CC-API CCC Proposal TechTalk

Intel: *Lu, Ken/Xia, Haidong/Yao, Jiewen*

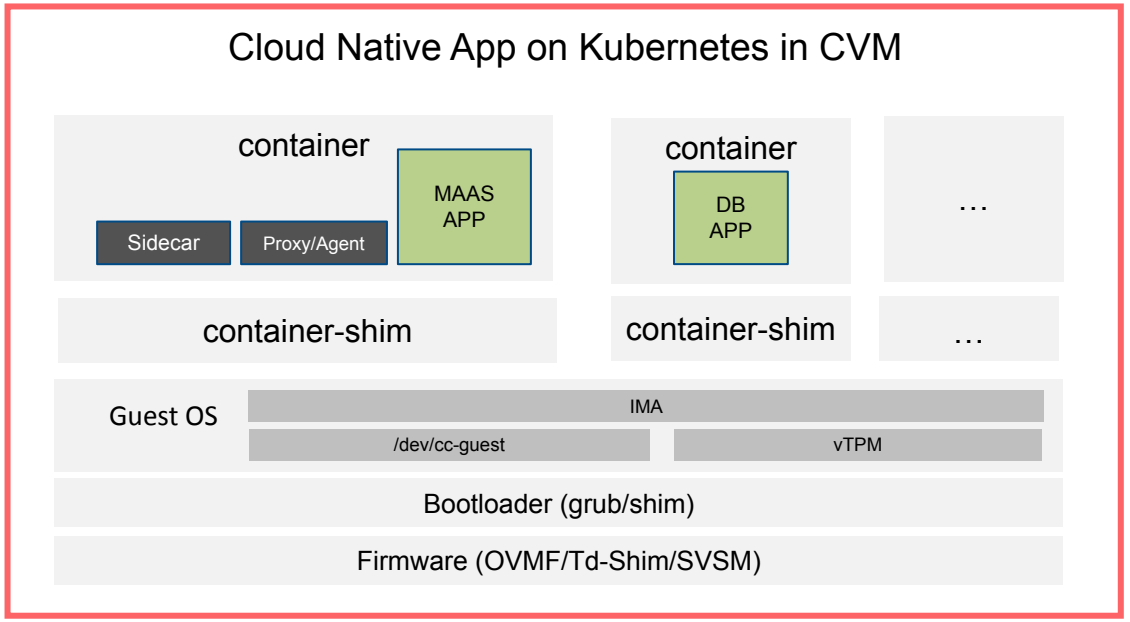
Bytedance/Tiktok: *Zhang, Wenhui/Jianjun Chen*

Contributors: Xie May; Ying Ruoyu; Dong, Xiaocheng; Chen, Hairong; Bhandaru Malini; Dan Middleton; Peng Liu; Hou Yu; Bing Duan; Trent Jaeger; Timothy Grance

Sponsor: Perez, Ronald

Motivation - Use Scenarios of TEE and Requirements

Confidential **Database** Service and Confidential **Model-as-a-Service** deployments commonly encompass multiple containers distributed across several virtual machines (VMs).



Microservice-like app on a confidential cluster

This deployment model necessitate attestation for userspace code:

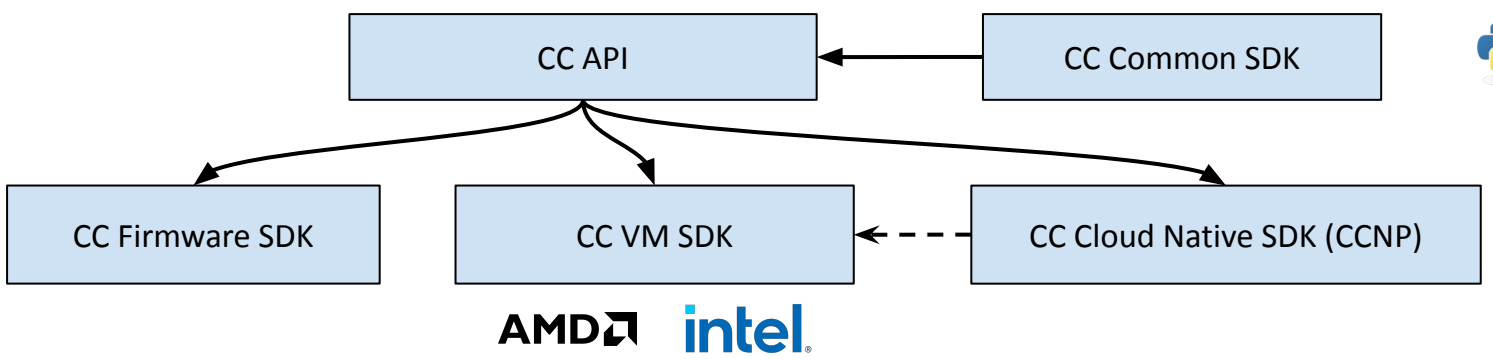
- **container launch time** integrity measurement and attestation
- **application initialization time** integrity measurement and attestation

