ICS 00.000
X XX

CIIPA

团　　体　　标　　准

T/CIIPA 0000X—2024

自主可控网络安全技术　机密计算安全服务接口规范

Autonomous and controllable network cybersecurity technology——Interface specification of confidential computing security services

（征求意见稿）

20XX-XX-XX 发布　　20XX-XX-XX 实施

中关村华安关键信息基础设施安全保护联盟　发布

目　　次

前　言

本文件按照《中关村华安关键信息基础设施安全保护联盟标准管理办法（暂行）》的要求，依据 GB/T 1.1—2020《标准化工作导则 第 1 部分：标准化文件的结构和起草规则》的规定起草。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别专利的责任。

本文件由中关村华安关键信息基础设施安全保护联盟提出。

本文件由中关村华安关键信息基础设施安全保护联盟网络安全标准专业委员会归口和解释。

本文件起草单位：麒麟软件有限公司、奇安信科技集团股份有限公司、飞腾信息技术有限公司、中关村华安关键信息基础设施安全保护联盟、北京源堡科技有限公司、中电（海南）联合创新研究院有限公司、华北电力大学、北京江民新科技术有限公司、北京国家金融科技认证中心有限公司、中电和瑞科技有限公司、北京神州绿盟科技有限公司。

本文件起草人：于博、纪胜龙、张大朋、战茅、靳佑鼎、王震、王舒、姬一文、张帆、岳佳圆、杨诏钧、吴彤、于晴晴、张建林、杨伟平、张坤、陈晓峰、常凯翔、郦文琪、廖菁菁、杨修、唐磊、黄江、刘涛、赵勇、王雪松、高翔。

本文件首次发布。

本文件在执行过程中的意见或建议反馈至中关村华安关键信息基础设施安全保护联盟（地址：北京市海淀区板井路 69 号世纪金源商务中心 607，100097，网址：http://www.ciipa.com，邮箱：guanbaolianmeng@cnciipa.com）。

引　言

自主可控网络安全技术，能够从根源上缓解信息技术产品存在“后门”、供应链不可控等安全风险，并通过在自主可控产品中植入安全模块、自主可控系统各安全功能协同等措施，提升网络安全防护的效率与能力。

安全模块的实现需要系统提供接口，而不同系统的安全功能协同、跨平台之间的互联互通、用户应用程序的迁移和维护都需要解决机密计算安全服务接口、参数、版本等的一致性和统一性问题，从而实现机密计算安全服务的标准化。安全服务接口是实现机密计算的入口，目的是屏蔽底层硬件架构和软件的开发接口差异，为上层应用程序提供统一的机密计算服务接口及安全服务。因此，为了更好地实现自主可控网络安全技术，制定机密计算安全服务接口规范势在必行。

自主可控网络安全技术机密计算安全服务接口规范

1 范围

本文件确立了网络安全服务机构面向关键信息基础设施的安全服务以及自身安全保障等应具备的安全服务能力相关要求和评价标准。

本文件适用于网络安全服务机构对关键信息基础设施提供安全服务的相关能力建设的参考。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB/T 25069-2022 信息安全技术 术语

GB/T 39204-2022 信息安全技术 关键信息基础设施安全保护要求

GB/T 32914-2023 信息安全技术 网络安全服务能力要求

GB/T 42446-2023 信息安全技术 网络安全从业人员能力基本要求

GB/T 42461-2023 信息安全技术 网络安全服务成本度量指南

GB/T 31496-2023 信息技术 安全技术 信息安全管理体系 指南

GB/T 43269-2023 信息安全技术 网络安全应急能力评估准则

GB/T 20986-2023 信息安全技术 网络安全事件分类分级指南

GB/T 35274-2023 信息安全技术 大数据服务安全能力要求

GB/T 31168-2023 信息安全技术 云计算服务安全能力要求

GB/T 40753-2021 供应链安全管理体系 ISO 28000 实施指南

GB/T 20984-2022 信息安全技术 信息安全风险评估方法

GB/T 36959-2018 信息安全技术 网络安全等级保护测评机构能力要求和评估规范

GB/T 30283-2022 信息安全技术 信息安全服务 分类与代码

GB/T 22080-2016 信息技术 安全技术 信息安全管理体系 要求

GB/T 30271-2013 信息安全技术 信息安全服务能力评估准则

国家网络安全事件应急预案（2017 年 1 月 10 日 中央网络安全和信息化领导小组办公室〔2017〕4 号公布）

网络产品安全漏洞管理规定（2021 年 7 月 12 日 工业和信息化部 国家互联网信息办公室 公安部〔2021〕66 号公布）

3 术语和定义

GB/T 32914、GB/T 25069、GB/T 39204、XXX 界定的以及下列术语和定义适用于本文件。

3.1 **关键信息基础设施** critical information infrastructure

公共通信和信息服务、能源、交通、水利、金融、公共服务、电子政务、国防科技工业等重要行业和领域的，以及其他一旦遭到破坏、丧失功能或者数据泄露，可能严重危害国家安全、国计民生、公共利益的重要网络设施、信息系统等。

3.2 **网络安全服务** cybersecurity service

根据服务协议，基于服务人员、技术、工具、管理和资金等资源，提供保障网络运行安全、网络信息安全等服务的相关过程。

3.3 **网络安全服务机构** cybersecurity service provider

提供网络安全服务的组织。
注：简称“服务机构”

3.4 **网络安全服务需求方** cybersecurity service qcquirer

获取外部所提供的网络安全服务，以满足网络安全需求，实现自身业务目标的组织或个人。
注：简称“服务需求方”或“需方”

3.5 **网络安全服务提供方** cyber security service provider

按照服务协议，通过专业的网络安全人员提供网络安全服务的组织或个人。
注：简称“服务提供方”或“供方”

3.6 **服务协议** service contract

服务开始前供需双方共同签署，并在服务过程中共同遵守的约定。
注：服务协议形式上是服务合同及其附属的工作说明书。

3.7 **服务水平** service level

在服务协议中对服务交付成果明确约定、可测量和文档化的一系列服务指标。

3.8 **服务要素** service factors

规划和实施服务所需的服务人员、服务流程、服务环境、服务方法、服务工具和服务保障等要素。

4 机密计算安全服务接口

4.1 **服务方案** service plans

基于服务目标，对服务各阶段中所需执行的过程、任务、活动以及相关服务要素、服务水平进行详细描述的文档。

4.2 **服务变更** service change

对服务范围、服务目标、服务依据、服务内容、服务水平、服务价格和服务要素等进行新增、修改或解除的活动。

5 缩略语

GE 通用飞地（General Enclave)

