Cerberus 项目：固件挑战规范

作者：

- 微软首席固件工程经理 Bryan Kelly
- Christopher Weimer，微软高级固件工程师
- Akram Hamdy，微软固件工程师

修订记录

- v0.01 (2017-08-28)
  - 初稿。
- v0.02 (2017-09-28)
  - 添加参考部分。
- v0.03 (2017-10-28)
  - 将消息交换从协议转移到基于寄存器。
- v0.04 (2018-12-02)
  - 添加 MCTP 支持和更新会话
- v0.05 (2018-30-04)
  - 纳入供应商反馈。
- v0.06 (2018-15-10)
  - 更新身份验证流程。
  - 将测量更改为 PMR 和认证集成。
- v0.07 (2019-10-01)
  - 由于对扩展的静态要求，将 PMR 命名更改为 PM。
- v0.08 (2019-15-02)
  - 添加固件恢复图像更新命令。
  - 澄清错误响应。
- v0.09 (2019-26-06)
  - 添加重置配置命令。
  - 识别受加密超时影响的命令。
- v0.10 (2019-05-08)
  - 更新 Cerberus 定义的 MCTP 消息定义。
- v0.11 (2019-21-10)
  - 添加有关设备制造商配对的详细信息。
  - 添加命令以获取 RIoT、芯片和主机重置信息。
- v0.12 (2019-27-12)
  - 关于必需和可选命令的说明。
- v0.13 (2020-17-03)
  - 添加命令以获取清单平台 ID 和 PMR 测量数据。
  - 更新开封和设备功能命令。
  - 对命令包格式的澄清。
  - 添加日志格式。
- v0.14 (2020-30-04)
  - 多个命令的更新格式，添加扩展更新状态。

    - 对证书的澄清。
    - 添加有关加密消息的详细信息。
- v0.15 (2020-22-05)
    - 添加解封 ECDH 种子参数。
    - 定义一系列保留命令。
- v1.00 (2020-19-08)
    - 更新会话建立和安全设备绑定。
    - 添加回 Rq 位。

(c) 2017 年微软公司。

合法的

自 2017 年 11 月 1 日起，以下个人或实体已根据开放网络基金会最终规范协议 (OWFa 1.0) 提供此规范，该协议可在 <http://www.openwebfoundation.org/legal/the-owf -1-0-协议/owfa-1-0>。

- 微软公司。

您可以在 http://www.opencompute.org/participate/legal-documents/ 查看本规范的 Open Web Foundation Agreement Version 1.0 的签名副本，其中可能还包括上面列出的其他方。

您对本规范的使用可能受其他第三方权利的约束。本规范"按原样"提供。贡献者明确否认任何保证（明示、暗示或其他方式），包括与规范相关的适销性、非侵权、特定用途适用性或所有权的暗示保证。实施或以其他方式使用规范的全部风险由规范实施者和用户承担。在任何情况下，任何一方均不对任何其他方因与本规范或其管理协议相关的任何类型的任何行为原因造成的利润损失或任何形式的任何性质的间接、特殊、偶然或后果性损害负责，无论基于违反合同、侵权行为（包括疏忽）或其他情况，

本规范的贡献者和许可人可能提到了某些仅在本规范中引用的技术，而不是根据 OWF CLA 或 OWFa 获得许可的技术。以下是仅供参考的技术列表： 智能平台管理接口（IPMI）；I2C 是 NXP SEMICONDUCTORS 的商标和技术；EPYC 是 ADVANCED MICRO Devices INC. 的商标和技术；ASPEED AST 2400/2500 系列处理器是 ASPEED TECHNOLOGY INC. 的一项技术；MOLEX NANOPITCH、NANO PICOBLADE 和 MINI-FIT JR 及相关连接器是 MOLEX LLC 的商标和技术；WINBOND 是 WINBOND ELECTRONICS CORPORATION 的商标；NVLINK 是 NVIDIA 的一项技术；英特尔至强可扩展处理器、英特尔 QUICKASSIST 技术、英特尔超线程技术、增强型英特尔 SPEEDSTEP 技术、英特尔虚拟化技术、英特尔服务器平台服务、英特尔管理引擎和英特尔可信执行技术是英特尔公司的商标和技术；SITARA ARM CORTEX-A9 处理器是德州仪器的商标和技术；来自 PENCOM 的导销；松下电池。这些技术的实施可能受其自身法律条款的约束。

概括

在本文档中，术语"处理器"是指所有中央处理器 (CPU)、片上系统 (SOC)、微控制单元 (MCU) 和微处理器架构。该文档详细说明了主动组件和平台 RoT 所需的挑战协议。处理器必须实现所有必需的功能以建立基于硬件的信任根。本质上无法满足这些要求的处理器必须实施闪存保护 Cerberus RoT 描述的物理闪存保护要求文档。

主动部件是包含处理器、微控制器或运行软逻辑的设备的附加卡和外围设备。

本文档描述了用于平台的 Active RoT 的固件证明测量的协议。该规范包括预启动、启动和运行时挑战以及平台固件完整性的验证。分层架构超越了典型的 UEFI 测量，包括所有主动器件固件的完整性测量。该文档描述了支持 Cerberus 项目的证明挑战所需的 API。

物理通信通道

典型的云服务器主板布局具有路由到所有主动部件的 I2C 总线。这些 I2C 总线通常由底板管理控制器 (BMC) 用于对主动部件进行热监控。在 Cerberus 板布局中，I2C 通道首先由平台 Cerberus 微控制器在启动和预启动期间使用，然后mux切换回 BMC 进行热管理。Cerberus 可以随时请求 BMC 放弃对运行时挑战和证明的控制。Cerberus 控制 I2C mux位置，并在运行时协调访问。Cerberus 还可以在运行时通过 BMC 代理命令，可选择链路加密和非对称密钥交换，使 BMC 对通信视而不见。主板上的 Cerberus 微控制器被称为平台主动信任根 (PA-RoT)。该微控制器是分层信任根平台设计的负责方，包含平台固件清单 (PFM) 和组件固件清单 (CFM) 中保存的所有平台固件的可证明散列。

大多数云服务器主板将 I2C 路由到主动部件以进行热监控，添加 mux 逻辑是对主板的唯一修改。除了添加额外的 mux，另一种替代做法是使用 BMC上的 I2C 建立安全的质询通道。一旦 BMC 被 Cerberus 加载和验证，它就可以充当 I2C 代理。这种方法不太可取，因为如果 BMC 认证失败，它会限制平台认证。在这两种方法中，主动部件接口的物理连接器都不需要更改，因为它们已经具有 I2C。

具有“处理器安全启动要求”文档中描述的固有安全属性的主动器件不需要将物理 Cerberus 微控制器放置在其处理器和闪存之间。不满足“处理器安全启动要求”文档中所述要求的主动器件需要在其处理器和闪存之间实施 Cerberus 微控制器，以建立所需的信任根。图 1 主板 I2C 通道图，表示主板 PA-RoT 和主动部件 RoT (AC-RoT) 之间的预启动和后启动测量挑战通道。

> TODO：图 1

Project Cerberus 固件认证是一个分层架构。现代服务器中的大多数主动器件在平台的主机处理器完成初始化并能够挑战设备之前启动到操作级别。在 Cerberus 设计中，平台保持在预上电或重置状态，从而隔离主动部件并对其固件测量进行质询。主动组件必须响应来自 PA-RoT 的挑战，以确认其固件的完整性，然后再将它们从隔离中取出。

在这个版本的 Cerberus 平台设计中，PFM 和 CFM 是静态的。清单可通过 PA-RoT 的通信接口进行编程。规范的未来版本将考虑主动组件的自动检测和 PFM/CFM 的计算。PFM 和 CFM 是允许的固件版本及其相应固件测量的清单。清单包含用于限制回滚的单调标识符。

PA-RoT 使用 CFM 中的测量值来挑战主动部件并比较它们的测量值。然后 PA-RoT 使用这些测量的摘要作为平台级测量，创建一个分层平台级摘要，可以证明平台和主动器件固件的完整性。

PA-RoT 将支持消息的身份验证、完整性和机密性。主动组件 RoT (AC-RoT) 将支持消息和质询的身份验证和完整性。为此，AC-RoT 需要支持证书认证。主动器件将支持用于身份验证的组件唯一 CA 签名质询证书。

> 注意：I2C 是一种低速链路，在优化协议消息和携带更高位数的强加密哈希算法之间存在性能权衡。不能支持证书身份验证的 RoT 需要支持散列算法和固件测量的 RSA 或 ECDSA 签名。

Power Control - 功率控制

在Cerberus主板设计中，电源和复位排序是由PA-RoT编排的。当电压施加到主板上时，它通过涌流电路传递到CPLD，该CPLD对电源轨进行时间敏感排序以确保稳定。一旦建立了电源良好水平，就认为平台已通电。Cerberus设计的平台在初始启动时，唯一启动的活动组件是PA- RoT。RoT首先安全地加载并解压缩其内部固件，然后通过测量BMC flash来验证Baseboard Management Controller (BMC)固件的完整性。当BMC固件通过认证后，Active RoT将使BMC上电。一旦BMC上电，Active RoT对平台UEFI的固件进行认证，在此期间，RoT序列上电到PCIe插槽并开始Active Component RoT挑战。当UEFI通过认证后，平台将处于系统复位状态，Active RoT将使系统保持复位状态，直到ac -RoT对测量挑战做出响应。任何不响应其测量挑战的PCIe端口将随后断

电。如果任何预期的活动组件未能响应测量挑战，Cerberus策略确定系统是否应该在活动组件断电的情况下启动，或者平台应该保持备用电源，同时通过OOB路径向数据中心管理软件报告测量失败。

Communication - 通信

Cerberus PA-RoT通过I2C与AC-RoT通信。该协议支持身份验证和测量质询。Cerberus PA-RoT生成一个安全的非对称密钥对，这是微控制器所独有的，紧跟DICE架构。私钥在Cerberus RoT的安全区域之外是不可访问的。密钥生成和链接遵循 RIoT 规范，在章节:9.3 DICE和RIoT密钥和证书中描述。派生的平台alias公钥可用于AC-RoT在挑战和震动期间的认证和通信，以建立通信。

