

- EMS restful interfaces
  - 1. 状态管理
  - 2 故障管理
    - 2.1 告警上报
    - 2.2 获取网元告警
  - 3. 配置管理
    - 3.1 参数配置表
    - 3.2 获取参数
    - 3.3 配置参数
    - 3.4 创建/修改网元
  - 4 性能管理
    - 4.1 任务管理
    - 4.2 订阅任务数据上报
    - 4.3 黄金指标上报
    - 4.4 测量数据主动上报
    - 4.5 测量数据获取/补采
  - 5 跟踪管理
    - 5.1 订阅管理
    - 5.2 消息上报
  - 6. 操作维护
    - 6.1 MML命令
      - 6.1.1 MML命令格式
      - 6.1.2 MML配置表
    - 6.2 MML接口
  - 7 北向接口(NBI)
    - 7.1 查询资源数据接口
  - 8 文件接口
    - 8.1 软件管理
      - 8.1.1 软件包管理接口
      - 8.1.2 网元软件包管理接口
    - 8.2 License管理
      - 8.2.1 License文件管理接口
      - 8.2.2 网元License管理接口
  - 9 4A接口
    - 9.1 从账号(user)管理接口
      - 9.1.1 查询全部从账号(user)接口
      - 9.1.2 单个从账号(user)接口

- 9.2 角色(role)管理接口
  - 9.2.1 查询全部角色(role)接口
  - 9.2.2 单个角色(role)接口
- 9.3 登录认证接口
  - 9.2.1 登录认证
- 10 核心网用户信息接口
  - 10.1 UDM签约用户
    - 10.1.1 查询/增加/修改/删除
  - 10.2 UDM鉴权用户
    - 10.2.1 查询/增加/修改/删除
  - 10.3 UE信息
    - 10.3.1 查询SMF在线用户数
    - 10.3.2 查询SMF在线用户
  - 10.4 IMS在线用户信息
    - 10.4.1 查询IMS在线用户信息
  - 10.5 基站信息
    - 10.5.1 查询某个AMF下的基站信息

# EMS restful interfaces

---

## 1. 状态管理

---

- URI:

```
/api/rest/systemManagement/v1/elementType/{elementTypeValue}/objectType/systemState
```

说明:

```
** elementTypeValue=smf/amf/..查询的网元网元类型
```

- Method:

GET

- Return:

## NF -> OMC BE的json结构

```
type SysState struct {
    HostName      string `json:"hostName"` // linux命令: hostname
    OsInfo        string `json:"osInfo"`   // linux命令: uname -a
    DbInfo        string `json:"dbInfo"`   // 网元如果有db, 显示数据库名和版本信息, OMC: mysql --version
    Version       string `json:"version"` // 软件版本信息: 16.1.1
    IpAddr        []string `json:"ipAddr"` // 网管的ipv4和ipv6
    Port          uint16 `json:"port"`    // 用于网管的port
    Capability     uint32 `json:"capability"`
    SerialNum     string `json:"serialNum"`
    ExpiryDate    string `json:"expiryDate"`
    HardwareInfo  struct {
        CPUs      int `json:"cpus"` // 主机(裸机/虚拟机)的cpu个数
        Memory    int `json:"memory"` // 主机(裸机/虚拟机): 配置的内存
    } `json:"hardwareInfo"`
    CpuUsage      struct {
        NfCpuUsage  uint16 `json:"nfCpuUsage"`
        SysCpuUsage uint16 `json:"sysCpuUsage"`
    } `json:"cpuUsage"`
    MemUsage      struct {
        TotalMem    uint32 `json:"totalMem"`
        NfUsedMem   uint32 `json:"nfUsedMem"`
        SysMemUsage uint16 `json:"sysMemUsage"`
    } `json:"memUsage"`
    DiskSpace     struct {
        PartitionNum uint8 `json:"partitionNum"`
        PartitionInfo []struct {
            Total uint32 `json:"total"` // MB
            Used  uint32 `json:"used"` // MB
        } `json:"partitionInfo"`
    } `json:"diskSpace"`
    //Timestamp string `json:"timestamp"`
}
```

- Example: OMC BE -> OMC FE

```
"data": [
  {
    "AMF_0": {
      "error": {
        "errorCode": "1",
        "errorInfo": "Internal server error, NF connect refused"
      },
      "ipAddress": "192.168.2.188"
    }
  },
  {
    "SMF_0": {
      "ipAddress": "192.168.1.232",
    }
  }
]
```

```
"systemState": {
  "capability": 10000,
  "cpuUsage": {
    "nfCpuUsage": 2,
    "sysCpuUsage": 52
  },
  "diskSpace": {
    "partitionInfo": [
      {
        "total": 1920,
        "used": 0
      },
      {
        "total": 393,
        "used": 13
      },
      {
        "total": 48700,
        "used": 32431
      },
      {
        "total": 1965,
        "used": 0
      },
      {
        "total": 5,
        "used": 0
      },
      {
        "total": 1965,
        "used": 0
      },
      {
        "total": 55,
        "used": 55
      },
      {
        "total": 63,
        "used": 63
      },
      {
        "total": 91,
        "used": 91
      },
      {
        "total": 49,
        "used": 49
      },
      {
        "total": 55,
        "used": 55
      },
      {
        "total": 73,
        "used": 73
      },
      {

