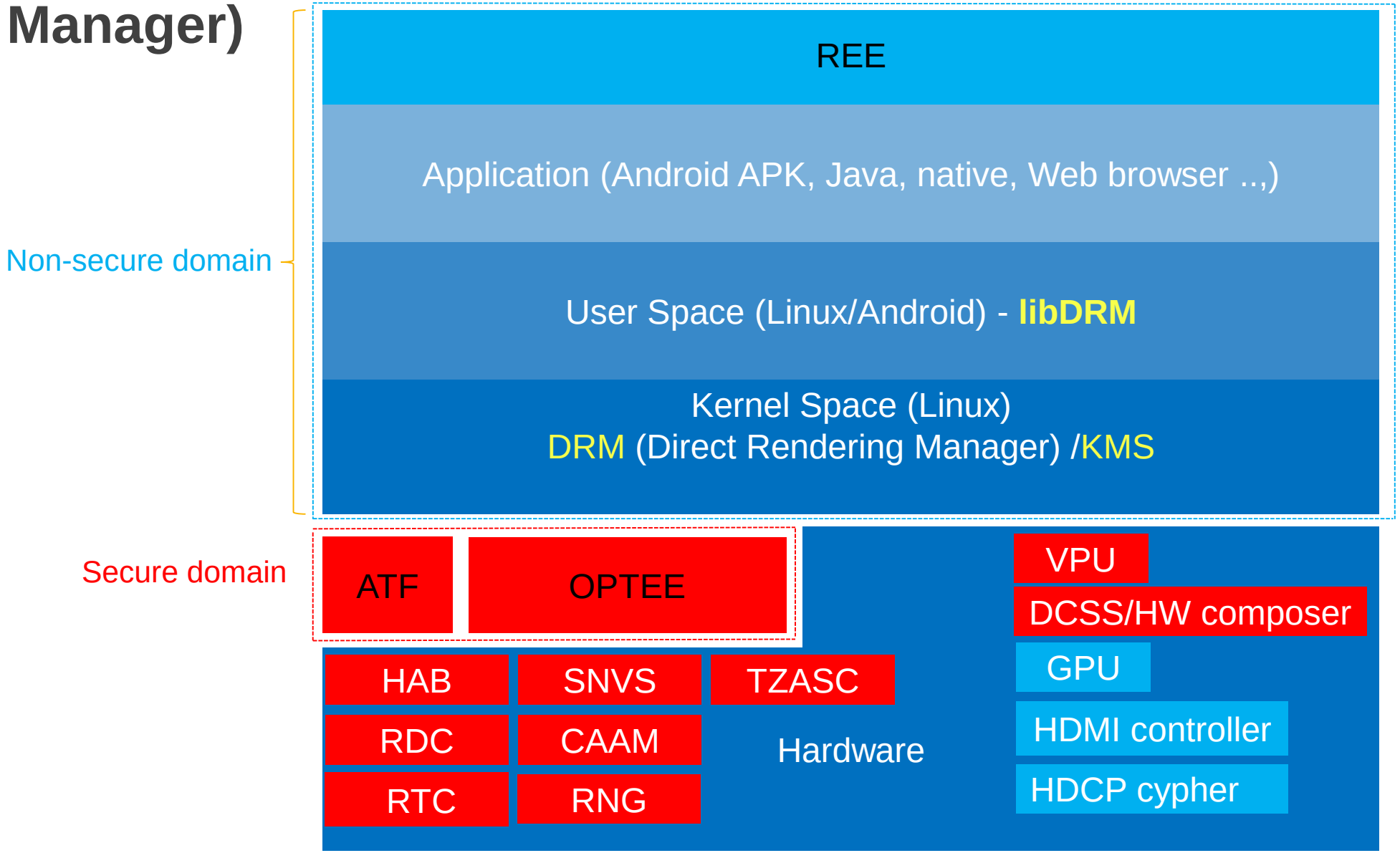
Problems:

RDC can protect the HDCP registers, but only by using static configuration of HDCP registers. This is not compliant with requirements, as we need to reconfigure HDCP configuration dynamically depending the display connected to the device, or the videob content to be displayed.