RSA/ECDSA 密钥生成

Cerberus 平台 Active RoT 应支持设备标识符组合引擎 (DICE) 架构。在 DICE 中，复合设备标识符 (CDI) 用作设备身份和证明的基础。源自唯一设备机密 (UDS) 的密钥和第一个可变代码的测量用于微控制器内的数据保护和上层固件层的证明。设备 ID 非对称密钥对是从 CDI 以加密方式导出的，并与设备 ID 证书相关联。CDI 使用从 PUF 派生的 UDS 和其他随机熵，包括微控制器唯一 ID 和固件安全描述符。

Cerberus 实现了用于证书生成和认证的 RIoT 核心架构。有关 DICE 和 RIoT 密钥生成的详细信息，请查看部分：9.3 DICE 和 RIoT 密钥和证书。

注意：CDI 是基于 UDS、微控制器安全描述符和第二阶段引导加载程序（可变）测量的复合密钥。第二阶段引导加载程序是可变代码，但通常不会随固件更新而更新。对第二阶段引导加载程序的更改将导致不同的 CDI，从而导致不同的非对称设备 ID 密钥生成。与以前的设备密钥关联的证书将失效，需要签署新的证书。

第二个非对称密钥对证书在 Cerberus 的 RIoT 核心层中创建并传递给 Cerberus 应用程序固件。该密钥对构成别名证书，派生自 CDI、Cerberus 固件安全描述符和下一阶段 Cerberus 固件的度量。

CDI 派生私钥的知识证明（称为设备 ID 私钥）用作加密协议中的构建块以识别设备。设备 ID 私钥用于签署别名证书，从而验证密钥的完整性。在初始配置期间，设备 ID 证书是由 Microsoft 证书颁发机构签署的 CA。配置时，设备 ID 密钥必须与先前签名的公钥相匹配。

> TODO：图 2

> 注意：CDI 和设备 ID 私钥在退出 RIoT Core 之前被安全擦除。

软件的每一层都可以使用其私钥证书为下一层签名并颁发新证书，每个后续层都延续这个链条。应用层的证书（Alias Certificate）可以在设备认证和建立安全通道时使用。证书还可以确定真实性。用于签署证书的非应用层私钥必须只能由它们生成的层访问。公钥被持久化或传递给上层，并最终与上下游实体交换。

Chained Measurements - 链式测量

Cerberus 固件测量基于设备标识符组合引擎 (DICE) 架构：https://trustedcomputinggroup.org/work-groups/dice-architectures

RoT 上的第一个可变代码是第二引导加载程序 (SBL)。CDI 是`HMAC(Device Secret Key + Entropy, H(SBL))`）的测量值。然后，此测量值传递给第二阶段引导加载程序，计算第三引导加载程序 (TBL) 的摘要。在 Cerberus RoT 上，这是应用程序固件：`HMAC(CDI, H(TBL))`。

> TODO：图 4

运行 Cerberus 应用程序固件的第三阶段引导加载程序 (TBL) 将对其保护的处理器的 SPI/QSPI 闪存进行额外的区域测量，测量活动和非活动区域。使用证明新鲜度种子验证和扩展 TBL 测量。最终测量结果被签名、密封并提供给挑战软件。

应用程序固件证明固件的种子可以从 Cerberus 固件测量扩展，或者使用 Alias 证书可以提供专用新鲜度种子来测量受保护的处理器固件。

测量值存储在 PA-RoT 中的固件或硬件寄存器值中。种子通常使用设备证书、别名证书或证明证书传输到设备。

协议和层次结构

以下部分描述了主板的平台主动 RoT (PA-RoT) 和主动组件建立平台级 RoT 的能力和所需的协议和应用程序编程接口 (API)。Cerberus 主动 RoT 和主动组件 RoT 需要支持以下 I2C 协议。

该协议源自 MCTP SMBus/I2C 传输绑定规范。为不支持 MCTP 的设备定义了协议的有限版本。如果 AC-RoT 通过 MCTP 实现证明协议，它也可以选择通过本地 SMBus/I2C 实现最小证明协议。

Attestation Message Interface - 认证消息接口

证明消息接口使用 MCTP over I2C 消息协议来传输证明有效负载。AC-RoT MCTP 管理端点应实现与 MCTP SMBus/I2C 传输绑定规范相关的管理组件传输协议 (MCTP) 基本规范中详述的所需行为。以下部分概述了对管理端点预期行为的其他要求：

- 消息接口请求和响应消息使用自定义消息类型进行传输。
- MCTP 消息将仅以同步请求和响应方式传输。端点 (AC-RoT) 永远不应向控制器 (PA-RoT) 发起请求消息。
- MCTP 端点必须严格遵守本规范中定义的响应超时。当端点收到标准消息时，它应该在 100 毫秒内传输响应。如果端点在 100 毫秒内没有开始发送响应消息，它应该丢弃消息而不响应。
- MCTP 端点必须严格遵守为加密命令公布的响应超时。加密命令包括消息签名生成和验证的传输。加密命令超时乘数在设备功能命令中协商。
- MCTP 将身份验证留给应用程序实现。本规范部分遵循 USB 身份验证规范流程，当身份验证已建立时，可以交换证明种子。
- 不需要管理端点响应 ARP 消息。AC-RoT 端点不应生成任何 ARP 消息来通知主机。设备应该知道它们通常在 I2C mux之后，并且不应在分配的时间之外控制 I2C 总线，它们被提供以响应 MCTP 请求消息。
- MCTP 端点设备应仅响应。
- 无论端点是否支持 ARP，它们都应该以不支持 ARP 的方式运行。
- MCTP 规范使用大端字节排序，而本规范使用小端字节排序。此排序不会更改字节在物理层上发送的有效负载顺序。
- 端点应支持固定地址；支持端点 ID 以允许单个物理地址后面有多个 MCTP 端点。
- 如 MCTP SMBus/I2C 传输绑定规范中所定义，端点应支持快速模式 400KHz。

- 不支持多主机的端点设备应在从机模式下运行。PA-RoT PCD 将识别设备的模式。主机将发出 MCTP 有效负载格式的 SMBUS 写块，然后主机将发出 I2C 读取作为响应。Master 读取最初的 12 个字节并使用字节 3 和 MCTP EOM 标头位来确定是否需要额外的 SMBUS 读取命令来收集响应消息的其余部分。
- 端点应使用 MCTP 设置端点 ID 控制消息支持 EID 分配。
- 端点应支持 MCTP Get Vendor Defined Message Support 控制消息以指示 Cerberus 协议的支持级别。

Platform Cerberus Active RoT 始终是 MCTP 主站。Active Component RoT's 可以配置为 Endpoint，或者 Endpoint and Master。主动部件 RoT 端点和主机应该连接到单独的物理总线。由于主从定义是分层建立的，因此不需要主仲裁。主动组件 RoT 成为端点和主控的唯一层次结构是存在下游子设备时，例如以下框图中描述的主机总线适配器 (HBA)：

> 待办事项：图 6

在此图中，HBA RoT 是平台活动 RoT 的端点和下游 HBA 扩展器的主节点。对于平台的主动 RoT，HBA 是一个端点 RoT。对于 HBA 扩展器，HBA 控制器是一个主 RoT。

消息传递协议包含与 MCTP SMBus/I2C 传输绑定规范相关的管理组件传输协议 (MCPT) 基本规范，其中主动器件 RoT 是端点，平台的活动 RoT 是主机。

Protocol Format - 协议格式

所有 MCTP 事务均基于 SMBus 块写入总线协议。下表显示了一个 MCTP 封装的消息；请注意，偏移量以位而不是字节为单位给出。

Payload	Description
7:0	I2C destination address.
15:8	Command code; must be `0x0f`.
23:16	Number of bytes in the packet payload.
31:24	I2C source address.
35:32	Reserved (should be zero).
39:36	MCTP Header version.
47:40	Destination EID.
55:48	Source EID.
56:56	Start of message (SOM) flag.
57:57	End of message (EOM) flag.
59:58	Sequence number.
60:60	Tag owner.
63:61	Message tag.
64+N:64	Packet payload.

Payload	Description
64+N+8:64+N	PEC

一个包应包含至少 1 个字节的有效负载，最大不超过协商的 MCTP 传输单元大小。字节计数表示事务中的后续字节数，不包括 PEC。

每笔交易结束时的 PEC 是使用标准 CRC-8 计算的，它使用多项式`x^8 + x^2 + x + 1`，初始值为 0，最终 XOR 为 0。此 CRC 独立于MCTP 协议定义的消息完整性检查。CRC 是在整个封装的 MCTP 数据包上计算的，其中包括所有标头和目标 I2C 地址，因为它是通过 I2C 总线发送的（图 7 MCTP 封装消息中的字节 1 到 N）。由于 I2C 从地址通常不作为事务数据的一部分包含在内，因此 CRC 等于 CRC8 (`7_bit_address << 1 || i2c_payload`)。例如，发送到具有 7 位 I2C 地址“0x41”的设备的 MCTP 数据包的 CRC 计算结果为 CRC8（“0x82 || 数据包”）。

数据包格式

物理媒体特定报头和物理媒体特定报尾由端口使用的 MCTP 传输绑定规范定义。

请参阅 MCTP 传输绑定规范。

兼容的管理端点应实现 MCTP 基本规范中定义的所有 MCTP 要求的功能。

基本协议的公共字段包括消息类型字段，该字段标识在 MCTP 协议中承载的更高层消息类别。

Transport Layer Header - 传输层报头

管理组件传输协议 (MCTP) 基本规范定义了 MCTP 数据包报头（字段说明请参阅 DSP0236）如下。

请注意，偏移量以位为单位，而不是字节。此表对应于上表中位“31”之后的数据。

Payload	Description
3:0	Reserved (should be zero).
7:4	MCTP Header version.
15:8	Destination EID.
23:16	Source EID.
24:24	Start of message (SOM) flag.
25:25	End of message (EOM) flag.
27:26	Sequence number.
28:28	Tag owner.
31:29	Message tag.
32:32	Integrity check flag.
39:33	MCTP message type.
Variable	MCTP header.

Payload	Description
Variable	MCTP message data.
Variable	Integrity check.

