全栈式风控引擎

业务客户端容灾方案

腾讯云

【版权声明】

【商标声明】

【服务声明】

本文档意在向您介绍腾讯云全部或部分产品、服务的当时的相关概况，部分产品、服务的内容可能不时有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或默示的承诺或保证。

【联系我们】

我们致力于为您提供个性化的售前购买咨询服务，及相应的技术售后服务，任何问题请联系 4009100100或95716。

业务客户端容灾方案

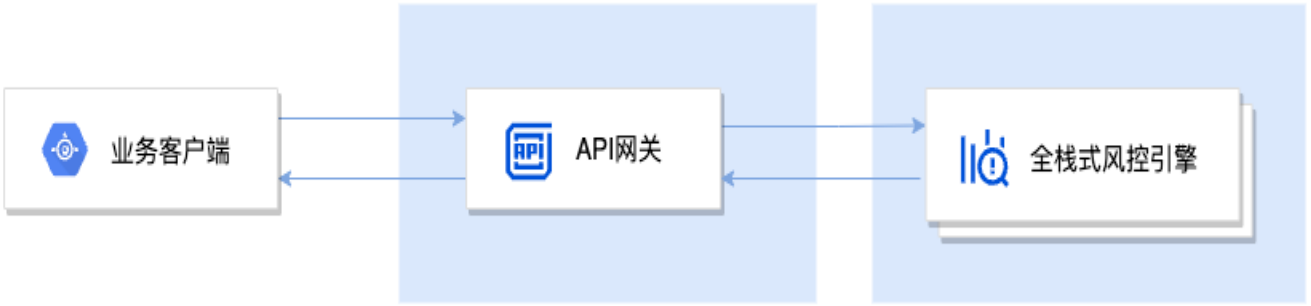
最近更新时间：2024-08-08 10:02:41

概述

全栈式风控引擎服务的业务客户端容灾方案是我们提供的一套有效的应对策略，以便您在使用全栈式风控引擎服务时，即便出现接口异常，也能保障您的线上业务流程不受影响，同时提升用户体验。在您面对各种可能的异常情况时，本方案将为您提供强有力的支持，助力业务的连续性和稳定性。

业务流程图

全栈式风控引擎的请求流程涉及到业务客户端、腾讯云API网关以及全栈式风控引擎服务端。这三者之间的交互调用时序图如下所示：



业务客户端（后端）容灾

当请求“全栈式风控引擎接口”发生异常时，业务客户端需要进行相应的业务异常处理，以确保不会因为接口返回结构异常、请求超时或服务未响应而阻碍业务流程。以下是几种异常返回的情况，**业务客户端需要进行容灾处理**：

- 接口返回内部错误，Code 错误码为 `InternalError` 。

```
{
    "Response": {
        "Error": {
            "Code": "InternalError",
            "Message": "An internal error has occurred. Retry your
request, but if the problem persists, contact us."
        },
        "RequestId": "9f347b06-****-****-****-****62ba5bf6"
    }
}
```

- 更多错误码详情，请查看 公共错误码 。

监控和报警

业务服务端可以对异常进行系统监控，以便在全栈风控引擎的服务端接口出现问题，能及时发现并制定相应的应急处理方案。如果错误率超过设定的阈值，应立即触发报警。

异常捕获与重试策略

对于某些类型的错误（如网络错误或服务暂时不可用），客户端可以捕获这些错误并根据实际业务需求实施重试策略。如果在执行完所有重试后，服务端仍无法正常响应，客户端可以选择跳过并返回默认结果，同时记录下这些问题数据，以便事后进行分析和处理。
例如，如果第一次请求失败，客户端可以选择在等待一段时间后再次尝试。需要注意的是，重试的次数和间隔应该有明确的限制，以避免陷入无限循环。

> **注意：**
> 以下伪代码仅供参考，请以实际业务代码为准。

```
def handle_response(response, retry_count=3, retry_interval=5):
    if "Error" in response["Response"]:
        error = response["Response"]["Error"]
        code = error["Code"]
        message = error["Message"]
        request_id = response["Response"]["RequestId"]

        logging.error(f"请求失败，错误代码：{code}，错误信息：{message}，请求
ID：{request_id}")

        if code == "InternalError":
                            # 对于内部错误，你可能需要重试请求
            print("发生了内部错误，正在重试请求...")
            for i in range(retry_count):
                print(f"重试次数：{i+1}")
                # 等待一段时间
                time.sleep(retry_interval)
                 # 重试请求的代码...
                                    # 如果重试成功，返回结果并退出函数
                                    # 如果重试失败，继续循环
                            # 如果所有重试都失败，返回默认值
            return 'pass'        elif code == "其他错误码":
            # 根据实际情况进行容灾处理
        else:
            print("发生了未知错误")
            return 'pass'
```

```
    else:
        # 如果没有发生错误，就正常处理响应
            # 处理响应的代码...
```