Conclusion:

we need to protect HDCP registers with TZASC, and make sure REE doesn’t disable HDCP protection while secure video is displayed.

i.MX 8MQ – HDCP today managed by libDRM (Direct Rendering Manager)



Access to HDCP configurations= and status done through “Content Protection” property / IOCTL by KMS/DRM

DRM_MODE_CONTENT_PROTECTION_UNDESIRED = 0

DRM_MODE_CONTENT_PROTECTION_DESIRED = 1

DRM_MODE_CONTENT_PROTECTION_ENABLED = 2

.

Secure Driver

Non secure driver

I.MX 8MQ – current HDCP management in Linux

- HDCP support for i.MX 8MQ: available since NXP Linux bsp 4.14.98-2.0.0_ga:
  $ repo init -u https://source.codeaurora.org/external/imx/imx-manifest -b imx-linux-sumo -m imx-4.14.98-2.0.0_ga.xml
  $ repo sync
  $ DISTRO=fsl-imx-xwayland MACHINE=imx8mqevk source ./fsl-setup-release.sh -b build-xwayland
  $ bitbake fsl-image-qt5

- HDCP configured and managed by Linux:
  ./imx8mqevk/kernel-source/drivers/gpu/drm/imx/hdp/imx-hdp.c
  ./imx8mqevk/kernel-source/drivers/gpu/drm/msm/hdmi/hdmi_hdcp.c

- HDCP IP is not available for all i.MX 8MQ socs.

- HDCP not enabled by default. Weston service shall be stop prior to enable it.
  systemctl stop weston.service
  modetest -w 46:"Content Protection":1
  systemctl start weston.service

- . Hardware IP from Cadence, with non-open source firmware.

HDCP MANAGEMENT BY OPTEE

HDCP managed by OPTEE

Prevent Linux to change HDCP configuration:

-> TZASC used to protect HDCP registers.

-> Trusted Application to write and read HDCP registers.

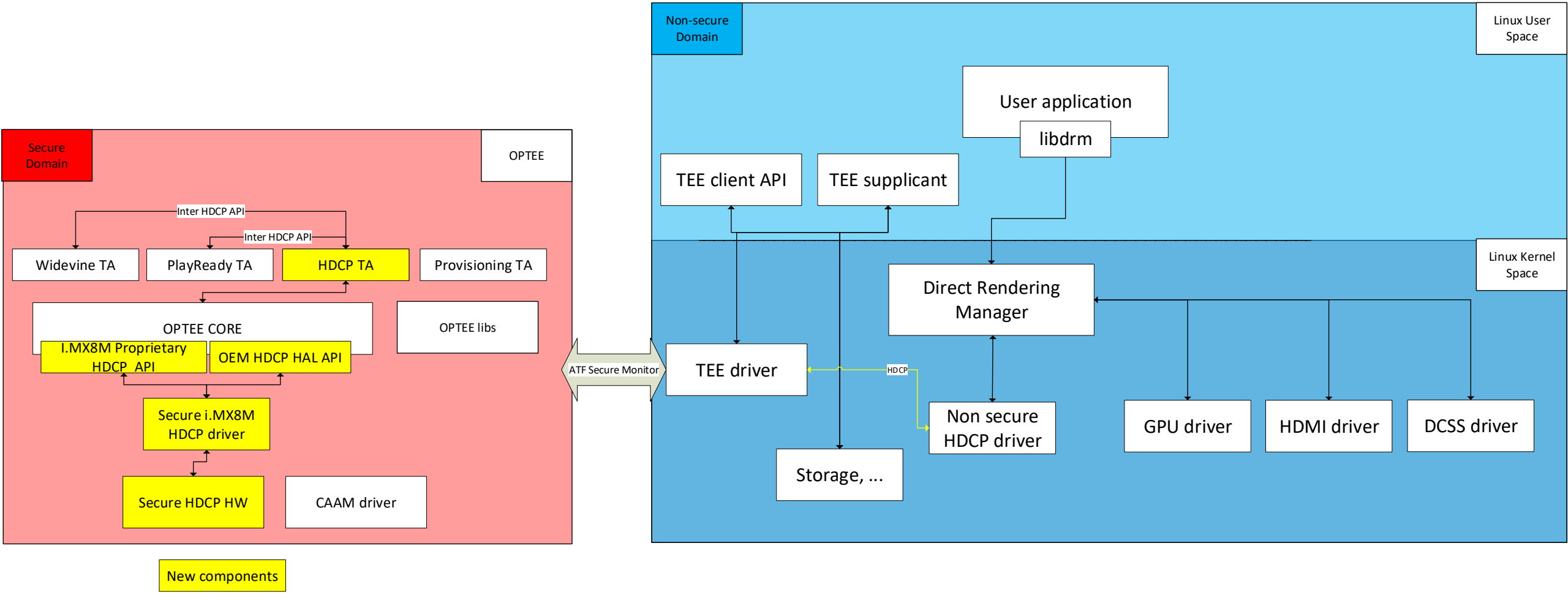
Provide a trusted mechanism for third party Trusted Application (Widevine/PlayReady) to get HDCP status/version, without relying on REE OS.