Motivation - Pain Points of the Deployments

- Different hardware vendors exhibit disparities when exposing hardware interfaces to attestors
  - Current Infrastructure as a Service (IaaS) typically does not completely shield these hardware differences. This might result in different approaches to application attestation on various hardware platforms.
  - For instance, Intel TDX supports attestation for applications through **RTMR**. Additionally, Intel provides an IMA backend to RTMR, which is supported by two non-upstream versions of patches. On the other hand, AMD SEV-SNP supports application attestation through **vTPM** and other means.
-
- Support for application attestation from different hardware vendors requires manual intervention by users.
  - Users need to **manually analyze application dependencies**, create attestation policies, and integrate them into the authentication process using specific interfaces (Intel uses the IMA interface, while AMD uses the vTPM interface).
  - Furthermore, the interface solutions provided by Intel and AMD differ. Intel TDX supports application attestation through the **IMA interface**, whereas AMD SEV-SNP supports it through the **HMAC_Update** interface, ARM supports this through RSI interfaces.

Goal

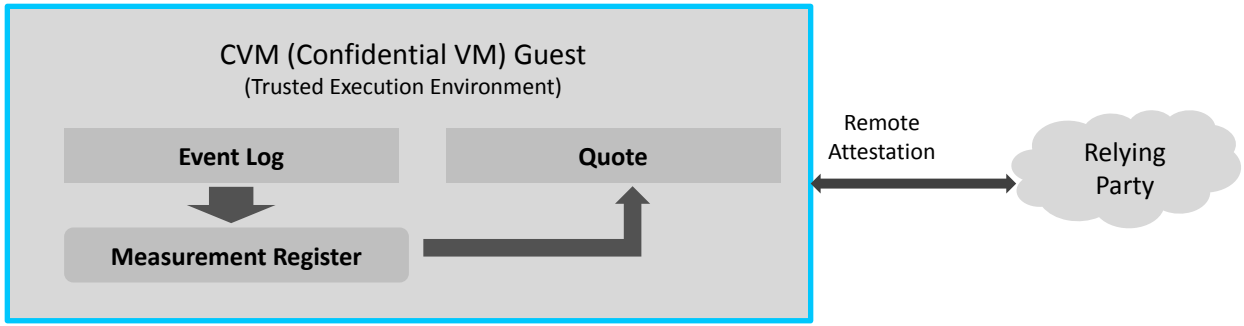
- Provide ***vendor agnostic*** and ***Trusted Computing Group (TCG) compliant*** APIs, which provides an abstraction layer above the various confidential primitives (i.e., measurement, event log, quote) for zero-trust design, supporting
  - *multiple deploy environments* (firmware/VM/cloud native cluster)
  - *multiple programming languages* within CVM (confidential virtual machine) guest.

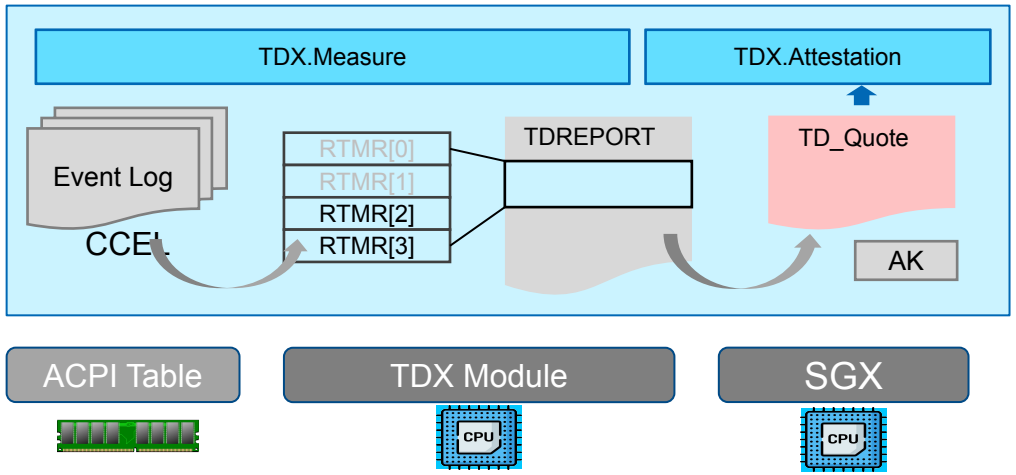


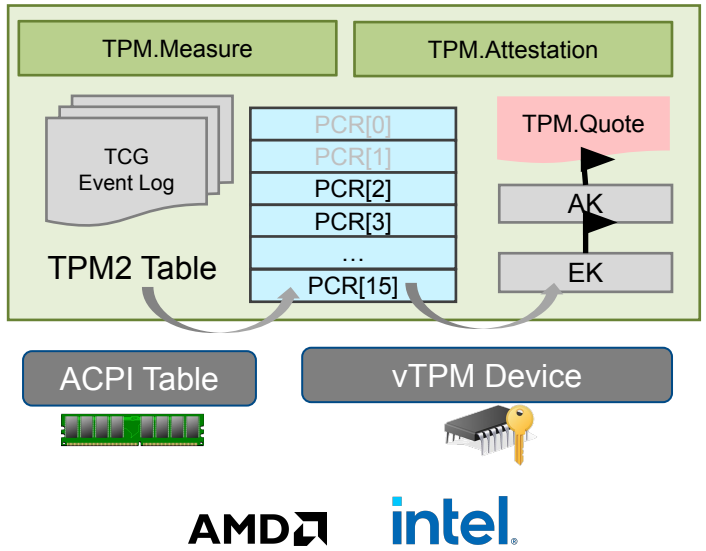
Problem Statement (1) – Diverging Confidential Primitive

