阿里云
游戏盾

常见问题

文档版本：20190812

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。
2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。
3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。
4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的”现状“、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。
5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含”阿里云”、Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。
6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **禁止:** 重置操作将丢失用户配置数据。
	该类警示信息可能导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **警告:** 重启操作将导致业务中断，恢复业务所需时间约10分钟。
	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	 **说明:** 您也可以通过按Ctrl + A选中全部文件。
>	多级菜单递进。	设置 > 网络 > 设置网络类型
粗体	表示按键、菜单、页面名称等UI元素。	单击 确定。
`courier` 字体	命令。	执行 `cd /d C:/windows` 命令，进入Windows系统文件夹。
*## *	表示参数、变量。	`bae log list --instanceid` *`Instance_ID`*
[]或者[a|b]	表示可选项，至多选择一个。	`ipconfig` *`[-all|-t]`*
{}或者{a|b}	表示必选项，至多选择一个。	`swich` *`{stand | slave}`*

目录

1 游戏盾如何配置RAM权限

游戏盾支持通过RAM系统进行权限的划分，本文介绍了具体的操作方法。

操作步骤

1. 参见创建RAM用户创建并配置RAM。
2. 分配游戏盾相关权限。

   目前，支持以下两个权限：

   - 管理云盾游戏盾完整操作权限：AliyunYundunGameShieldFullAccess
   - 管理云盾游戏盾只读访问权限：AliyunYundunGameShieldReadOnlyAccess

2 转发规则配置常见问题

游戏盾支持哪些转发规则和端口?

支持端口范围：80、443、1025-65535

支持转发规则：TCP

游戏盾端口转发条目数限制?

游戏盾单IP最多支持50个端口的转发，超过50个端口的情况则可以通过创建多个业务分组（每个业务50条转发、业务下所有IP转发规则相同）来满足需求。

如果开启游戏应用网关，可以联系游戏盾团队启用SDK接入，并通过4层端口复用来满足业务需求。（这种模式下程序需要做较大改造）

游戏盾如何支持HTTP/HTTPS协议?

请使用TCP协议转发代替HTTP、HTTPS协议使用。

单条转发后端服务器支持多个吗?

单条转发规则可以支持20个源站，默认会根据连接数启用负载均衡和会话保持功能。

3 如何解决HTTP(S)协议接入游戏盾后HOST不匹配问题

问题描述

终端通过HTTP/HTTPS协议接入的情况下，如果将URL中的HOST配置直接从域名替换为服务器IP后，由于不通过域名而是通过IP直接进行请求。这种情况下，服务端获取到的域名信息为服务器IP，因此请求报文内容中的Host信息为IP，而对于这种请求，服务端的响应结果将取决于服务端的具体配置：

- 如果服务端只服务单个域名，则可能无视Host的值，返回正确的页面。
- 如果服务端服务多个域名，则通常会返回404或403错误。

另外，如果通过HTTPS协议接入，服务端可能无法找到匹配的证书，只能返回默认证书或者不返回。而客户端在进行证书校验时，也会因为域名不匹配的问题（证书是域名，而校验的是服务器IP），导致SSL证书校验失败 。

传统解决方案

HTTP协议接入解决方案

针对HTTP协议接入情况的解决方案相对简单。一般来说，第三方库都提供相应接口支持修改HTTP请求Header的HOST信息，只需要开发人员将HTTP Header的HOST改为对应的域名即可。

HTTPS协议接入解决方案

- **Android系统解决方案**

  1. **证书HOST校验问题**

     终端在SSL握手过程中会校验当前请求URL的HOST是否在服务端证书的可选域名列表中。例如，假设原本需要请求的URL为`https://a.b.com`，使用服务器IP直连后实际请求的URL变成`https://1.2.3.4`。

     由于请求的HOST被替换成服务器IP，底层在进行证书的HOST校验时失败，最终导致请求失败。

     一般来说，系统都提供相应接口，允许终端设置证书HOST校验实现。因此，利用该接口，将底层默认实现中取终端传入URL的HOST信息（即服务器IP）替换回对应的域名即可解决证书HOST校验问题。

     JAVA代码示例：

```
HostnameVerifier hnv = new HostnameVerifier() {
@Override
public boolean verify(String hostname, SSLSession session) {
//示例
if("yourhostname".equals(hostname)){
return true;
} else {
HostnameVerifier hv =
HttpsURLConnection.getDefaultHostnameVerifier();
return hv.verify(hostname, session);
}
}
};

HttpsURLConnection.setDefaultHostnameVerifier(hnv);
```

  2. **SNI问题**

     由于不通过域名而是通过IP直接进行请求。这种情况下，服务端获取到的域名信息为服务器IP，因此请求报文内容中的Host信息为IP，而服务端配置了多个域名，导致无法正确选择域名。

     一般来说，系统都提供相应接口，允许终端传入自定义SSLSocketFactory，SSLSocketFactory是用来创建SSLSocket的工厂，SSLSocket是Socket协议的拓展，具有SSL握手功能，且系统提供解决SNI问题的实现类SSLCertificateSocketFactory。因此，利用该方法解决SNI问题。

     JAVA代码示例

```
conn.setSSLSocketFactory(new SSLSocketFactory(){
@Override
public Socket createSocket(Socket s, String host, int port,boolean
 autoClose) throws IOException{
```

```
SSLCertificateSocketFactory sslSocketFactory = (SSLCertifi
cateSocketFactory)SSLCertificateSocketFactory.getDefault(0);
SSLSocket sslSocket = (SSLSocket)sslSocketFactory.createSocket(s,
realHost,port,autoClose);
sslSocket.setEnableProtocols(sslSocket.getSupportedProtocols());
sslSocketFactory.setHostname(sslSocket, realHost);
return sslSocket;
}
});
```

