

什么是OSS ?

对象存储服务 (Object Storage Service , OSS) 是一种海量、安全、低成本、高可靠的云存储服务 , 适合存放任意类型的文件。容量和处理能力弹性扩展 , 多种存储类型供选择 , 全面优化存储成本。

什么是AccessKey ?

AccessKey包括AccessKeyId和AccessKeySecret两部分 , AccessKeyId用于标识用户 , AccessKeySecret用于验证用户的密钥 , 主要用于程序方式调用云服务API。

我们来看一个简单的测试案例 , 当测试某个上传点时 , 获取到一个HTML表单 :

Request

RawParamsHeadersHex

-----WebKitFormBoundaryoeAATQBaaavlasNQ
Content-Disposition: form-data; name="name"

3.png
-----WebKitFormBoundaryoeAATQBaaavlasNQ
Content-Disposition: form-data; name="key"

a/20190425/b23d5d798b61a9e4872063d2f83cd12d1572011366578.png
-----WebKitFormBoundaryoeAATQBaaavlasNQ
Content-Disposition: form-data; name="policy"

eyJleHBpcmF0aW9uIjoiaWJyYmC0wMS0wMVQxMjowMDowMC4wMDBaliwiY29uZGI0aW9ucyI6VW1siY29udGVudC1sZW5ndGgtcmFuZ2UuLDAsMTA0ODU3NjAwMF1dfQ==
-----WebKitFormBoundaryoeAATQBaaavlasNQ
Content-Disposition: form-data; name="OSSAccessKeyId"

6MKQqxGiGU4AUk44
-----WebKitFormBoundaryoeAATQBaaavlasNQ
Content-Disposition: form-data; name="success_action_status"

200
-----WebKitFormBoundaryoeAATQBaaavlasNQ
Content-Disposition: form-data; name="signature"

Xq2V7XmCvwSrm5K5MY80QbkJous=
-----WebKitFormBoundaryoeAATQBaaavlasNQ
Content-Disposition: form-data; name="file"; filename="3.png"
Content-Type: image/png

请求的主机名xxxx.aliyuncs.com和表单的OSSAccessKeyId参数 , 基本可以确认系统使用 OSS 作为上传文件存储 , 即使上传恶意脚本文件也无法成功解析 , 面对这种情况如何破局 ?

通过查阅相关文件 , 我们可以知道使用表单上传文件到 OSS的技术方案里 , 有三种实现方式 :

oss产品文档 : https://help.aliyun.com/document_detail/31923.html

1. 在客户端通过JavaScript代码完成签名 , 然后通过表单直传数据到OSS。
2. 在服务端完成签名 , 然后通过表单直传数据到OSS。

3. 在服务端完成签名，并且服务端设置了上传后回调，然后通过表单直传数据到OSS。OSS回调完成后，再将应用服务器响应结果返回给客户端。

当采用JavaScript客户端直接签名时，AccessKeyId和AccessKeySecret会暴露在前端页面，存在严重的安全隐患。

通过翻找js文件，可发现AccessKey就写在js文件里面。

```
accessid= 'LTAI4[REDACTED]31CUPw';
accesskey= 'Lr4:[REDACTED]9GYfKexwqCl4h';
host = 'http://[REDACTED]-hangzhou.aliyuncs.com';

g_dirname = ''
g_object_name = ''
g_object_name_type = ''
now = timestamp = Date.parse(new Date()) / 1000;

var policyText = {
  "expiration": "2020-01-01T12:00:00.000Z", //设置该Policy的失效时间，超过这个失效时间之后，就设
  "conditions": [
    ["content-length-range", 0, 1048576000] // 设置上传文件的大小限制
  ]
};

var policyBase64 = Base64.encode(JSON.stringify(policyText))
message = policyBase64
var bytes = Crypto.HMAC(Crypto.SHA1, message, accesskey, { asBytes: true });
var signature = Crypto.util.bytesToBase64(bytes);
```