To achieve zero-trust design, remote attestation process requires signed evidence (quote) reporting the integrity measurement, and event log originated from a RTM.

However, different vendors provide different API primitives.

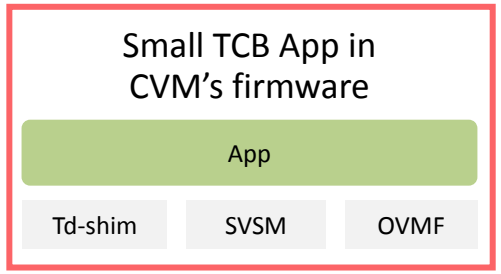






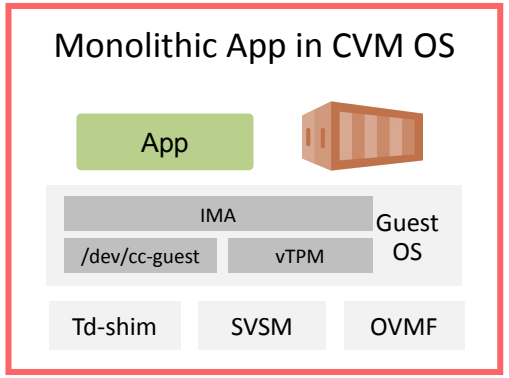
Problem Statement (2) – Complex Deploy Environment

FW level



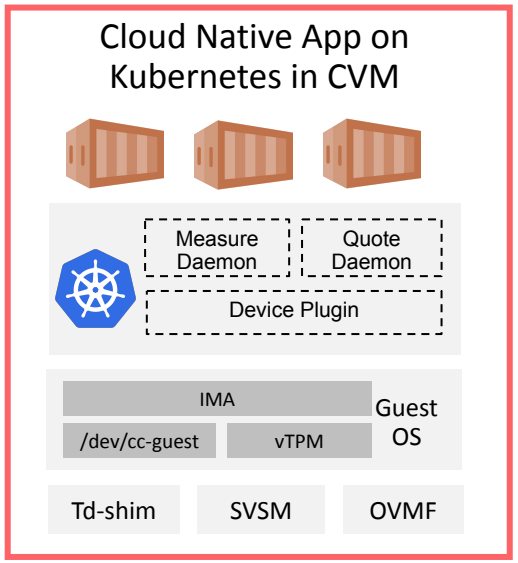
Small TCB application
e.g, Mig-TD, vTPM-TD