CA　普通应用（Client Application）
TA　可信应用（Trusted Application)
API　应用编程接口 (Application Programming Interface)
TEE　可信执行环境 (Trusted Execution Environment)

6　机密计算通用框架

GB/T xxxxx-xxxx《信息安全技术 机密计算通用框架》对于机密计算通用框架的定义如图 1 所示，包括硬件层、系统软件层、系统服务层、应用层和安全管理五个部分。

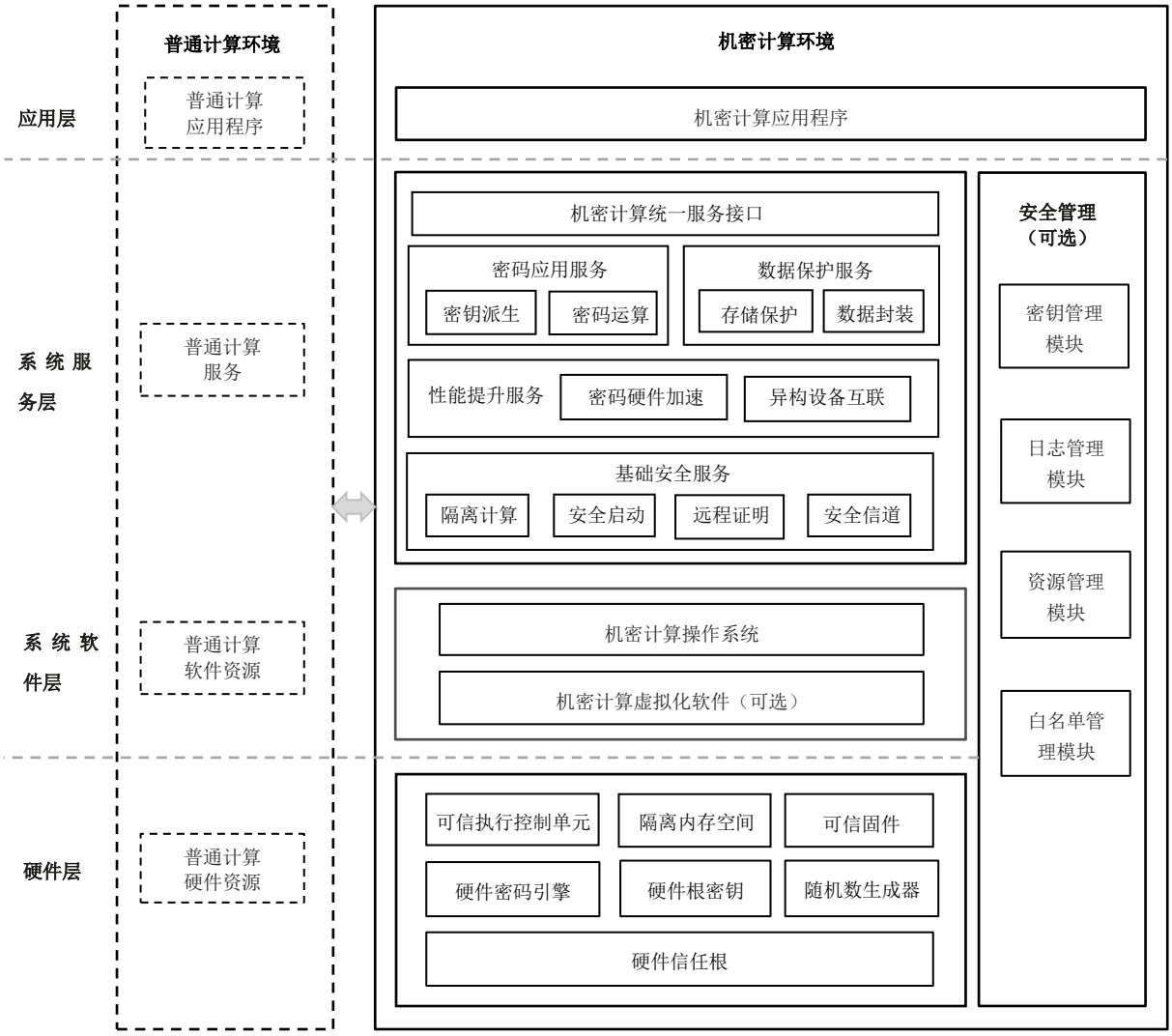

图1 机密计算通用框架图

其中，硬件层基于硬件隔离实现受保护的资源不被开放系统访问，并基于硬件安全功能为机密计算提供受信任的硬件基础；系统软件层为机密计算提供基于软件的隔离机制、必要的硬件资源和基础服务；服务层为上层应用程序提供统一的机密计算服务接口及安全服务，系统服务是由底层的系统软件和硬件以及管理模块交互形成，机密计算统一服务接口用以屏蔽底层硬件架构和软件的开发接口差异；应用层是直接面向结果需求方的应用程序，需求方通过应用程序执行计算操作；安全管理为开展机密计算活动提供必要的管理模块。

服务层中的“机密计算统一服务接口”，可供机密计算应用程序调用，属于 T/ZSA xxx-xxxx《自主可控网络安全技术 安全框架》中基础软件安全能力的“操作系统自有安全机制”部分。

7 机密计算安全服务接口

7.1 隔离计算类接口

7.1.1 创建机密计算 enclave

原型：

```
ge_result_t ge_host_create_enclave(
    const char *path,
    ge_enclave_type_t type,
    uint32_t flags,
    const ge_enclave_features_t *features,
const uint32_t features_count,
    ge_enclave_t    *enclave
);
```

描述：创建机密计算应用程序，创建 enclave。成功返回后，enclave 完全初始化并准备使用。

参数：path [in] 要加载的 enclave 路径

type [in] 支持的可信执行环境类型

flags [in]enclave 的运行方式，保留

features [in] 指向 enclave 类型的附加属性数组

features_count [in] 附加属性的数量

enclave [out] 输出创建的 enclave

返回值： GE_SUCCUSS 成功

其他 失败，返回错误代码

7.1.2 销毁机密计算 enclave

原型：

```
ge_result_t ge_host_destroy_enclave(
      ge_enclave_t *enclave
);
```

描述：销毁机密计算应用程序，终止一个 enclave 并收回其资源。

参数：enclave [in] 已经创建需销毁的 enclave

返回值： GE_SUCCUSS 成功

其他 失败，返回错误代码

7.2 远程证明类接口

7.2.1 请求指定 TA 的证明报告

原型：

```
ge_result_t ge_ra_host_get_report(
      ge_get_ra_report_input_t *in,
      ge_ra_buf_t *report
);
```

