厦门市算力调度服务平台
统一接口文档

2025 年 11 月

文档信息

文档名称	厦门市算力调度服务平台统一接口文档
文件目的	统一接口对接标准规范

文件控制

版本记录

日期	作者	版本	变更说明
202511	Geyl	1.0	文档新增

目录

1. 每日指标报送

1.1. 指标设计

1.1.1. 基础指标

序号	基础指标分类	编码前缀	适用范围
1	资源池信息	ZYC	
2	GPU 服务器信息	GPUFWQ	所有资源池
3	存储服务器信息	CCFWQ	所有资源池
4	算力卡信息	SLK	所有资源池
5	客户信息	KHXX	所有资源池
6	用户使用量	YHSYL	所有资源池
7	交易情况	JYQK	所有资源池
8	实时状态	SSZT	所有资源池
9	客户资源使用情况	KHZYSYQK	所有资源池

1.1.2. 业务指标

资源池信息（ZYC）

序号	业务指标编码	指标名称	指标含义	单位	数据类型	说明
1	ZYC_JSON_ARRAY	资源池信息 JSON 数组	资源池的信息，以 JSON 数组的格式上报	Json 数组	String	

【指标说明：所有资源池的信息，数组里有两个属性。poolId 代表 ID，poolName 代表名称】

ZYC_JSON_ARRAY 上报 json 格式说明：

```
[
  {
    "poolId": "1771424516106551298",//资源池 ID
    "poolName": "资源池一"/资源池名称
  },
  {
```

```
    "poolId ": "1771444040981741570",//资源池 ID
    "poolName": "资源池二"/资源池名称
  }
]
```

GPU 服务器信息（GPUFWQ）

序号	业务指标编码	指标名称	指标含义	单位	数据类型	说明
1	GPUFWQ_JSON_ARRAY	GPU 服务器信息 JSON 数组	GPU 服务器的信息，以 JSON 数组的格式上报	Json 数组	String	

【指标说明：所有资源池的信息，数组里有六个属性。gpuServerId 代表 ID，gpuServerName 代表名称，poolId 代表所属资源 ID，cpuNum 代表 CPU 数量，cpuCore 代表 CPU 总核心数，memory 代表内存大小（GB）】

GPUFWQ_JSON_ARRAY 上报 json 格式说明：

```
[
  {
    "gpuServerId": "1771421843957415938",//GPU 服务器 ID
    "gpuServerName": "服务器名称一",//GPU 服务器名称
    "poolId": "1771424516106551298",//所属资源池 ID
    "cpuNum": "32",//CPU 数量
    "cpuCore": "512",//CPU 总核心数
    "memory": "512"//内存大小，单位 GB
  },
  {
    "gpuServerId ": "1771358657971421186",//GPU 服务器 ID
    "gpuServerName": "服务器名称二",//GPU 服务器名称
    "poolId": "1771444040981741570",//所属资源池 ID
    "cpuNum": "32",//CPU 数量
    "cpuCore": "512",//CPU 总核心数
    "memory": "512"//内存大小，单位 GB
  }
]
```

存储服务器信息（CCFWQ）

序号	业务指标编码	指标名称	指标含义	单位	数据类型	说明
1	CCFWQ_JSON_ARRAY	存储服务器信息 JSON 数组	存储服务器的信息，以 JSON 数组的格式上报	Json 数组	String	

【指标说明：所有资源池的信息，数组里有四个属性。storageServerId 代表 ID,storageServerName 代表名称，storageCapacity 代表存储量(TB),poolId 代表所属资源 ID】

CCFWQ_JSON_ARRAY 上报 json 格式说明：

```
[
  {
    "storageServerId": "1773889513445261313",//存储服务器 ID
    "storageServerName": "存储服务器名称一",//存储服务器名称
    "storageCapacity": 2000,//存储量，单位 TB
    "poolId": "1771424516106551298"//所属资源池 ID
  },
  {
    "storageServerId ": "1771733838103511042",//存储服务器 ID
    "storageServerName ": "存储服务器名称二",//存储服务器名称
    "storageCapacity ": 2000,//存储量，单位 TB
    "poolId": "1771444040981741570"//所属资源池 ID

  }
]
```

算力卡信息（SLK）

序号	业务指标编码	指标名称	指标含义	单位	数据类型	说明
1	SLK_JSON_ARRAY	算力卡情况 JSON 数组	算力卡的全部信息,以 JSON 数组的格式上报	Json 数组	String	

【指标说明：以每张算力卡为单位，汇总算力卡的信息，数组里有五个属性。cardId 代表算力卡 ID，gpuServerId 代表所属 GPU 服务器 ID，brandName 代表品牌，modelCode 代表型号，computingPower 代表算力值（T）】

SLK_JSON_ARRAY 上报 json 格式说明：

```
[
  {
    "cardId": "1771421843957415000",//算力卡 ID
    "gpuServerId": "1771421843957415938",//所属 GPU 服务器 ID
    "brandName": "华为",//品牌
    "modelCode":"Atlas 300T Pro",//型号
    "computingPower": 280,//单卡算力值，如果是区间则取区间最大值，单位：T
  },
  {
```

```
        "cardId": "1771421843957415001",//算力卡 ID
        "gpuServerId": "1771421843957415948",//所属 GPU 服务器 ID
        "brandName": "华为",//品牌
        "modelCode":"Atlas 300I Duo",//型号
        "computingPower ": 280,//单卡算力值，单位：T
    }
]
```

