Luca De Fulgentis
Offensive Security Adept
Chief Technology Officer at Secure Network
Consuming brain power with InfoSec since 2001
INTRODUCING THE FREAK SHOW
Install apps from Windows Phone Store

Extract binaries from unlocked device

Decompile binaries

Analyze decompiled source code

~60 XAPs

~160 AppXs

Killer apps

Statistics for MTT 2015

Catalog of insecure APIs

Amazing freaks
freaks* in terms of vulnerable examples of code

* no toy code, real world examples only
SELF DEFENDING APPS
“A lack of binary protections results in a mobile app that can be analyzed, reverse-engineered, and modified by an adversary in rapid fashion”

1 https://www.owasp.org/index.php/Mobile_Top_10_2014-M10
A lack of binary protections results in a mobile app that can be analyzed, reverse-engineered, and modified by an adversary in rapid fashion.

1 https://www.owasp.org/index.php/Mobile_Top_10_2014-M10
A lack of binary protections results in a mobile app that can be analyzed, reverse-engineered, and modified by an adversary in rapid fashion.

1 https://www.owasp.org/index.php/Mobile_Top_10_2014-M10
A lack of binary protections results in a mobile app that can be analyzed, reverse-engineered, and modified by an adversary in rapid fashion.¹

A lack of binary protections results in a mobile app that can be analyzed, reverse-engineered, and modified by an adversary in rapid fashion.

- Lack of resources integrity verification and jailbreak detection mechanisms
- Lack of anti-debugging mechanisms and runtime-tampering detection
- Lack of code obfuscation and code encryption
- Intellectual property theft

1 https://www.owasp.org/index.php/Mobile_Top_10_2014-M10
A lack of binary protections results in a mobile app that can be analyzed, reverse-engineered, and modified by an adversary in rapid fashion.

~95%* of analyzed apps lack proper binary protections

* 223 out of 235 assessed apps
private void CordovaBrowser_Loaded(object sender, RoutedEventArgs e)
{
    this.resourceStreamInformation = Application.GetResourceStream(new Uri(Resource1.WWWPath, UriKind.Relative));
    // 
    string encodedPassword = Resource1.EncodedPassword;

    this.strPasswordDecodingSecond = CordovaView.Base64Decode(CordovaView.Base64Decode(encodedPassword.Substring(0, encodedPassword.Length - 10)));
    this.passwordLength = this.strPasswordDecodingSecond.Length;
    this.stream = this.resourceStreamInformation.Stream;
    this.filebytes = Convert.FromBase64String(CordovaView.StreamToString(this.stream));
    this.Unzip(new MemoryStream(this.Decrypt(this.filebytes, this.strPasswordDecodingSecond, this.passwordLength)));

    this.RetrievePage(); // CordovaView.uri setting
    this.CordovaBrowser.Navigate(CordovaView.uri); // Navigate unzipped app index.html page
}
Weak custom code encryption

```csharp
private void CordovaBrowser_Loaded(object sender, RoutedEventArgs e)
{
    this.resourceStreamInformation = Application.GetResourceStream(new Uri(Resource1.WWWPath, UriKind.Relative));

    // [...]
    string encodedPassword = Resource1.EncodedPassword;

    this.strPasswordDecodingSecond = CordovaView.Base64Decode(CordovaView.Base64Decode(encodedPassword.Substring(0, encodedPassword.Length - 10)));

    this.passwordLength = this.strPasswordDecodingSecond.Length;
    this.stream = this.resourceStreamInformation.Stream;
    this.filebytes = Convert.FromBase64String(CordovaView.StreamToString(this.stream));

    this.Unzip(new MemoryStream(this.Decrypt(this.filebytes, this.strPasswordDecodingSecond, this.passwordLength)));

    this.RetrievePage(); // CordovaView.uri setting
    this.CordovaBrowser.Navigate(CordovaView.uri); // Navigate unzipped app index.html page
}
```