```

```
        "total": 91,
        "used": 91
    },
    {
        "total": 1475,
        "used": 206
    },
    {
        "total": 49,
        "used": 49
    },
    {
        "total": 393,
        "used": 13
    },
    {
        "total": 393,
        "used": 0
    },
    {
        "total": 73,
        "used": 73
    },
    {
        "total": 63,
        "used": 63
    }
    ],
    "partitionNum": 19
},
"expiryDate": "2025-02-28",
"memUsage": {
    "nfUsedMem": 163992,
    "sysMemUsage": 1345,
    "totalMem": 4025608
},
"serialNum": "13740126",
"version": "1.5.3.2"
}
}
},
{
    "SMF_1": {
        "ipAddress": "192.168.1.173",
        "systemState": {
            "capability": 10000,
            "cpuUsage": {
                "nfCpuUsage": 0,
                "sysCpuUsage": 69
            },
            "diskSpace": {
                "partitionInfo": [
                    {
                        "total": 3966,
                        "used": 0
                    },
                ],
            }
        }
    }
}
```

```
        "total": 797,  
        "used": 0  
    },  
    {  
        "total": 200559,  
        "used": 5968  
    },  
    {  
        "total": 3987,  
        "used": 0  
    },  
    {  
        "total": 5,  
        "used": 0  
    },  
    {  
        "total": 3987,  
        "used": 0  
    },  
    {  
        "total": 797,  
        "used": 0  
    }  
    ],  
    "partitionNum": 7  
},  
"expiryDate": "2024-12-31",  
"memUsage": {  
    "nfUsedMem": 212136,  
    "sysMemUsage": 720,  
    "totalMem": 8167360  
},  
"serialNum": "13740272",  
"version": "1.5.3.3"  
}  
}  
}  
]  
}
```

## 2 故障管理

### 2.1 告警上报

- URI

```
/api/rest/faultManagement/v1/elementType/{elementTypeValue}/objectType/alarms
```

- Method

POST

- Relations

NF->OMC

- Body

```
type Alarm struct {
    AlarmSeq      int    `json:"alarmSeq"`
    AlarmId       string `json:"alarmId"`
    NeId          string `json:"neId"`
    AlarmCode     int    `json:"alarmCode"`
    AlarmTitle    string `json:"alarmTitle"`
    EventTime     string `json:"eventTime"`
    AlarmType     string `json:"alarmType"`
    OrigSeverity  string `json:"origSeverity"`
    PVFlag        string `json:"pvFlag"`
    NeName        string `json:"neName"`
    NeType        string `json:"neType"`
    ObjectName    string `json:"objectName"`
    LocationInfo  string `json:"locationInfo"`
    Province      string `json:"province"`
    AlarmStatus   int    `json:"alarmStatus"`
    SpecificProblem string `json:"specificProblem"`
    SpecificProblemID string `json:"specificProblemID"`
    AddInfo       string `json:"addInfo"`
}
```

## 2.2 获取网元告警

- URI

```
/api/rest/faultManagement/v1/elementType/{elementTypeValue}/objectType/alarms
```

- Method

GET

- Relations

OMC->NF

- Body

n/a

- Return

```
type Alarms struct {
    Alarms []struct {
        AlarmSeq      int    `json:"alarmSeq"`
        AlarmId        string `json:"alarmId"`
        NeId           string `json:"neId"`
        AlarmCode      int    `json:"alarmCode"`
        AlarmTitle     string `json:"alarmTitle"`
        EventTime      string `json:"eventTime"`
        AlarmType      string `json:"alarmType"`
        OrigSeverity   string `json:"origSeverity"`
        PVFlag         string `json:"pvFlag"`
        NeName         string `json:"neName"`
        NeType         string `json:"neType"`
        ObjectName     string `json:"objectName"`
        LocationInfo   string `json:"locationInfo"`
        Province      string `json:"province"`
        AlarmStatus    int    `json:"alarmStatus"`
        SpecificProblem string `json:"specificProblem"`
        SpecificProblemID string `json:"specificProblemID"`
        AddInfo       string `json:"addInfo"`
    } `json:"Alarms"`
}
```

## 3. 配置管理

### 3.1 参数配置表

- 类型定义（type）

- string

filter指定字符串长度，如：`"filter": "6~100"`，字符串的长度范围，如果单个数字表示最大长度

- ipv4

filter忽略

- ipv6



## filter忽略

- int

filter指定整型数的范围，如："filter": "100~999"

- enum

"filter": '{"0": "http", "1": "https"}'

- bool

"filter": '{"0": "false", "1": "true"}'

- regex

filter为正则表达式

- 可选 (optional)