描述：请求证明报告，创建用于本地或远程认证的报告。

在 host 程序中调用，in 参数中指定待证明 enclave TA 的 taid（uuid）。ge_host_ra_get_report 函数直接连接机密计算环境中的远程证明特权 TA（RAQTA）。远程证明特权 TA 中调用远程证明模块生成对应的远程证明报告。

参数：in [in]　认证报告的输入参数。

report [out]　指向输出报告数据的缓冲区。

返回值：　GE_SUCCUSS 成功

其他　失败，返回错误代码

7.2.2　请求当前 TA 的认证报告

原型：

```
ge_result_t ge_ra_enclave_get_report(
    ge_get_ra_report_input_t *in,
    ge_ra_buf_t *report
);
```

描述：请求证明报告，创建用于本地或远程认证的报告。

该函数由 enclave TA 调用，in 参数中无需传入 TA 对应的 taid（uuid）,生成的是当前调用该函数的 enclave TA 的证明数据。

参数：in [in]　认证报告的输入参数。

report [out]　指向输出报告数据的缓冲区。

返回值：　GE_SUCCUSS 成功

其他　失败，返回错误代码

注意：enclave 不存在 verify 函数，verify 需要在 enclave 外进行。

7.2.3　验证证明报告

原型：

```
ge_result_t ge_ra_host_verify_report(
    ge_ra_buf_t *report,
    const ge_ra_report_verify_t * check_data,
);
```

描述：验证证明报告。

参数：report [in]　待验证的认证报告。

check_data [out]　指向认证基准值信息的指针，认证信息包括 nonce、hash 等。

返回值：　GE_SUCCUSS 成功

其他　失败，返回错误代码

7.3　安全信道类接口

7.3.1　远程客户端建立安全通道

原型：

```
ge_result_t ge_sc_client_init(
    ge_sc_algo_t algo,
    ge_sc_ctx_t *ctx
);
```

描述：客户端（远程连接机密计算环境的客户端）函数。

安全信道初始化，建立通信双方安全可信的传输通道。

参数：algo [in] 指定安全信道算法类型

ctx [out] 安全通道实例上下文

返回值： GE_SUCCUSS 成功

其他 失败，返回错误代码

7.3.2 远程客户端销毁安全通道

原型：

```
ge_result_t ge_sc_client_deinit(
    ge_sc_ctx_t *ctx
);
```

描述：客户端（远程连接机密计算环境的客户端）函数。

安全通道销毁，销毁已建立的通信双方的传输通道。

参数：ctx [out] 安全通道实例上下文

返回值： GE_SUCCUSS 成功

其他 失败，返回错误代码

7.3.3 安全通道传送数据加密

原型：

```
ge_result_t ge_sc_client_encrypt(
    ge_sc_ctx_t *ctx,
    void *plain,
    size_t plain_len,
    void *encrypt,
    size_t *encrypt_len
);
```

描述：客户端（远程连接机密计算环境的客户端）函数。

对传入的明文数据进行加密。加密后的数据才可以通过安全通道进行传输。该函数仅仅作为加密数据使用。

参数：ctx [in] 安全通道实例上下文

plain [in]待加密的明文敏感数据

plain_len [in]plain 缓冲区中明文数据的长度

encrypt [in] 加密后的密文数据。

如果该参数为 NULL，则将所需要的加密缓冲区的长度计算后在 encrypt_len 参数中返回。

encrypt_len [out] 加密后密文数据的长度

返回值： GE_SUCCUSS 成功

其他 失败，返回错误代码

7.3.4 安全通道传送数据解密

原型：

```
ge_result_t ge_sc_client_encrypt(
    ge_sc_ctx_t *ctx,
    void *encrypt,
    size_t *encrypt_len
    void *plain,
    size_t plain_len,
```

);
描述：客户端（远程连接机密计算环境的客户端）函数。
对传入的密文数据进行解密。密文数据由机密计算的 enclave 传回。
参数：ctx [in] 安全通道实例上下文
encrypt [in] 密文数据
encrypt_len [in] 密文数据的长度
plain [out] 解密后的明文数据
如果该参数为 NULL，则不执行解密操作，仅将所需要的加密缓冲区的长度计算后在 plain_len 参数中返回。
plain_len [out] 解密后明文数据的长度
返回值： GE_SUCCUSS 成功
其他 失败，返回错误代码

7.3.5 机密计算服务端建立安全通道

机密计算服务端是指在机密计算服务器上提供服务的应用，位于普通环境（CA）中。
原型：

```
ge_result_t ge_sc_server_init(
    ge_sc_server_ctx_t *ctx
);
```

描述：服务端（提供机密计算服务）CA 函数调用该函数。
安全通道服务端初始化，建立通信双方安全可信的传输通道。
参数：ctx [out] 服务端安全通道实例上下文
返回值： GE_SUCCUSS 成功
其他 失败，返回错误代码

7.3.6 机密计算服务端销毁安全通道

原型：

```
ge_result_t ge_sc_server_deinit(
    ge_sc_server_ctx_t *ctx
);
```

描述：服务端（提供机密计算服务）CA 函数调用该函数。
安全通道服务端销毁，销毁通信双方安全可信的传输通道。
参数：ctx [out] 服务端安全通道实例上下文
返回值： GE_SUCCUSS 成功
其他 失败，返回错误代码

7.3.7 机密计算服务端回调函数

原型：

```
ge_result_t ge_sc_server_callb(
    ge_sec_chl_conn_ctx_t *ctx,
    void *buf,
    size_t buf_len
);
```

描述：服务端（提供机密计算服务）CA 函数调用该函数。
安全通道协商消息处理函数，处理安全通道协商过程中，客户端发送给服务端的消息。

参数：ctx [in]　安全通道实例上下文

buf [in]　接收到来自客户端的消息数据的缓冲区

buf_len [in]　接收到的数据长度

返回值：　GE_SUCCUSS 成功

其他　失败，返回错误代码

7.3.8　机密计算 enclave 数据加密

原型：

```
ge_result_t ge_sc_client_encrypt(
    size_t session_id,
    void *plain,
    size_t plain_len,
    void *encrypt,
    size_t *encrypt_len
);
```