客户信息（KHXX）

序号	业务指标编码	指标名称	指标含义	单位	数据类型	说明
1	KHXX_JSON_ARRAY	客户信息 JSON 数组	算力中心所有的客户信息，以 JSON 数组格式上报	Json 数组	String	

【指标说明：以企业为单位，提供已租赁算力的企业信息,数组里有三个属性。entName 代表企业名称，orgCode 代表组织机构编码，industryCode 代表所属行业分类】

KHXX_JSON_ARRAY 上报 json 格式说明：

```
[
  {
    "entName": "厦门某某公司一",//企业名称
    "orgCode":"91330101MA2Y3ABCDE" ,//企业组织机构编码，填社会信用统一代码
    "industryCode":"C" ,//所属行业：详见附录(行业分类编码表)
  },
  {
    "entName": "厦门某某公司二",//企业名称
    "orgCode":"91330101MA2Y3ABCDE" ,//企业组织机构编码，填社会信用统一代码
    "industryCode":"J" ,//所属行业：详见附录(行业分类编码表)
  }
]
```

用户使用量（YHSYL）

序号	业务指标编码	指标名称	指标含义	单位	数据类型	说明
1	YHSYL_JSON_ARRAY	用户使用量 JSON 数组	用户使用量全部信息,以 JSON 数组的格式上报	JSON 数组	String	每日上报当日所有用户的使用量信息

【指标说明：以客户为单位，每日提供租赁客户的算力租赁总数（P）,数组里有四个属性。entName 代表企业名称，orgCode 代表组织机构编码，computeUsage 代表算力使用量，statDate 代表统计日期】

YHSYL_JSON_ARRAY 上报 json 格式说明：

```
[
  {
    "entName": "厦门某某公司一",//用户名称
    "orgCode":"91330101MA2Y3ABCDE",//组织机构编码填社会信用统一代码
    "computeUsage":100,//算力使用量：单位 P
    "domestic":60,//国产算力使用量：单位 P
    "nvidia":40,//英伟达算力使用量：单位 P
    "statDate":"2025-07-01"//统计日期 格式：YYYY-MM-DD
  },
  {
    "entName": "厦门某某学校",//用户名称
    "orgCode":"91330101MA2Y3ABCDE",//组织机构编码填社会信用统一代码
    "computeUsage":100,//算力使用量：单位 P
    "domestic":100,//国产算力使用量：单位 P
    "nvidia":0,//英伟达算力使用量：单位 P
    "statDate":"2025-07-01"//统计日期 格式：YYYY-MM-DD
  }
]
```

交易情况（JYQK）

序号	业务指标编码	指标名称	指标含义	单位	数据类型	说明
1	JYQK_YSSL	已售算力	已销售的算力值	P	Double	
2	JYQK_SYSL	剩余算力	剩余的算力值	P	Double	
3	JYQK_ZKHS	总客户数	总的客户数量	家	Integer	
4	JYQK_BY_XZKHS	本月新增客户数	本月新增的客户数量	家	Integer	按月统计

【指标说明：已租赁的算力、企业信息的汇总情况】

实时状态（SSZT）

序号	业务指标编码	指标名称	指标含义	单位	数据类型	说明

1	SSZT_GPU_SYL	GPU 使用率	算力中心当前 GPU 使用率	%	Double	取值范围 0-100
2	SSZT_CPU_SYL	CPU 使用率	算力中心当前 CPU 使用率	%	Double	取值范围 0-100
3	SSZT_MEM_SYL	内存使用率	算力中心当前内存使用率	%	Double	取值范围 0-100
4	SSZT_DISK_SYL	存储使用率	算力中心当前存储使用率	%	Double	取值范围 0-100

【指标说明：使用率指的是已分配率，如资源池 GPU 共 100P，租赁给客户 50P，指使用率为 50%】

客户资源使用情况（KHZYSYQK）

序号	业务指标编码	指标名称	指标含义	单位	数据类型	说明
1	KHZYSYQK_JSON_ARRAY	客户资源使用情况 JSON 数组	客户资源使用情况全部信息，以 JSON 数组的格式上报	JSON 数组	String	每日上报当日所有用户的使用量信息

【指标说明：以客户为单位，每日提供租赁客户的资源使用情况。entName 代表企业名称，orgCode 代表组织机构编码，details 代表需上传的资源使用情况，每日从 1 点开始至 24 点需传 24 条（其中：hour 代表时，gpuUsageRate 代表 gpu 利用率，cpuUsageRate 代表 cpu 利用率，memoryUsageRate 代表内存利用率，storageUsageRate 代表存储利用率，gpuVramUsageRate 代表 GPU 显存利用率，networkUsageRate 代表网络带宽利用率）】