- true，默认值，表示该字段为可选填，没有写该属性项时optional为true（兼容以前配置文件）
- false，表示必填项

- Example

```
UDM:
system:
  display: "System"
  list:
    - name: "serviceIP"
      type: "ipv4"
      value: "172.16.5.140"
      optional: "false"
      access: "read-write"
      filter: ''
      display: "Service IP"
      comment: ""
    - name: "servicePort"
      type: "int"
      value: "8080"
      access: "read-write"
      filter: "0~65535"
      display: "Service Port"
      comment: "0~65535"
  subsSmfSelection:
    display: "Subs Smf Selection"
    array:
      - name: "index"
```

```

    type: "int"
    value: "0"
    access: "read-write"
    filter: '0~15'
    display: "Index"
    comment: "0~15"
- name: "name"
  type: "string"
  value: 'def_ambr'
  access: "read-write"
  filter: '^.{1,32}$'
  display: "Name"
  comment: "0~32"
- name: "snssai"
  type: "string"
  value: '1-000001'
  access: "read-write"
  filter: '^\\d{1,3}[A-Fa-f0-9]{6}$'
  display: "Snssai"
  comment: ""
- name: "dnnList"
  type: "int"
  value: '0'
  access: "read-write"
  filter: '0~3'
  display: "Dnn List"
  comment: ""
  array:
    - name: "index"
      type: "int"
      value: "0"
      access: "read-write"
      filter: '0~15'
      display: "index"
      comment: "0~15"
    - name: "dnn"
      type: "string"
      value: 'cmnet'
      access: "read-write"
      filter: '^.{1,32}$'
      display: "Dnn"
      comment: "0~32"
    - name: "defaultDnnInd"
      type: "bool"
      value: 'true'
      access: "read-write"
      filter: 'false;true;'
      display: "default Dnn Indicator"
      comment: ""

```

完整文件具体请参考 `udm_param_config.yaml`

## 3.2 获取参数

- URI

```
/api/rest/systemManagement/v1/elementType/{elementTypeValue}/objectType/config/{paraName}?loc={index0}/{paraName1}/{index1}/...
```

### 说明

```
elementTypeValue: udm, smf, amf... 网元类型  
udm paraName: system, subsUEAmbr, subsSmfSelection ...  
非Array的参数, 忽略loc
```

- Params

- loc 多层表的定位信息

- Method

GET

- Return

```
/api/rest/systemManagement/v1/elementType/udm/objectType/config/system
```

```
{  
  "data": [  
    {  
      "serviceIP": "172.16.5.140",  
      "servicePort": "8080",  
      "...": "..."  
    }  
  ]  
}
```

```
/api/rest/systemManagement/v1/elementType/udm/objectType/config/subsSmfSelection?loc=1/dnnList
```

```
{  
  "data": [  
    {
```

```
        "index": "0",
        "dnn": "cmnet",
        "...": "..."
    },
    {
        "index": "1",
        "dnn": "ims",
        "...": "..."
    }
]
}
```

## 3.3 配置参数

- URI

```
/api/rest/systemManagement/v1/elementType/{elementTypeValue}/objectType/config/{paraName}?loc={index0}/{paraName1}/{index1}
```

- Params

- loc 多层表的定位信息

- Method

POST/PUT/DELETE

说明:

单层表不支持POST/DELETE操作

- Body

```
{
    "serviceIP": "172.16.5.140",
    "servicePort": "8080",
    "...": "..."
}
```

说明:

DELETE操作不带Body

## 3.4 创建/修改网元

- URI

```
/api/rest/systemManagement/v1/elementType/{elementTypeValue}/objectType/config  
/omcNeConfig
```

- Params

N/A

- Method

PUT

- Body

```
type OmcNeConfig struct {  
    NeId      string `json:"neId" xorm:"ne_id"`           // 网元标识(内部),  
    RmUID     string `json:"rmUID" xorm:"rm_uid"`       // neUID/rmUID 网元  
    唯一标识  
    NeName    string `json:"neName" xorm:"ne_name"`     // 网元名称/友好名称  
    (北向资源/性能等使用)  
    PvFlag    string `json:"pvFlag" xorm:"pv_flag"`     // 网元虚实性标识  
    VNF/PNF: 虚拟/物理  
    Province  string `json:"province" xorm:"province"` // 网元所在省份  
    VendorName string `json:"vendorName" xorm:"vendor_name"` // 厂商名称  
    // ManagedBy string `json:"managedBy" xorm:"managed_by"` // 管理  
    ManagedElement的ManagementNode对象类的DN值  
    Dn string `json:"dn" xorm:"dn"` // 资源里边的ManagedBy, 性能的Dn, 网络唯一标识  
}
```

## 4 性能管理

### 4.1 任务管理

- URI

```
/api/rest/performanceManagement/v1/elementType/{elementTypeValue}/objectType/measurementTask?id={taskId1}&id={taskId2}
```

- Method

POST/PUT/DELETE/PATCH

- Params

taskId=1&taskId=2

- POST: 增加测量任务，激活任务，不带id参数，id在body
- PUT: 修改测量任务，激活任务，不带id参数，id在body
- DELETE: 删除测量任务，不需要带body，带id参数，可带多个
- PATCH: 暂停测量任务，不需要带body，带id参数，可带多个