描述：服务端（提供机密计算服务）对应的安全应用（TA）调用该函数。

对传入的明文数据进行加密。加密后的数据才可以通过安全通道传输给客户端程序。

参数：ctx [in]　安全通道实例上下文

plain [in]待加密的明文敏感数据

plain_len [in]plain 缓冲区中明文数据的长度

encrypt [in]　加密后的密文数据。

如果该参数为 NULL，则将所需要的加密缓冲区的长度计算后在 encrypt_len 参数中返回。

encrypt_len [out] 加密后密文数据的长度

返回值：　GE_SUCCUSS 成功

其他　失败，返回错误代码

7.3.9　机密计算 enclave 数据解密

原型：

```
ge_result_t ge_sc_enclave_decrypt(
    size_t session_id,
    void *encrypt,
    size_t *encrypt_len
    void *plain,
    size_t plain_len,
);
```

描述：服务端（提供机密计算服务）对应的安全应用（TA）调用该函数。

对传入的密文数据进行解密。密文数据由客户端程序传递过来。

参数：session_id [in]安全通道索引

encrypt [in]　　密文数据

encrypt_len [in]　　密文数据的长度

plain [out]解密后的明文数据

如果该参数为 NULL，则不执行解密操作，仅将所需要的加密缓冲区的长度计算后在 plain_len 参数中返回。

plain_len [out]解密后明文数据的长度

返回值： GE_SUCCUSS 成功
其他 失败，返回错误代码

7.4 密钥派生类接口

7.4.1 根据策略获取派生公钥

原型：
ge_result_t ge_get_public_key_by_policy(
ge_seal_policy_t seal_policy,
const ge_asymmetric_key_params_t* key_params,
uint8_t** key_buffer,
size_t* key_buffer_size,
uint8_t** key_info,
size_t* key_info_size
);
描述：安全应用（TA）调用该函数。
根据 enclave 的标识和指定策略获取相关联的公钥。
参数：seal_policy [in] 用于派生密钥的标识属性的策略
key_params [in] 非对称密钥派生的参数
key_buffer [out] 指向缓冲区的指针，该缓冲区成功时包含所请求的公钥
key_buffer_size [out] 缓冲区 key_buffer 的大小
key_info [out] 指向缓冲区的可选指针，用于存放特定于 enclave 的 key info，该信息可用于稍后在较新的安全版本中通过调用 ge_get_public_key 检索相同的 key
key_info_size [out] 缓冲区 key_info 的大小
返回值： GE_SUCCUSS 成功
其他 失败，返回错误代码

7.4.2 获取派生公钥

原型：
ge_result_t ge_get_public_key(
const ge_asymmetric_key_params_t* key_params,
const uint8_t* key_info,
size_t key_info_size,
uint8_t** key_buffer,
size_t* key_buffer_size
);
描述：安全应用（TA）调用该函数。
返回与 enclave 的标识关联的公钥。
参数: key_params[in]非对称密钥派生的参数
key_info[in] 用于派生密钥的特定于 enclave 的密钥信息。
该信息可以是来源于 ge_get_public_key_by_policy 函数调用后 key_info 返回的信息。
key_info_size [in]key_info 缓冲区的大小
key_buffer [out] 指向缓冲区的指针，该缓冲区成功时包含所请求的公钥
key_buffer_size [out] key_buffer 的大小
返回值： GE_SUCCUSS 成功
其他 失败，返回错误代码

7.4.3 根据策略获取派生私钥

原型：
ge_result_t ge_get_private_key_by_policy(
ge_seal_policy_t seal_policy,
const ge_asymmetric_key_params_t* key_params,
uint8_t** key_buffer,
size_t* key_buffer_size,
uint8_t** key_info,
size_t* key_info_size
);
描述：安全应用（TA）调用该函数。
根据 enclave 的标识和指定策略获取相关联的私钥。
参数：seal_policy [in] 用于派生非对称密钥的标识属性的策略
key_params [in] 非对称密钥派生的参数
key_buffer [out] 指向缓冲区的指针，该缓冲区成功时包含所请求的私钥
key_buffer_size [out] 缓冲区 key_buffer 的大小
key_info [out] 指向缓冲区的可选指针，用于存放特定于 enclave 的 key info，该信息可用于稍后在较新的安全版本中检索相同的 key
key_info_size [out] 缓冲区 key_info 的大小
返回值： GE_SUCCUSS 成功
其他 失败，返回错误代码

7.4.4 获取派生私钥

原型：
ge_result_t ge_get_private_key(
const ge_asymmetric_key_params_t* key_params,
const uint8_t* key_info,
size_t key_info_size,
uint8_t** key_buffer,
size_t* key_buffer_size
);
描述：返回与 enclave 的标识关联的私钥。
参数：key_params [in] 非对称密钥派生的参数
key_info[in] 用于派生密钥的特定于 enclave 的密钥信息。
key_info_size [in]key_info 缓冲区的大小
key_buffer [out] 指向缓冲区的指针，该缓冲区成功时包含所请求的私钥。
key_buffer_size [out] key_buffer 缓冲区的大小
返回值： GE_SUCCUSS 成功
其他 失败，返回错误代码

7.4.5 释放获取的派生密钥

原型：
void ge_free_key(
uint8_t* key_buffer,
size_t key_buffer_size,

uint8_t* key_info,
size_t key_info_size
);
描述：释放给定的键和/或键信息。
在释放之前，该函数将清空键缓冲区以避免泄露任何机密数据。
参数：key_buffer [in] 如果不是 NULL，则释放 key_buffer 缓冲区
key_buffer_size [in] key_buffer 的大小
key_info [in] 如果不是 NULL，则释放 key_info 缓冲区。
key_info_size [in]key_info 的大小。
返回值： 无

7.5 存储保护类接口

7.5.1 打开安全存储数据

原型：
ge_ss_handle* ge_enclave_ss_open(
const char* name,
const char* mode,
const ge_secure_storage_key *key
);
描述：该函数创建或者打开一个安全存储数据。
参数：filename [in] 数据存储的文件名或者 ID 名
mode [in] 数据存储的模式字符串，可以是'r', 'w'的组合或者'a'，同时支持'+' 属性。数据始终以二进制'b'存储。
key [in] 存储数据对应的加密密钥。该密钥将作为密钥派生密钥被使用，用于派生出真正的数据加密密钥。
如果使用 ge_enclave_ss_open 创建安全数据时使用了该密钥，需要保证该密钥在后续的文件操作中始终可用。
当 key 为 NULL 时，使用从密钥存储服务获取的存储密钥对数据进行加密。
返回值： NULL 失败，错误码保存在 errno 中
其他值成功，返回文件句柄其他