VM level



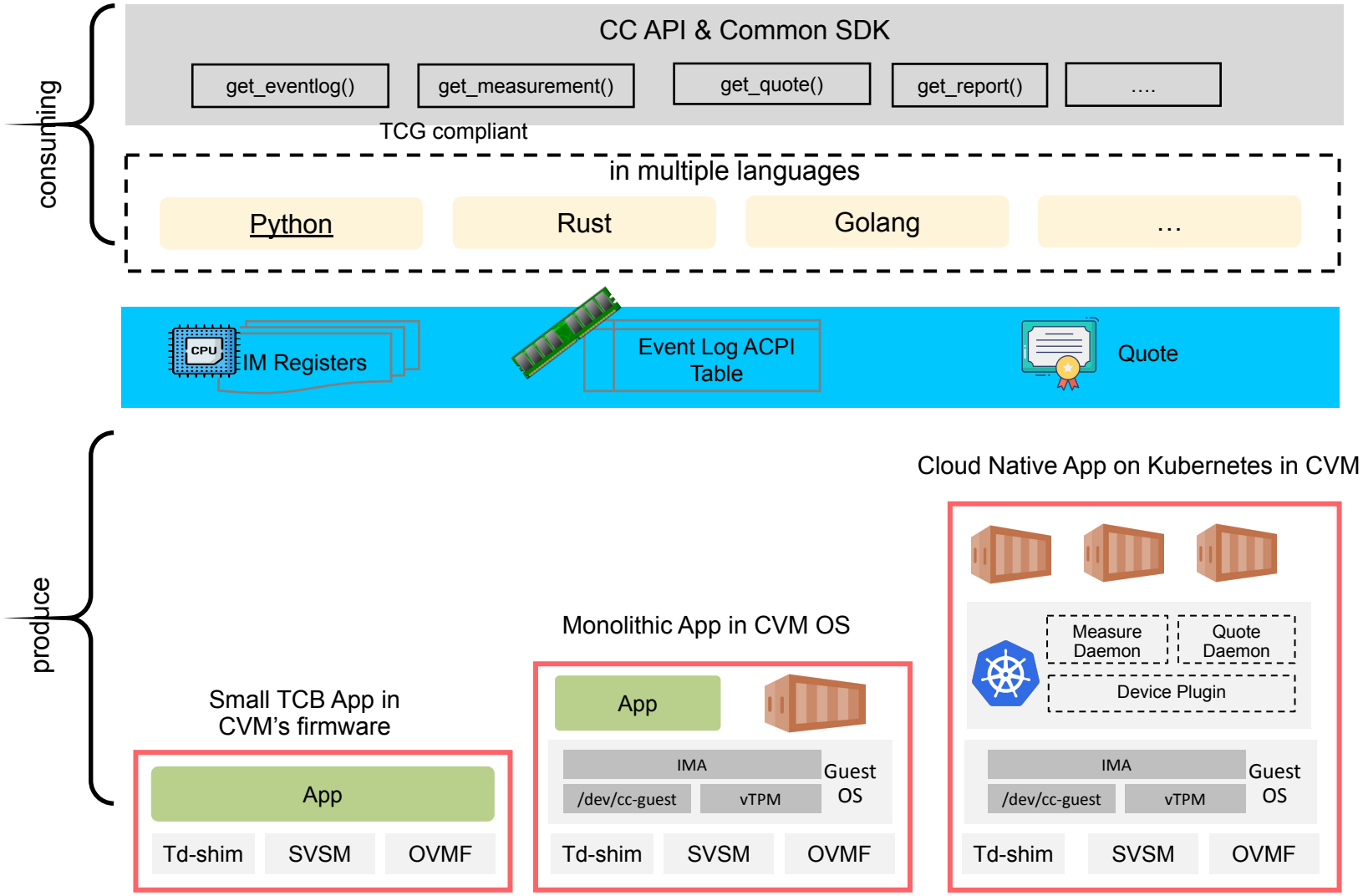
Traditional VM based cloud
application using IMA

Cloud Native Cluster Level



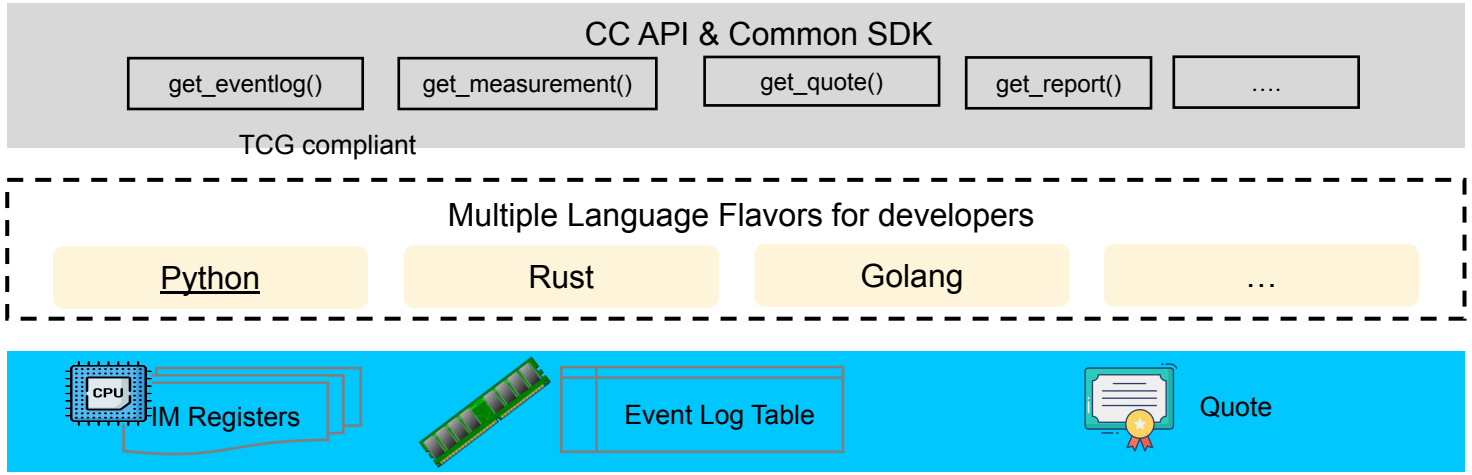
Microservice-like app on a
confidential cluster

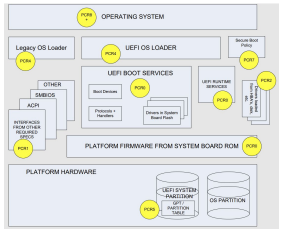
Unified CC API & SDK For applications



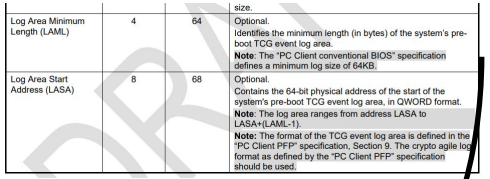
- Supporting different HW vendor in multiple programming languages
- Supporting different deployment with different TCBs (firmware/VM/cloud native cluster) via different approaches (TDVMCALL/device node/daemonset)

Unified CC API & SDK For applications - cont.