KHZYSYQK_JSON_ARRAY 上报 json 格式说明：

```
[
  {
    "entName": "厦门某某公司一",//用户名称
    "orgCode":"91330101MA2Y3ABCDE",//组织机构编码填社会信用统一代码
    "details": [
      {
        "hour": "1", //代表时（每日从 1 开始至 24 需传 24 条）
        "gpuUsageRate":"85",//gpu 利用率：单位%
        "cpuUsageRate":"75.5",//cpu 利用率：单位%
        "memoryUsageRate":"60",//内存利用率：单位%
        "storageUsageRate":"45.5",//存储利用率：单位%
        "gpuVramUsageRate":"80",//GPU 显存利用率：单位%
        "networkUsageRate":"44.5"//网络带宽利用率：单位%

      },
```

```
        {
          "hour": "2", //代表时（每日从 1 开始至 24 需传 24 条）
          "gpuUsageRate":"85",//gpu 利用率：单位%
          "cpuUsageRate":"75.5",//cpu 利用率：单位%
          "memoryUsageRate":"60",//内存利用率：单位%
          "storageUsageRate":"45.5"//存储利用率：单位%
          "gpuVramUsageRate":"80",//GPU 显存利用率：单位%
          "networkUsageRate":"44.5"//网络带宽利用率：单位%
        }
      ]
    },
    {
      "entName": "厦门某某学校",//用户名称
      "orgCode":"91330101MA2Y3ABCDE",//组织机构编码填社会信用统
一代码
      "details": [
        {
          "hour": "1", //代表时（每日从 1 开始至 24 需传 24 条）
          "gpuUsageRate":"85",//gpu 利用率：单位%
          "cpuUsageRate":"75.5",//cpu 利用率：单位%
          "memoryUsageRate":"60",//内存利用率：单位%
          "storageUsageRate":"45.5",//存储利用率：单位%
          "gpuVramUsageRate":"80",//GPU 显存利用率：单位%
          "networkUsageRate":"44.5"//网络带宽利用率：单位%
        },
        {
          "hour": "2", //代表时（每日从 1 开始至 24 需传 24 条）
          "gpuUsageRate":"85",//gpu 利用率：单位%
          "cpuUsageRate":"75.5",//cpu 利用率：单位%
          "memoryUsageRate":"60",//内存利用率：单位%
          "storageUsageRate":"45.5",//存储利用率：单位%
          "gpuVramUsageRate":"80",//GPU 显存利用率：单位%
          "networkUsageRate":"44.5"//网络带宽利用率：单位%
        }
      ]
    }
  ]
```

1.2. 接口概述

本接口用于各算力中心向算力调度服务平台同步数据，实现算力资源的集中管理。

1.3. 接口基础信息

1.3.1. 接口协议

采用 HTTPS 协议，确保数据传输安全。

1.3.2. 接口地址

https://[算力调度服务平台 IP:端口]/app-api/metricReport

1.3.3. 请求方法

POST

1.3.4. 数据格式

请求体和响应体均采用 JSON 格式。

1.4. 请求头参数

序号	参数名称	是否必选	类型	描述
1	Content-Type	是	String	固定值：application/json
2	Authorization	是	String	身份认证信息，格式为：Bearer {appId}:{signature}
3	X-Request-ID	是	String	唯一请求标识，建议使用 UUID，用于防重放攻击和请求追踪
4	X-Pool-Type	是	String	资源池类型，固定值：public
5	X-Timestamp	是	Long	请求时间戳（精确到秒），用于防重

				放攻击
6	X-Expires	是	Long	请求过期时间（精确到秒），建议设置为 timestamp + 300（即 5 分钟）
7	X-Sign-Algorithm	是	String	签名算法，固定值：SM3
8	X-Encrypt-Algorithm	是	String	加密算法，固定值：SM4-CBC
9	X-Sign-Nonce	是	String	随机字符串，用于签名生成，每次请求需不同
10	X-Metric-Type	是	String	指标类型，固定值：business

参数中带有中文的，需要进行 UTF-8 编码。

1.5. 请求体结构

1.5.1. 通用结构

RequestBody 请求体 Json 格式：

```
{
  "appId": "平台为每一个对接方分配的唯一标识",
  "sign": "7c4a8d09ca3762af61e59520943dc26494f8941b8f20012872e9",
  "timestamp": 1731234567,
  "data": "加密后的数据（Base64 编码）"
}
```

1.5.2. 字段说明

序号	参数名称	是否必选	类型	描述
1	appId	是	String	平台为每个请求方分配的唯一标识，用于身份验证
2	sign	是	String	签名信息，使用 SM3 算法生成，用于验证请求的合法性和数据完整性
3	timestamp	是	Long	请求时间戳（精确到秒），需与请求头中的 X-Timestamp 一致
4	data	是	String	加密后的数据，使用 SM4-CBC 算法加密，Base64 编码

1.5.3. 加密前的数据结构

```
{
  "metrics": [
```

```
    {
      "metricCode": " JBXX_ZYCSL ", //具体的业务指标编码
      "value": 16, //根据业务指标编码对应的值的数据类型传值
      "reportTime": "2025-07-08 12:00:00" //上报时间
    },
    {
      "metricCode": " SLQK_JSON_ARRAY ",
       "value": "[
                  {
                     "brandName": "华为",//品牌
                     "modelCode":"Atlas 300T Pro" ,//型号
                     "quantity": 100,//数量，单位：张
                     "singleCardPerformance ": 280,//单卡算力值，如
              果是区间值则去区间最大值，单位：T
                  },
                  {
                     "brandName": "华为",//品牌
                     "modelCode":"Atlas 300I Duo" ,//型号
                     "quantity": 80,//数量，单位：张
                     "singleCardPerformance ": 280,//单卡算力值，单
              位：T
                  }
               ]" , //单位是 JSON 数组类型的值示例
      "reportTime": "2025-07-08 12:00:00"
    }
  ]
}
```

1.6. 响应体结构

```
{
  "code": 200,
  "msg": "指标上报成功",
  "requestId": "e6a7e2d3-9f8a-4f5b-9b6c-8f7d5e8f4a3b"
}
```