Unzip() calls the UnzipAndSaveFiles() method
public void UnzipAndSaveFiles(Stream stream) {
    // [...] 
    using (ZipInputStream zipInputStream = new ZipInputStream(stream)) {
        storeForApplication.CreateDirectory(Resource1.WWWDirectory);
        ZipEntry nextEntry;
        while ((nextEntry = zipInputStream.GetNextEntry()) != null) {
            // [...] 
            str1 = Path.Combine(Resource1.WWWDirectory, strArray[index]);
            if (!storeForApplication.DirectoryExists(str1)) {
                storeForApplication.CreateDirectory(str1);
            }
        }
    }
}

unzipped file content is saved in the SANDBOX
On apps encryption

- Windows Phone Store apps are downloaded as encrypted files
  - Packages are then decrypted during the installation phase

- A privileged access to the file system allows binaries extraction
  - Apps’ bytecode can be easily decompiled with publicly available utilities
    - ILSpy, .NET Reflector and JetBrains dotPeek are examples of available decompilers

- Code obfuscation and encryption represent solid strategies to mitigate
  - Intellectual theft
  - App behavior analysis while increasing malicious users effort
## Comparison by feature by package format

### In summary...

<table>
<thead>
<tr>
<th>Feature</th>
<th>XAP Phone</th>
<th>XAP 8.1 Phone</th>
<th>AppX Phone</th>
<th>AppX Windows</th>
</tr>
</thead>
<tbody>
<tr>
<td>Platform Targeting</td>
<td>7.x and later</td>
<td>8.1 and later</td>
<td>8.1 and later</td>
<td>8.0 and later</td>
</tr>
<tr>
<td>Package Encryption</td>
<td>Yes</td>
<td>Yes</td>
<td>No, not yet.</td>
<td>No, not yet.</td>
</tr>
<tr>
<td>Package Bundling</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Debug Package Signing</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Differential Download/Update</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Application File Single Instancing</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Formal Versioning Requirements</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>External Volume (SD) App Installation</td>
<td>Yes on 8.1</td>
<td>Yes</td>
<td>Yes</td>
<td>No, not yet.</td>
</tr>
</tbody>
</table>

* Slide taken from a Microsoft’s Build presentation*
Secure mindset

- Apps should be securely designed to mitigate binary attacks
  - OWASP RE and Code Modification Prevention Project provides secure design principles
- Adopt tools such as dotFuscator and ConfuserEx to protect binaries
- Certificate pinning should be implemented as well – see later
- Binary protections simply mitigate, but do not solve, binary attacks
  - They represent a further layer of security (obscurity?)
  - Consider that every protection can be bypassed with proper time and motivation
  - The idea is raising the bar to increase attacker’s effort
DATA TRANSPORT SECURITY
Transport security

- Confidentiality in the app-to-backend communication
  - http-based communication ∈ WP-supported mechanisms

- Common issues
  - Communication over an unencrypted channel – e.g., http instead of https → MiTM attacks
  - Communication over a poorly encrypted channel – e.g., use of weak encryption mechanisms
  - Issues related to digital certificates handling
## Hunting for transport issues

<table>
<thead>
<tr>
<th>Category</th>
<th>Namespaces</th>
<th>Classes, Methods or Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>GetAsync()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PostAsync()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PutAsync()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GetStreamAsync()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GetStringAsync()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SendAsync()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SocketProtectionLevel.PlainSocket - property</td>
</tr>
<tr>
<td></td>
<td></td>
<td>StreamSocket.UpgradeToSslAsync()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DatagramSocket.ConnectAsync()</td>
</tr>
</tbody>
</table>
Hunting for transport issues

<table>
<thead>
<tr>
<th>Category</th>
<th>Namespaces</th>
<th>Classes, Methods or Properties</th>
</tr>
</thead>
</table>
| Web            | Microsoft.Phone.Controls                        | WebBrowser.Navigate()  
                              WebBrowser.Source property|
|                | Windows.UI.Xaml.Controls                       | WebView.Navigate()  
                              WebView.Source property|
|                | Microsoft.Phone.Tasks                          | WebBrowserTask.Uri property |
|                | Windows.System                                 | Launcher.LaunchUriAsync(uri) |
                              StreamWebSocket.ConnectAsync() – with ws:// uri scheme |
# Hunting for transport issues