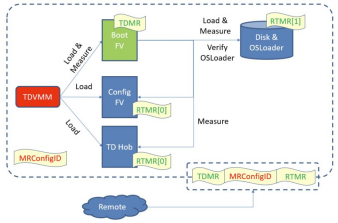




PCR: TCG PC PFP spec



TPM2 table: TCG_TPM2 ACPI Table spec



RTMR: TDVF spec

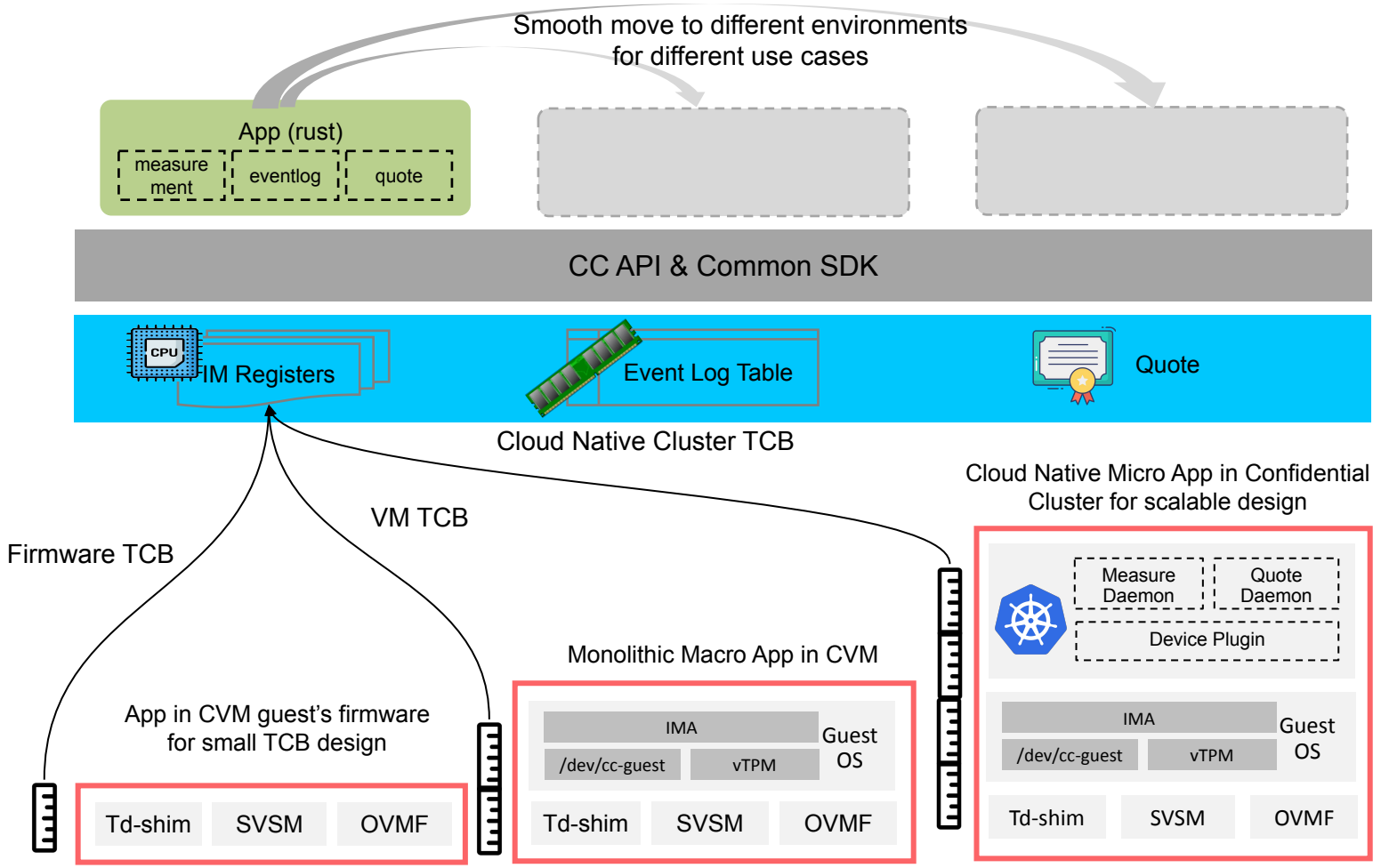
pcrIndex	UINT32
eventType	UINT32
digests	TPM_DIGEST
event	BYTE[]

Event log: TCG PC PFP spec

TCG Specification compliant data structure:

- IMR(Integrity Measurement Register): PCR for vTPM, RTMR for TDX
- Event Log ACPI table: TPM2 table for vTPM, CCEL for TDX

Unified CC API & SDK For applications - cont.