1.7. 安全机制

1.7.1. 数据加密

✓ 采用 SM4-CBC 算法加密业务指标数据

✓ 密钥长度 128 位（16 字节）

✓ 初始化向量（IV）长度 16 字节

✓ 加密模式：CBC

✓ 填充方式：PKCS7Padding

1.7.2. 数据签名

✓ 采用 SM3 算法生成签名

✓ 签名原始数据：appId + timestamp + 加密后数据 + 应用密钥

✓ 签名结果转换为 16 进制字符串

1.7.3. 密钥管理

✓ 加密密钥和应用密钥通过安全通道预先分发

✓ 定期更换密钥

1.8. 请求方示例代码

```
import org.bouncycastle.crypto.digests.SM3Digest;
import org.bouncycastle.crypto.engines.SM4Engine;
import org.bouncycastle.crypto.modes.CBCBlockCipher;
import org.bouncycastle.crypto.paddings.PaddedBufferedBlockCipher;
import org.bouncycastle.crypto.params.KeyParameter;
import org.bouncycastle.crypto.params.ParametersWithIV;
import org.bouncycastle.jce.provider.BouncyCastleProvider;
import org.bouncycastle.util.encoders.Hex;
import com.google.gson.Gson;
```

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLEncoder;
import java.nio.charset.StandardCharsets;
import java.security.Security;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.*;


public class MetricReportClient {
    // 静态初始化 BouncyCastle 加密库
    static {
        Security.addProvider(new BouncyCastleProvider());
    }

    // 配置参数（实际使用时从安全配置中心获取）
    private static final String APP_ID = "app_5f8d3e7a9b2c"; // 平台分配的唯一标识
    private static final String SECRET_KEY = "sk_9a4b6c8d2e0f1g3h5j7k"; // 应用密钥
    //private static final String SM4_KEY = "a1b2c3d4e5f6a7b8"; // 16 字节 SM4 密钥
    //private static final String SM4_IV = "a9b8c7d6e5f4a3b2"; // 16 字节初始化向量
    private static final String SM4_KEY = "a1b2c3d4e5f6a7b8a1b2c3d4e5f6a7b8"; //16
字节 SM4 密钥 32 字符=16 字节
    private static final String SM4_IV = "a9b8c7d6e5f4a3b2a9b8c7d6e5f4a3b2"; //16 字
节初始化向量 32 字符=16 字节
    private static final String API_URL =
"http://127.0.0.1:48080/app-api/metricReport";//测试环境 用 http  生产环境用 https
    private static final String POOL_TYPE = "public"; // 资源池类型

   private static final Gson GSON = new Gson();
    private static final DateTimeFormatter DATE_FORMATTER =
DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss");

    public static void main(String[] args) {
        try {
            // 1. 构建业务指标数据
            List<Metric> metrics = buildMetrics();

            // 2. 加密数据
            Map<String, Object> dataMap = new HashMap<>();
            dataMap.put("metrics", metrics);
```

```
            String plainData = GSON.toJson(dataMap);
            String encryptedData = encrypt(plainData, SM4_KEY, SM4_IV);

            // 3. 生成签名
            long timestamp = System.currentTimeMillis() / 1000; // 秒级时间戳
            String signature = generateSignature(APP_ID, timestamp, encryptedData,
SECRET_KEY);

            // 4. 构建请求体
            Map<String, Object> requestBody = new HashMap<>();
            requestBody.put("appId", APP_ID);
            requestBody.put("sign", signature);
            requestBody.put("timestamp", timestamp);
            requestBody.put("data", encryptedData);

            // 5. 发送请求
            sendRequest(requestBody, timestamp);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    // 构建业务指标数据示例
    private static List<Metric> buildMetrics() {
        List<Metric> metrics = new ArrayList<>();
        LocalDateTime yesterday = LocalDateTime.now().minusDays(1);
        String reportTime = DATE_FORMATTER.format(yesterday);

        // 资源池信息
        String zyc =
"[{\"poolId\":\"3b9a7c8d-5e6f-4a2b-9c1d-7e8f0a1b2c3e\",\"poolName\":\"百度资源池一
\"}," +

"{\"poolId\":\"8e7d6c5b-4a3f-5d2e-8c1f-0b9a8c7d6e5e\",\"poolName\":\"百度资源池二
\"}]";


        metrics.add(new Metric("ZYC_JSON_ARRAY", zyc, reportTime));


        // Gpu服务器
        String gpu =
"[{\"gpuServerId\":\"5f4e3d2c-1b0a-6789-abcd-ef1234567890b\",\"gpuServerName\":\"百
```

```
度 GPU 服务器一
\",\"poolId\":\"3b9a7c8d-5e6f-4a2b-9c1d-7e8f0a1b2c3e\",\"cpuNum\":32,\"cpuCore\":51
2,\"memory\":1024}," +

"{\"gpuServerId\":\"a1b2c3d4-e5f6-7890-1234-567890abcdefb\",\"gpuServerName\":\"百
度 GPU 服务器二
\",\"poolId\":\"8e7d6c5b-4a3f-5d2e-8c1f-0b9a8c7d6e5e\",\"cpuNum\":32,\"cpuCore\":51
2,\"memory\":1024}]";



        metrics.add(new Metric("GPUFWQ_JSON_ARRAY", gpu, reportTime));

        // 存储服务器
        String cc =
"[{\"storageServerId\":\"d9c8b7a6-5432-1fed-cba0-9876543210fe\",\"storageServerName
\":\"百度存储服务器一
\",\"poolId\":\"3b9a7c8d-5e6f-4a2b-9c1d-7e8f0a1b2c3e\",\"storageCapacity\":1000},"
+

"{\"storageServerId\":\"2a4c6e8g-1357-2468-abcd-ef0123456789\",\"storageServerName\
":\"百度存储服务器二
\",\"poolId\":\"8e7d6c5b-4a3f-5d2e-8c1f-0b9a8c7d6e5e\",\"storageCapacity\":2000}]";

        metrics.add(new Metric("CCFWQ_JSON_ARRAY", cc, reportTime));