测量任务创建/修改后暂时存储在OMC数据库，激活任务时再下发到网元

- Relations

OMC -> NF/NF -> OMC

- Body

下发测量任务的报文结构

```
type MeasureTask struct {
    Tasks []struct {
        Id int `json:"Id"`

        StartTime string `json:"StartTime"`
        EndTime    string `json:"EndTime"`

        Schedule struct {
            Type    string `json:"Type"` // 计划类型: Weekly/Monthly, 如果type
            // 为"", 则任务以StartTime和EndTime为条件进行统计, 否则以Schedule方式进行
            Days    []int  `json:"Days"` // Weekly: [0,1,...5,6], 0~6表示星期日
            // ~星期六, Monthly: [1,2,3,...,30,31]一个月的几天
            Periods []struct {
                Start string `json:"Start"` // 零点或者零点加测量粒度的整数倍
                // 00:15:00
                End    string `json:"End"` // 零点加测量粒度的整数倍 08:45:00,
                // 16:15:00
            } `json:"Periods"`
        } `json:"Schedule"`

        GranulOption string `json:"GranulOption"` // 测量粒度选项:
    }
}
```

15M/30M/60M/24H

```
    KPISet []struct {
        Code string `json:"Code"` // 统计编码 如: SMFHA01
        KPIs []string `json:"KPIs"` // 指标项集合
    } `json:"KPISet"`
} `json:"Tasks"`

    NotifyUrl string `json:"NotifyUrl"` /* 数据上报URL
"http://x.x.x.x:xxxx/api/rest/performanceManagement/v1/elementType/smf/objectType/measureReport" */
}
```

- Return

## 4.2 订阅任务数据上报

- URI

```
/api/rest/performanceManagement/v1/elementType/{elementTypeValue}/objectType/measureReport
```

- Method

POST

- Relations

NF -> OMC

- Body

网元返回测量数据的报文结构

```
type MeasureReport struct {
    Id          int    `json:"Id"`
    Timestamp   string `json:"TimeStamp"`
    NeName      string `json:"NeName"`
    RmUID       string `json:"rmUID"`
    NeType      string `json:"NeType"`

    Report struct {
        Period struct {
            StartTime string `json:"StartTime"`
            EndTime   string `json:"EndTime"`
        }
    }
}
```

```

    } `json:"Period"`

    Datas []struct {
        Code string `json:"Code"` // 统计编码 如: SMFHA01
        KPIs []struct {
            KPIID string `json:"KPIID"` // 指标项, 如:
SMF.AttCreatePduSession._Dnn
            KPIValues []struct {
                Name string `json:"Name"` // 单个的写"Total", 或者指标项有
多个测量项, 如Dnn的名称写对应的Dnn"cmnet"/"ims"
                Value int64 `json:"Value"`
            } `json:"KPIValues"`
        } `json:"KPIs"`
    } `json:"Datas"`
} `json:"Report"`
}

```

## 4.3 黄金指标上报

- URI

```

/api/rest/performanceManagement/v1/elementType/{elementTypeValue}/objectType/k
piReport/{index}

```

index取值范围: 0-1439

- Method

POST

- Body

```

type KpiReport struct {
    Timestamp string `json:"TimeStamp"`
    Task struct {
        Period struct {
            StartTime string `json:"StartTime"`
            EndTime string `json:"EndTime"`
        } `json:"Period"`
        NE struct {
            NEName string `json:"NEName"`
            RmUID string `json:"rmUID"`
            NeType string `json:"NeType"`
            KPIs []struct {
                KPIID string `json:"KPIID"`
                Value int `json:"Value"`
                Err string `json:"Err"`
            }
        }
    }
}

```



```
    } `json:"KPIs"`
  } `json:"NE"`
} `json:"Task"`
}
```

## 4.4 测量数据主动上报

- URI

```
/api/rest/performanceManagement/v1/elementType/{elementTypeValue}/objectType/measurement/{index}
```

- Method

POST

- Relations

NF -> OMC

- Params

NA

- Body

网元主动上报测量数据的报文结构

```
type Measurement struct {
    Index      int    `json:"Index"` // 1天当中测量时间粒度(如15分钟)的切片索引: 0~95
    Timestamp  string `json:"TimeStamp"`
    NeName     string `json:"NeName"` // UserLabel
    RmUID      string `json:"rmUID"`
    NeType     string `json:"NeType"` // 网元类型
    PmVersion  string `json:"PmVersion"` // 性能数据版本号
    Dn         string `json:"Dn"` // (???)网元标识, 如:RJN-CMZJ-TZ,SubNetwork=5GC88,ManagedElement=SMF53456,SmfFunction=53456
    Period     string `json:"Period"` // 测量时间粒度选项: 5/15/30/60
    TimeZone   string `json:"TimeZone"` // 时区, 如: "UTC+8"
    StartTime  string `json:"StartTime"`

    Datas []struct {
        ObjectType string `json:"ObjectType"` // 网络资源类别名称, Pm指标项列表中为空间粒度 如: SmfFunction
    }
}
```

```

        KPIs []struct {
            KPIID      string `json:"KPIID"` // 指标项, 如:
SMF.AttCreatePduSession._Dnn
            KPIValues []struct {
                Name string `json:"Name"` // 单个的写"Total", 或者指标项有多个
测量项, 如Dnn的名称写对应的Dnn"cmnet"/"ims"
                Value int64  `json:"Value"`
            } `json:"KPIValues"`
        } `json:"KPIs"`
    } `json:"Datas"`
}

```

- Return

Code: 204, no content

## 4.5 测量数据获取/补采

- URI

```

/api/rest/performanceManagement/v1/elementType/{elementTypeValue}/objectType/measurement/{index}