- Back-end SDK helps to produce the diverse launch time and runtime measurement for different TCBs.
- Simplify the deployment effort cross complex environments

Backup

TDX Event Log Structure ~= TCG

```
///
/// Event Log Entry Structure Definition
///
typedef struct tdTCG_PCR_EVENT {
  TCG_PCRINDEX      PCRIndex;           ///< PCRIndex event extended to
  TCG_EVENTTYPE     EventType;          ///< TCG EFI event type
  TCG_DIGEST        Digest;             ///< Value extended into PCRIndex
  UINT32            EventSize;          ///< Size of the event data
  UINT8             Event[1];           ///< The event data
} TCG_PCR_EVENT;

#define TSS_EVENT_DATA_MAX_SIZE  256

///
/// TCG_PCR_EVENT_HDR
///
typedef struct tdTCG_PCR_EVENT_HDR {
  TCG_PCRINDEX      PCRIndex;
  TCG_EVENTTYPE     EventType;
  TCG_DIGEST        Digest;
  UINT32            EventSize;
} TCG_PCR_EVENT_HDR;
```

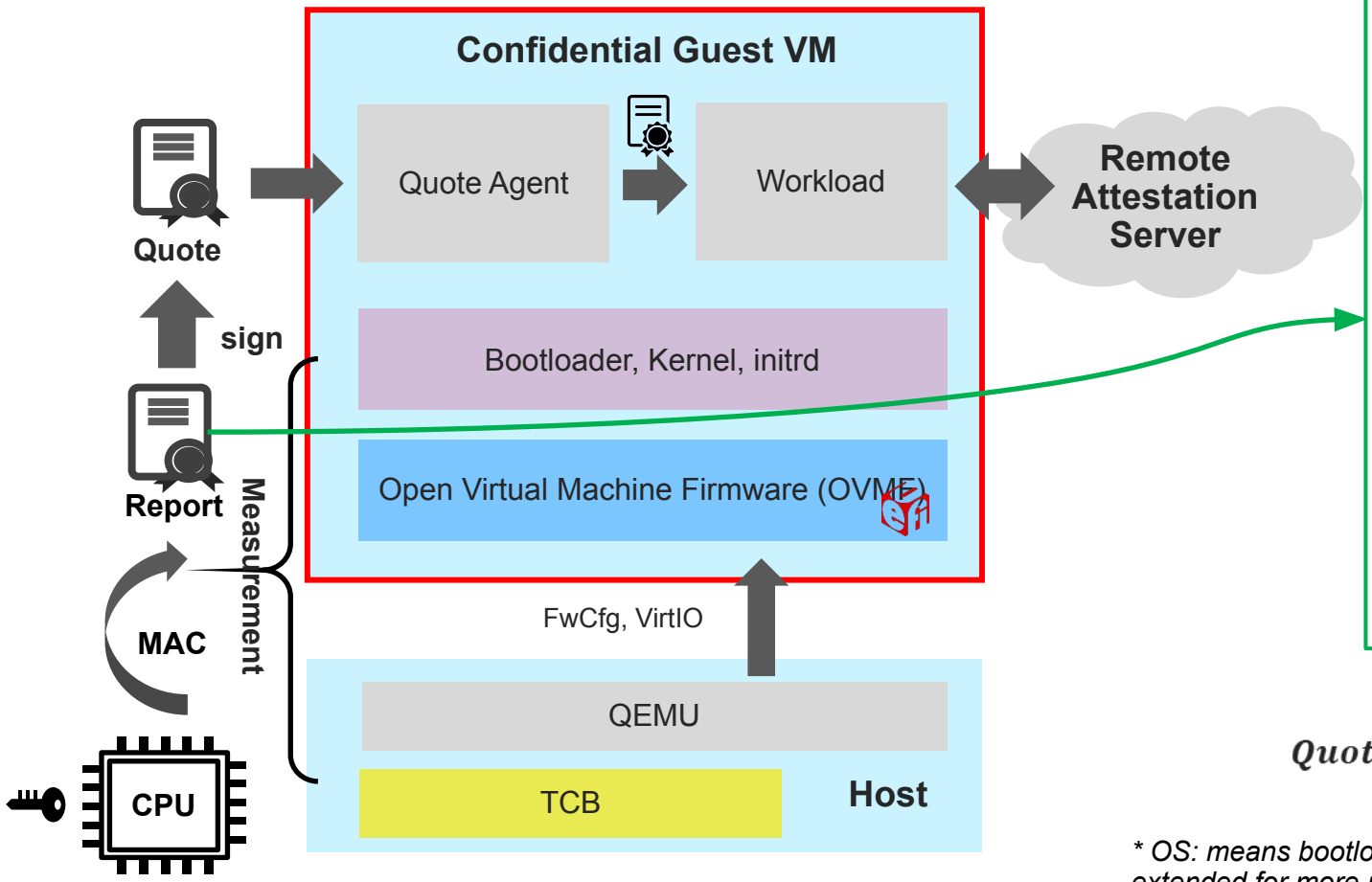
```
==== TDX Event Log Entry - 14 [0x7CBA771F] ====
RTMR             : 1
Type             : 0x80000007 (EV_EFI_ACTION)
Length           : 106
Algorithms ID    : 12 (TPM_ALG_SHA384)
Digest[0] :
00000000  77 A0 DA B2 31 2B 4E 1E 57 A8 4D 86 5A 21 E5 B2  w...1+N.W.M.Z!..
00000010  EE 8D 67 7A 21 01 2A DA 81 9D 0A 98 98 80 78 D3  ..gz!.*.......x.
00000020  D7 40 F6 34 6B FE 0A BA A9 38 CA 20 43 9A 8D 71  .@.4k....8. C..q
RAW DATA: ------------------------------------------------
7CBA771F  02 00 00 00 07 00 00 80 01 00 00 00 0C 00 77 A0  ..............w.
7CBA772F  DA B2 31 2B 4E 1E 57 A8 4D 86 5A 21 E5 B2 EE 8D  ..1+N.W.M.Z!....
7CBA773F  67 7A 21 01 2A DA 81 9D 0A 98 98 80 78 D3 D7 40  gz!.*.......x..@
7CBA774F  F6 34 6B FE 0A BA A9 38 CA 20 43 9A 8D 71 28 00  .4k....8. C..q(.
7CBA775F  00 00 43 61 6C 6C 69 6E 67 20 45 46 49 20 41 70  ..Calling EFI Ap
7CBA776F  70 6C 69 63 61 74 69 6F 6E 20 66 72 6F 6D 20 42  plication from B
7CBA777F  6F 6F 74 20 4F 70 74 69 6F 6E                    oot Option
RAW DATA: ------------------------------------------------
```