        // 算力情况指标（JSON 数组）

        String slqkJson = "[{\"brandName\":\"华为\",\"modelCode\":\"Atlas 300T
Pro\",\"computingPower\":300,\"cardId\":\"7c3e5a8b-2d4f-910e-6b3a-8c7d2e1f0a9b\",\"
gpuServerId\":\"5f4e3d2c-1b0a-6789-abcd-ef1234567890b\"}," +
                "{\"brandName\":\"华为\",\"modelCode\":\"Atlas 300I
Duo\",\"computingPower\":200,\"cardId\":\"1f2e3d4c-5b6a-7890-f1e2-d3c4b5a67890\",\"
gpuServerId\":\"a1b2c3d4-e5f6-7890-1234-567890abcdefb\"}]";

        metrics.add(new Metric("SLK_JSON_ARRAY", slqkJson, reportTime));

        // 客户信息指标（JSON 数组）

        String khxxJson = "[{\"entName\":\"厦门百度客户一
\",\"orgCode\":\"91330101MA2Y3ABCDC\",\"industryCode\":\"A\"}," +
                "{\"entName\":\"厦门百度客户二
\",\"orgCode\":\"91330101MA2Y3ABCDD\",\"industryCode\":\"B\"}]";
```

```
        metrics.add(new Metric("KHXX_JSON_ARRAY", khxxJson, reportTime));

        // 用户使用量指标（JSON数组）
        String yhsylJson = "[{\"entName\":\"厦门百度客户一
\",\"orgCode\":\"91330101MA2Y3ABCDC\",\"computeUsage\":100,\"statDate\":\"2025-07-0
1\",\"domestic\":40,\"nvidia\":60}," +
                "{\"entName\":\"厦门百度客户二
\",\"orgCode\":\"91330101MA2Y3ABCDD\",\"computeUsage\":100,\"statDate\":\"2025-07-0
1\",\"domestic\":40,\"nvidia\":60}]";



        metrics.add(new Metric("YHSYL_JSON_ARRAY", yhsylJson, reportTime));

        // 交易情况指标
        metrics.add(new Metric("JYQK_YSSL", 512.3, reportTime)); // 已售算力
        metrics.add(new Metric("JYQK_SYSL", 512.2, reportTime)); // 剩余算力
        metrics.add(new Metric("JYQK_ZKHS", 200, reportTime)); // 总客户数
        metrics.add(new Metric("JYQK_BY_XZKHS", 10, reportTime)); // 本月新增客户数



        // 实时状态指标
        metrics.add(new Metric("SSZT_GPU_SYL", 40, reportTime)); // GPU使用率
        metrics.add(new Metric("SSZT_CPU_SYL", 45.2, reportTime)); // CPU使用率
        metrics.add(new Metric("SSZT_MEM_SYL", 70.8, reportTime)); // 内存使用率
        metrics.add(new Metric("SSZT_DISK_SYL", 55.3, reportTime)); // 存储使用率


        String KHZYSYQK = "[{\"entName\":\"厦门客户一
\",\"orgCode\":\"91330101MA2Y3ABCDC\"," +
                "\"details\":[" +

"{\"hour\":\"1\",\"gpuUsageRate\":\"82\",\"cpuUsageRate\":\"70.5\",\"memoryUsageRat
e\":\"65\",\"storageUsageRate\":\"42.3\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"2\",\"gpuUsageRate\":\"78\",\"cpuUsageRate\":\"68.2\",\"memoryUsageRat
e\":\"63\",\"storageUsageRate\":\"41.8\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"3\",\"gpuUsageRate\":\"65\",\"cpuUsageRate\":\"55.7\",\"memoryUsageRat
```

```
e\":\"58\",\"storageUsageRate\":\"40.5\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"4\",\"gpuUsageRate\":\"52\",\"cpuUsageRate\":\"45.3\",\"memoryUsageRat
e\":\"52\",\"storageUsageRate\":\"39.2\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"5\",\"gpuUsageRate\":\"48\",\"cpuUsageRate\":\"42.1\",\"memoryUsageRat
e\":\"50\",\"storageUsageRate\":\"38.7\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"6\",\"gpuUsageRate\":\"55\",\"cpuUsageRate\":\"50.6\",\"memoryUsageRat
e\":\"53\",\"storageUsageRate\":\"39.5\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"7\",\"gpuUsageRate\":\"68\",\"cpuUsageRate\":\"62.4\",\"memoryUsageRat
e\":\"59\",\"storageUsageRate\":\"40.8\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"8\",\"gpuUsageRate\":\"85\",\"cpuUsageRate\":\"75.8\",\"memoryUsageRat
e\":\"68\",\"storageUsageRate\":\"43.2\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"9\",\"gpuUsageRate\":\"92\",\"cpuUsageRate\":\"82.3\",\"memoryUsageRat
e\":\"75\",\"storageUsageRate\":\"45.6\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"10\",\"gpuUsageRate\":\"95\",\"cpuUsageRate\":\"88.7\",\"memoryUsageRa
te\":\"79\",\"storageUsageRate\":\"46.3\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"11\",\"gpuUsageRate\":\"93\",\"cpuUsageRate\":\"86.2\",\"memoryUsageRa
te\":\"77\",\"storageUsageRate\":\"45.9\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"12\",\"gpuUsageRate\":\"88\",\"cpuUsageRate\":\"80.5\",\"memoryUsageRa
te\":\"73\",\"storageUsageRate\":\"44.8\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"13\",\"gpuUsageRate\":\"82\",\"cpuUsageRate\":\"75.3\",\"memoryUsageRa
te\":\"69\",\"storageUsageRate\":\"43.5\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"14\",\"gpuUsageRate\":\"89\",\"cpuUsageRate\":\"83.1\",\"memoryUsageRa
```