通常支持空 (`0x00`) 源和目标 EID，但是具有多个 MCTP 端点的 AC-RoT 设备可以指定大于 `0x07` 且小于 `0xff` 的 EID 值。PA-RoT 不广播任何 MCTP 消息。

请注意，上面的最后三个字段取决于 MCTP 消息类型。

MCTP 消息

一条 MCTP 消息由一个或多个 MCTP 数据包组成。通常有两种类型的消息，MCTP 控制消息和 MCTP Cerberus 消息。

控制消息是标准的 MCTP 消息，最大消息体为 64 字节。

Cerberus 消息是本规范中定义的消息。这些消息的最大消息正文为 4096 字节，但可以根据设备功能将此大小协商为更小。

消息类型

根据管理组件传输协议 (MCTP) 基本规范，消息类型应为“0x7e”。消息类型用于支持供应商定义的消息，其中供应商由基于 PCI 的供应商 ID 标识。初始消息标头在管理组件传输协议 (MCTP) 基本规范中指定，并在下面详细说明以确保完整性：

> 表 2 供应商定义的消息

Message Header	Byte	Description
Request Data	1:2	PCI/PCIe Vendor ID. The MCTP Vendor Id formatted per 0x00 Vendor ID format offset.
	3:N	Vendor-Defined Message Body. 0 to N bytes.
Response Data	1:2	PCI/PCIe Vendor ID, the value is formatted per 0x00 Vendor ID offset.
	3:M	Vendor-Defined Message Body. 0 to M bytes.

Vendor ID 是 16 位无符号整数，在 PCI 2.3 规范中有描述。该值标识设备制造商。

表 4 MCTP 消息格式中描述了消息体和内容。

消息字段

MCTP 消息的格式由前两个字节的消息头、随后的消息数据和以消息完整性检查结尾的消息组成。

消息头包含由 MCTP 基本规范定义的消息类型 (MT) 字段和完整性检查 (IC)。消息类型字段指示

消息完整性检查

消息完整性检查字段包含根据消息内容计算的 32 位 CRC。

数据包组装成消息

一个 MCTP 消息可以拆分成多个 MCTP 数据包有效载荷并作为一系列数据包发送。请参阅 MCTP Base Specification 了解打包和消息组装规则。

请求消息

请求消息是由主 MTCP 控制器生成并发送到 MCTP 端点的消息。请求消息指定端点要执行的操作。请求消息是控制消息或 Cerberus 消息。

响应消息

响应消息是当 MCTP 端点完成对先前发出的请求消息的处理时生成的消息。响应消息必须在分配的时间内完成或丢弃。

EID 分配

BMC 会将 EID 分配给不同的 Active Component RoT 设备。所有主动器件 RoT 设备都应支持 MCTP 设置端点 ID 控制请求和响应消息。Platform Active RoT 的静态 EID 为 0x0b。

Certificates - 证书

PA-RoT 和 AC-Rot 将至少有两个证书：Device Id Certificate（通常由离线 CA 签名的 CA）和 Alias Certificate（由 Device Id Certificate 签名）。

PA-RoT 也可能有一个由设备 Id 证书签名的附加证明证书。

证书遵循 9.3 DICE 和 RIoT 密钥和证书。

格式

所有证书都应使用 X.509 v3 ASN.1 结构。所有证书都应使用 ASN.1 的二进制 DER 编码。所有证书都应符合 RFC5280。

为了促进证书链身份验证，必须存在权限和主题密钥标识符扩展。9.3 DICE 和 RIoT 密钥和证书中定义了进一步的证书要求。超出 RFC5280 或 DICE 要求的扩展是允许的，只要它们符合适当的标准。自定义扩展必须标记为非关键。

文本格式

证书中包含的所有文本 ASN.1 对象均应指定为`UTF8String`、`PrintableString`或`IA5String`。任何文本对象的长度不得超过 64 个字节，不包括 DER 类型和 DER 长度编码。

专有名称

专有名称由许多唯一标识设备的属性组成。可分辨名称的唯一性可以通过包括序列号等属性来实现。

对象标识符

对象标识符应遵循 9.3 DICE 和 RIoT 密钥和证书

序列号

根据 9.3 DICE 和 RIoT 密钥和证书，每个别名证书的证书序列号在统计上必须是唯一的。

如果安全处理器有熵源，则可以使用 8 字节随机数。

如果安全处理器没有熵源，则可以使用基于密钥的加密安全密钥派生函数生成一个 8 字节序列号，例如 SP800-108 [9.6 NIST 特别出版物 800- 108]。每个生成的证书的序列号必须是唯一的。对于别名证书，这应该通过将 FWID 合并到密钥派生过程中来实现（例如作为 SP-800-108 中的上下文值）

如果生成的 8 字节序列号不是正数，则可以用额外的八位字节填充它以保持与 RFC5280 的一致性。

Certificate Chains - 证书链

最大证书链大小为 4096 字节。证书链不需要额外的编码，因为每个证书都可以单独检索。

证书签名应使用 ECDSA 和未压缩点形式的 NIST P256 `secp256r1` 曲线。应使用“SHA2-256”算法执行散列。

Authentication - 验证

身份验证是用于建立对特定设备的信任并证明设备固件完整性的过程。一旦设备通过身份验证，就可以选择建立安全会话以提供机密性。对于某些设备，安全会话也可用于启用可用于额外安全性或启用其他功能的加密绑定。

设备只需要支持来自另一个端点的单个会话。在单会话设备中，新会话的建立将取代和终止现有会话以允许新会话。先前的会话将在收到指定请求的密钥交换的会话外（未加密）`GET_DIGESTS` 请求时终止。在会话中收到的`GET_DIGESTS`请求（加密）或未请求密钥交换不会导致活动会话终止。

PA-RoT 和 AC-RoT 认证

设备使用 Alias 证书链和签名固件测量进行身份验证。证书链由签署设备 ID 证书的证书颁发机构认可。证书层次结构在 TCG [基于隐式身份的设备证明](https://trustedcomputinggroup.org/wp-content/uploads/TCG-DICE-Arch-Implicit-Identity-Based-Device-Attestation-v1-rev93。 pdf) 参考。

证书身份验证流程紧跟 USB 身份验证架构和身份验证消息。本规范中定义的命令结构源自 USB-C 身份验证协议。规范相关章节如下：

- 第 3 节：身份验证架构，*USB 身份验证规范*
- 第 4 节：身份验证协议，*USB 身份验证规范*
- 第 5 节：身份验证消息，*USB 身份验证规范*

认证序列从主机（例如 PA-RoT）发出 Get Digests 命令开始。从设备（例如 AC-RoT）使用设备别名证书链中每个证书的 SHA256 哈希列表进行响应。

如果 master 已经缓存了证书链，它可以选择跳过请求这些证书。

如果主服务器没有正确的证书，它将为从服务器的别名证书链中的每个证书发出获取证书请求。

从站将使用请求的证书进行响应，该证书必须来自受信任的 CA。

Master 将验证 Slave 的 Alias 证书链。如果验证失败或证书被吊销，则认证失败。PA-RoT 和 AC-RoT 可以使用吊销补丁进行更新，有关详细信息，请参阅固件更新规范。

一旦证书链通过验证，主机将通过质询请求验证固件完整性。从设备将生成一个挑战响应，该响应使用其别名密钥签名，其中包括设备固件的测量值。主机可以根据组件固件清单 (CFM) 或其他参考来验证此测量，以确认固件有效。

认证流程如下：

1. Master 为 Slot 0 发出 `GET_DIGESTS` 命令
2. Slave 响应 Alias 证书链的摘要
3. 对于链中的每个证书，Master 检查证书是否已被缓存。
4. 如果证书没有被缓存，Master发出`GET_CERTIFICATE`请求。
5. 从站用请求的证书进行响应。
6. Master 根据受信任的根 CA 验证 Alias 证书链。
7. Master 发出包含 nonce `RN1` 的 Challenge 命令。
8. Slave 生成包含的响应
    - 随机数，`RN2`。
    - 集体固件测量，`PMR0`。
    - 使用别名密钥在 (`request || response`) 对上签名。
9. Master验证签名和固件测量

> 待办事项：图 9

安全会话建立

Cerberus 设备可选择支持安全会话建立，以提供两个设备之间或外部数据中心软件与设备之间的机密性。会话建立利用基本身份验证流程和额外的密钥交换来建立安全会话。

设备身份验证使用身份证书链，而会话密钥交换基于临时 ECDH 密钥。Device Capabilities 命令将确定设备是否支持安全会话。

在查询设备功能后，主机将通过发出 Get Digests 请求开始身份验证序列。当使用身份验证建立安全会话时，`GET_DIGESTS`请求还将指示最终将使用的密钥交换类型。如果设备不支持请求的密钥交换，它将以错误响应。

身份验证完成后，master 将生成一个临时密钥并将其发送给 slave。从机将生成自己的临时会话密钥并完成密钥交换。会话密钥将通过在挑战中包含的会话密钥和随机数上运行 ECDH 和后续 KDF 来派生。

用于派生会话和 HMAC 密钥的 KDF 是 NIST SP800-108 计数器模式。

会话状态是不稳定的。会话的每一端都必须能够处理会话上下文的丢失并支持会话重建。

会话建立顺序

会话建立流程从第 5.1.1 节中描述的标准身份验证开始。一旦从站被验证，密钥交换用于建立会话。

1. Master 生成一个临时 ECC 密钥对，并发送一个包含公钥 (`PKreq`) 类型 0 的 `Key Exchange` 请求。
2. 从站生成一个具有同等强度的临时 ECC 密钥对 (`PKresp`)。

3. 从站使用 NIST SP800-108 计数器模式生成一个 256 位 AES 会话密钥 (`K_S`)，参数如下：
    - `K_I = ECDH(PKreq, PKresp)`
    - `label = RN1`
    - `context = RN2`
    - `r = 32`
4. Slave 使用 NIST SP800-108 计数器模式生成一个 256 位 MAC 密钥 (`K_M`)，参数如下（注意 `label` 和 `context` 被交换）：
    - `K_I = ECDH(PKreq, PKresp)`
    - `label = RN2`
    - `context = RN1`
    - `r = 32`
5. 从站发送一个密钥交换响应，包括：
    1. 从属会话公钥 (`PKresp`)。
    2. 在“PKreq”和“PKresp”上使用别名密钥的签名。
    3. 使用“K_M”的别名密钥证书的 HMAC。
6. Master 使用密钥交换响应中的公钥生成相同的会话和 MAC 密钥。
7. Master 验证响应中的签名和 HMAC。
8. 现在可以使用共享会话密钥使用 256 位 AES-GCM 对消息进行加密。

> TODO：图 10

Secure Device Binding - 安全设备绑定

在某些情况下，有必要在设备之间进行额外级别的身份验证。这种额外的身份验证机制用于配对两个设备，以便检测到任何一个设备的更换。检测到未经授权的部件更换被标记为安全违规。