- **iOS系统解决方案**

1. **证书HOST校验问题**

   在`NSURLSession`**的证书校验代理方法**`URLSession:didReceiveChallenge:completionHandler`**中增加前置处理，将待验证的** `domain`**由原本的服务器IP转换为其对应的域名，然后再进行后续处理。**

   Objective-C代码示例

```
- (void)URLSession:(NSURLSession *)session
didReceiveChallenge:(NSURLAuthenticationChallenge *)challenge
completionHandler:(void (^)(NSURLSessionAuthChallengeDisposition
disposition, NSURLCredential *credential))completionHandler
{
NSURLSessionAuthChallengeDisposition disposition = NSURLSessi
onAuthChallengePerformDefaultHandling;
NSURLCredential *credential = nil;

// 证书验证前置处理
NSString *domain = challenge.protectionSpace.host; // 获取当前请求
的 host（域名或者 IP），假设此时为: 1.2.3.4
NSString *testHostIP = self.tempDNS[self.testHost];
// 此时服务端返回的证书里的 CN 字段（即证书颁发的域名）与上述 host 可能不一
致，
// 因为上述 host 在发请求前已经被替换为 IP，所以校验证书时会发现域名不一致而
无法通过，导致请求被取消
// 所以，需要在校验证书前进行替换处理。
if ([domain isEqualToString:testHostIP]) {
domain = self.testHost; // 替换为对应域名: a.b.com
}

// 以下逻辑与 AFNetworking -> AFURLSessionManager.m 里的代码一致
if ([challenge.protectionSpace.authenticationMethod isEqualToS
tring:NSURLAuthenticationMethodServerTrust]) {
if ([self evaluateServerTrust:challenge.protectionSpace.serverTrus
t forDomain:domain]) {
// 上述evaluateServerTrust:forDomain方法用于验证 SSL 握手过程中服务端返
回的证书是否可信任，
// 以及请求的 URL 中的域名与证书里声明的的 CN 字段是否一致。
credential = [NSURLCredential credentialForTrust:challenge.
protectionSpace.serverTrust];
if (credential) {
disposition = NSURLSessionAuthChallengeUseCredential;
} else {
disposition = NSURLSessionAuthChallengePerformDefaultHandling;
}
} else {
disposition = NSURLSessionAuthChallengeCancelAuthenticationChall
enge;
```

```
}
} else {
disposition = NSURLSessionAuthChallengePerformDefaultHandling;
}

if (completionHandler) {
completionHandler(disposition, credential);
}
}
```

其中，关于evaluateServerTrust:forDomain方法的定义，可参考 AFNetworking中 AFSecurityPolicy模块的代码，Objective-C代码示例如下所示。

```
- (BOOL)evaluateServerTrust:(SecTrustRef)serverTrust forDomain:(
NSString *)domain {
// 创建证书校验策略
NSMutableArray *policies = [NSMutableArray array];
if (domain) {
// 需要验证请求的域名与证书中声明的 CN 字段是否一致
[policies addObject:(__bridge_transfer id)SecPolicyCreateSSL(true,
 (__bridge CFStringRef)domain)];
} else {
[policies addObject:(__bridge_transfer id)SecPolicyCreateBasicX509
()];
}

// 绑定校验策略到服务端返回的证书 (serverTrust)
SecTrustSetPolicies(serverTrust, (__bridge CFArrayRef)policies);

// 评估当前 serverTrust 是否可信任,
// 根据苹果官方文档 https://developer.apple.com/library/ios/
technotes/tn2232/_index.html
// 当 result 为 kSecTrustResultUnspecified 或 kSecTrustResultProce
ed 的情况下，serverTrust 可以被验证通过。
SecTrustResultType result;
SecTrustEvaluate(serverTrust, &result);
return (result == kSecTrustResultUnspecified || result ==
kSecTrustResultProceed);
}
```

2. SNI问题

通过使用基于原生支持设置SNI字段的更底层的库（libcurl），解决SNI问题。

Objective-C代码示例

```
//{HTTPS域名}:443:{IP地址}
NSString *curlHost = ...;
_hosts_list = curl_slist_append(_hosts_list, curlHost.UTF8String);
curl_easy_setopt(_curl, CURLOPT_RESOLVE, _hosts_list);
```

其中，curlHost形（如{HTTPS域名}:443:{IP地址}，_hosts_list）是结构体类型 hosts_list，可设置多个IP与Host域名间的映射关系。通过在curl_easy_setopt方法中传入CURLOPT_RESOLVE，将该映射设置到HTTPS请求中，即可达到设置SNI的目的。

游戏盾解决方案

利用游戏盾自身机制，阿里云提供一种更好的解决方案：

1. 将被访问网站的DNS域名解析到127.0.0.1。

   > **说明:**
   > 更改DNS域名解析需要确认该域名没有其它线上业务。

2. HTTP、HTTPS请求时，使用域名：代理端口的形式替换原先的`127.0.0.1：代理端口`接入方式来访问web服务器。其中的代理端口即`getProxyTcpByDomain`接口返回的端口。

通过上述操作，就可以完美解决SSL认证、单IP多HTTPS服务器等问题。使用这种方式，无需改动代码，且能更好地保护源站服务器。

同时，与传统解决方案相比，游戏盾解决方案在安全性方面也更完善。由于传统解决方案在代码中暴露域名信息，如果域名配置了源站服务器IP，攻击者很容易找到源站服务器直接进行攻击；而采用游戏盾解决方案，即使攻击者发现源站域名，也无法获取源站服务器IP。

因此，无论是从兼容性、简单性、还是安全性角度，推荐您使用游戏盾解决方案解决HOST不匹配问题。