```

- Method

GET

- Relations

OMC -> NF

- Params

NA

- Body

NA

- Return

网元返回测量数据的报文结构

```

type Measurement struct {
    Index      int    `json:"Index"`           // 1天中测量时间粒度(如15分钟)的切片索引: 0~95
    Timestamp  string `json:"TimeStamp"`
    NeName     string `json:"NeName"` // UserLabel
    RmUID      string `json:"rmUID"`
    NeType     string `json:"NeType"` // 网元类型
    PmVersion  string `json:"PmVersion"` // 性能数据版本号
    Dn        string `json:"Dn"` // (???)网元标识, 如:RJN-CMZJ-TZ,SubNetwork=5GC88,ManagedElement=SMF53456,SmfFunction=53456
    Period     string `json:"Period"` // 测量时间粒度选项: 5/15/30/60
    TimeZone  string `json:"TimeZone"`
    StartTime  string `json:"StartTime"`

    Datas []struct {
        ObjectType string `json:"ObjectType"` // 网络资源类别名称, Pm指标项列表中为空间粒度 如: SmfFunction
        KPIs []struct {
            KPIID      string `json:"KPIID"` // 指标项, 如: SMF.AttCreatePduSession._Dnn
            KPIValues []struct {
                Name      string `json:"Name"` // 单个的写"Total", 或者指标项有多个测量项, 如Dnn的名称写对应的Dnn"cmnet"/"ims"
                Value     int64  `json:"Value"`
            } `json:"KPIValues"`
        } `json:"KPIs"`
    } `json:"Datas"`
}

```

## 5 跟踪管理

### 5.1 订阅管理

- 创建订阅

- URI

```
/api/rest/traceManagement/v1/subscriptions
```

- Method

POST

- Relations

OMC front-end->OMC back-end, OMC -> NF

- Body

OMC front-end->OMC back-end: id 不帶 OMC->NF: id必选

```
type TraceTask struct {
    Id          int      `json:"id"`
    TraceType   string   `json:"traceType"`
    StartTime   string   `json:"startTime"`
    EndTime     string   `json:"endTime"`
    Imsi        string   `json:"imsi"`
    Msisdn      string   `json:"msisdn"`
    SrcIp       string   `json:"srcIp"`
    DstIp       string   `json:"dstIp"`
    SignalPort  int16    `json:"signalPort"`
    NeType      string   `json:"neType"`
    NeId        string   `json:"neId"`
    UeIp        string   `json:"ueIp"`
    Interfaces  []string `json:"interfaces"`
    NotifyUrl   string   `json:"notifyUrl"`
}
```

Example:

```
{
  "id": 3,
  "traceType": "Interface",
  "startTime": "2023-07-04 13:00:00",
  "endTime": "2023-07-04 19:00:00",
  "neType": "UDM",
  "neId": "SZ_01",
  "interfaces": ["N8", "N10"],
  "notifyUrl": "gtp:192.168.0.229:2152",
}
```

- Return

Code=204 non-content

- 修改订阅

- URI

```
/api/rest/traceManagement/v1/subscriptions
```

- Method

PUT

- Relations

OMC front-end->OMC back-end, OMC -> NF

- Body

```
type TraceTask struct {
    Id          int      `json:"id"`
    TraceType   string   `json:"traceType"`
    StartTime   string   `json:"startTime"`
    EndTime     string   `json:"endTime"`
    Imsi        string   `json:"imsi"`
    Msisdn      string   `json:"msisdn"`
    SrcIp       string   `json:"srcIp"`
    DstIp       string   `json:"dstIp"`
    SignalPort  int16    `json:"signalPort"`
    NeType      string   `json:"neType"`
    NeId        string   `json:"neId"`
    UeIp        string   `json:"ueIp"`
    Interfaces  []string `json:"interfaces"`
    NotifyUrl   string   `json:"notifyUrl"`
}
```

Example:

```
{
  "id": 3,
  "traceType": "Interface",
  "startTime": "2023-07-04 13:00:00",
  "endTime": "2023-07-04 19:00:00",
  "neType": "UDM",
  "neId": "SZ_01",
  "interfaces": ["N8", "N10", "N11"],
  "notifyUrl": "gtp:192.168.0.229:2152",
}
```

- Return

Code=204 non-content

- 删除订阅

- URI

```
/api/rest/traceManagement/v1/subscriptions?id={id1}&id={id2}
```

- Method

DELETE

- Relations

OMC front-end->OMC back-end, OMC -> NF

- Params

id: 订阅任务id, 支持多个

- Body

NA

- Return

Code=204, non-content

- 查询订阅 (暂不实现, 直接从数据库查询)

- URI 查询单个订阅:

```
/api/rest/traceManagement/v1/subscriptions?id={id}
```

查询所有订阅:

```
/api/rest/traceManagement/v1/subscriptions
```

- Method

GET

- Body

NA

- Return

```
{
  "data": [
    {
      "id": 1,
      "traceType": "Interface",
      "startTime": "2023-07-04 13:00:00",
      "endTime": "2023-07-04 19:00:00",
      "neType": "UDM",
      "neId": "SZ_01",
      "interfaces": ["N8", "N10", "N11"]
    }
  ]
}
```