7.5.2 关闭已打开的安全存储数据

原型：
ge_result_t ge_enclave_ss_close(
ge_ss_handle* handle,
);
描述：该函数关闭一个打开的安全存储数据文件。
参数：handle [in] 已打开的数据存储的文件的句柄。
返回值： GE_SUCCESS 成功
GE_ERROR_INVALID_PARAMETER 输入参数有误
GE_ERROR_BASE 通用错误

7.5.3 读取数据

原型：
size_t ge_enclave_ss_read(

void* ptr,
size_t size,
size_t count,
ge_ss_handle* handle,
);
描述：该函数从数据存储文件中读取指定长度的数据。
参数：ptr [out] 指向接收缓冲区的指针，至少能容乃 size*count 字节数据
size [in] 每块数据的大小
count [in]读取的块数量
handle [in] 已打开的数据存储的文件的句柄
返回值： 实际从存储数据文件中读取的数据块的数量。

1. 如果返回值等于请求读取的元素个数，则表示读取操作成功。
2. 如果返回值小于请求读取的元素个数，则表示读取操作未完成或发生错误。这时，可以通过调用 ge_enclave_ss_error 函数来检查是否发生了错误。
3. 如果返回值为 0，则表示已经到达文件末尾，没有更多的元素可供读取。

7.5.4 写入数据

原型：
size_t ge_enclave_ss_write(
const void* ptr,
size_t size,
size_t count,
ge_ss_handle* stream,
);
描述：该函数将数据块写入数据存储文件中。
参数：ptr [out] 指向写入数据块的指针，至少能容乃 size*count 字节数据
size [in] 每块数据的大小
count [in]读取的块数量
handle [in] 已打开的数据存储的文件的句柄
返回值： 实际写入的数据块的数量。
返回值应该等于 count，除非发生了错误。
如果返回值小于 count，那么可能发生了写入错误或达到了文件末尾。如果发生错误，可以通过调用 ge_enclave_ss_error 函数来检查错误信息。

7.5.5 获取数据的当前位置

原型：
long int ge_enclave_ss_tell(
ge_ss_handle* handle,
);
描述：该函数获取文件指针的当前位置。
参数：handle [in] 已打开的数据存储的文件的句柄
返回值： -1 错误，错误码通过 errno 获取
其他 返回一个从文件开头到当前文件指针位置的偏移量。以字节为单位，表示当前位置与文件开头的距离。

7.5.6 设置文件数据位置

原型：

```
int ge_enclave_ss_seek(
    ge_ss_handle*   stream ,
    long offset,
    int whence
);
```

描述：设置流 stream 的文件位置为给定的偏移 offset，参数 offset 意味着从给定的 whence 位置查找的字节数。

参数：handle [in]　已打开的数据存储的文件的句柄

offset [in]　新设置的偏移量的值

whence [in]　设置的偏移量相对的位置，有三种取值

1. SEEK_SET：这个值表示文件的开始。因此，offset 将被解释为从文件开始到文件指针应移动的字节数。

2. SEEK_CUR：这个值表示当前的偏移量。因此，offset 将被解释为从当前位置到文件指针应移动的字节数。

3. SEEK_END：这个值表示文件的结束。因此，offset 将被解释为从文件结束到文件指针应移动的字节数。

返回值：　0 成功

其他　错误，可以通过调用 ge_enclave_ss_error 函数来检查是否发生了错误。

7.5.7 删除存储数据

原型：

```
int32_t ge_enclave_ss_remove(
    ge_ss_handle*   handle
);
```

描述：该函数删除数据存储文件。

参数：handle [in]　已打开的数据存储的文件的句柄

返回值：　0 操作成功

其他　失败，通过 errno 获取错误原因。

7.5.8 获取错误码

原型：

```
int32_t ge_enclave_ss_error(
    ge_ss_handle*   handle
);
```

描述：该函数获取操作错误码。

参数：handle [in]　已打开的数据存储的文件的句柄

返回值：　上一次文件操作的错误码，0 表示无错误。

7.6 密码运算类接口

7.6.1 杂凑运算上下文初始化

原型：

```
ge_crypto_result_t ge_enclave_crypto_md_init(
```

ge_crypto_md_context_t *ctx
);
描述：该函数初始化并返回一个计算哈希值的上下文。
init、setup start update...update、final 是一个连续处理输入数据计算哈希的过程。
参数：ctx [in]　杂凑算法上下文
返回值：　GE_SUCCUSS 成功
其他　失败，返回错误代码

7.6.2　设置杂凑运算上下文

原型：
ge_crypto_result_t ge_enclave_crypto_md_setup(
ge_crypto_md_context_t *ctx,
const ge_crypto_md_info_t *md_info,
int hmac
);
描述：该函数根据输入的杂凑算法信息设置杂凑算法上下文。
参数：ctx [out] 杂凑算法上下文
md_info [in]　杂凑算法信息
hmac [in] 是否计算 hmac
1 - 计算 hmac
0 - 不计算 hmac
返回值：　GE_SUCCUSS 成功
其他　失败，返回错误代码

7.6.3　杂凑运算开始

原型：
ge_crypto_result_t ge_enclave_crypto_md_init(
ge_crypto_md_context_t *ctx,
);
描述：该函数开始进行杂凑计算。Init、Update...Update、Final 是一个连续处理输入数据计算哈希的过程。
参数：ctx [out] 杂凑算法上下文
返回值：　GE_SUCCUSS 成功
其他　失败，返回错误代码

7.6.4　杂凑运算数据更新

原型：
ge_crypto_result_t ge_enclave_crypto_md_update(
ge_crypto_md_context_t *ctx,
const uint8_t* input,
size_t ilen,
);
描述：更新杂凑算法的输入数据。
参数：ctx [in, out] 哈希算法上下文
input [in]输入数据

ilen [in] 输入数据

返回值： GE_SUCCUSS 成功

其他 失败，返回错误代码

7.6.5 杂凑运算结束

原型：

```
ge_crypto_result_t ge_enclave_crypto_md_finish(
    ge_crypto_md_context_t *ctx,
    uint8_t *output,
    uint32_t *olen,
);
```