Open Source HDCP Trusted Application, monitoring HDCP register access, to block configuration changes while Widevine/PlayReady session are open.

Open Source OEM HDCP API allowing soc manufacturers to add support of their own HDCP hardware IP

Most part of HDCP driver remains at REE OS level. A small OPTEE proprietary driver shall allow TEE OS to read and write into the registers.

i.MX 8MQ – HDCP proposal managed by libDRM and OPTEE



HDCP TA API 1/3

HDCP_Result **ta_hdcp_get_capability**(uint32_t **param_types**, TEE_Param **params**[4]);

Description:

Returns the **maximum** HDCP version supported by the device, and the **current** HDCP version supported by the device and any connected display.

This function shall return HDCP_ERROR_INTERNAL_ERROR for any errors returned from OEM internal secure HDCP API.

```
typedef enum HDCP_Version
{
    HDCP_NONE  = 0,          // No HDCP supported,
    HDCP_V1_X  = 1,          // HDCP Version 1.X
    HDCP_V2_0  = 2,          // HDCP Version 2.0
    HDCP_V2_1  = 3,          // HDCP Version 2.1
    HDCP_V2_2  = 4,          // HDCP Version 2.2 type 1
    HDCP_V2_3  = 5,          // HDCP Version 2.3 type 1
    HDCP_NO_DIGITAL_OUTPUT = 0xff    // No digital output.
} HDCP_Version;
```

Parameters:

param_types (in): TEE_PARAM_TYPES(TEE_PARAM_TYPE_VALUE_OUTPUT,
TEE_PARAM_TYPE_NONE,
TEE_PARAM_TYPE_NONE,
TEE_PARAM_TYPE_NONE);

params[0].value.a (out) : this is the current **HDCP_Capability**, based on the device itself, and the display to which it is connected.

params[0].value.b (out) : this is the maximum supported **HDCP_Capability** version for the device, ignoring any attached device.

Threading:

This function may be called simultaneously with any other functions.

Returns:

HDCP_SUCCESS
HDCP_ERROR_BAD_PARAMETERS
HDCP_ERROR_INTERNAL_ERROR

HDCP TA API 2/3

HDCP_Result **ta_hdcp_open_session**(uint32_t **param_types**, TEE_Param **params**[4]);

Description:

Shall be called by Widevine and PlayReady TA to open an HDCP Session, and notify HDCP TA a secure video path is used. This function shall support the maximum number of session supported by Widevine and PlayReady TA -> (50 for Widevine, 1 for PlayReady). By calling this function, Widevine and PlayReady TA register their session to the HDCP TA.
If maximum of session the HDCP TA is able to manage has been reached,
HDCP_ERROR_TOO_MANY_SESSIONS shall be returned.

Parameters:

•param_types (in): TEE_PARAM_TYPES(TEE_PARAM_TYPE_VALUE_OUPUT,
TEE_PARAM_TYPE_NONE,
TEE_PARAM_TYPE_NONE,
TEE_PARAM_TYPE_NONE);

•params[0].value.a (out) **HDCP_SESSION**

Threading:

This function may be called simultaneously with any other functions.