```
te\":\"76\",\"storageUsageRate\":\"45.2\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"15\",\"gpuUsageRate\":\"94\",\"cpuUsageRate\":\"87.6\",\"memoryUsageRa
te\":\"78\",\"storageUsageRate\":\"46.1\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"16\",\"gpuUsageRate\":\"92\",\"cpuUsageRate\":\"85.4\",\"memoryUsageRa
te\":\"77\",\"storageUsageRate\":\"45.8\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"17\",\"gpuUsageRate\":\"88\",\"cpuUsageRate\":\"81.2\",\"memoryUsageRa
te\":\"74\",\"storageUsageRate\":\"44.9\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"18\",\"gpuUsageRate\":\"80\",\"cpuUsageRate\":\"73.5\",\"memoryUsageRa
te\":\"68\",\"storageUsageRate\":\"43.1\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"19\",\"gpuUsageRate\":\"75\",\"cpuUsageRate\":\"68.9\",\"memoryUsageRa
te\":\"65\",\"storageUsageRate\":\"42.4\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"20\",\"gpuUsageRate\":\"70\",\"cpuUsageRate\":\"63.2\",\"memoryUsageRa
te\":\"62\",\"storageUsageRate\":\"41.7\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"21\",\"gpuUsageRate\":\"65\",\"cpuUsageRate\":\"58.7\",\"memoryUsageRa
te\":\"59\",\"storageUsageRate\":\"40.9\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"22\",\"gpuUsageRate\":\"58\",\"cpuUsageRate\":\"52.3\",\"memoryUsageRa
te\":\"55\",\"storageUsageRate\":\"39.8\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"23\",\"gpuUsageRate\":\"50\",\"cpuUsageRate\":\"46.5\",\"memoryUsageRa
te\":\"51\",\"storageUsageRate\":\"38.9\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"24\",\"gpuUsageRate\":\"45\",\"cpuUsageRate\":\"41.2\",\"memoryUsageRa
te\":\"48\",\"storageUsageRate\":\"38.2\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}" +
                "]},{\"entName\":\"厦门客户二
\",\"orgCode\":\"91330101MA2Y3ABCDD\"," +
```

```
                "\"details\":[" +

"{\"hour\":\"1\",\"gpuUsageRate\":\"78\",\"cpuUsageRate\":\"65.3\",\"memoryUsageRat
e\":\"62\",\"storageUsageRate\":\"40.5\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"2\",\"gpuUsageRate\":\"72\",\"cpuUsageRate\":\"60.1\",\"memoryUsageRat
e\":\"59\",\"storageUsageRate\":\"39.8\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"3\",\"gpuUsageRate\":\"60\",\"cpuUsageRate\":\"52.4\",\"memoryUsageRat
e\":\"55\",\"storageUsageRate\":\"38.7\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"4\",\"gpuUsageRate\":\"50\",\"cpuUsageRate\":\"43.2\",\"memoryUsageRat
e\":\"50\",\"storageUsageRate\":\"37.6\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"5\",\"gpuUsageRate\":\"45\",\"cpuUsageRate\":\"39.8\",\"memoryUsageRat
e\":\"48\",\"storageUsageRate\":\"37.2\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"6\",\"gpuUsageRate\":\"52\",\"cpuUsageRate\":\"47.5\",\"memoryUsageRat
e\":\"51\",\"storageUsageRate\":\"38.1\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"7\",\"gpuUsageRate\":\"65\",\"cpuUsageRate\":\"58.3\",\"memoryUsageRat
e\":\"56\",\"storageUsageRate\":\"39.4\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"8\",\"gpuUsageRate\":\"80\",\"cpuUsageRate\":\"72.6\",\"memoryUsageRat
e\":\"65\",\"storageUsageRate\":\"41.8\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"9\",\"gpuUsageRate\":\"88\",\"cpuUsageRate\":\"80.2\",\"memoryUsageRat
e\":\"72\",\"storageUsageRate\":\"44.3\",\"gpuVramUsageRate\":\"65\",\"networkUsage
Rate\":\"42.3\"}," +

"{\"hour\":\"10\",\"gpuUsageRate\":\"92\",\"cpuUsageRate\":\"85.7\",\"memoryUsageRa
te\":\"76\",\"storageUsageRate\":\"45.5\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"11\",\"gpuUsageRate\":\"90\",\"cpuUsageRate\":\"83.4\",\"memoryUsageRa
```