AccessKey泄露，如何进行漏洞利用呢？

AccessKey是访问阿里云API的密钥，将会造成什么样的风险呢。

1、通过API接口

AccessKey ID和AccessKey Secret 就是打开这扇门的钥匙，通过调用API实现资源接管，这种方式对编程能力有一定要求。

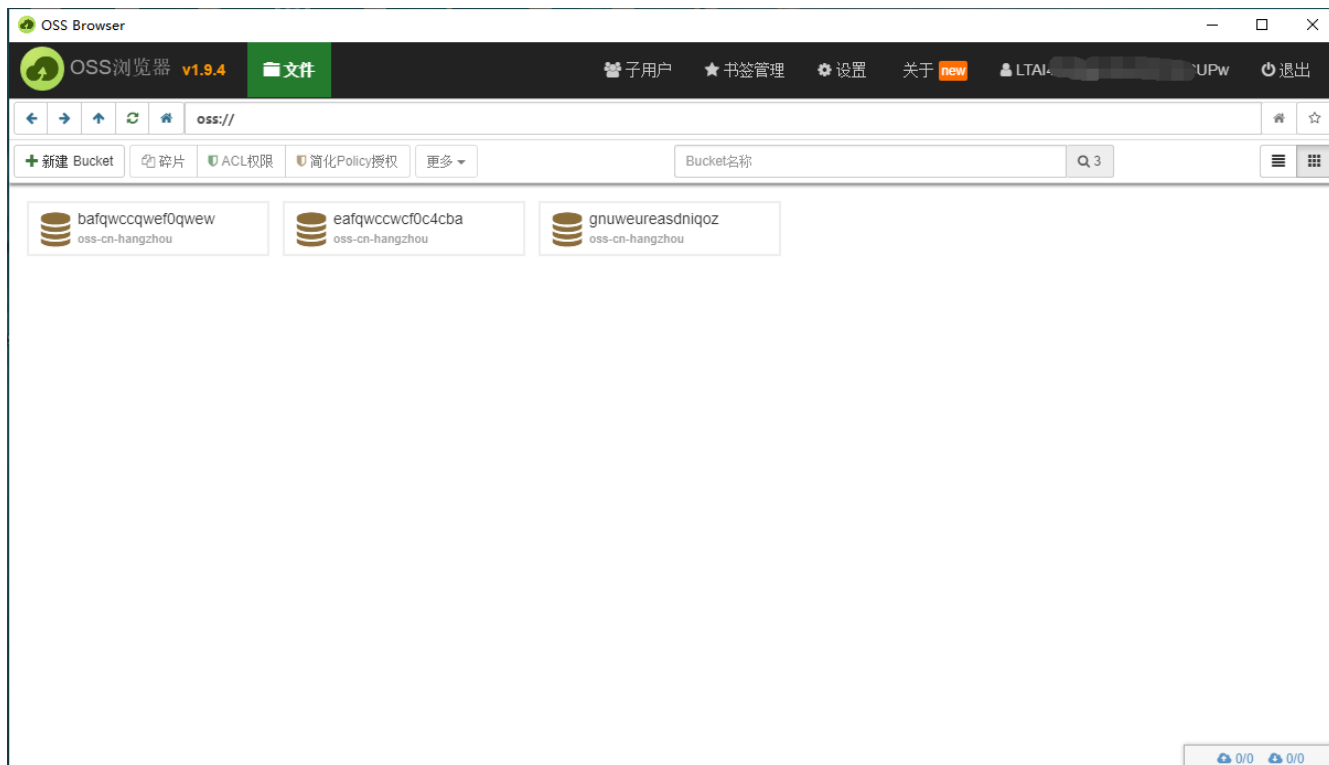
API参考：https://help.aliyun.com/document_detail/64844.html

2、通过第三方管理工具

- OSSBrowser

ossbrowser 是 OSS 官方提供的图形化管理工具，提供类似 Windows 资源管理器的功能，使用 ossbrowser，您可以方便地浏览、上传、下载和管理文件。