## 5.2 消息上报

Example:

```
{
  "id": [1,2],
  "timestamp": "20230413 16:02:27.523496",
  "imsi": "4600001000000001",
  "msisdn": "12307550001",
  "srcAddr": "192.168.1.172:51034",
  "dstAddr": "192.168.1.187:8080",
  "neType": "AMF",
  "neId": "SZ_0",
  "interface": "N8",
  "data": "00002f01040000007f418d0be25c2e3cb8570bcedc780f038345",
  "diagnosis": "It is external debug information",
}
```

## 6. 操作维护

### 6.1 MML命令

#### 6.1.1 MML命令格式

- 命令格式

```
oper object:parameter1={value1},parameter2={value2},parameter3={value3};
```

- 操作(operation)

- 根据实际操作可选用如下动作，如果没有合适的可自行增加，要做到简洁直观

```
add: 增加
mod(set): 修改/设置
del(rmv): 删除
dsp(lst): 查询显示
bak: 备份
exp: 导出
imp: 导入
bat: 批量
exec(run): 执行
act/dea: 激活/去激活
```

- 与对象之间采用空格分隔

- 对象(object)

- 操作的对象，如签约数据(sub/udmuser)/鉴权数据(auth/authdat), n7接口(n7server)
- 对象名称用所操作对象的英文名(或缩写), 为字母或者数字的组合, 不含空格, "-", "\_"等特殊字符
- 使用":"与参数进行分隔

- 参数(param)

- 参数名采用英文常用名/约定俗成的缩写/缩略语等, 如imsi, msisdn, ip, port等, 为字母或者数字的组合, 不含空格, "-", "\_"等特殊字符
- 参数值为字符串, 如有":", ",", ";", "\""字符, 需加\"进行转义
- 参数之间用","进行分隔

- 命令结束符";", 操作/对象/参数均采用小写字母 (HW采用的都是大写字母)

- 目前现有的各个网元的命令格式

除了UDM签约数据/鉴权数据的MML格式和上述格式基本一致, 各个网元的系统参数MML都不一样, 需统一成上述格式

- UDM鉴权数据/签约数据



```
add
authdat:imsi=460000100000030,ki=805DADC6E8A54A0D59D622C7A04D08E0,amf=8000
,algo=0,opc=CF7FD414E05754CFE08B4FE7F2EF2A36
```

- UDM系统参数

```
set n8ip 172.16.5.130
```

- AMF系统参数

```
set n8_ip 192.168.1.121
```

- SMF/UPF系统参数

```
set n7 server <http|https> <ip> <port>
```

## 6.1.2 MML配置表

- 类型定义 (type)

- string

filter指定字符串长度，如："filter": "6~100"，字符串的长度范围，如果单个数字表示最大长度

- ipv4

filter忽略

- ipv6

filter忽略

- int

filter指定整型数的范围，如："filter": "100~999"

- enum

"filter": '{"0": "http", "1": "https"}'

- o bool

"filter": '{"0": "false", "1": "true"}'

- o regex

filter为正则表达式

- Example

```
udm:
  authdataManagement:
    display: "Authentication Data Management"
    mml:
      - operation: "dsp"
        object: "authdat"
        display: "Display Auth Data"
        params:
          - name: "imsi"
            type: "string"
            optional: "false"
            filter: ""
            display: "IMSI"
            comment: ""
  subscriberManagement:
    display: "Subscriber Management"
    mml:
      - operation: "dsp"
        object: "authdat"
        display: "Display Auth Data"
        params:
          - name: "imsi"
            type: "string"
            optional: "false"
            filter: ""
            display: "IMSI"
            comment: ""
  systemManagement:
    display: "System Management"
    mml:
      - operation: "set"
        object: "n8ip"
        display: "Set N8 IP Address"
        params:
          - name: "ip"
            type: "ipv4"
            optional: "false"
            filter: ""
            display: "IP Address"
            comment: ""
```

## 6.2 MML接口

- URI

```
/api/rest/operationManagement/v1/elementType/{elementTypeValue}/objectType/mml  
?ne_id={neId}
```

- Relations

OMC front-end -> OMC back-end

OSS -> OMC (北向接口)

- Params

ne\_id={neId}

- Method

POST

- Body

```
{  
  "mml": [  
    "date",  
    "list ver",  
    "list lic",  
    "list comm"  
  ]  
}
```

- Return

```
{  
  "data": [  
    "2023-05-11 17:52:32.37333745 +0800 CST m=+28762.188435351\n",  
    "16.1.1\n",  
    "Expiry date: 2024-12-31, sn: 13740272\n",  
    "COMMAND NOT FOUND, opr: list, obj: comm\n"  
  ]  
}
```

# 7 北向接口(NBI)

## 7.1 查询资源数据接口

- URI

```
/api/rest/resourceManagement/{apiVersion}/elementType/{elementTypeValue}/objectType/{objectTypeValue}?rmUIDs={rmUIDValues}&fields={attributeNames}
```

- Relations

OMC -> NF

OSS -> OMC (北向接口)