<table>
<thead>
<tr>
<th>Category</th>
<th>Namespaces</th>
<th>Classes, Methods or Properties</th>
</tr>
</thead>
</table>
| XAML Object Element Usage        | -                                 | «Source» property for WebBrowser and WebView  
|                                  |                                   | «uri» property for WebBrowserTask  
|                                  |                                   | “NavigateUri” for HyperlinkButton                                                             |
| Push Notifications               | Microsoft.Phone.Notivation        | HttpNotificationChannel(string)                                                                |

Windows.Web.Syndication classes/methods should be reviewed as well
Video
Attacking unencrypted communication – take I
Hijacking CNN’s app news
Video
Attacking unencrypted communication – take II
Stealing Instagram app authorization token
Phishing has never been so easy

<phone:PhoneApplicationPage
    x:Class="App.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
    xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    mc:Ignorable="d"
    d:DesignWidth="480" d:DesignHeight="768"
    FontFamily="{StaticResource PhoneFontFamilyNormal}"
    FontSize="{StaticResource PhoneFontSizeNormal}"
    Foreground="{StaticResource PhoneForegroundBrush}"
    SupportedOrientations="PortraitOrLandscape" Orientation="Portrait"
    shell:SystemTray.IsVisible="True">
    <phone:WebBrowser
        Height="Auto" IsScriptEnabled="true"
        HorizontalAlignment="Stretch" Name="WONT-SAY"
        VerticalAlignment="Stretch" Width="Auto"
        Margin="-12,0,0,0"
        Grid.ColumnSpan="2"
        Source="http://m.WONT-SAY.com/login1.html?continua=true" />
</phone:PhoneApplicationPage>

an attacker can replace the login page with a malicious one
Phishing has never been so easy

```csharp
public CordovaView()
{
    this.InitializeComponent();
    if (DesignerProperties.IsInDesignTool)
        return;

    // [...]
    if (this.configHandler.ContentSrc != null)
        this.StartPageUri = !Uri.IsWellFormedUriString(this.configHandler.ContentSrc, UriKind.Absolute) ?
            new Uri(CordovaView.AppRoot + "www/" + this.configHandler.ContentSrc, UriKind.Relative) :
            new Uri(this.configHandler.ContentSrc, UriKind.Absolute);
    // [...]

<!-- ... -->

<link rel="stylesheet" href="style.css" />
<link rel="stylesheet" href="style-icons.css" />

<script type="text/javascript" src="http://maps.googleapis.com/maps/api/js?v=3.4&key=ABCDE[...]&libraries=places"></script>
<!-- ... -->
```

www/index.html loads a remote JS via http == XSS
Video

Attacking unencrypted communication – take III
The *Italian job* (or how to manipulate a banking app UI)
<Button x:Name="uxSignUpButton" Margin="12,180,0,240" VerticalAlignment="Bottom" Style="{StaticResource LinkedInPhoneButton}"
    toolKit:TiltEffect.IsTiltEnabled="True" Tag="http://www.linkedin.com/reg/join" Tap="openHyperlink_Click">
    <StackPanel Orientation="Horizontal">
        <TextBlock Text="{Binding Path=LocalizedResources.SignIn_SignUp1, Source={StaticResource LocalizedStrings}}"
            Margin="0,0,0,0" HorizontalAlignment="Left" VerticalAlignment="Top" Style="{StaticResource LinkedInTextAccentStyle}" />
        <TextBlock Text="{Binding Path=LocalizedResources.SignIn_SignUp2, Source={StaticResource LocalizedStrings}}"
            Margin="6,0,0,0" HorizontalAlignment="Left" VerticalAlignment="Top" Style="{StaticResource PhoneTextSubtleStyle}" />
    </StackPanel>
</Button>

private void openHyperlink_Click(object sender, RoutedEventArgs e) {
    Button button = sender as Button;
    if (button == null)
        return;

    string uriString = button.Tag.ToString();
    if (string.IsNullOrEmpty(uriString) || uriString.Length < 8)
        return;

    if (!uriString.ToLower().StartsWith("http"))
        uriString = "http://" + uriString;

    try
    {
        this._wtask.Uri = new Uri(uriString);
        this._wtask.Show();
    }
}
private void aj(object A_0, RoutedEventArgs A_1)
{
    // [...]
    switch (num2 == num3)
    {
        case true:
            int num4 = 0;
            num4 = 0;
            if (num4 == 0)
            {
                num4 = 1;
                if (num4 == 0);
            }
            FbNavService.Current.ShowTask((CustomLauncherBase) new CustomWebBrowserTask()
            {
                Url = "http://m.facebook.com/r.php"
            });
    }
}

Somewhere in the code..