描述：该函数返回计算得到的 hash 值，同时释放哈希计算上下文资源。

参数：ctx [in] 哈希算法上下文

output [out] 指向输出的哈希值的缓冲区，不能为 NULL

hash_len [in,out] in：output 缓冲区的长度

out：输出的哈希数据的实际长度，单位为字节

返回值： GE_SUCCUSS 成功

其他 失败，返回错误代码

7.6.6 杂凑运算计算杂凑值

原型：

```
ge_result_t* ge_enclave_crypto_digest(
    const ge_crypto_md_info_t *md_info,
    const uint8_t* p_src,
    uint32_t src_len,
    unsigned char* p_hash,
    uint32_t *hash_len,
);
```

描述：该函数对输入数据计算杂凑。

参数：md_info [in] 哈希算法信息

p_src [in]输入数据缓冲区

src_len [in] 输入数据缓冲区长度

p_hash [out] 指向输出的哈希值的缓冲区，不能为 NULL

hash_len [in,out] in：p_hash 缓冲区的长度

out：输出的哈希数据的实际长度，单位为字节

返回值： GE_SUCCUSS 成功

其他 失败，返回错误代码

7.6.7 hmac 计算开始

原型：

```
ge_crypto_result_t ge_enclave_crypto_md_hmac_starts(
    ge_crypto_md_context_t *ctx,
    const unsigned char *key,
    size_t keylen,
);
```

描述：该函数开始进行 hmac 计算。
参数：ctx [in]　哈希算法上下文
　　key [in]　hmac 计算时的密钥
　　key_len [in]　hmac 密钥长度
返回值：　GE_SUCCUSS 成功
　　其他　失败，返回错误代码

7.6.8　hmac 数据更新

原型：

```
ge_crypto_result_t ge_enclave_crypto_md_hmac_update(
    ge_crypto_md_context_t *ctx,
    const unsigned char *input,
      size_t ilen
);
```

描述：更新 hmac 的输入数据。
参数：ctx [in]　哈希算法上下文
　　input [in]hmac 算法输入数据
　　ilen [in] hmac 算法输入数据长度
返回值：　GE_SUCCUSS 成功
　　其他　失败，返回错误代码

7.6.9　hmac 计算结束

原型：

```
ge_crypto_result_t ge_enclave_crypto_md_hmac_finish(
    ge_crypto_md_context_t *ctx,
    unsigned char *output,
    size_t *olen,
);
```

描述：该函数返回计算得到的 hash 值，同时释放哈希计算上下文资源。
参数：ctx [in]　哈希算法上下文
　　output [out]　指向输出的哈希值的缓冲区，不能为 NULL
　　olen [in, out]in：output 缓冲区的长度
　　out：输出的哈希数据的长度，单位为字节
返回值：　GE_SUCCUSS 成功
　　其他　失败，返回错误代码

7.6.10　对称算法设置秘钥

原型：

```
ge_crypto_result_t ge_enclave_crypto_cipher_setkey(
    ge_crypto_cipher_context_t *ctx,
    const unsigned char *key,
  int key_bitlen,
  ge_cipher_operation_t operation
);
```

描述：该函设置密钥及对称密钥操作模式（加密/解密）。

参数：ctx [in]　对称算法运算的上下文
key [in]　指向存放对称密钥的缓冲区
key_bitlen [in]　对称密钥的 bit 长度
operation [in]对称密钥的密钥操作模式（加密/解密）
返回值：　GE_SUCCUSS 成功
其他　失败，返回错误代码

7.6.11　对称算法设置初始向量 iv

原型：

```
ge_crypto_result_t ge_enclave_crypto_cipher_setiv(
    ge_crypto_cipher_context_t *ctx,
    const unsigned char *iv,
  size_t iv_len
);
```

描述：该函数设置对称密钥运算的初始向量。
某些对称算法模式不需要初始向量，此时该函数无效。
参数：ctx [out] 对称算法运算的上下文
iv [in]　指向存放对称密钥初始向量的缓冲区
iv_len [in]　对称密钥初始向量长度
返回值：　GE_SUCCUSS 成功
其他　失败，返回错误代码

7.6.12　对称算法加解密

原型：

```
ge_crypto_result_t ge_enclave_crypto_cipher_cyrpt(
    ge_crypto_cipher_context_t *ctx,
    const unsigned char *iv,
    size_t iv_len,
    const unsigned char *input,
    size_t ilen,
    unsigned char *output,
    size_t *olen
);
```

描述：该函数对输入数据进行加解密操作。
该函数调用前，需要先调用 ge_enclave_crypto_cipher_set_key 函数设置对称密钥。
参数：ctx [in, out] 对称算法运算的上下文
iv [in]　对称算法计算时所使用的初始向量
iv_len [in]　对称算法计算时所使用的初始向量的长度
input [in]指向对称算法计算时的输入数据
ilen [in] 对称算法计算时的输入数据长度
output [out]　指向对称算法计算时的输出数据
olen [in，out]in：对称算法计算时的输出数据缓冲区 output 长度
out：对称算法计算后的输出数据长度
返回值：　GE_SUCCUSS 成功
其他　失败，返回错误代码

7.6.13 对称算法上下文初始化

原型：

```
ge_crypto_result_t ge_enclave_crypto_cipher_init(
    ge_crypto_cipher_context_t *ctx,
);
```

描述：该函数初始化并返回一个对称加解密运算的上下文。

参数：ctx [out] 对称算法运算的上下文

cipher_info [in] 对称算法信息

返回值： GE_SUCCUSS 成功

其他 失败，返回错误代码

7.6.14 设置对称算法上下文

原型：

```
ge_crypto_result_t ge_enclave_crypto_cipher_setup(
    ge_crypto_cipher_context_t *ctx,
    const ge_crypto_cipher_info_t *cipher_info,
);
```

描述：该函数根据输入的对称算法信息设置对称算法上下文。

参数：ctx [out] 对称算法上下文

cipher_info [in] 对称算法信息

返回值： GE_SUCCUSS 成功

其他 失败，返回错误代码

7.6.15 对称算法数据更新

原型：

```
ge_crypto_result_t ge_enclave_crypto_cipher_update(
    ge_crypto_cipher_context_t *ctx,
  const unsigned char *input,
  size_t ilen,
    unsigned char *output,
  size_t *olen
);
```