- Params

- rmUIDs={rmUIDValues}

可携带多额rmUID(统一资源定位符), OMC->NF不带

- fields={attributeNames}

属性域集合={属性名列表}, 指定资源对象多个属性名的英文逗号分割, 一个属性名时无英文逗号。

- Method

GET

- Body

```
GET /api/rest/resourceManagement/v1/elementType/SMF/objectType/ManagedElement?
rmUIDs=1101AGTHXSMF0000015704000100&fields=UserLabel HTTP/1.1
accessToken: 52661fbd-6b84-4fc2-aa1e-17879a5c6c9b
Host: serverIP:port
Content-Type: application/json; charset=UTF-8
Content-Length:...
{
}
```

- Return

```
HTTP/1.1 200 OK
Content - Type: application/json
Content-Length:...
{
  "data": [{
    "rmUID": "1101AGTHXSMF0000015704000100",
    "UserLabel": "SMFRJBJJC01",
    ...
  }]
}
```

## 8 文件接口

---

### 8.1 软件管理

#### 8.1.1 软件包管理接口

- URI

```
/api/rest/systemManagement/v1/{neType}/software/{version}?md5Sum={md5Sum}
```

- Relations

OMC front-end -> OMC back-end

- Params

md5Sum={md5Sum}

- Method

- POST: upload to OMC, 上传文件到OMC, content\_type=multipart/form-data
- GET: download from omc, 下载文件到OMC, content\_type=multipart/form-data
- DELETE: delete from omc

- Body

POST: 软件包文件, OMC按网元类型存储文件, 如果文件名相同则会覆盖

- Return
  - POST/DELETE: code=204
  - GET: code=200, 返回文件

### 8.1.2 网元软件包管理接口

- URI

```
/api/rest/systemManagement/v1/{neType}/software/{version}/{neId}
```

- Relations

OMC front-end -> OMC back-end

- Params

- Method

- POST: distribute to NF
- PUT: active
- PATCH: rollback

- Body

NA

- Return

Code: 204 Not content

## 8.2 License管理

### 8.2.1 License文件管理接口

- URI

```
/api/rest/systemManagement/v1/{neType}/license
```

- Relations

OMC front-end -> OMC back-end

- Params

- Method

- POST: upload to OMC, 上传文件到OMC, content\_type=multipart/form-data
- GET: download from omc, 下载文件到OMC, content\_type=multipart/form-data
- DELETE: delete from omc

- Body

POST: 携带License文件, OMC按网元类型存储文件, 如果文件名相同则会覆盖

- Return

Code: 204 Not content

## 8.2.2 网元License管理接口

- URI

```
/api/rest/systemManagement/v1/{neType}/license/{neId}
```

- Relations

OMC front-end -> OMC back-end

- Params

- Method

- POST: distribute to NF
- PUT: active
- PATCH: rollback

- Body

NA

- Return

Code: 204 Not content

## 9 4A接口

---

### 9.1 从账号(user)管理接口

#### 9.1.1 查询全部从账号(user)接口

- URI

```
/api/rest/aaaa/v1/security/users
```

- Relations

4A -> OMC

- Params

- Method

GET

- Body

- Return

#### 9.1.2 单个从账号(user)接口

- URI



```
/api/rest/aaaa/v1/security/users/{id}
```

- Relations

4A -> OMC

- Params

- Method

GET/POST/PUT/DELETE

- Body

- Return

## 9.2 角色(role)管理接口

### 9.2.1 查询全部角色(role)接口

- URI

```
/api/rest/aaaa/v1/security/roles
```

- Relations

4A -> OMC

- Params

- Method

GET

- Body

- Return

## 9.2.2 单个角色(role)接口

- URI

```
/api/rest/aaaa/v1/security/roles/{id}
```

- Relations

4A -> OMC

- Params

- Method

GET/POST/PUT/DELETE

- Body

- Return

## 9.3 登录认证接口

### 9.2.1 登录认证

- URI

```
/api/rest/aaaa/v1/security/authentication/token
```

- Relations

4A -> OMC

- Params

- Method

GET

- Body

- Return

## 10 核心网用户信息接口

---

### 10.1 UDM签约用户

#### 10.1.1 查询/增加/修改/删除

用MML接口

- Uri

```
/api/rest/ueManagement/v1/elementType/UDM/objectType/subData/{imsi}?neId={neId}
```

- Relations

OMC -> UDM

- Params

OMC-FE -> OMC-BE, neld={neld}, 指定UDM的neld

- Method

GET

- Body

NA

- Return

```
// SmfUEInfo SMF在线用户信息
type SubData struct {
    IMSI      string `json:"imsi"`           // SIM卡号
    MSISDN    string `json:"msisdn"`
    Amf       string `json:"amf"`           // Amf
    Status    string `json:"status"`       // 状态
    Ki        string `json:"ki"`           // ki
    AlgoIndex string `json:"algoIndex"` //
    Opc       string `json:"opc"`
}
```

```
{
  "data":
  {
    "imsi": "460000100000010",
    "msisdn": "12307550010",
    "...": "..."
  }
}
```

## 10.2 UDM鉴权用户

### 10.2.1 查询/增加/修改/删除

用MML接口

## 10.3 UE信息

### 10.3.1 查询SMF在线用户数

- Uri

```
/api/rest/ueManagement/v1/elementType/smf/objectType/ueNum?neId={neId}
```

