苹果为什么要禁止JSPatch等热更新技术?

最近不少 iOS 开发者都收到了苹果的警告邮件，在邮件中，苹果称开发者使用了动态代码更新技术，要求开发者删除相关代码，并重新提交一个新的 App 版本以供审核。

邮件原文翻译如下:

> 尊敬的开发者，
>
> 您的应用，扩展程序和/或链接框架似乎包含明确设计的代码，能够在应用审核批准后更改应用的行为或功能，这不符合 Apple 开发人员计划许可协议和应用的第 3.3.2 节商店审查指南 2.5.2。此代码与远程资源相结合，可以帮助对应用程序的行为进行重大更改，与最初对 App Store 进行审核相比。虽然

> 当前可能不使用此功能，但它可能会加载私有框架，私有方法，并支持未来的功能更改。
>
> 这包括将任意参数传递给动态方法（如 dlopen()、dlsym()、respondingToSelector:、performSelector:、method_exchangeImplementations()）和运行远程脚本以便更改应用程序行为或调用 API 的任何代码，下载的脚本。即使远程资源不是故意恶意的，它也可能很容易被劫持通过中间人（MiTM）攻击，这可能对您的应用程序的用户造成严重的安全漏洞。
>
> 请对您的应用执行深入审核，并删除符合上述功能的任何代码、框架或 SDK，然后再提交下一个更新以供审核。

而上文提到的苹果开发者协议 3.3.2 节具体内容如下:

> “一个应用程序不应该下载或安装任何可执行代码。解释执行的代码可以在应用内使用，如果所有的脚本、代码，和解释器都被打包在应用内而没有被下载。前述内容的唯一的例外在于下载的脚本和代码使用了 Apple 内置的 WebKit 框架或 JavaScriptCore，并且对应的脚本或代码并没有改变这个应用提供功能和特性的主要目的，与提交到 App Store 的版本以及相应的宣传描述相符。

苹果警告邮件波及甚广，在 GitHub 的 JSPatch 和 react-native 项目下非常多的 iOS 开发者在讨论这件事，也因为苹果并没有具体指明，导致了大家的各种猜测。今天，React-Native 官方已经辟谣，确认不是 React-Native 的问题，而是 JSPatch / Rollout 的问题，而 JSPatch 作者 bang 也发文对此进行了回应，并表示:

> 动态化还是处于灰色地带，严格来说 RN 是不符合规则的，但还是被允许，只要不给苹果添麻烦，苹果就不会管，JSPatch 因为前面提到的两点风险被管

> 了，怎样做到使用并不给苹果添麻烦呢？
>
> 1. 减少使用人数，降低影响面；
> 2. 禁止 SDK 接入；
> 3. 接入保证传输安全和只用于修复 bug。
>
> 第一点警告邮件和代码检查使得使用门槛变高了，显然会减少使用人数。第二点第三点只要有一个平台来管控，是可以做到的，可能的话希望能跟苹果审核团队协商。

至此，我们不禁要思考，苹果是不是完全禁止热更新技术了？为什么要这么做？对于这几个问题，白鹭引擎架构师王泽撰文分享了他的观点，并阐述了此事件是否会对手游开发产生影响，具体如下：

苹果是不是完全禁止了热更新技术？

并不是，目前为止收到警告邮件的开发者绝大部分使用了 JS-Patch 或 Rollout 类库，剩下未直接使用这些类库的开发者，目前初步估计很可能是在集成的第三方 SDK 中使用了上述框架。而未采用上述框架的热更新技术，目前为止并未收到影响。而绝大部分游戏引擎由于并没有调用这些类库，也自然没有受到影响。

当然，后续事态会不会进一步扩大，还需要看苹果接下来的策略。但是笔者认为，游戏中的热更新技术并不会受到苹果的禁止，作为一名技术人员，我们不讨论产品、商业等问题，只从技术角度来看，为什么 JSPatch 苹果认为是不允许的，而游戏引擎的热更新技术，苹果目前认为是可以的。

苹果为什么要禁止 JSPatch 等热更新技术？

JSPatch 的原理是，开发者编写 JavaScript 代码，利用苹果内置的 JavaScriptCore.Framework 执行，以实现热更新功能。这一点看似也符合标准，但是在技术上，存在着重大安全隐患，参考 JSPatch 的业务逻辑：

```
require('UIView')var view = UIView.alloc().init()view.setBackgroundColor
```

简单理解，JSPatch 可以理解为所有的 Objective-C 的 API 进行了映射，允许开发者在 JS 端调用任意原生代码，这显然是极其危险的。假设这段代码是通过热更新技术下载执行的，如果在中间存在黑客，把这段代码动态替换掉，比如修改为获取用户通讯录并上传到黑客的服务器，就会造成重大的安全问题。

为什么游戏热更新技术可以被理解为是安全的?

与 JSPatch 不同的是，游戏热更新技术主要的实现方式是把动态脚本下载之后，让动态脚本调用游戏引擎提供的接口实现缺陷修复。与 JSPatch 不同的是，动态脚本并不能任意调用全部原生代码，而是只能根据游戏引擎提供的接口调用相关功能。在这个过程中，游戏引擎的原生端作为一个安全沙箱，提供了一个安全的保护层，只要游戏引擎不要对外提供获取通讯录的接口，黑客就无法通过替换动态脚本的方式获取用户的隐私资料。进而可以被认为是安全的，自然不在苹果的禁止范围内。

小结

1. 苹果认为热更新技术容易被黑客利用，造成重大安全问题。在官方警告邮件中，也是在进行如此描述。
2. JSPatch 这种基于反射，允许获取全部系统接口的方式，确实存在着一定的安全风险。虽然可以通过安全策略去防范，但是苹果决定一刀切，严格禁止。
3. 游戏引擎由于不是利用反射机制实现的热更新，不能获取全部系统接口，所以目前苹果认为是安全的，无需警告。

最后，笔者作为一名技术人员，以上所有内容，都是基于客观的技术层面进行讨论，请勿上升到“商业模式”、“生态闭环”等层面的“高度”，欢迎技术层面的交流讨论。

本文文字及图片出自 微信公众号

分享这篇文章：

相关文章：

1. 苹果自研ARM处理器 恰似乔布斯当年转投英特尔
2. Mac迁移至苹果自主芯片 全面Arm架构时代或将到来
3. “我写代码赚的钱，凭啥让苹果白拿30%？”
4. 苹果祭出大招：史上最强 Mac 发布，iPad OS 惊艳问世
5. WWDC19 苹果宣布全新 UI 框架 SwiftUI
6. 别和苹果技术顾问斗嘴
7. 苹果向中国开发者宣战了，两万余APP遭下架
8. 一位JSPatch开发者谈来自苹果的警告
9. swift语言之父已确认被电动汽车公司特斯拉挖走！
10. 苹果工程师讲述初代iPhone开发经历：不知道是啥

你的反应是：

请关注我们：

JSPATCH 苹果