绑定来自在会话密钥上运行的附加 KDF，以派生最终会话密钥。参与紧密耦合配对的每个设备都包含一个公共 HMAC 密钥，该密钥在 KDF 中用于设备配对会话密钥。此公共 HMAC 密钥源自两个设备首次建立安全会话时可用的会话数据。

Secure Device Binding Sequence - 安全设备绑定序列

具有设备绑定的配对会话从标准安全会话开始。

建立会话后，将使用额外的密钥交换来更新会话密钥。这个额外的密钥交换必须加密。

1. Master加载配对密钥`K_P`。
    1. 如果这是第一次配对请求，Master 使用 NIST SP800-108 Counter Mode 使用以下参数生成密钥：
        - `K_I = K_S`
        - `label = "pairing"`
        - No `context`
        - `r = 32`
    2. 如果主设备已经与从设备配对，则主设备从安全存储中加载密钥。
2. Master 使用 NIST SP800-108 计数器模式生成一个新的会话密钥“K_S”。
    - `K_I = K_P`
    - `label = K_S`
    - No `context`
    - `r = 32`

3. Master 发送类型 1 的密钥交换请求，其中包括使用K_M的配对密钥的长度和 HMAC。此消息使用当前会话密钥K_S加密。
4. 从机加载K_P
    1. 如果这是来自该主机的第一个配对请求，则从机使用与主机相同的输入生成密钥。
    2. 如果从设备已经与主设备配对，则从设备从安全存储中加载密钥。
5. Slave 计算配对密钥 HMAC 并将其与接收到的数据进行比较。
    - 如果这是这个 Master 的第一个配对请求，Slave 安全地 存储“K_P”。
6. Slave 生成一个新的会话密钥（K_S'）。
7. Slave 生成一个用K_S'加密的密钥交换响应。
8. Master 使用K_S'验证响应。无法使用K_S'解密 将要求主服务器使用K_S解密，因为从服务器不会在失败时更新会话密钥。
9. 安全会话继续使用K_S'。任何一方都不会处理使用 K_S 加密的消息。

> 待办事项：图 11

Command Format - 命令格式

以下部分描述了支持身份验证、质询和证明协议的 MCTP 消息格式。请求/响应消息体描述封装在 MCTP 传输中的供应商定义的 MCTP 消息。本节不介绍MCTP Transport Header，包括MCTP Header Version、Destination Endpoint ID等MCTP协议定义的字段。MCTP 消息封装在 3.2 节中描述

MCTP Get Vendor Defined Message Support 命令可以发现支持哪些端点供应商定义的消息。该发现标识供应商组织和定义的消息类型。此请求的格式在 MCTP 基本协议规范中进行了描述。

对于 Cerberus 协议，应返回以下信息以响应 MCTP "Get Vendor Defined Message Support"请求：

- Vendor ID Format: `0x00`
- PCI Vendor ID: `0x1414`
- Command Set Version: `0x04`

Attestation Protocol Format - 证明协议格式

从 PA-RoT 到 AC-RoT 的消息将具有以下格式。

这嵌入在第 3.4 节中定义的结构中。

Payload	Description
0:0	Integrity check flag.
7:1	MCTP message type (always `0x7e`).
23:8	MCTP PCI Vendor ID (always `0x1414`).
24:24	Request type.
25:25	Reserved (should be zero).
26:26	Set if encrypted (see below).
31:27	Reserved.

Payload	Description
39:32	Cerberus command byte.
Variable	The command payload.

协议标头字段仅包含在多数据包 MCTP 消息的第一个数据包中。消息体重构后，协议头将用于解释消息内容。保留字段必须设置为 0。

“请求类型”字段指示消息中包含什么类型的请求。本规范中定义的消息应将此字段设置为 0。

将此位设置为 1 提供了一种机制，除了不同的供应商 ID 外，还支持特定于设备的命令集。如果此位为 1，则不支持任何其他命令的设备将返回错误。

加密消息

如果在协议头中设置了 crypt 字段，则消息体包含加密数据。从单个 MCTP 数据包重建的命令代码字节和完整的消息有效载荷是加密的。第 5 节中描述的会话建立流程用于生成加密密钥。加密消息将在消息正文末尾以明文形式包含 16 字节的 GCM 身份验证标记和 12 字节的初始化向量。下表显示了带有加密尾部的加密 Cerberus 消息的正文。请注意，偏移量以位而不是字节为单位给出。

Payload	Description
0:0	Integrity check flag.
7:1	MCTP message type (always `0x7e`).
23:8	MCTP PCI Vendor ID (always `0x1414`).
24:24	Request type.
25:25	Reserved (should be zero).
26:26	Set if encrypted (always, in this case).
31:27	Reserved.
Variable	Ciphertext; corresponds to the command byte and its payload above.
N+15:N	AES-GCM tag.
N+31:N+15	AES-GCM initialization vector.

RoT 命令

下表总结了根据本规范定义的消息，并带有以下图例：

- "Enc" 表示命令必须在会话中加密。
- “慢” 意味着命令执行加密操作，并且应该使用更长的消息超时。
- “字节” 是标识此消息的命令字节。
- "Required" 是消息是否*必须*实现：`R` 表示 “在所有情况下都需要”；`O` 表示“必要时执行”；`M` 的意思是”认证Master所必需的”。

Message Name	Enc	Slow	Byte	Required	Description

Message Name	Enc	Slow	Byte	Required	Description
ERROR			0x7f	R	Default response message.
Firmware Version			0x01	R	Retrieves firmware version information.
Device Capabilities			0x02	R	Retrieves device capabilities.
Device Id			0x03	R	Retrieves device id.
Device Information			0x04	R	Retrieves device information.
Export CSR			0x20	R	Exports CSR for device keys.
Import Certificate		X	0x21	R	Imports CA signed Certificate.
Get Certificate State			0x22	R	Checks the state of the signed Certificate chain.
GET DIGESTS		X	0x81	R	Retrieves certificate chain digests.
GET CERTIFICATE			0x82	R	Retrieves certificate chain.
CHALLENGE		X	0x83	R	Authenticates the collective measurement.
Key Exchange		X	0x84	O	Exchanges session keys and mfg device pairing keys.
Session Sync	X	X	0x85	O	Checks status of a secure session.
Get Log Info			0x4f	O	Retrieves logging information.
Get Log			0x50	O	Retrieves debug, attestation, and tamper logs.
Clear Log			0x51	O	Clears device logs.
Get Attestation Data			0x52	O	Retrieves raw data from the attestation log.
Get Host State			0x40	O	Gets reset state of the host processor.
Get PFM Id			0x59	O	Gets PFM information.
Get PFM Supported			0x5a	O	Retrieves the PFM.
Prepare PFM			0x5b	O	Prepares an incoming PFM update.
Update PFM			0x5c	O	Uploads part of a new PFM.
Activate PFM			0x5d	O	Activates a newly-staged PFM.
Get CFM Id			0x5e	M	Gets CFM information.
Prepare CFM			0x5f	M	Prepares an incoming CFM update.
Update CFM			0x60	M	Uploads part of a new CFM.

Message Name	Enc	Slow	Byte	Required	Description
Activate CFM			0x61	M	Activates a newly-staged CFM.
Get CFM Supported			0x8d	M	Retrieve supported CFM IDs.
Get PCD Id			0x62	M	Gets PCD information.
Prepare PCD			0x63	M	Prepares an incoming PCD update.
Update PCD			0x64	M	Uploads part of a new PCD.
Activate PCD			0x65	M	Activates a newly-staged PCD.
Prepare Firmware Update			0x66	O	Prepares an incoming firmware update.
Update Firmware			0x67	O	Uploads part of a new firmare image.
Update Status			0x68	M	Firmware or manifest update status
Extended Update Status			0x8e	M	Firmware or manifest extended status
Activate Firmware Update			0x69	O	Activates a newly-staged firmware update.
Reset Configuration		X	0x6a	O	Resets configuration to default state.
Get Config IDs		X	0x70	M	Gets authenticated manifest IDs.
Recovery Firmware			0x71	O	Restores firmware index using backup.
Prepare Recovery Image			0x72	O	Prepares an incoming recovery image update.
Update Recovery Image			0x73	O	Uploads part of a new recovery image.
Activate Recovery Image			0x74	O	Activate newly-staged recovery image.
Get Recovery Image Id			0x75	O	Get recovery image information.
Get PMR		X	0x80	O	Gets a Platform Measurement Register.
Update PMR	X	X	0x86	O	Extends a Platform Measurements Register.
Reset Counter			0x87	R	Reset Counter.
Unseal Message		X	0x89	O	Begin an unsealing challenge.
Unseal Message Result			0x8a	O	Get unsealing status and result.

命令字节“0xf0”到“0xff”永远不会被 Cerberus 分配，并且可能被视为“专用”区域。

关于所需消息的注释：

1. 如果支持任何“Enc”消息，则需要 `Key Exchange` 和 `Session Sync`。
2. 如果支持证明日志，则需要`Get Attestation Data`。
3. 如果支持任何更新命令（例如，`Prepare`/`Update`/`Activate` 命令），则需要`Update Status` 和 `Extended Update Status`。
4. 如果支持任何清单类型，则需要“获取配置 ID”。

消息体结构

以下部分描述了 MCTP 消息体的结构。

错误信息

当命令未按建议完成时，为命令响应返回错误命令，它还充当无响应命令的通用状态，其中“无错误”代码将指示成功。Message Tag、Sequence Number 和 Command 匹配对相应请求的响应。

Message Body 返回如下：

Payload	Description
1	Error code
2:5	Error data

“错误代码”是以下之一： 表 9 错误代码

Error Code	Value	Description	Data
No Error	0x00	Success	0x00
Invalid Request	0x01	Invalidated data in the request	0x00
Busy	0x03	Device is busy processing other commands	0x00
Unspecified	0x04	Unspecified error occurred	Vendor defined
Invalid Checksum	0xf0	Invalid checksum	Checksum
Out of Order Message	0xf1	EOM before SOM	0x00
Authentication	0xf2	Authentication not established	0x00
Out of Sequence Window	0xf3	Message received out of Sequence Window	0x00
Invalid Packet Length	0xf4	Packet received with unexpected size	Packet Length
Message Overflow	0xf5	Message exceeded maximum length	Message Length