TDX Event Log Sample

Refer

- *HashCompleteAndExtend implementation for TDX*

Guest Measurement Chain



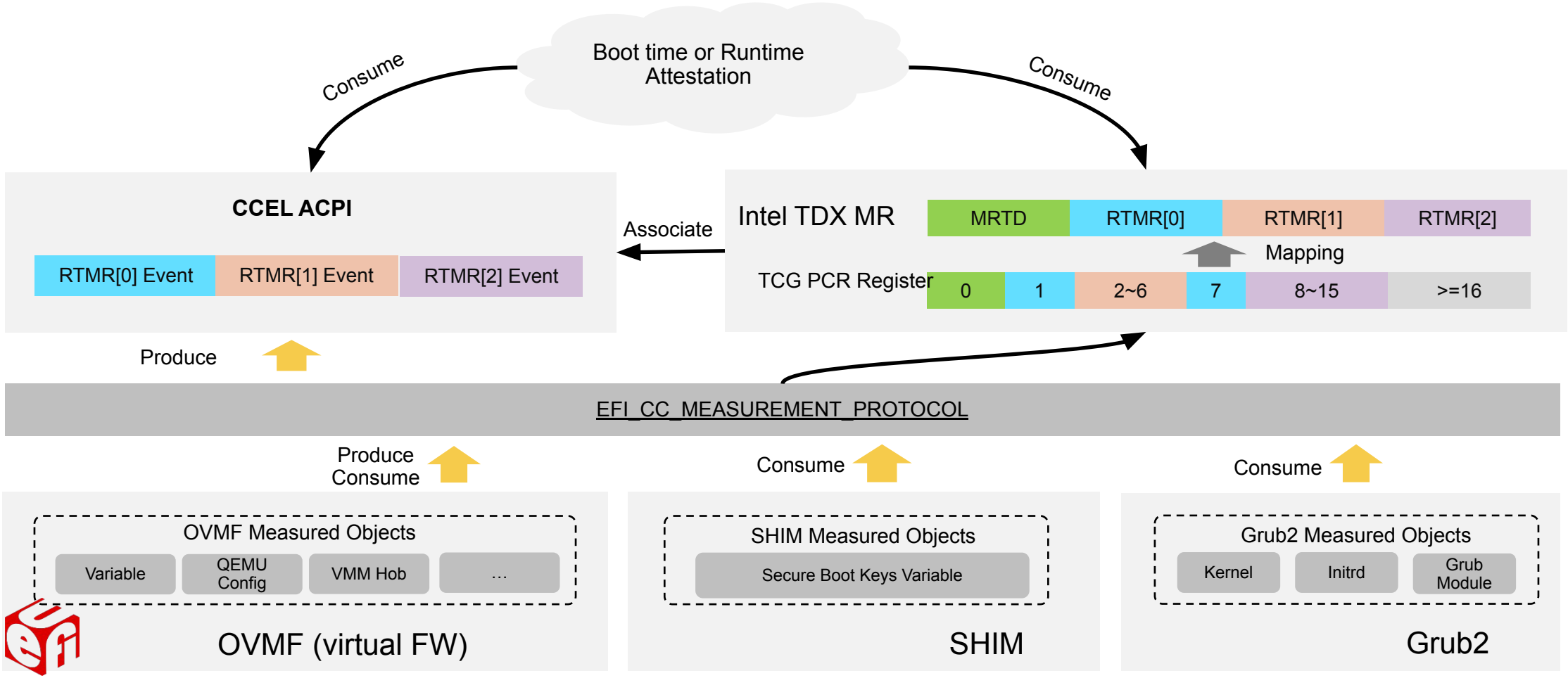
TD_REPORT (here)

