/data/logs/nginx/
${appId}.log

标准json字符串
标准json字符串
……

```
log_format msgLog  '{'
  '"msgId":"$msg_id"'
  ',"msgVersion":"1.0"'
  ',"msgSite":"$msg_site"'
  ',"msgSource":"ngx_log"'
  ',"msgFormat":"json"'
  ',"msgSignFlag":$msg_sign_flag'
  ',"msgBody":{'
    '"svr_host":"$host"'
    ',"svr_req_method":"$request_method"'
    ',"svr_req_url":"$request_uri"'
    ',"svr_content_type":"$content_type"'
    ',"svr_remote_addr":"$remote_addr"'
    ',"svr_forwarded_for":"$http_x_forwarded_for"'
    ',"svr_receive_time":$msg_receive_time'
    ',"appId":"$msg_app_id"'
    ',"body":$msg_req_body'
    '}'
  '}'
;
```

log_raw_${产品线}

KEY:未定义
VALUE:

```
{
  "@timestamp":"2017-04-25T07:31:46.128Z",
  "message":"json字符串",
  "type":"log"
}
```

log_origin_${appId}.log

KEY:

```
  {
    "rawTs":"归集层接收时间",
    "rawParId":"归集层分区ID",
    "rawOffset":"归集层offset",
    "oriTs":"应用层接收时间"
  }
```

VALUE:

```
{
  "msgId":"",
  "msgVersion":"",
  "msgSite":"",
  "msgSource":"",
  "msgFormat":"",
  "msgSignFlag":"",
  "logId":"",
  "logVersion":"",
  "logTime":"",
  "logSignFlag":"",
  "appId":"",
  "logBody":{ }
}
```

目录: /data_warehouse/ods_origin.db/tmp_log_origin/
SequenceFile:
timeKey=${logTime:yyyyMMddHH}
backupper: ${appId}_${timeKey}_ori_${oriParId_oriOffset}.seq
forest: ${appId}_${timeKey}_raw_${rawParId_rawOffset}.seq

```
{
  "_sync":{
    "rawTs":"归集层接收时间",
    "rawParId":"归集层分区ID",
    "rawOffset":"归集层offset",
    "oriTs":"应用层接收时间"
    "oriParId":"应用层分区ID",
    "oriOffset":"应用层offset",
    "odsTs":"ods接收时间"
  },
  "msgId":"",
  "msgVersion":"",
  "msgSite":"",
  "msgSource":"",
  "msgFormat":"",
  "msgSignFlag":"",
  "logId":"",
  "logVersion":"",
  "logTime":"${yyyy-MM-dd HH:mm:ss}",
  "logSignFlag":"",
  "appId":"",
  "logBody":{ }
}
```

目录: /data_warehouse/ods_origin.db/log_origin/
key_appId=${appId}/key_day=${yyyymmdd}/key_hour=${HH}
SequenceFile:
timeKey=${logTime:yyyyMMddHH}
backupper: ${appId}_${timeKey}_ori_${oriParId_oriOffset}.seq
forest: ${appId}_${timeKey}_raw_${rawParId_rawOffset}.seq

```
{
  "_sync":{
    "rawTs":"归集层接收时间",
    "rawParId":"归集层分区ID",
    "rawOffset":"归集层offset",
    "oriTs":"应用层接收时间"
    "oriParId":"应用层分区ID",
    "oriOffset":"应用层offset",
    "odsTs":"ods接收时间"
  },
  "msgId":"",
  "msgVersion":"",
  "msgSite":"",
  "msgSource":"",
  "msgFormat":"",
  "msgSignFlag":"",
  "logId":"",
  "logVersion":"",
  "logTime":"${yyyy-MM-dd HH:mm:ss}",
  "logSignFlag":"",
  "appId":"",
  "logBody":{ }
}
```

nginx文件:
> 文件以appId命名
> 日志格式统一为`msgLog`

归集层kafka:
> 由filebeat写入,消息体结构由filebeat决定
> 按产品线进行归集

forest处理:
> 日志编码转换
> 旧版日志签名校验
> 日志平展化

应用层kafka:
> 在KEY中记录同步信息
> VALUE为forest处理后的JSON字符串

HDFS文件:
> _sync字段记录同步信息
> 每个分区对应单独文件, 从而满足文件单线程写入
> logTime为业务时间,即日志行为发生时间
> 数据分区以logTime为基准,保证物理分区和业务分区的一致性

hive表:
> 按appId,天,小时进行分区
> 建立appId到表名的映射关系,可选方案: log_origin_${产品线}_${应用名}
> 可遵循上述JSON结构建表,logBody映射为Map结构