如果未为以下部分中的命令定义定义显式响应，则错误消息是带有“No Error”的预期响应。

错误消息可以作为对任何处理失败的命令的响应而出现。

固件版本

此命令获取目标固件的版本。

Request - 请求

Payload	Description
1	Area Index

区域索引如下：

- 0x00：整个固件
- 0x01: RIoT 核心
- 由供应商定义的附加索引。

Response - 响应

Payload	Description
1:32	Firmware Version Number (ASCII)

Device Capabilities - 设备能力

设备功能提供有关设备功能的信息。消息和数据包最大值是根据共享功能协商的。在确定适当的最大尺寸时，还需要考虑一些额外的因素：

1. 根据 MCTP 规范，数据包负载大小必须不小于 64 字节。
2. 根据第 3.5 节，消息负载大小不得超过 4096 字节。
3. 设备支持的总数据包大小将包括3.2节中定义的封装，不包括目标地址的第一个字节。 例如，最大数据包负载为 247 字节将导致每个数据包传输 256 字节：
    - 1 字节目的地址。
    - 3 字节 SMBus 标头。
    - 4 字节 MCTP 标头。
    - 247 字节有效负载。
    - 1 字节 CRC-8。
4. 部分命令不支持跨消息分隔。设备必须支持与受支持命令的预期有效负载兼容的最大消息大小。
5. 主设备，如 PA-RoT，应支持最大 4096 字节的消息大小，以确保与任何从设备完全兼容。

Request - 请求

Payload	Description
1:2	Maximum Message Payload Size
3:4	Maximum Packet Payload Size
5	Mode: [7:6]

	00 = AC-RoT 01 = PA-RoT 10 = External 11 = Reserved [5:4] Master/Slave 00 = Unknown 01 = Master 10 = Slave 11 = both master and slave [3] Reserved [2:0] Security 000 = None 001 = Hash/KDF 010 = Authentication [Certificate Auth] 100 = Confidentiality [AES]
6	[7] PFM support [6] Policy Support [5] Firmware Protection [4-0] Reserved
7	PK Key Strength: [7] RSA [6] ECDSA [5:3] ECC 000: None 001: 160bit 010: 256bit 100: Reserved [2:0] RSA: 000: None 001: RSA 2048 010: RSA 3072 100: RSA 4096
8	Encryption Key Strength: [7] ECC [6:3] Reserved [2:0] AES: 000: None 001: 128 bit 010: 256 bit 100: 384 bit

Response - 响应

Payload	Description

1:2	Maximum Message Payload Size
3:4	Maximum Packet Payload Size
5	Mode: [7:6] 00 = AC-RoT 01 = PA-RoT 10 = External 11 = Reserved [5:4] Master/Slave 00 = Unknown 01 = Master 10 = Slave 11 = both master and slave [3] Reserved [2:0] Security 000 = None 001 = Hash/KDF 010 = Authentication [Certificate Auth] 100 = Confidentiality [AES]
6	[7] PFM support [6] Policy Support [5] Firmware Protection [4-0] Reserved
7	PK Key Strength: [7] RSA [6] ECDSA [5:3] ECC 000: None 001: 160bit 010: 256bit 100: Reserved [2:0] RSA: 000: None 001: RSA 2048 010: RSA 3072 100: RSA 4096
8	Encryption Key Strength: [7] ECC [6:3] Reserved [2:0] AES: 000: None 001: 128 bit 010: 256 bit 100: 384 bit

9	Maximum Message timeout: multiple of 10ms
10	Maximum Cryptographic Message timeout: multiple of 100ms

设备ID

八字节响应。

Request - 请求

空的数据体。

Response - 响应

Payload	Description
1:2	Vendor ID; LSB
3:4	Device ID; LSB
5:6	Subsystem Vendor ID; LSB
7:8	Subsystem ID; LSB

设备信息

此命令获取有关目标设备的信息。

Request - 请求

Payload	Description
1	Information Index

信息指标如下：

- 0x00：唯一芯片标识符。
- 附加索引由供应商定义。

Response - 响应

Payload	Description
1:N	Requested identifier blob

导出 CSR

导出设备标识证书自签名证书签名请求。初始证书是自签名的，直到 CA 签名并导入。

将 CA 签名版本的证书导入设备后，将替换自签名证书。

Request - 请求

Payload	Description
1	Index: Default = 0

Response - 响应

Payload	Description
1:N	Certificate Signing Request

导入证书

将签名的设备 ID 证书链导入设备。

每个导入命令将链中的单个证书发送到设备，并且对必须导入这些证书的顺序没有要求。在验证完整链后，设备将被密封，并且在不更改固件的情况下不能进行进一步的导入。在使用新证书执行证书链验证之前返回响应消息，并且仅指示证书是否被设备接受。

可以使用获取证书状态请求查询证书供应的完整状态。

收到证书后，设备将开始验证存储的证书链。发送多个证书时，可能需要确保在发送新证书之前已完成先前的身份验证步骤。可以使用 Get Certificate State 命令检查身份验证状态。

Request - 请求

Payload	Description
1	Type
2:3	Certificate Length
4:N	Certificate

证书类型如下：

- 0x00: 设备 ID 证书
- 0x01: 根 CA 证书
- 0x02: 中级 CA 证书

获取证书状态

确定已发送到设备的签名证书的证书链状态。此命令的请求不包含其他有效负载。

Request - 请求

空的数据体。

Response - 响应

Payload	Description
1	State

Payload	Description
2:4	Error details, if chain validation has failed.

可能的状态：

- 0x00：已提供有效链。
- 0x01：尚未提供有效链。
- 0x02：正在验证存储链。

获取摘要

此命令源自类似的 USB Type C-Authentication Message。不包括协议版本字节，消息类型存在于 MCTP 消息的命令字节中。请参阅：表 4 MCTP 消息格式。响应是不同的，因为证书处理偏离 USB 类型 C，并且保留字节被重新调整用途。

可以随时发送此命令。如果先前已建立身份验证，此命令将重新协商/覆盖先前的身份验证和会话建立。字节 2，保留 USB Type C – 认证规范，用于描述密钥交换算法。当请求者和响应者支持多种密钥交换算法时，这是相关的。

不包含有效证书链的插槽将生成包含 0 摘要的响应。有效载荷字节 2 将指示没有返回任何摘要。

Request - 请求

Payload	Description
1	Slot Number (0 to 7) of the target Certificate Chain to read.
2	Key Exchange Algorithm

潜在的密钥交换算法是：

- 0x00：无
- 0x01：ECDH

Response - 响应

Payload	Description
1	Capabilities Field (always 0x01).
2	Number of digests returned.
3:N	SHA256 digests of the certificates in the chain, starting from the root.

获取证书

此命令检索 AC-RoT 的公共证明证书链。

它严格遵循 USB C 型身份验证规范。

如果设备没有请求的槽或索引的证书，则响应中的证书内容将为空。

Request - 请求

Payload	Description
1	Slot Number (0 to 7) of the target Certificate Chain to read.
2	Certificate to read, zero-indexed from the root Certificate.
3:4	Offset: offset in bytes from start of the Certificate to read.
5:6	Length: number of bytes to read.

Response - 响应

Payload	Description
1	Slot Number of the target Certificate Chain returned.
2	Certificate index of the returned certificate
3:N	Requested contents of target Certificate Chain.

CHALLENGE - 挑战

PA-RoT 将发送此命令，提供密钥交换中的第一个随机数。

Request - 请求

Payload	Description
1	Slot Number (0 to 7) of the target Certificate Chain to read.
2	Reserved
3:35	Random nonce

Response - 响应

Payload	Description
1	Slot Number of the Certificate Chain used.
2	Certificate slot mask
3	MinProtocolVersion supported by device
4	MaxProtocolVersion supported by device
5:6	Reserved
7:38	Random nonce
39	Number of components used to generate the PMR0 measurement
40	Length of PMR0 (L)

Payload	Description
41:40+L	Value of PMR0 (Aggregated Firmware Digest)
41+L:N	Signature over concatenated request and response message payloads.

固件摘要是目标组件的安全描述符和固件的度量。此固件测量数据不包括 Cerberus PCD、CFM 或 PFM。这些测量值在 PMR1 中返回。这些数字在 6.44 获取配置 ID 中检索。USB-C 上下文哈希不包含在挑战中。这将替换为设备的上下文测量。> 注意：证明证书派生将包括固件和安全描述符的测量。PMR0 预计是变化最少的 PMR，因为它包含安全描述符和设备初始引导加载程序的度量。

密钥交换

密钥交换用于与设备建立加密通道。会话建立流程在第 5 节身份验证中有详细说明。

收到此密钥类型为 0 的消息后，双方可以建立加密会话。如果是 Key Type 1，配对的密钥也会被比较，如果验证匹配预期的密钥，配对的功能就会被解锁。Key Type 1 只能在加密会话下执行，因为 HMAC 需要会话密钥。当最初计算设备绑定的配对密钥 (K<sub>P) 时，会话的密钥类型 0 请求中指定的 HMAC 将与 KDF 一起使用。

关闭已建立的会话（密钥类型 2）时，如果会话成功终止，响应将以纯文本形式传输。

Request - 请求

Payload	Description
1	Key Type
2:N	Key data. Format is defined by the type of request.

可能的密钥类型：

- 0x00：会话密钥。
- 0x01：配对密钥 HMAC。
- 0x02: 销毁会话。

会话密钥数据

Payload	Description
1	HMAC Algorithm
2:N	ASN.1 DER encoded ECC public key (`PKreq`)

可能的 HMAC 哈希算法：

- 0x00：SHA-256
- 0x01: SHA-384
- 0x02：SHA-512

此消息中指定的 HMAC 类型适用于已建立会话的所有 HMAC 操作，包括任何后续配对消息。由于会话密钥（`K_S` 和 `K_M`）是 256 位密钥，因此无论用于密钥交换消息的 HMAC 类型如何，它们将始终使用 SHA-256 生成。

配对密钥 HMAC 数据

Payload	Description
1:2	Length in bytes of the pairing key
3:N	HMAC of the pairing key: `HMAC(K_M, K_P)`

销毁会话数据

Payload	Description
1:N	HMAC of session key: `HMAC(K_M, K_S)`

Response - 响应

Payload	Description
1	Key Type from request
2:N	Response data. Format is defined by the type of request.