Returns:

HDCP_SUCCESS
HDCP_ERROR_BAD_PARAMETERS
HDCP_ERROR_TOO_MANY_SESSIONS

HDCP TA API 3/3

HDCP_Result **ta_hdcp_close_session**(uint32_t **param_types**, TEE_Param **params**[4]);

Description:

Shall be called by Widevine or PlayReady TA to close an existing HDCP Session, and notify HDCP TA a secure video path is still needed or not. If session doesn't exist, HDCP_ERROR_CLOSE_SESSION_FAILED shall be returned.

Parameters:

param_types (in): TEE_PARAM_TYPES(TEE_PARAM_TYPE_VALUE_INPUT,
TEE_PARAM_TYPE_NONE,
TEE_PARAM_TYPE_NONE,
TEE_PARAM_TYPE_NONE);

params[0].value.a (in) **HDCP_SESSION** to close.

Threading:

This function may be called simultaneously with any other functions.

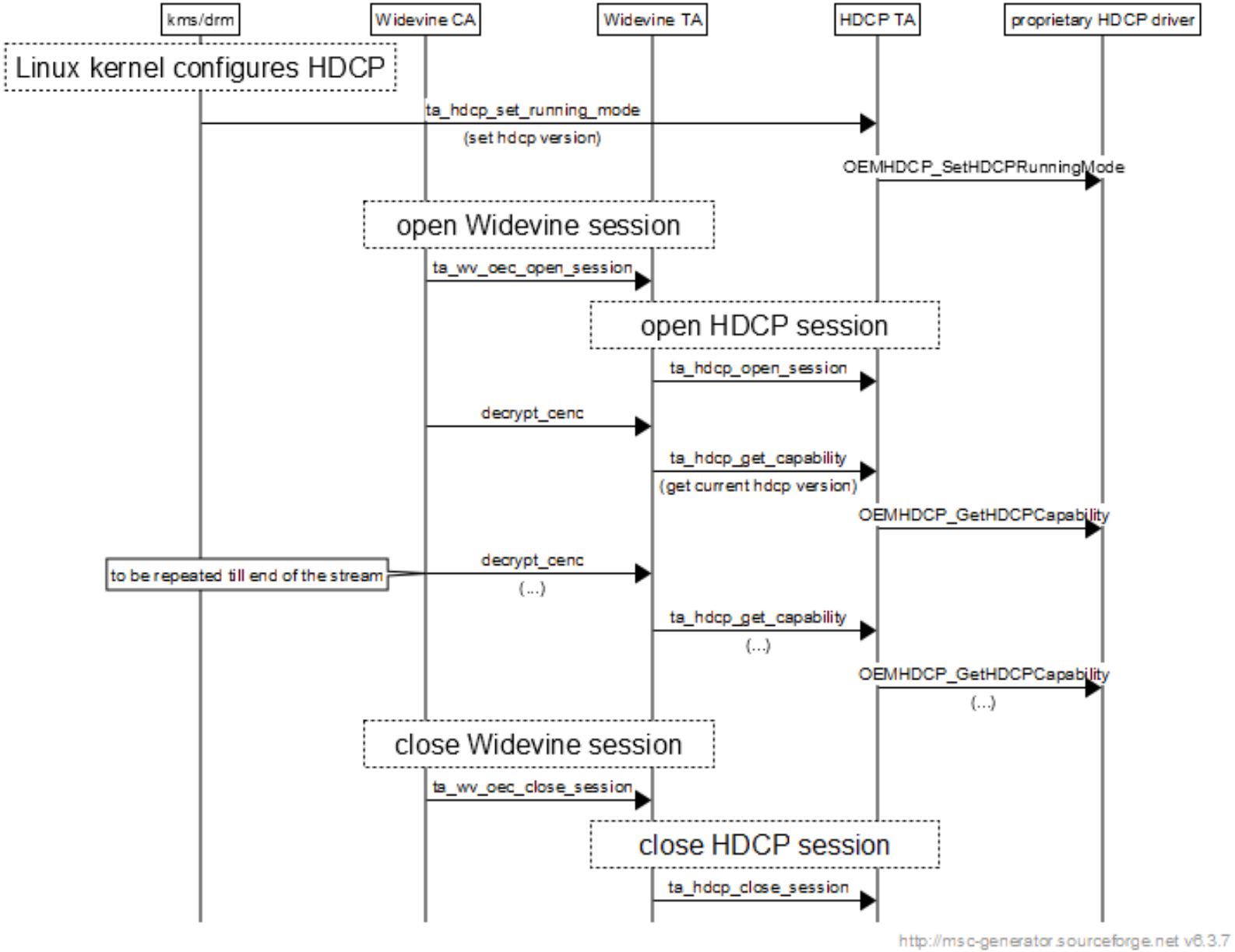
Returns:

HDCP_SUCCESS
HDCP_ERROR_BAD_PARAMETERS
HDCP_ERROR_CLOSE_SESSION_FAILED

HDCP Proprietary API

- Shall provide read and write function to HDCP registers
- When HDCP sessions are on going, shall block any write access changing HDCP version.

Widevine


kms/drm
Widevine CA
Widevine TA
HDCP TA
proprietary HDCP driver
Linux kernel configures HDCP
ta_hdcp_set_running_mode
(set hdcp version)
OEMHDCP_SetHDCPRunningMode
open Widevine session
ta_wv_oec_open_session
open HDCP session
ta_hdcp_open_session
decrypt_cenc
ta_hdcp_get_capability
(get current hdcp version)
OEMHDCP_GetHDCPCapability
to be repeated till end of the stream
decrypt_cenc
(...)
ta_hdcp_get_capability
(...)
OEMHDCP_GetHDCPCapability
(...)
close Widevine session
ta_wv_oec_close_session
close HDCP session
ta_hdcp_close_session
http://msc-generator.sourceforge.net v6.3.7

Widevine decrypt_cenc

Code to manage HDCP version is already there.

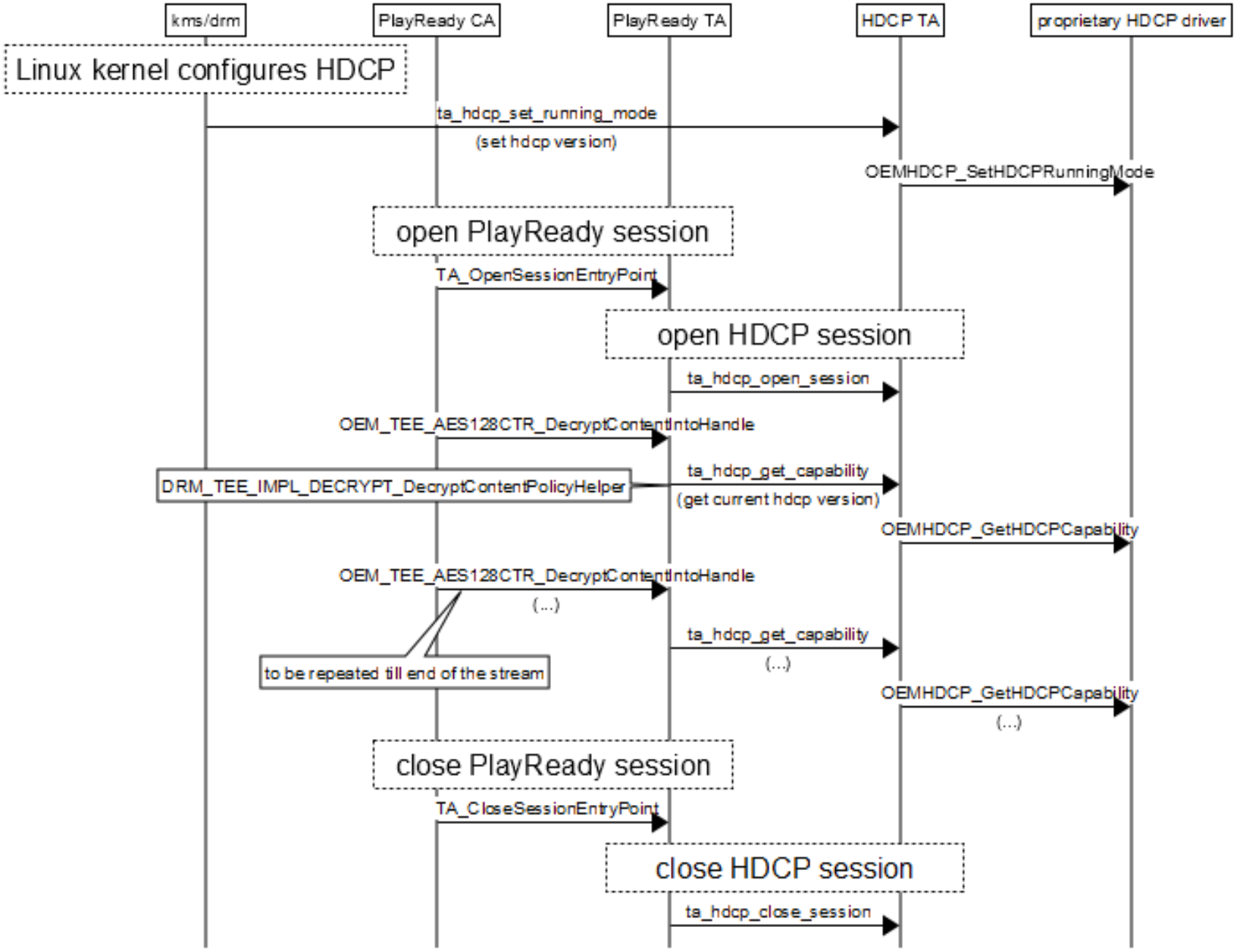
```
required_hdcp = ((control->control_bits & controlhdcpversionmask) >> CONTROL_HDCP_VERSIONSHIFT);
if ( current_hdcp_capability() == 0 ||   required_hdcp > current_hdcp_capability())
{
        return OEMCrypto_ERROR_INSUFFICIENT_HDCP;
}
....
```

