SECURITY CHALLENGES OF HYBRID MOBILE APPLICATIONS
BY MIKKO SAARIO

Twitter: @midisFI
(http://twitter.com/midisfi)
THIS IS NOT YOUR UNCLE OLAF'S
"OWASP TOP 10 MOBILE SECURITY"
TALK
AGENDA

- Hybrid Mobile Apps - what are they?
- Hybrid environments mix HTML+JS with "Native" code
- Hybrid environments mix managed code with unmanaged code
- Some difference between a "browser" and a "webview"
- Some ways of 'leaking info by accident'
- Bunch of technical details (small but perhaps important)
ME

- Nokia (since 2001)
- Security Manager for Sales & Marketing services
  - Nokia.com etc.
- Past:
  - Ovi Store, Maps.nokia.com
  - Nokia Account
  - N-Gage v2
- Founded OWASP Helsinki in 2006
- Motto: "the more you learn, the less you seem to know"
- More of a Defender & Breaker than a Builder
POLL
"DISCLAIMER"

- Examples are mainly for Windows Phone 8
- With some Qt (v4.8.x)/QML thrown in (Qt 5 is out, but...)
- Android and iPhone guys - just be cool
  (it's probably not that different)
TRADITIONAL AIR GAP
A PRETTY EFFECTIVE SECURITY CONTROL
WEB HAD NO ACCESS TO DEVICE API
SECURITY USED TO BE
SIGNING APPS WITH
CAPABILITIES

Permission, privilege, you name it...
So, installation handled the "rights"
BUFFER OVERFLOWS
SMS OF DEATH
MALWARE
AND SO FORTH
AND STILL IS...
THEY DIDN'T DISAPPEAR

There is just more of everything
+ all the legacy code
BRIDGING THE AIR GAP

NATIVE + WEB

HYBRID
import QtWebKit 1.0

WebView {
    url: "https://www.owasp.org"
    preferredWidth: 490
    preferredHeight: 400
    scale: 0.5
    smooth: false
}
A HYBRID ENVIRONMENT

- Native mobile apps utilize Web technologies inside the app
  - HTML, CSS and JavaScript embedded in / utilized by native code (C#, VB, objective-C, C++, "java")
- Typically utilizing

  ```
  /\(ui)?Web\(View|Browser)/g class
  ```

  For the rest of us: a "WebBrowser", "uiWebView", or just plain "WebView"

- Windows Phone 8: WebBrowser control (Windows 8 is very similar)
- Rendering engine without the "chrome" (Browser UI)
- According to Microsoft, nearly all Top 50 WP7 apps used the WebBrowser Control (a.k.a. WebView)
COMMON

- Qt/QML [Qt Quick] multiplatform
- Widgets (W3C)
- Android
- Mac OS X, iOS
- Windows OS / Phone
- Apache Cordova / PhoneGap
In native XAML, expose an interface to JS

```xml
<phone:WebBrowser
    ScriptNotify="alert_ScriptNotify"
    IsScriptEnabled="True"
/>
```

JavaScript calls the parent native app

```javascript
function AlertSilverlight(data)
{
    window.external.notify(data);
}
AlertSilverlight(1);
```

Listener picks it up and executes

```csharp
private void alert_ScriptNotify(object sender, NotifyEventArgs e)
{
    MessageBox.Show(e.Value);
}
```

Code: WinPhone 8 C#/Silverlight/XAML/JavaScript
AND IN QML

In this case, JS can be included in QML (inline or external)

```javascript
WebView {
    javascriptWindowObjects: QtObject {
        WebView.windowObjectName: "qmlJS" // expose from native to JS

        function qmlFunc() {
            // native QML or JavaScript code here
            console.log("Hello from QML!");
        }
    }
}

// e.g. a JS file embedded in the QML project build can refer to an ext JS
Qt.include("http://mydomain/import.js")

JavaScript calls the exposed method

window.qmlJS.qmlFunc();
```

Code: QML(Qt markup) + HTML/JavaScript
Inside QML, imported/inline JavaScript is NOT sandboxed

```javascript
import "http://localhost/mikko/qt/javascript/unsafe.js" as ExtScript
// Beware of evil content
function fakeFunc() {
    var xhr = new XMLHttpRequest;
    xhr.open("GET", "file:///c:/setup.log"); // Read files on the local disk
    xhr.onreadystatechange = function() {
        if (xhr.readyState == XMLHttpRequest.DONE) {
            var a = xhr.responseText;
            console.log(a); // Send someplace else
        }
    }
    xhr.send();
}
// Invoke
ExtScript.fakeFunc();
```

WebView JavaScript is sandboxed

```javascript
import QtWebKit 1.0
WebView {
    url: "https://www.owasp.org"
}
```
**QML XHR RESULT**

**Local file was read via XHR**

```javascript
[InstallShield Silent]
Version=v7.00
File=Log File
[ResponseResult]
resultCode=0
[Application]
Name=RICOH Media Driver ver.2.10.00.04
Version=2.10.00.04
Company=RICOH
Lang=0009
```