会话密钥数据

Payload	Description
1	Reserved. Set to 0.
2:3	Key Length
4:N	DER-encoded ECC public key (`PKresp`)
N+1:N+2	Signature Length
N+3:M	Signature over session keys: `SIGN(PKreq || PKresp)`
M+1:M+2	HMAC Length
M+3:H	HMAC of the Alias certificate: `HMAC(KM, alias_cert)`

配对密钥 HMAC 数据

空的数据体。

销毁会话数据

空的数据体。

会话同步

检查加密会话的状态。消息必须始终加密。

Request - 请求。

Payload	Description
1:4	Random number (`RNreq`)

Response - 响应。

Payload	Description
1:N	HMAC of the nonce: `HMAC(KM, RNreq)`

获取日志信息

获取 RoT 的内部日志信息。

Request - 请求

空的数据体。

Response - 响应

Payload	Description
1:4	Debug Log (0x01) Length in bytes
5:8	Attestation Log (0x02) Length in bytes
9:12	Tamper Log (0x03) Length in bytes

获取日志

获取 RoT 的内部日志。有 3 种类型的可用日志： 调试日志，其中包含 Cerberus 应用程序信息和机器状态。

Attestation measurement log，这种日志格式就像TCG日志，还有Tamper日志。无法清除或重置篡改计数器。

Request - 请求

Payload	Description
1	Log Type
2:5	Offset

可能的日志类型：

- 0x01：调试日志。
- 0x02: 证明日志。
- 0x03：篡改日志。

Response - 响应

Payload	Description
1:N	Contents of the log.

由日志结尾决定的长度，或基于设备功能的数据包大小，请参阅部分：6.7 设备功能。如果响应跨越多个 MCTP 消息，则响应结束将由 MCTP 消息确定，该消息的有效载荷小于两个设备支持的最大有效载荷。为了保证响应永远不会恰好落在最大负载边界上，响应者必须发回一个负载为零的额外数据包。

鉴证日志格式

证明日志将报告每个 PMR 的所有组件的单独测量值。每个测量将是日志中的一个条目。整个日志由按顺序连接的每个条目组成。条目的结构与 TCG 事件非常相似。请参阅 TCG PC 客户端平台固件配置文件规范。

日志条目标题：

Offset	Description
1	Log entry start marker: [7:4]: 0x0c [3:0]: Header format, 0x0b per this specification.
2:3	Total length of the entry, including the header
4:7	Unique entry identifier

证明条目格式：

Offset	Description
1:7	Log Entry Header
8:11	TCG Event Type
12	Measurement index within a single PMR.
13	Index of the PMR for the measurement.
14:15	Reserved, set to 0.
16	Number of digests
17:19	Reserved, set to 0.
20:21	Digest algorithm Id (`0x0b`, SHA-256)
22:53	SHA-256 digest used to extend the measurement.
54:57	Measurement length
58:89	Measurement

调试日志格式

设备报告的调试日志没有指定格式，因为这在不同设备之间可能会有所不同，并且不需要证明。预计设备的诊断实用程序将能够理解公开的日志信息。此处提供了推荐的条目格式。

建议的调试条目格式：

Offset	Description
1:7	Log Entry Header
8:9	Format version of the entry.
10	Severity of the entry.
11	ID of the source of the message.
12	ID for the entry message type.
13:16	Message specific argument.
17:20	Message specific argument.

清除调试/证明日志

清除 RoT 中的日志。请注意，无法清除篡改日志。

Request - 请求

Payload	Description
1	Log Type

注意：在清除证明日志时，它会使用当前测量值自动重新创建。

Get Attestation Data - 获取鉴权数据

获取用于生成证明日志中报告的测量的原始数据。

并非所有测量都有可用的原始数据，因为这主要用于通过少量数据或文本生成的测量。对不提供测量数据的条目的请求将生成具有空负载的正常响应。对无效条目的请求将生成错误响应。

虽然此命令旨在支持应适合单个 MCTP 消息的小块数据，但请求中包含偏移参数以支持所需数据太大的情况。

Request - 请求

Payload	Description
1	Platform Measurement Register
2	Entry Index
3:6	Offset

Response - 响应

Payload	Description
1:N	The measured data.

获取主机状态

检索受 Cerberus 保护的主机处理器的重置状态。

Request - 请求

Payload	Description
1	Port Id

Response - 响应

Payload	Description
1	Host Reset State

可能的状态：

- 0x00 - 主机正在运行。
- 0x01 - 主机处于复位状态。
- 0x02 - 主机未运行，也未处于重置状态。

获取PFM ID

检索 PFM 标识符。

Request - 请求

Payload	Description
1	Port Id
2	PFM Region: 0x00 = Active, 0x01 = Pending
3 (optional)	Identifier: 0x00 = Version Id (default), 0x01 = Platform Id

Response - 响应

Payload	Description
1	PFM Valid (0x00 or 0x01)
2:5	PFM Version Id

Payload	Description
1	PFM Valid (0 or 1)
2:N	PFM Platform Id as null-terminated ASCII

获取 PFM 支持的固件

Request - 请求

Payload	Description
1	Port Id
2	PFM Region: 0x00 = Active, 0x01 = Pending
3:6	Offset

Response - 响应

Payload	Description
1	PFM Valid (0x00 or 0x01)
2:5	PFM Version Id
6:N	PFM supported FW versions

如果响应跨越多个 MCTP 消息，则响应结束将由 MCTP 数据包确定，该数据包的有效载荷小于两个设备支持的最大有效载荷。

为了保证响应永远不会恰好落在最大负载边界上，响应者应该发回一个负载为零的额外数据包。

准备PFM

为传入的 PFM 提供 RoT。

Request - 请求

Payload	Description
1	Port Id
2:5	Total size

更新PFM

闪存描述符结构描述了设备的闪存区域。

Request - 请求

Payload	Description
1	Port Id
2:N	PFM Payload

PFM 有效载荷包括 PFM 签名和单调仅前向 Id。PFM 签名在收到所有 PFM 有效载荷后进行验证。PFM 根据激活命令激活。请注意，如果系统在收到 PFM 后重新启动，则 PFM 会自动激活。要在重新引导之前激活，请发出 Activate PFM 命令。

激活 PFM

在有效的 PFM 更新后，更新命令密封 PFM 提交方法。如果立即提交，闪存读取和写入应在发出此命令时暂停。RoT 将掌握 SPI 总线并验证新更新的 PFM。此命令只能遵循有效的 PFM 更新。

Request - 请求

Payload	Description
1	Port Id
2	Activation: 0x00 = Reboot only, 0x01 = Immediately

如果仅发出重启，则在更新新的 PFM 之前，“立即”提交 PFM 的选项不可用。

获取 CFM ID

检索组件固件清单 ID

Request - 请求

Payload	Description
1	Port Id
2	CFM Region: 0x00 = Active, 0x01 = Pending
3 (optional)	Identifier: 0x00 = Version Id (default), 0x01 = Platform Id

Response - 响应

Payload	Description
1	CFM Valid (0x00 or 0x01)
2:5	CFM Version Id

Payload	Description
1	CFM Valid (0 or 1)
2:N	CFM Platform Id as null-terminated ASCII

准备CFM

为传入的组件固件清单提供 RoT。

Request - 请求

Payload	Description
1:5	Total size

更新CFM

闪存描述符结构描述了设备的闪存区域。

Request - 请求

Payload	Description
1:N	CFM Payload

CFM 有效载荷包括 CFM 签名和单调仅前向 Id。CFM 签名在收到所有 CFM 有效负载后进行验证。CFM 根据激活命令激活。请注意，如果系统在收到 CFM 后重新启动，则挂起的 CFM 将被验证并自动激活。要在重新引导之前激活，请发出 Activate CFM 命令。

激活CFM

在有效的 CFM 更新后，更新命令密封 CFM 提交方法。RoT 将掌握 I2C 并根据 CFM 证明平台配置数据中的组件。

Request - 请求

Payload	Description
1	Activation: 0x00 = Reboot only, 0x01 = Immediately

获取 CFM 组件 ID

Request - 请求

Payload	Description
1	CFM Region: 0x00 = Active, 0x01 = Pending
2:5	Offset

Response - 响应

Payload	Description
1	CFM Valid (0x00 or 0x01)
2:5	CFM Version Id
6:N	CFM supported component IDs

如果响应跨越多个 MCTP 消息，则响应结束将由 MCTP 数据包确定，该数据包的有效载荷小于两个设备支持的最大有效载荷。为了保证响应永远不会恰好落在最大负载边界上，响应者应该发回一个负载为零的额外数据包。

获取PCD ID

检索 PCD Id。

Request - 请求

Payload	Description
1 (optional)	Identifier: 0x00 = Version Id (default), 0x01 = Platform Id

Response - 响应

Payload	Description
1	PCD Valid (0x00 or 0x01)
2:5	PCD Version Id

Payload	Description
1	PCD Valid (0x00 or 0x01)
2:N	PCD Platform Id as null-terminated ASCII

准备PCD

为传入的平台配置数据提供 RoT。

Request - 请求

Payload	Description
1:4	Total size

更新PCD

闪存描述符结构描述了设备的闪存区域。

Request - 请求

Payload	Description
1:N	PCD Payload

PCD 有效载荷包括 PCD 签名和单调仅前向 Id。PCD 签名在收到所有 PCD 有效载荷后进行验证。PCD 根据激活命令被激活。请注意系统是否在收到 PCD 后重新启动。

激活PCD

在有效的 PCD 更新后，激活命令会密封 PCD 承诺。

Request - 请求

空的数据体。

平台配置

下表描述了平台配置数据结构。

<table>
<tr><th>Payload</th><th>Description</th></tr>
<tr><td>1:3</td><td>Platform Configuration Data Id</td></tr>
<tr><td>4:5</td><td>Length</td></tr>
<tr><td>6</td><td>Policy Count</td></tr>
<tr><td>7:N</td><td>Each AC-RoT has 1 entry. The Configuration Data determines the feature enablement and attestation
<table>
<tr><th>Byte</th><th>Description</th></tr>
<tr><td>1</td><td>Device Id</td></tr>
<tr><td>4</td><td>Channel</td></tr>
<tr><td>5</td><td>Slave Address</td></tr>
<tr><td>6</td><td>[7:5] Threshold Count
[4] Power Control
0 = Disabled
1 = Enabled
[3] Debug Enabled
0 = Disabled
1 = Enabled
[2] Auto Recovery
0 = Disabled
1 = Enabled
[1] Policy Active
0 = Disabled
1 = Enabled
[0] Threshold Active
0 = Disabled
1 = Enabled</td></tr>
<tr><td>7</td><td>Power Ctrl Index</td></tr>
<tr><td>8</td><td>Failure Action</td></tr>
</table></td></tr>
<tr><td>N:N</td><td>Signature of payload</td></tr>
</table>

电源控制索引通知 PA-RoT 分配给组件电源序列的索引。这会通知 PA-RoT 需要在平台电源定序器中声明哪个控制寄存器。

可能的故障操作：

- 0x00：平台定义。
- 0x01：仅报告。
- 0x02: 自动恢复。
- 0x03：电源控制。

准备固件更新

为传入的固件更新提供 RoT。

Request - 请求

Payload	Description
1:4	Total size

更新固件

闪存描述符结构描述了设备的闪存区域。

Request - 请求

Payload	Description
1:N	Firmware Update payload, header signature.

