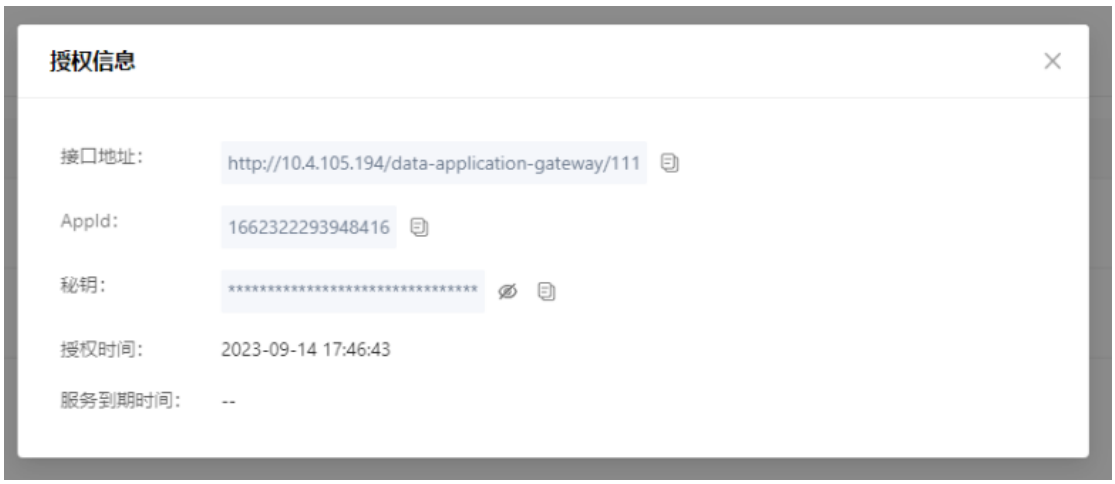
1. 获取 AppId 和 AppSecret（密钥）



2. 构造签名串

签名串一共有 6 行，每一行为一个参数。结尾以\n（换行符，ASCII 编码值为 0x0A）结束，包括最后一行。如果参数本身以\n 结束，也需要附加一个\n。

```
HTTP 请求方法\n
请求时间戳 timestamp（秒级，请求时间与服务器时间相差不能超过 1 分钟）\n
请求随机串 nonce（每个 nonce 只能使用一次，重复使用无效）\n
请求 Path (不包括域名部分，不包含 QueryString)\n
请求 QueryString (按 key 升序排序 k1=v1&k2=v2&k3=v3)\n
请求 Body (序列化后的 json 字符串，如果没有 Body，此行为空，只保留换行符)\n
```

签名串的示例：

```
95088180
78b577e0-b460-4b67-af2b-6e109be17462
/data-application-gateway/111
a=1&b=2&c=3
{
    "version": 1,
    "dirty": 1
}
```

3. 计算签名值

接口服务使用 hmac-sha256 方式进行签名计算，以下是一些语言的代码示例：

```
// JavaScript 生成签名，message 为签名串，appSecret 为密钥
const signature = CryptoJS.HmacSHA256(message, appSecret).toString();
```

```
// Golang 生成签名，message 为签名串，appSecret 为密钥
h := hmac.New(sha256.New, []byte(appSecret))
h.Write([]byte(message))
signature = hex.EncodeToString(h.Sum(nil))
```

4. 设置 HTTP Header

把签名信息放在 Authorization 头中，格式为：

```
Authorization: ANYFABRIC-HMAC-
SHA256 appid=${appId},timestamp=${timestamp},nonce=${nonce},signature=${signature}
```

其中认证类型固定为 ANYFABRIC-HMAC-SHA256，后边拼接签名信息

Authorization 的示例：

```
Authorization: ANYFABRIC-HMAC-
SHA256 appid=1662322293948416,timestamp=1695089020,nonce=a5de3b96-e448-409c-b263-
ae4902e92222,signature=69595bc03836fc5c5cf898848534162ed8deb609b6edc951850ca5f70450
1531
```