```
te\":\"74\",\"storageUsageRate\":\"45.1\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"12\",\"gpuUsageRate\":\"85\",\"cpuUsageRate\":\"78.6\",\"memoryUsageRa
te\":\"70\",\"storageUsageRate\":\"44.2\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"13\",\"gpuUsageRate\":\"79\",\"cpuUsageRate\":\"73.2\",\"memoryUsageRa
te\":\"67\",\"storageUsageRate\":\"43.0\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"14\",\"gpuUsageRate\":\"86\",\"cpuUsageRate\":\"81.5\",\"memoryUsageRa
te\":\"73\",\"storageUsageRate\":\"44.8\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"15\",\"gpuUsageRate\":\"90\",\"cpuUsageRate\":\"84.7\",\"memoryUsageRa
te\":\"75\",\"storageUsageRate\":\"45.3\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"16\",\"gpuUsageRate\":\"88\",\"cpuUsageRate\":\"82.3\",\"memoryUsageRa
te\":\"73\",\"storageUsageRate\":\"44.9\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"17\",\"gpuUsageRate\":\"83\",\"cpuUsageRate\":\"77.8\",\"memoryUsageRa
te\":\"70\",\"storageUsageRate\":\"44.0\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"18\",\"gpuUsageRate\":\"76\",\"cpuUsageRate\":\"70.5\",\"memoryUsageRa
te\":\"65\",\"storageUsageRate\":\"42.6\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"19\",\"gpuUsageRate\":\"70\",\"cpuUsageRate\":\"65.9\",\"memoryUsageRa
te\":\"62\",\"storageUsageRate\":\"41.5\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"20\",\"gpuUsageRate\":\"65\",\"cpuUsageRate\":\"60.3\",\"memoryUsageRa
te\":\"59\",\"storageUsageRate\":\"40.7\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"21\",\"gpuUsageRate\":\"58\",\"cpuUsageRate\":\"54.6\",\"memoryUsageRa
te\":\"55\",\"storageUsageRate\":\"39.6\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"22\",\"gpuUsageRate\":\"52\",\"cpuUsageRate\":\"48.9\",\"memoryUsageRa
```

```
te\":\"52\",\"storageUsageRate\":\"38.8\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"23\",\"gpuUsageRate\":\"47\",\"cpuUsageRate\":\"43.5\",\"memoryUsageRa
te\":\"49\",\"storageUsageRate\":\"38.0\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}," +

"{\"hour\":\"24\",\"gpuUsageRate\":\"42\",\"cpuUsageRate\":\"39.2\",\"memoryUsageRa
te\":\"46\",\"storageUsageRate\":\"37.5\",\"gpuVramUsageRate\":\"65\",\"networkUsag
eRate\":\"42.3\"}" +
                "]}]";


        metrics.add(new Metric("KHZYSYQK_JSON_ARRAY", KHZYSYQK, reportTime));

        return metrics;
    }

    // SM4-CBC 加密
    private static String encrypt(String plainText, String key, String iv) throws
Exception {
        byte[] keyBytes = Hex.decode(key);
        byte[] ivBytes = Hex.decode(iv);
        byte[] plainBytes = plainText.getBytes(StandardCharsets.UTF_8);

        PaddedBufferedBlockCipher cipher = new PaddedBufferedBlockCipher(new
CBCBlockCipher(new SM4Engine()));
        cipher.init(true, new ParametersWithIV(new KeyParameter(keyBytes), ivBytes));

        byte[] output = new byte[cipher.getOutputSize(plainBytes.length)];
        int length = cipher.processBytes(plainBytes, 0, plainBytes.length, output, 0);
        length += cipher.doFinal(output, length);

        return Base64.getEncoder().encodeToString(Arrays.copyOf(output, length));
    }

    // SM3 签名生成
    private static String generateSignature(String appId, long timestamp, String
encryptedData, String secretKey) {
        String rawData = appId + timestamp + encryptedData + secretKey;
        byte[] dataBytes = rawData.getBytes(StandardCharsets.UTF_8);

        SM3Digest digest = new SM3Digest();
        digest.update(dataBytes, 0, dataBytes.length);
```

```
        byte[] hash = new byte[digest.getDigestSize()];
        digest.doFinal(hash, 0);

        return Hex.toHexString(hash);
    }

    // 发送 HTTP 请求
    private static void sendRequest(Map<String, Object> requestBody, long timestamp)
throws Exception {
        String requestJson = GSON.toJson(requestBody);
        String requestId = UUID.randomUUID().toString();
        String nonce = UUID.randomUUID().toString();
        long expires = timestamp + 300; // 5 分钟过期

        URL url = new URL(API_URL);
        HttpURLConnection connection = (HttpURLConnection) url.openConnection();
        connection.setRequestMethod("POST");
        connection.setRequestProperty("Content-Type", "application/json");
        connection.setRequestProperty("Authorization", "Bearer " + APP_ID + ":" +
requestBody.get("sign"));
        connection.setRequestProperty("X-Request-ID", requestId);
        connection.setRequestProperty("X-Pool-Type", POOL_TYPE);
        connection.setRequestProperty("X-Timestamp", String.valueOf(timestamp));
        connection.setRequestProperty("X-Expires", String.valueOf(expires));
        connection.setRequestProperty("X-Sign-Algorithm", "SM3");
        connection.setRequestProperty("X-Encrypt-Algorithm", "SM4-CBC");
        connection.setRequestProperty("X-Sign-Nonce", nonce);
        connection.setRequestProperty("X-Metric-Type", "business");
        connection.setDoOutput(true);

        try (OutputStream os = connection.getOutputStream()) {
            byte[] input = requestJson.getBytes(StandardCharsets.UTF_8);
            os.write(input, 0, input.length);
        }

        int responseCode = connection.getResponseCode();
        String responseBody;
        try (BufferedReader br = new BufferedReader(
                new InputStreamReader(connection.getInputStream(),
StandardCharsets.UTF_8))) {
            StringBuilder response = new StringBuilder();
            String responseLine;
            while ((responseLine = br.readLine()) != null) {
                response.append(responseLine.trim());
```

```
            }
            responseBody = response.toString();
        }

        System.out.println("响应状态码: " + responseCode);
        System.out.println("响应内容: " + responseBody);
    }

    // 指标数据模型
    static class Metric {
        private String metricCode;
        private Object value;
        private String reportTime;

        public Metric(String metricCode, Object value, String reportTime) {
            this.metricCode = metricCode;
            this.value = value;
            this.reportTime = reportTime;
        }