下载地址：<http://gosspublic.alicdn.com/oss-browser/1.9.4/oss-browser-win32-x64.zip>



- 护卫神.云备份

一键备份数据到阿里云OSS，支持Bucket管理，支持鼠标拖放，支持剪贴板，支持断点续传，支持统计目录大小，支持文件搜索。

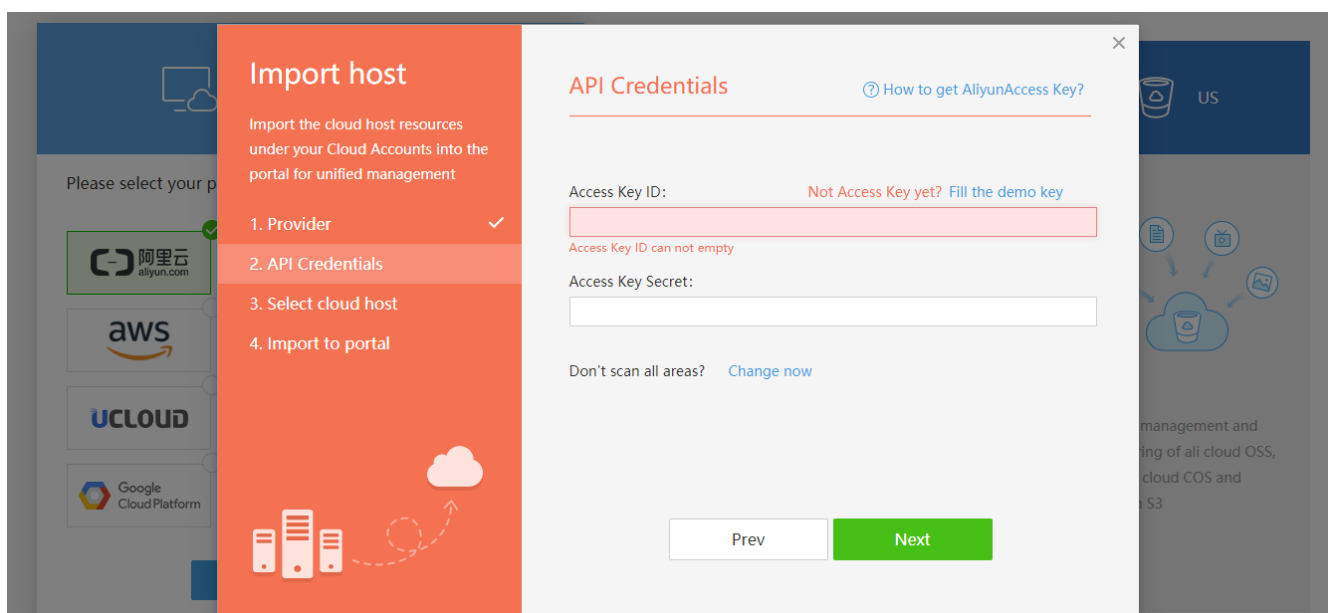
下载地址：<https://d.hws.com/free/HwsOSS.zip>



• 行云管家

多云管理平台，提供跨云厂商的云计算管理方案，AccessKey导入，可重置服务器密码。

官方地址：<https://yun.cloudbility.com/login.html>



常见问题解答

1. OSS的AccessKey 在什么情况下会出现泄露？

采用JavaScript客户端直接签名时，AccessKeyId和AccessKeySecret会暴露在前端页面，存在严重的安全隐患。或者通过其他方式获取webshe11，查看相关配置文件找到AccessKey配置。

2. 前端OSS AccessKey 泄露，代码如何修复？

采用JavaScript客户端签名直传存在严重安全风险，建议采用服务端签名后直传。

3. 泄露了访问OSS的AccessKey该如何补救？

最安全的办法就是更换AccessKey，毕竟它只能创建或删除，启用或禁用，是没有给你修改密码的机会的。

4. 测试时，如何简单地来判断OSS的AccessKey是否储存在前端？

可以通过上传操作时，抓取的HTTP请求数量来做简单的判断。

当采用JavaScript客户端直接签名，用户直接上传数据到OSS，一次请求即可完成。

当采用服务端签名后直传的方式，需要用户向应用服务器请求上传Policy，再将数据上传到OSS，至少需要两次请求。

企业上云已成趋势，面对云平台的部署架构，不管是开发、安全或是运维，都将面临新的风险和挑战。云上丰富的产品矩阵，为用户提供了各种实例的选项，但技术方案的实现，云上的安全策略及服务，RAM精准的权限控制，每一步都与安全有关。