更新状态

更新状态报告更新负载状态。更新状态将是请求的最后一个操作的状态。此状态将保持不变，直到执行另一项操作或重置 Cerberus。

Request - 请求

Payload	Description
1	Update Type
2	Port Id

可能的更新类型：

- 0x00：固件
- 0x01：PFM
- 0x02：CFM
- 0x03：PCD
- 0x04：主机固件
- 0x05：恢复固件
- 0x06：重置配置

Response - 响应

Payload	Description
1:4	Update Status. See firmware update specification for details.

扩展更新状态

扩展更新状态报告更新有效负载状态以及预期的剩余更新字节数。更新状态将是请求的最后一个操作的状态。此状态将保持不变，直到执行另一项操作或重置 Cerberus。

Request - 请求

Payload	Description
1	Update Type
2	Port Id

可能的更新类型：

- 0x00：固件
- 0x01：PFM
- 0x02：CFM
- 0x03：PCD
- 0x04：主机固件
- 0x05：恢复固件
- 0x06：重置配置

Response - 响应

Payload	Description
1:4	Update Status. See firmware update specification for details.
5:8	Expected update bytes remaining.

激活固件更新

提醒 Cerberus 更新字节的发送已完成，应该开始更新验证。此命令没有有效负载，预计错误响应为零。

Request - 请求

空的数据体。

重置配置

将配置参数重置为默认状态。根据请求参数的不同，可以擦除不同数量的类型配置，并且每种类型的配置可能需要不同级别的授权才能完成。

如果完成操作需要授权，则响应将包含特定于设备的一次性使用授权令牌，必须使用 PFM 密钥签名才能解锁操作。授权令牌具有以下行为：

1. 在不提供签名授权令牌的情况下对相同操作的请求将生成一个使任何旧令牌无效的新令牌。即使尚未使用旧令牌也是如此。
2. 授权令牌被用于解锁操作后，将永远无法再次使用。必须请求新令牌。即使请求的操作无法成功完成也是如此。

3. 提供签名令牌时未能授权请求不会使设备中的当前授权令牌无效。

如果不需要授权，或者请求是使用签名令牌发送的，将返回一个标准错误响应来指示状态。

Request - 请求

Payload	Description
1	Type of reset operation to request.
2:N	(Optional) Device-specific authorization token, signed with PFM key.

可能的重置类型：

- 0x00：通过擦除所有 PFM 和 CFM 将设备恢复到不受保护（旁路）状态。
- 0x01：通过删除所有配置执行出厂重置。这不包括签名的设备证书。

Response - 响应

Payload	Description
1:N	Device-specific authorization token

获取配置 ID

此命令检索 PFM Id、CFM Id、PCD Id 以及请求随机数和响应 id 的签名摘要。

Request - 请求

Payload	Description
1:32	Random Nonce

Response - 响应

Payload	Description
1:32	Random Nonce
33	Number of PFM Ids (P)
34	Number of CFM Ids (C)
35:(*P4* + *C4* + 4) (V')	PFM, CFM, PCD Version IDs
V'+1:M	PFM, CFM, PCD Platfomrm IDs
M+1:SIGN	`SIGN(request || response)`

恢复固件

启动设备的固件恢复过程。并非所有设备都支持所有类型的恢复。实现是特定于设备的。

Request - 请求

Payload	Description
1	Port Id
2	Firmware image to use: 0x00 = Exit Recovery, 0x01 = Enter Recovery

准备恢复镜像

为端口的传入恢复映像提供 RoT。

Request - 请求

Payload	Description
1	Port Id
2:5	Total size

更新恢复镜像

闪存描述符结构描述了设备的闪存区域。

Request - 请求

Payload	Description
1	Port Id
2:N	Recovery Image Payload

激活恢复映像

信号恢复图像已完全发送，图像验证应开始。映像通过验证后，即可用于主机固件恢复。

Request - 请求

Payload	Description
1	Port Id

获取恢复映像 ID

检索恢复映像标识符。

Request - 请求

Payload	Description
1	Port Id

Payload	Description
2 (optional)	Identifier: 0x00 = Version Id (default), 0x01 = Platform Id

Response - 响应

Payload	Description
1:N	Recovery Image Platform Id as null-terminated ASCII

获取平台测量寄存器

返回 Cerberus 平台测量寄存器 (PMR)，它是 Cerberus 固件、PFM、CFM 和 PCD 的摘要。PMR0 中包含的此信息是 Cerberus 固件。

PMR1-2预留给PFM/CFM/PCD等配置。

PMR3-4 保留供外部使用。

证明要求至少保留 PMR0，因为这是质询响应消息中报告的测量值。PMR0 至少应包含设备的任何安全配置和所有固件组件。这包括任何正在运行的固件以及为启动设备而运行的固件。为了便于证明，PMR0 旨在仅包含静态信息。如果还将测量可变数据，则应通过 PMR1-2 公开这些测量结果。

Request - 请求

Payload	Description
1	Platform Measurement Number
2:33	Random Nonce

Response - 响应

Payload	Description
1:32	Random Nonce
33	Measurement length (L)
34:33+L	Platform Measurement Value
34+L:N	`SIGN(request ++ response)`

PMR1-4 在组件复位时被清除。PMR0 在 Cerberus 复位时被清除并重建。

更新平台测量寄存器

允许对 PMR3-4 进行外部更新。尝试更新 PMR0-2 将导致错误。测量扩展仅支持 SHA2。SHA1 和 SHA3 不适用。注意：测量只能通过经过身份验证和安全的通道进行更新。

Request - 请求

Payload	Description
1	Platform Measurement Number
2:N	Measurement Extension

重置计数器

自上电以来提供 Cerberus 和组件复位计数器。

Request - 请求

Payload	Description
1	Reset Counter Type
2	Port Id

可能的重置计数器类型：

- 0x00：本地设备
- 0x01：受保护的外部设备（如果适用）。 这些不包括设备挑战的外部 AC-RoT。
- 其他值由供应商定义。

Response - 响应

Payload	Description
1:2	Reset Count

消息解封

此命令开始解封证明消息。密文仅限于可以与开封所需的其他部分一起放入单个消息中的内容。

Request - 请求

Payload	Description
1	[7:5] Reserved [4:2] HMAC Type: 000 – SHA256 [1:0] Seed Type: 00 – RSA: Seed is encrypted with an RSA public key 01 – ECDH: Seed is an ECC public key, ASN.1/DER encoded
2	Additional Seed Parameters RSA: [7:3] Reserved [2:0] Padding Scheme: 000 – PKCS#1 v1.5 001 – OAEP using SHA1

	010 – OAEP using SHA256 ECDH: [7:1] Reserved [0]: Seed Processing: 0 – No additional processing. Raw ECDH output is the seed. 1 – Seed is a SHA256 hash of the ECDH output.
3:4	Seed Length (S)
5:4+S (S')	Seed
S'+1:S'+2	Cipher Text Length (C)
S'+3:S'+2+C (C')	Cipher Text
C'+1:C'+2	HMAC Length (H)
C'+3:C'+2+H (H')	HMAC
H'+1:H'+64 (P0')	PMR0 Sealing, 0's to ignore. Unused bytes are first and must be set to 0.
P0'+1:P0'+64 (P1')	PMR1 Sealing, 0's to ignore. Unused bytes are first and must be set to 0.
P1'+1:P1'+64 (P2')	PMR2 Sealing, 0's to ignore. Unused bytes are first and must be set to 0.
P2'+1:P2'+64 (P3')	PMR3 Sealing, 0's to ignore. Unused bytes are first and must be set to 0.
P3'+1:P3'+64	PMR4 Sealing, 0's to ignore. Unused bytes are first and must be set to 0.

消息解封结果

此命令检索解封过程的当前状态。

Request - 请求

空的数据体。

Response - 响应

Payload	Description
1:4	Unsealing status

Payload	Description
1:4	Unsealing status
5:6	Encryption Key Length
7:N	Encryption Key

Seal/Unseal 流程在 Cerberus 证明集成规范中进行了描述。

平台主动 RoT (PA-RoT)

PA-RoT 负责挑战 AC-RoT 并收集它们的固件测量值。PA-RoT 保留了主动器件的私有清单，其中包括地址、总线、固件版本、摘要和固件拓扑。

清单通知 PA-RoT 有关系统中所有主动器件的信息。它提供了它们的 I2C 地址，以及有关如何根据已知或预期状态验证其测量的信息。平台 RoT 中配置的策略决定了在测量验证失败时应该采取什么行动。

在 Cerberus 设计的主板中，PA-RoT 协调上电。只有挑战清单中列出的主动部件，通过验证才会从上电复位中释放。

平台固件清单 (PFM) 和组件固件清单

PA-RoT 包含一个平台固件清单 (PFM)，它描述了平台上允许的固件。组件固件清单 (CFM) 描述了平台中允许的组件固件。特定于每个 SKU 的平台配置数据 (PCD) 描述了平台中组件类型的数量及其各自的位置。

注意：PFM 和 CFM 不同于处理器安全启动要求规范中描述的启动密钥清单。PFM 和 CFM 描述允许跨平台和主动器件在系统中运行的固件。

CFM 是 PFM 的补充，由 PA-RoT 生成，用于系统中组件的测量比较。此补充作为 Reported Firmware Manifest (RFM)，类似于 TCG 日志。PFM 和 RFM 加密存储在 PA-RoT 中。PA-RoT 的对称加密密钥是硬件生成的，并且对每个微控制器都是唯一的。PA-Rot 中的对称密钥不可导出或固件不可读；并且只能由加密引擎访问以进行加密/解密。AES Galois/Counter Mode (GCM) 在应用程序级别和持久存储级别对清单的任何更改加密一个唯一的可审计标签。