        // Getter 和 Setter (Gson 序列化需要)
        public String getMetricCode() { return metricCode; }
        public Object getValue() { return value; }
        public String getReportTime() { return reportTime; }
    }

}
```

本示例中的具体指标名称要以 1.2 章节的为准。

1.9. 附录

1.9.1. 行业分类编码表（GB/T 4754-2017）

序号	编码	名称
1	A	农、林、牧、渔业
2	B	采矿业
3	C	制造业
4	D	电力、热力、燃气及水生产和供应业
5	E	建筑业
6	F	批发和零售业
7	G	交通运输、仓储和邮政业

8	H	住宿和餐饮业
9	I	信息传输、软件和信息技术服务业
10	J	金融业
11	K	房地产业
12	L	租赁和商务服务业
13	M	科学研究和技术服务业
14	N	水利、环境和公共设施管理业
15	O	居民服务、修理和其他服务业
16	P	教育
17	Q	卫生和社会工作
18	R	文化、体育和娱乐业
19	S	公共管理、社会保障和社会组织
20	T	国际组织

1.9.2. 错误码列表

错误码	含义	描述
200	成功	数据同步成功
400	错误请求	请求参数缺失或格式错误
401	未授权	身份验证失败，可能是 appId 或签名错误
403	禁止访问	没有权限访问该接口
404	未找到	请求的资源池不存在
408	请求超时	请求处理时间超过限制
500	内部服务器错误	服务器处理请求时发生错误
501	不支持的方法	请求方法不被支持
503	服务不可用	服务器暂时无法处理请求

2. 合同信息报送

2.1. 接口概述

该接口用于向指定服务端提交合同报告数据及相关文件，支持数据加密和签名验证，确保传输安全性。

2.2. 接口基础信息

2.2.1. 接口协议

采用 HTTPS 协议，确保数据传输安全。

2.2.2. 接口地址

https://[算力调度服务平台 IP:端口]/app-api/contractReport

2.2.3. 请求方法

POST

2.2.4. 内容类型

multipart/form-data

2.3. 请求头参数

序号	参数名称	是否必选	类型	描述
1	Content-Type	是	String	固定值：multipart/form-data; boundary=xxx（xxx 为随机生成的分隔符）
2	Authorization	是	String	身份认证信息，格式为：Bearer {appId}:{signature}
3	X-Request-ID	是	String	唯一请求标识，建议使用 UUID，用于防重放攻击和请求追踪
4	X-Pool-Type	是	String	资源池类型，固定值：public
5	X-Timestamp	是	Long	请求时间戳（精确到秒），用于防重放攻击
6	X-Expires	是	Long	请求过期时间（精确到秒），建议设置为 timestamp + 300（即 5 分钟）
7	X-Sign-Algorithm	是	String	签名算法，固定值：SM3
8	X-Encrypt-Algorithm	是	String	加密算法，固定值：SM4-CBC

9	X-Sign-Nonce	是	String	随机字符串，用于签名生成，每次请求需不同
10	X-Metric-Type	是	String	指标类型，固定值：contract

2.4. 请求体结构

2.4.1. 表单文字字段

序号	参数名称	是否必选	类型	描述
1	appId	是	String	平台为每个请求方分配的唯一标识，用于身份验证
2	sign	是	String	签名信息，使用 SM3 算法生成，用于验证请求的合法性和数据完整性
3	timestamp	是	Long	请求时间戳（精确到秒），需与请求头中的 X-Timestamp 一致
4	data	是	String	加密后的数据，使用 SM4-CBC 算法加密，Base64 编码

2.4.2. 文件字段

序号	参数名称	是否必选	类型	描述
1	files	是	File	需上传的文件（支持多文件），文件名格式为：{contractCode}_{原文件名}（用于关联合同编号）

2.4.3. 加密前数据结构

```
{
  "contracts": [
    {
      "contractCode": "合同编号",
      "entName": "企业名称",
      "orgCode": "机构代码",
      "signDate": "签约日期（yyyy-MM-dd）",
      "contractAmount": "合同金额",
      "contractStartTime": "合同开始时间（yyyy-MM-dd）",
      "contractEndTime": "合同结束时间（yyyy-MM-dd）",
```

```
      "computeScenarios": "计算场景",
      "description": "描述信息"
    }
  ]
}
```

2.5. 响应体结构

```
{
  "code": 200,
  "msg": "合同信息上报成功",
  "requestId": "e6a7e2d3-9f8a-4f5b-9b6c-8f7d5e8f4a3b"
}
```

2.6. 安全机制

2.6.1. 数据加密

✓ 采用 SM4-CBC 算法加密业务指标数据

✓ 密钥长度 128 位（16 字节）

✓ 初始化向量 (IV) 长度 16 字节

✓ 加密模式：CBC

✓ 填充方式：PKCS7Padding

2.6.2. 数据签名

✓ 采用 SM3 算法生成签名

✓ 签名原始数据：appId + timestamp + 加密后数据 + 应用密钥

✓ 签名结果转换为 16 进制字符串

2.6.3. 密钥管理

✓ 加密密钥和应用密钥通过安全通道预先分发

✓ 定期更换密钥

2.7. 附录

2.7.1. 应用场景编码表

序号	编码	名称
1	0	智慧交通
2	1	智能制造
3	2	智慧医疗
4	3	智慧政务
5	4	智慧金融
6	5	智慧教育
7	6	智慧文旅
8	7	智慧城市
9	8	智慧商务
10	9	智慧供应链
11	10	智慧农业
12	11	智慧海洋
13	12	智慧园区