**Need to be EXTRA careful with untrusted JavaScript/QML inside .qml files**
IDENTIFYING NATIVE METHODS IN JS

JavaScript is looking for native QML methods

```javascript
for (var i in window) {
    document.write("Window property(" + i + "): " + window[i]);
}
```

Where we see:

- `Window property(qmlJS): QObject_QML_0(name = ")

And then iterating through window.qmlJS:

- `Window property(qmlFunc()): undefined`
- `Window property(qmlFunc2(QVariant)): undefined`
- `Window property(qmlFunc3(QVariant,QVariant)): undefined`

Code: QML/Qt + JavaScript
Think whether the exposed object enables the web environment access things that shouldn't be open, and whether the web content loaded by that web page comes from a trusted source. In general, when exposing native QObjects that give the web environment access to private information or to functionality that's potentially harmful to the client, such exposure should be balanced by limiting the web page's access to trusted URLs only with HTTPS, and by utilizing other measures as part of a security strategy.
OWASP TOP TEN
YES, IT'S ALL VERY VALID
BOTH OF THEM
UI CONTROLS

- A lot of the usual UI controls are missing
- E.g. Windows Phone will just fail self-signed or untrusted SSL cert (need to manually import)
- No SSL "lock" visible & no mixed mode (http + https content) in WP
- Warnings (WP gives SSL warning, but prevents from loading page), popups etc.
DO NOT TRACK

Windows Phone 8

- IE 10 "Do Not Track" setting does not apply in WP8 WebBrowser - NO user control over this
DEMO
SAME ORIGIN POLICY

Notes for Windows Phone 8

- Content loaded from isolated storage is not restricted by SOP (file:///)
- Content created via NavigateToString is not restricted by SOP

```javascript
var html = "<html><script>...Something...</script></html>

webBrowser1.NavigateToString(html);
```

- JavaScript called via native InvokeScript can be loaded from any domain
- Some differences on desktop vs mobile Silverlight
Qt/QML

- Qt WebView has SOP limits on file://

```html
Origin: file://
```

- Qt WebView has normal SOP limits for JavaScript

```html
<html>  // This runs in the WebKit WebView
<script>xhr();</script>
</html>  // Normal cross-origin (CORS) rules apply
```

- Qt QML inline JavaScript / XMLHttpRequest does not enforce the same origin policy

```javascript
// Script source: http://some_site/some.js
// This is inside filename.qml and inline JavaScript
xhr.open("GET", "http://another_site/?&id=123456");
```

- Imported inline JS has 'full access'

```javascript
import "http://untrusted.com/js/unsafe.js" as ExtScript
```
EVAL IS STILL EVIL
ARE YOU EVAL'ING SOME UNTRUSTED INPUT?

Or native:

// QML
WebviewID.evaluateJavaScript("something")

// Qt
QString js = input;
mainFrame()->evaluateJavaScript(something);

// C#
WP_browserInvokeScript("eval", "something");
UTILIZING C/C++
WINDOWS PHONE 8 - "GOING NATIVE"

- Most likely usages are: to redeploy existing code & squeeze out max performance
- I have deliberate use of non-secure versions of functions
- E.g. "strcpy" vs "strcpy_s" (Microsoft)

// Call from managed C# code to C++ (file.cs)
OWASP_WP_RT_CPP.StringCharacterCounter sccMain = new OWASP_WP_RT_CPP.StringCharacterCounter();
answer.Text = sccMain.GetLength(answer_text.Text).ToString() + " characters were found!";

// Native C++ (file.cpp)
unsigned int StringCharacterCounter::GetLength(String^ strToParse)
{
    std::wstring stlString = strToParse->Data();
    return stlString.length();
}
The above killed Nokia N9 (MeeGo) Twitter client (Qt C++)

we can't know how much space we need to allocate...
...for this pathological string we are generating
two glyphs for each character.
Demo
INJECTIONS

- SQL injection

```cpp
QSqlQuery q("SELECT something FROM " + untrusted_data);
```

- XML Query / XPath injection
  Dynamic queries with user-supplied input could leave you vulnerable in Qt
  Use parameter binding for XML & SQL

- XSS special cases (Qt rich text)
  QLabel default setting is QLabel::AutoText ['MIME type sniffing']

  ```cpp
  QLabel::AutoText
  ```

Output can be manipulated via a limited set of supported HTML tags. Encode output, or use:

```cpp
QLabel::PlainText
```
Palveluun ei saada yhteyttä

Almost done!
Tap 'Submit' to save your review. Once saved, your review cannot be edited.