描述：更新对称算法的输入数据。

参数：ctx [in, out] 对称算法运算的上下文

input [in]指向对称算法计算时的输入数据

ilen [in] 对称算法计算时的输入数据长度

output [out] 指向对称算法计算时的输出数据

olen [in，out]in：对称算法计算时的输出数据缓冲区 output 长度

out：对称算法计算后的输出数据长度

返回值： GE_SUCCUSS 成功

其他 失败，返回错误代码

7.6.16 对称算法计算结束

原型：

```
ge_crypto_result_t ge_enclave_crypto_cipher_finish(
```

ge_crypto_cipher_context_t *ctx,
unsigned char *output,
size_t *olen
);
描述：该函数结束对称算法计算过程，返回剩余部分的解密数据。
注意该函数不会释放计算过程上下文，需要调用 ge_enclave_crypto_cipher_free 进行释放。
参数：ctx [in, out] 对称算法运算的上下文
output [out] 指向对称算法计算时的输出数据
olen [in，out]in：对称算法计算时的输出数据缓冲区 output 长度
out：对称算法计算后的输出数据长度
返回值： GE_SUCCUSS 成功
其他 失败，返回错误代码

7.6.17 对称算法释放上下文

原型：
ge_crypto_result_t ge_enclave_crypto_cipher_finish(
ge_crypto_cipher_context_t *ctx
);
描述：该函数释放对称算法计算上下文资源。
参数：ctx [in] 对称算法运算的上下文
返回值： GE_SUCCUSS 成功
其他 失败，返回错误代码

7.6.18 非对称算法上下文初始化

原型：
ge_crypto_result_t ge_enclave_crypto_pk_init(
ge_crypto_pk_context *ctx
);
描述：该函数释放公钥运算的上下文。
注意 ctx 结构体由调用方管理。
参数：ctx [out] 公钥运算的上下文
返回值： GE_SUCCUSS 成功
其他 失败，返回错误代码

7.6.19 设置非对称算法上下文

原型：
ge_crypto_result_t ge_enclave_crypto_pk_setup(
ge_crypto_pk_context *ctx,
const ge_crypto_pk_info_t *pk_info,
);
描述：该函数根据输入的非对称算法信息设置非对称算法上下文。
参数：ctx [out] 非对称算法上下文
pk_info [in] 非对称算法信息
返回值： GE_SUCCUSS 成功

其他　失败，返回错误代码

7.6.20　非对称算法计算签名

原型：

```
ge_crypto_result_t ge_crypto_pk_sign(
        ge_crypto_pk_context *ctx,
        ge_crypto_md_type_t md_alg,
    const unsigned char *hash,
        size_t hash_len,
unsigned char *sig,
        size_t* sig_len
);
```

描述：该函数对输入数据进行签名。

参数：ctx [in]　非对称算法运算的上下文

md_alg [in]　签名中使用的杂凑算法

hash [in]　指向签名源数据的杂凑缓冲区

hash_len [in]　签名源数据的杂凑长度

sig [in]　指向签名数据的缓冲区

sig_len [in，out] in：缓冲区 sig 长度

out：签名后签名数据长度

返回值：　GE_SUCCUSS 成功

其他　失败，返回错误代码

7.6.21　非对称算法验证签名

原型：

```
ge_crypto_result_t ge_crypto_pk_verify(
        ge_crypto_pk_context *ctx,
        ge_crypto_md_type_t md_alg,
    const unsigned char *hash,
        size_t hash_len,
const unsigned char *sig,
        size_t sig_len
);
```

描述：该函数对输入数据进行签名验证。

参数：ctx [in]　非对称算法运算的上下文

md_alg [in]　签名中使用的杂凑算法

hash [in]　指向签名源数据的杂凑缓冲区

hash_len [in]　签名源数据的杂凑长度

sig [in]　指向签名数据的缓冲区

sig_len [in]　签名数据长度

返回值：　GE_SUCCUSS 成功

其他　失败，返回错误代码

7.6.22　非对称算法加密

原型：

ge_crypto_result_t ge_crypto_pk_encrypt(
ge_crypto_pk_context *ctx,
const unsigned char *input,
size_t ilen,
unsigned char *output,
size_t *olen,
);
描述：该函数对输入数据进行加密。
参数：ctx [in]　公钥运算的上下文
input [in]指向待加密数据缓冲区
ilen [in] 待加密数据长度
output [out]　指向加密后输出数据数据缓冲区
olen [in，out]in：输出数据缓冲区的长度
out：加密后数据长度
返回值：　GE_SUCCUSS 成功
其他　失败，返回错误代码

7.6.23　非对称算法解密

原型：
ge_crypto_result_t ge_crypto_pk_decrypt(
ge_crypto_pk_context *ctx,
const unsigned char *input,
size_t ilen,
unsigned char *output,
size_t *olen,
);
描述：该函数对输入数据进行解密。
参数：ctx [in]　非对称算法运算的上下文
input [in]指向待加密数据缓冲区
ilen [in] 待加密数据长度
output [out]　指向加密后输出数据数据缓冲区
olen [in，out]in：输出数据缓冲区的长度
out：加密后数据长度
返回值：　GE_SUCCUSS 成功
其他　失败，返回错误代码

7.6.24　释放非对称算法上下文

原型：
ge_crypto_result_t ge_enclave_crypto_pk_free(
ge_crypto_pk_context *ctx,
);
描述：该函数释放非对称运算上下文资源。
参数：ctx [in]　非对称算法运算的上下文
返回值：　GE_SUCCUSS 成功
其他　失败，返回错误代码

7.6.25 获取随机数

原型：

```
ge_crypto_result_t ge_enclave_crypto_pk_gen_random(
   unsigned char *rand,
      size_t rand_len
);
```

描述：该函数生成指定长度的随机数。

参数：random [out] 指向输出随机数的缓冲区

random_len [out] 待生成随机数的长度，单位为字节

返回值： GE_SUCCUSS 成功

其他 失败，返回错误代码

7.7 数据封装类接口

7.7.1 封装数据

原型：

```
ge_result_t ge_enclave_sd_seal_data(
      const uint8_t *seal_data,
      uint32_t seal_data_len,
      uint8_t *sealed_data,
      uint32_t sealed_data_len,
      const uint8_t *additional_text,
      uint32_t additional_text_len
);
```

描述：该函数使用 AES-GCM（或者 SM4-GCM）对输入数据进行加密。

ge_enclave_sd_seal_data 函数从安全区派生出唯一的数据密封密钥，并使用该密钥加密数据缓冲区。该功能可用于安全区被销毁后保存机密数据。密封的数据 blob 可以再安全区重新实例化时解封。