下表列出了每个主动器件的 PFM 中存储的属性：

Attribute	**Description**
Description	Device Part or Description
Device Type	Underlying Device Type of AC-RoT
Remediation Policy	Remediation actions for integrity failure.
Firmware Version	List of firmware versions
Flash Areas/Offsets	List of offset and digests, used and unused
Measurement	Firmware Measurements
Measurement Algorithm	Algorithm used to calculate measurement.
Public Key	Public keys in the key manifest.
Digest Algorithm	Algorithm used to calculate.
Signature	Firmware signature(s)

PA-RoT 主动从平台固件中测量闪存，PFM 提供元数据，指示 RoT 进行测量和签名验证。PA-RoT 将测量结果存储在 RFM 中。然后 PA-Rot 使用平台配置数据挑战 AC-RoT 的测量。它将 AC-RoT 的测量值与 CFM 的测量值进行比较，同时在 RFM 中记录测量值。

平台固件和组件固件的测量结果与 PFM 和 CFM 进行了比较。如果发生不匹配，PA-RoT 将生成事件日志并调用为平台和/或组件定义的策略操作。对于 PFM/CFM 挑战失败，可以自动执行多种操作。动作在 CFM 和 PCD 文件中定义。

注意：PA-RoT 和 AC-RoT 强制执行安全启动，并且只允许下载数字签名和未撤销的固件。PFM 或 CFM 不匹配只会在固件完整性受到质疑时发生。

RoT对外通信接口

PA-RoT 通过 SPI、QSPI 连接到平台，具体取决于主板。尽管 PA-RoT 物理连接到 SPI 总线，但微处理器对主机来说似乎是透明的，因为它只提供一个闪存接口。PA-RoT 和 AC-RoT 的管理接口是通过底板管理控制器 (BMC) 引导的 I2C 总线。BMC 可以到达平台内所有的AC-RoT。BMC 将 PA-RoT 桥接到机架管理器，机架管理器又将机架桥接到数据中心管理网络。PA-RoT 的接口如下：

> TODO：图 12

数据中心管理 (DCM) 软件可以通过机架管理器与 PA-RoT 带外 (OOB) 通信。机架管理器允许通过隧道连接到底板管理控制器，后者通过 I2C 连接到 PA-RoT。该通道被认为是不安全的，这就是所有通信都经过身份验证和加密的原因。数据中心管理软件可以通过这个安全通道收集 RFM 测量和其他挑战数据。也可以通过此渠道进行安全更新。

主机接口

主机可以通过 BMC 主机接口与 PA-RoT 和 AC-RoT 进行通信。与 OOB 路径类似，BMC 将主机端 LPC/eSPI 接口桥接到 RoT 上的 I2C 接口。通过 BMC 的主机是一个不安全的通道，因此需要身份验证和机密性。

带外 (OOB) 接口

OOB 接口对于在开机期间报告潜在的固件危害至关重要。如果在开机期间发生固件损坏，则 OOB 通道可以在 CPU 保持复位状态时与 DCM 软件通信。

如果恢复策略确定系统应保持关闭状态，DCM 软件仍然可以询问 PA-RoT 的详细状态并确定补救措施。

与 Cerberus 的 OOB 通信需要 TLS 和证书身份验证。

旧版界面

遗留接口被定义为与不支持 MCTP 的设备向后兼容。这些设备必须提供一个寄存器集，其中包含用于设备功能的特定偏移量、接收别名证书、接受 Nonce 并为签名固件测量提供偏移量。有效载荷结构将与 MCTP 协议版本紧密匹配。旧版接口不支持基于会话的身份验证，但允许签名测量。

协议格式

传统协议利用 SMBus 写/读字和块命令。

该接口是基于寄存器的，使用 I2C 设备的类似读写子程序。数据传输和接收要求为 32 字节或更大。

可以跨多个块读取或写入命令递归地截断和检索大的有效负载。

块读取 SMBUS 命令在 SMBUS 规范中指定。从机地址写入和命令代码字节由主机传输，然后重复开始，最后读取从机地址。当从机响应所选数据时，主机保持计时。命令代码字节可以被认为是寄存器空间。

PEC处理

SMBus 遗留协议实现可以利用 8 位 SMBus 数据包错误检查 (PEC) 来确保事务数据的完整性。PEC 由每个数据包的发送器和接收器使用读取或写入总线事务的 8 位循环冗余校验 (CRC-8) 计算得出。PEC 累积在开始条件之后发送或接收的所有字节。

接收到无效 PEC 的活动 RoT 可以选择 NACK 携带不正确 PEC 值的字节或丢弃事务和任何进一步事务（读取或写入）的数据，直到接收到下一个有效读取或写入 Start 事务。

消息拆分

该协议支持 Write Block 和 Read Block 命令。标准 SMBus 事务限制为 32 字节的数据。预计一些具有固有 Cerberus 功能的主动组件 RoT 可能具有围绕 SMBus 协议设计的有限 I2C 消息缓冲区，将它们限制为 32 字节。

为了克服消息长度的硬件限制，功能寄存器包括一个缓冲区大小，用于确定消息的最大数据包大小。

这允许平台的 Active RoT 发送大于 32 字节的消息。

如果主动器件 RoT 只允许 32 字节的数据，则平台的活动 RoT 可以将读取或写入块分成多个数据包，总计整个消息。每个段都包括递减的数据包编号，该编号按顺序标识整个消息的一部分。为了保持在协议长度内，每个消息段不得超过 255 个字节。

有效载荷格式

SMBus 写入和读取块的有效负载部分将封装本规范中定义的协议。为了简化，规范的这一部分省略了 SMBus START 和 STOP 帧以及 ACK/NACK 位条件。要查看 START 和 STOP 数据包帧以及 ACK/NACK 条件的细节，请参阅 SMBus 规范。

写入和读取命令的数据块将封装消息有效负载。封装的有效负载包括一个 uint16 寄存器偏移量和数据部分。

注册格式

SMBUS 命令字节索引寄存器，而额外的写入偏移索引在寄存器空间内。偏移量和相应的响应被封装到 I2C 写入和读取块命令的数据部分中。PA-RoT 始终是 I2C master，因此 Write 和 Read 命令是从 I2C master 的角度来描述的。

当跨多个命令写入寄存器时，某些寄存器可能包含部分或临时数据。寄存器写入的完成或密封可以通过将密封寄存器写入零偏移量来执行。

下图描述了大寄存器空间的寄存器读取访问流程：

> 待办事项：图 14

下图描述了大寄存器空间的寄存器写入访问流程，需要密封（更新完成位）：

> TODO：图 15

遗留主动器件 RoT 命令

下表描述了 Active Component RoT 接受的命令。

所有命令均由主机启动。命令编号不代表连续的内存空间，而是相应寄存器的索引

Register Name	Command	Length	R/W	Description
Status	0x30	2	R	Command Status
Firmware Version	0x32	16	R/W	Retrieve firmware version information
Device Id	0x33	8	R	Retrieves Device Id
Capabilities	0x34	9	R	Retrieves Device Capabilities
Certificate Digest	0x3c	32	R	SHA256 of Device Id Certificate
Certificate	0x3d	4096	R/W	Certificate from the AC-Rot
Challenge	0x3e	32	W	Nonce written by RoT
Platform Configuration Register	0x03	0x5e	R	Reads firmware measurement, calculated with S Nonce

遗留命令格式

以下部分描述了不实现 SMBUS 并遵守传统测量交换协议的 AC-RoT 的寄存器格式。

地位

SMBUS 读取命令读取有关错误状态的详细信息。状态寄存器在写入挑战随机数和读取测量值之间发出。导出测量的延迟时间必须符合能力命令。

Payload	Description
1	Status: 0x00 = Complete, 0x01 = In Progress, 0x02 = Error
2	Error Data or Zero

固件版本

SMBUS 写命令负载设置索引。随后的 SMBUS 读取命令读取响应。有关寄存器有效负载描述，请参阅响应： 表 11 固件版本响应

设备ID

SMBUS 读取命令读取响应。有关寄存器负载描述，请参阅响应：表 1 字段定义。

设备能力

SMBUS 读取命令读取响应。有关寄存器有效负载描述，请参阅响应：表 13 设备功能响应

证书摘要

SMBUS 读取命令读取响应。有关寄存器负载描述，请参阅响应：表 24 `GET DIGEST` 响应

PA-Rot 将使用摘要来确定它是否已经缓存了证书。与 MCTP 不同，仅支持别名和设备 ID 证书。

因此，它必须由相互信任的 CA 签名，因为 CA 公共证书不存在

证书

SMBUS 写命令将偏移量写入寄存器空间。有关寄存器有效负载描述，请参阅响应：表 26 `GET CERTIFICATE` 响应

与 MCTP 不同，仅支持别名和设备 ID 证书。因此，它必须由相互信任的 CA 签名，因为 CA 公共证书不存在于减少的质询中

SMBUS 写入命令为测量新鲜度写入随机数。

Payload	Description
1:32	Random 32 byte nonce chosen by PA-RoT

测量

SMBUS 读取命令，该命令使用上述 hallenge 中的随机数读取带符号的测量值。PA-RoT 必须在发出质询之后和读取测量之前轮询状态寄存器是否完成。

Payload	Description
1	Length (L) of following hash digest.
2:33	`H(Challenge Nonce ** H(PMR0))`
34:N	Signature of HASH [2:33]

参考

1. DICE架构 https://trustedcomputinggroup.org/work-groups/dice-architectures
2. 暴动 https://www.microsoft.com/en-us/research/publication/riot-a-foundation-for-trust-in-the-internet-of-things
3. DICE 和 RIoT 密钥和证书 https://www.microsoft.com/en-us/research/publication/device-identity-dice-riot-keys-certificates
4. USB Type C认证规范 http://www.usb.org/developers/docs
5. PCIe 设备安全增强规范 https://www.intel.com/content/www/us/en/io/pci-express/pcie-device-security-enhancements-spec.html
6. NIST 特别出版物 800-108 - 使用伪随机函数推导密钥的建议。 http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-108.pdf
7. TCG PC客户端平台固件配置文件规范 https://trustedcomputinggroup.org/resource/pc-client-specific-platform-firmware-profile-specification