0x000	REPORTMACSTRUCT	REPORTTYPE	RESERVER	0x200	TDINFO	ATTRIBUTES	XFAM
0x010		CPUSVN		0x210		MRTD	
0x020		TEE_TCB_INFO_HASH		0x220			
0x030				0x230			
0x040				0x240			
0x050				0x250		MRCONFIGID	
0x060		TEE_INFO_HASH		0x260			
0x070				0x270			
0x080				0x280		MROWNER	
0x090				0x290			
0x0A0		REPORTDATA		0x2A0			
0x0B0				0x2B0		MROWNERCONFIG	
0x0C0				0x2C0			
0x0D0				0x2D0			
0x0E0		RESERVE		0x2E0		RTMR[0]	
0x0F0		MAC		0x2F0			
0x100	TEE_TCB_INFO	VALID	TEE_TCB_SVN	0x300		RTMR[1]	
0x110		TEE_TCB_SVN	MRSEAM	0x310			
0x120		MRSEAM		0x320			
0x130				0x330		RTMR[2]	
0x140		MRSEAM	MRSIGNERSEAM	0x340			
0x150		MRSIGNERSEAM		0x350			
0x160				0x360		RTMR[3]	
0x170		MRSIGNERSEAM	ATTRIBUTES	0x370			
0x180		RESERVED		0x380			
0x190				0x390			
0x1A0				0x3A0		RESERVED	
0x1B0				0x3B0			
0x1C0				0x3C0			
0x1D0				0x3D0			
0x1E0				0x3E0			
0x1F0	RESERVED	RESERVED		0x3F0			

$$\boldsymbol{Quote} = Sign\left(\sum M_{TCB} + M_{OVMF} + \mathrm{M}_{OS}\right)$$

** OS: means bootloader (like shim/grub), kernel, initrd. It can be also extended for more runtime measurement like dynamic kernel module.*

CC Measurement Protocol



- UEFI 2.10
- ACPI 6.5

PyTdxMeasure

A Python tool running in Intel TDX guest to dump and verify the RTMR, Intel TDX event log:

- Launch an Intel TDX guest
- Install pytdxmeasure
  python3 -m pip install pytdxmeasure
- Dump TDREPORT
  tdx_tdreport
- Dump Intel TDX event log
  tdx_eventlog
- Verify RTMR
  tdx_verify_rtmr

```
==== TDX Event Log Entry - 71 [0x1C331F50] ====
RTMR              : 2
Type              : 0xD (EV_IPL)
Length            : 161
Algorithms ID     : 12 (TPM_ALG_SHA384)
Digest[0] :
00000000  E8 09 21 B2 EE 7B D3 C9 24 2D 7D 2B 60 CA D2 8A  ..!..{..$-}+`...
00000010  7F E6 8E 4C 8E A3 5A AE F3 E6 21 81 3B 60 E8 FF  ...L..Z...!.;`..
00000020  BE BE 2A AF ED 50 03 23 BE D6 4E 99 B9 71 6A 70  ..*..P.#..N..qjp
RAW DATA: ---------------------------------------------
1C331F50  03 00 00 00 0D 00 00 00 01 00 00 00 0C 00 E8 09  ................
1C331F60  21 B2 EE 7B D3 C9 24 2D 7D 2B 60 CA D2 8A 7F E6  !..{..$-}+`.....
1C331F70  8E 4C 8E A3 5A AE F3 E6 21 81 3B 60 E8 FF BE BE  .L..Z...!.;`....
1C331F80  2A AF ED 50 03 23 BE D6 4E 99 B9 71 6A 70 5F 00  *..P.#..N..qjp_.
1C331F90  00 00 67 72 75 62 5F 63 6D 64 20 69 6E 69 74 72  ..grub_cmd initr
1C331FA0  64 20 28 68 64 30 2C 6D 73 64 6F 73 33 29 2F 62  d (hd0,msdos3)/b
1C331FB0  6F 6F 74 2F 69 6E 69 74 72 61 6D 66 73 2D 35 2E  oot/initramfs-5.
1C331FC0  31 35 2E 30 2D 53 50 52 2E 4D 56 50 2E 50 43 2E  15.0-SPR.MVP.PC.
1C331FD0  76 31 30 2E 34 2E 6D 76 70 34 30 2E 65 6C 38 2E  v10.4.mvp40.el8.
1C331FE0  78 38 36 5F 36 34 2B 67 75 65 73 74 2E 69 6D 67  x86_64+guest.img
1C331FF0  00                                               .
RAW DATA: ---------------------------------------------
```