额外的数据缓冲不会被加密，它将作为 MAC 的一部分参与到加密数据中。这些数据可能包括应用程序、版本、数据等信息用于标志密封的数据 blob。

参数：seal_data [in]指向待加密原始数据的指针。该参数不能为 NULL，且指向的数据必须在安全区内

seal_data_len [in]待加密原始数据的长度。该参数必须为非 0 值

sealed_data [out] 指向密封数据输出缓冲区

sealed_data_len [in] 分配的 ge_enclave_sealed_data_t 数据缓冲区的长度。该长度需要调用辅助函数 ge_enclave_get_sealed_data_size 提前确定。

additional_text [in] 指向附加的 MAC 数据。该数据是可选的，可以为 NULL（当为 NULL 时，对应的 additional_text_len 应为 0）

additional_text_len [in] 附加 MAC 数据的长度，单位为字节

返回值： GE_SUCCUSS 成功

其他 失败，返回错误代码

7.7.2 根据策略封装数据

原型：

```
ge_result_t ge_enclave_sd_seal_data_ex(
      uint32_t key_policy,
```

const uint8_t *seal_data,
uint32_t seal_data_len,
ge_enclave_sealed_data_t *sealed_data,
uint32_t sealed_data_len,
const uint8_t *additional_text,
uint32_t additional_text_len
);

描述： 使用 AES-GCM（或者 SM4-GCM）对输入数据进行加密。该函数是 ge_enclave_sd_seal_data 函数的高级形式。

参数：key_policy [in] 描述加密所使用的的数据密封密钥的派生策略

seal_data [in]指向待加密原始数据的指针。该参数不能为 NULl，且指向的数据必须在安全区内

seal_data_len [in]待加密原始数据的长度。该参数必须为非 0 值

sealed_data [out] 指向密封数据输出缓冲区

sealed_data_len [in] 分配的 ge_enclave_sealed_data_t 数据缓冲区的长度。该长度需要调用辅助函数 ge_enclave_get_sealed_data_size 提前确定。

additional_text [in] 指向附加的 MAC 数据。该数据是可选的，可以为 NULL（当为 NULL 时，对应的 additional_text_len 应为 0）

additional_text_len [in] 附加 MAC 数据的长度，单位为字节

返回值： GE_SUCCUSS 成功

其他 失败，返回错误代码

7.7.3 解封数据

原型：

ge_result_t ge_enclave_sd_unseal_data(
const ge_enclave_sealed_data_t *sealed_data,
uint8_t *decrypted_data,
uint32_t *decrypted_data_len,
uint8_t *additional_text,
uint32_t *additional_text_len
);

描述：该函数使用 AES-GCM（或者 SM4-GCM）对输入数据进行解密。

参数：seal_data [out] 指向已密封数据

decrypted_data[in] 指向存放解密数据数据缓冲区的指针。缓冲区长度由 decrypted_data_len 确定

decrypted_data_len[in，out] 存放解密数据缓冲区的长度，单位为字节。需要提前调用函数 ge_enclave_get_encrypted_text_size 确定解密数据缓冲区的最小长度。ge_enclave_unseal_data 会通过该参数返回实际的解密数据的长度

additional_text [out] 指向附加的 MAC 数据缓冲区。该数据是可选的，可以为 NULL（当为 NULL 时，对应的 additional_text_len 应为 0）

additional_text_len [in，**out]** 附加 MAC 数据的长度，单位为字节。需要提前调用辅助函数 ge_enclave_get_add_mac_text_size 确定 MAC 数据缓冲区的最小长度。ge_enclave_sd_unseal_data 会通过该参数返回实际的附加数据的长度。

返回值： GE_SUCCUSS 成功
其他 失败，返回错误代码

7.7.4 获取数据封装后长度

原型：
uint32_t ge_enclave_sd_get_unsealed_data_size(
uint32_t add_mac_text_len,
uint32_t seal_data_len
);
描述：该函数返回数据封装后长度。
参数：add_mac_text_len [in] 可选的附件数据的长度，单位为字节
seal_data_len[in] 待加密的源数据的长度，单位为字节
返回值： UINT32_MAX 失败
其他 成功，返回的是需要为结构体 ge_enclave_sealed_data_t 分配的最小的缓冲区长度

7.7.5 获取数据解封后长度

原型：
uint32_t ge_enclave_get_encrypted_text_size(
const ge_enclave_sealed_data_t *sealed_data
);
描述：该函数返回存放解封数据缓冲区的需要的最小长度。
参数：seal_data[in] 指向已加密数据的指针
返回值： UINT32_MAX 失败
其他 成功，返回的是源数据长度

7.8 硬件加速类接口

7.8.1 密码硬件加速设置

原型：
uint32_t ge_cae_set_status(
uint32_t cae_id,
uint32_t status
);
描述：设置密码加速引擎的启用状态。
该状态会影响密码运算类接口的运行，当密码加速引擎设置为开启时，系统将使用密码加速引擎对密码运算类接口进行密码运算。
参数：**cae_id [in]** 密码加速引擎 ID，0 表示由系统分配
status**[in]**密码加速引擎的启用状态，0 表示关闭，1 表示开启
返回值： GE_SUCCUSS 成功

附　　录　A

（规范性）

通用错误码

机密计算安全服务定义的通用错误码见表A.1。

表 A.1 通用错误码

名称	含义
GE_SUCCUSS	成功
GE_ERROR_INVALID	参数非法
GE_ERROR_OUT_OF_MEMORY	安全区内存不足
GE_ERROR_UNEXPECTED	未定义的错误，可能由算法库错误等产生
GE_ERROR_INVALID_ENCLAVE	加载可信应用失败
GE_ERROR_INVALID_PATH	路径非法
GE_ERROR_READ_DATA	读数据错误
GE_ERROR_WRITE_DATA	写数据错误
GE_ERROR_SEEK_DATA	定位数据错误
GE_ERROR_SYNC_DATA	同步数据错误
GE_ERROR_CRYPTO_INVALID_TYPE	算法类型错误
GE_ERROR_PCL_MAC_MISMATCH	解密时 MAC 错误
GE_ERROR_ENCLAVE_MAXIMUM	单个应用创建的 enclave 数量达到最大
GE_ERROR_BASE	通用错误
GE_ERROR_SHORT_BUFFER	传入 buffer 过小
GE_ERROR_INVALID_KEYNAME	无效的 KEYNAME
GE_ERROR_CAE_NOT_SUPPORT	不支持密码加速引擎