But implementation of current_hdcp_capability() needs to be improved:

```
bool local_display(void)
{
    return true;
}
static OEMCrypto_HDCP_Capability current_hdcp_capability(void)
{
     return local_display() ? HDCP_NO_DIGITAL_OUTPUT : HDCP_V1;
}
```

We need to connect current_hdcp_capability() to OEMHDCP_GetHDCPCapability () and no more use local_display() ? HDCP_NO_DIGITAL_OUTPUT : HDCP_V1;

PlayReady


kms/drm
PlayReady CA
PlayReady TA
HDCP TA
proprietary HDCP driver
Linux kernel configures HDCP
ta_hdcp_set_running_mode
(set hdcp version)
OEMHDCP_SetHDCPRunningMode
open PlayReady session
TA_OpenSessionEntryPoint
open HDCP session
ta_hdcp_open_session
OEM_TEE_AES128CTR_DecryptContentIntoHandle
DRM_TEE_IMPL_DECRYPT_DecryptContentPolicyHelper
ta_hdcp_get_capability
(get current hdcp version)
OEMHDCP_GetHDCPCapability
OEM_TEE_AES128CTR_DecryptContentIntoHandle
(...)
ta_hdcp_get_capability
(...)
to be repeated till end of the stream
OEMHDCP_GetHDCPCapability
(...)
close PlayReady session
TA_CloseSessionEntryPoint
close HDCP session
ta_hdcp_close_session
http://msc-generator.sourceforge.net v6.3.7

PlayReady OEM_TEE_AES128CTR_DecryptContentIntoHandle

- **HDCP version check is not yet implemented** and shall be added in all DRM_CALL DRM_TEE_AES128CTR_DecryptContent like functions, ideally before each call to Oem_Aes_CtrProcessDataIntoOutput().

- PlayReady use OPL (Output Protection Level)

Mapping proposal for OPL and HDCP versions:

If current HDCP_Version between device and display is	OPL shall be in below range to allow PlayReady TA to decrypt video data
HDCP_NONE / Analog video	[0 -200[
HDCP_V1_X	[201-250[
HDCP_V2_0	[201-250[
HDCP_V2_1	[201 -250[
HDCP_V2_2 type 0	[201 -250[
HDCP_V2_3 type 0	[201 -250[
HDCP_V2_2 type 1	[251 -300[
HDCP_V2_3 type 1	[251 -300[

QUESTIONS