Great game
★ ★ ★ ★ ★ by midis | v-1 | N9

Fun to play”>

Edit review  Submit
DENIAL OF SERVICE

- Isolated storage on Windows Phone has no size restrictions - single app may consume large amount of space
- Also, HTML5 AppCache can easily consume disk space by caching content locally

Content-type: text/cache-manifest

- XML query resource exhaustion attacks in Qt
- Qt 4.8.x also has various JS crash cases in very large arrays (Qt 5.0 & newer WebKit more robust)
QML JavaScript

- QMLv1 has problems handling very large JavaScript arrays
- Crashes the app
- Unlikely to be seen in "normal" app use

```javascript
function bigArray() {
    var a = new Array(1000000000); // lbn or something 'very big'
}
```

Compare to e.g. Firefox, Chrome (no effect)
Memory Leak in Nokia Suite (Qt)
QML DoS Demo
Capabilities

- Platform configurations are your 'enhanced capabilities'
- Your mileage will vary greatly
- Each framework has their own options and defaults
/* Qt default: true */
QWebSettings::JavascriptEnabled

QWebSettings::JavascriptCanOpenWindows
QWebSettings::JavascriptCanAccessClipboard

<!-- Windows Phone 8 default: disabled -->
<Grid x:Name="LayoutRoot">
    <phone:WebBrowser Name="mybrowser"
        IsScriptEnabled="True"
        Source="index.html"
        ScriptNotify="pokeMyScript" />
</Grid>
PLUGINS

Java, Flash and other plugins (NPAPI, Qt plugins)

/* Qt */
QWebSettings::JavaEnabled
QWebSettings::PluginsEnabled

/* No plugins in WP8 */
WebKit XSS Auditor

If you have it, why not use it?

```cpp
/* Control the Qt WebKit XSS auditor (not available in QML/Qt Quick 1) */
QWebSettings::XSSAuditingEnabled

QWebView view;
view.settings() -> setAttribute(QWebSettings::XSSAuditingEnabled, false);
view.load(QUrl("http://yourdomain/xssaudit.php?xss=<script>alert(1)</script>"));
```
HEADERS

Security header [overall HTML5] support finally becoming up-to-date
With WP8 + IE10 and Qt 5.0 + WebKit2

- Content Security Policy
- X-XSS-Protection
- Strict Transport Security for SSL
- X-Frame-Options to prevent framing
AND SO ON

- File access

```cpp
QWebSettings::LocalContentCanAccessRemoteUrls
QWebSettings::LocalContentCanAccessFileUrls

Can **file:** access **http:**
Can **qrc:** access **file:**
```

- Misc.

```cpp
QWebSettings::PrivateBrowsingEnabled
QWebSettings::DeveloperExtrasEnabled

With this setting, any site can access user's location
```

```cpp
// Windows Phone
IsGeolocationEnabled = "true"
```
DEMO

Windows Phone 8 native methods
Unlike Windows Phone, Qt requires you to define to which frame native methods are published.

As seen in the demo, any iframe in WP can access exposed native methods.

"Mind your security boundary"

WP does not advertise available methods (it just takes a string).

```c++
// Only myFrame has access to the native object
QWebFrame *myFrame = myWebPage->mainFrame();
myFrame->addToJavaScriptWindowObject("myObject", theObject);
```
QML XHR

You know this

```javascript
var xhr = new XMLHttpRequest();
...
xhr.open("GET", "http://target");
xhr.setRequestHeader("Whatever","ValueHere");
xhr.send();
```

Tried this instead

```javascript
xhr.setRequestHeader("Referer","http://somedomain.com");
```

Obviously fails!

Ok how about this?

```javascript
xhr.setRequestHeader("Origin","http://www.google.fi");
```

```
ORIGIN: http://www.google.fi
Connection: Keep-Alive
Accept-Encoding: gzip
Accept-Language: en-US,*
User-Agent: Mozilla/5.0
```
A Little Tweaking Later

xhr.setRequestHeader("anything","buubaa\nReferer: http://www.google.fi/yourpath");

ANYTHING: buubaa
Referer: http://www.google.fi/yourpath
Connection: Keep-Alive
Accept-Encoding: gzip
Accept-Language: en-US,*
User-Agent: Mozilla/5.0

Just needed a newline there ("\n", "\012", "\x0a")

To add "restricted" headers
Call it the 'Power of C++'

The grey area between the mainstream browser and your own apps
WebView uses WebKit - didn't have it

- Changing the Host header is not supported as Qt constructs only a single HTTP request
- This is not sandboxed code
- Currently classified as a bug (https://bugreports.qt-project.org/browse/QTBUG-27570)
SUMMARY

Usual secure development practices apply (OWASP etc.)
Hybrid techs are catching up, security-wise
User has less or no control
This is not "a browser" - updates etc. are lagging
HTML5 & new platforms are bridging the air gap
Controls such as Same Origin Policy may function differently
Perform robust checks when exposing sensitive platform features (camera, location)
QUESTIONS?

**OWASP APPSEC APAC**
(HTTPS://TWITTER.COM/SEARCH/OWASP)
MOBILE SECURITY
(HTTPS://TWITTER.COM/SEARCH/OWASP)

- **Owasp_Ven**
  (https://twitter.com/intent/user?

- **proggitarticles**
  (https://twitter.com/intent/user?
  screen_name=proggitarticles) Why you shouldn't use the OWASP Top 10 as a list of software security requirements:

- **biosshadow**
  (https://twitter.com/intent/user?
  screen_name=biosshadow) Here a bit early for the OWASP meet up. Anyone else here early. Also being stuck in a wheelchair.

- **clerkendweller**
  (https://twitter.com/intent/user?
  screen_name=clerkendweller) Vote for #owasp.