- Relations

OMC -> SMF

- Params

OMC-FE -> OMC-BE, neId={neId}, 指定SMF的neId

- Method

GET

- Body

NA

- Return

```
// SmFUENum SMF在线用户数
type SmFUENum struct {
    UENum int `json:"ueNum"` // 当前在线用户数
}
```

```
{
  "data":
  {
    "ueNum": 6
  }
}
```

### 10.3.2 查询SMF在线用户

说明：为了避免在线用户太多，SMF可做限制，如返回不超过100条

- Uri

```
/api/rest/ueManagement/v1/elementType/smf/objectType/ueInfo?imsi={imsi}&msisdn={msisdn}&neId={neId}
```

- Relations

OMC -> SMF

- Params

- OMC-FE -> OMC-BE, neId={neId}, 指定SMF的neId
- OMC -> SMF, imsi={imsi} 或者 msisdn={msisdn}, 可选, 如果都不指定, 则查询全部ueInfo

- Method

GET

- Body

NA

- Return

```
// SmfUEInfo SMF在线用户信息
type SmfUEInfo []struct {
    IMSI          string `json:"imsi"`
    MSISDN        string `json:"msisdn"`
    PduSessionInfo []struct {
        PduSessionID int    `json:"pduSessionID"`
        IPv4          string `json:"ipv4"`
        IPv6          string `json:"ipv6"`
        Dnn           string `json:"dnn"`
        Tai           string `json:"tai"`
        SstSD         string `json:"sstSD"`
        UpfN3IP       string `json:"upfN3IP"`
        RanN3IP       string `json:"ranN3IP"`
        Activetime   string `json:"activeTime"`
    } `json:"pduSessionInfo"`
}
```

```
{
  "data": [
    {
      "imsi": "460000100000010",
      "msisdn": "12307550010",
    }
  ]
}
```

```

    "pduSessionInfo": [
      {
        "pduSessionID": "1",
        "ipv4": "172.16.5.5",
        "...": "..."
      },
      {
        "pduSessionID": "2",
        "ipv4": "172.16.5.6",
        "...": "..."
      }
    ]
  },
  {
    "imsi": "460000100000238",
    "msisdn": "12307550238",
    "pduSessionInfo": [
      {
        "pduSessionID": "3",
        "ipv4": "172.16.5.7",
        "...": "..."
      },
      {
        "pduSessionID": "4",
        "ipv4": "172.16.5.8",
        "...": "..."
      }
    ]
  }
]
}

```

## 10.4 IMS在线用户信息

### 10.4.1 查询IMS在线用户信息

说明：为了避免在线用户太多，IMS可做限制，如返回不超过100条

- Uri

```

/api/rest/ueManagement/v1/elementType/ims/objectType/ueInfo?imsi=
{imsi}&msisdn={msisdn}&neId={neId}

```

- Relations

OMC -> IMS

- Params

- OMC-FE -> OMC-BE, neld={neld}, 指定IMS的neld
- OMC -> IMS, imsi={imsi} 或者 msisdn={msisdn}, 可选, 如果都不指定, 则查询全部ueInfo

- Method

GET

- Body

NA

- Return

```
// ImsUEInfo IMS在线用户信息
type ImsUEInfo []struct {
    IMSI      string `json:"imsi"`
    MSISDN    string `json:"msisdn"`
    IMPU      string `json:"impu"`
    Barring   int    `json:"barring"`
    RegState  int    `json:"regState"`
    Activetime string `json:"activeTime"`
}
```

```
{
  "data": [
    {
      "imsi": "460000100000010",
      "msisdn": "12307550010",
      "impu": "sip:12307550010@ims.mnc000.mcc460.3gppnetwork.org",
      "barring": 0,
      "regState": 1,
      "activeTime": "2023-07-11 18:26:46"
    },
    {
      "imsi": "460000100000238",
      "msisdn": "12307550038",
      "impu": "sip:12307550238@ims.mnc000.mcc460.3gppnetwork.org",
      "barring": 0,
      "regState": 1,
      "activeTime": "2023-07-11 18:26:46"
    }
  ]
}
```

## 10.5 基站信息



## 10.5.1 查询某个AMF下的基站信息

- Uri

```
/api/rest/ueManagement/v1/elementType/amf/objectType/nbInfo?nbId={nbId}&neId={neId}
```

- Relations

OMC -> AMF

- Params

- OMC-FE -> OMC-BE, neId={neId}, 指定AMF的neId
- OMC -> AMF, nbId={nbId}, 可选, 如果不指定则查询全部基站信息

- Method

GET

- Body

NA

- Return

```
// AmfNBInfo AMF的NodeB信息
type AmfNBInfo []struct {
    ID      string `json:"id"`      //NodeB ID
    Name    string `json:"name"`    // NodeB name
    Address string `json:"address"` // 基站地址
    UENum   int    `json:"ueNum"`  // UE数量
}
```

```
{
  "data": [
    {
      "id": "6001",
      "name": "NB6001",
      "address": "192.168.1.245:36412",
      "ueNum": 2
    },
    {
      "id": "6002",
      "name": "NB6002",

```

```
"address": "192.168.1.246:36412",  
"ueNum": 6
```

```
}
```

```
]
```

```
}
```