Clear-text subscription
public static async void GoToForgotPasswordWebPage()
{
    bool tContinue = NavigationHelper.ShowExitMessageDialog();
    string urlAdjusted = TrackingManager.Instance.WrapUriForPaidAppTracking(
        EbaySettings.Instance.CurrentSite.ForgotPasswordSite, "forgotpassword-core"
    );
}

public string WrapUriForPaidAppTracking(string inURL, string inMfe)
{
    StringBuilder stringBuilder = new StringBuilder();
    stringBuilder.Append(
        string.Format("http://rover.ebay.com/rover/\{0\}/\{1\}/\{2\}?mfe=\{3\}&mpre=\{4\}&mpt=\{5\}". "1", 
        this.GetRoverId(EbaySettings.Instance.IsoCodeFromSiteId(EbaySettings.Instance.CurrentSiteId)), "4", inURL, this.MptCacheBusterValue()));
}

What’s your pretext today?
Secure coding tips

- SSL/TLS everywhere!
  - Just adopt proper the `https://` scheme when using `Uri()` object
- WP 8.0 automagically discards invalid certificates
  - No programmatic way to disable the behavior
- WP 8.1 introduced the `IgnorableServerCertificateErrors` class
  - Selective ignore of certificate errors – not all exceptions can be discarded
- Are we completely safe from MiTM attacks? **Nope!**
  - An attacker can still hack into a Certificate Authority (CA) and forge valid certificates
  - An attacker can induce the victim to install a malicious certificate
- So What?
Implementing certificate pinning

- Windows Phone 8.0 apps require 3rd parties libraries (e.g., EldoS SecureBlackbox)
- Windows Phone 8.1 provides the `StreamSocket.Information` that returns the `StreamSocketInformation` object
  - `StreamSocketInformation.ServerCertificate` allows getting the remote server digital certificate

```csharp
public async void verifyCertificate(string url) {
    HostName host = new HostName(url);
    StreamSocket socket = new StreamSocket();

    await socket.ConnectAsync(host, "443");
    await socket.UpgradeToSslAsync(SocketProtectionLevel.Ssl, host);

    var cert = socket.Information.ServerCertificate;
    checkCertEntries(cert));
}
```

1 http://www.slideshare.net/iazza/certificate-pinning-in-mobile-applicationsprosconsv10
DATA STORAGE SECURITY
Device disk encryption

BitLocker disk encryption (AES 128) is supported since WP 8
BitLocker is disabled by default
It can be enabled via Exchange ActiveSync policy `RequiredDeviceEncryption`

Device physical memory attacks allow file system content extraction
sensitive data
should never be stored on device, even if encrypted*

*I know, this may damage the user experience
Storage Locations

File System
- From WP 8.0
  BitLocker encryption technology is supported
- BitLocker is disabled by default

Secure Digital (SD) Cards
- Files are NOT encrypted
- An app can access files if it has previously registered as an handler for that file type (via manifest specification)

Cloud Storage
- Data transport security
- Data confidentiality preservation
### Storage locations and physical paths

<table>
<thead>
<tr>
<th>Locations</th>
<th>Windows Runtime Apps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local data store</td>
<td><code>ApplicationData.Current.LocalFolder</code> - URI - <code>ms-appdata:///local/</code></td>
</tr>
<tr>
<td></td>
<td><code>C:\Data\Users\DefApps\APPDATA\Local\Packages\%packageName%\LocalState</code></td>
</tr>
<tr>
<td>Roaming data store</td>
<td><code>ApplicationData.Current.RoamingFolder</code> - URI - <code>ms-appdata:///roaming/</code></td>
</tr>
<tr>
<td></td>
<td><code>C:\Data\Users\DefApps\APPDATA\Local\Packages\%packageName%\RoamingState</code></td>
</tr>
<tr>
<td>Temporary data store</td>
<td><code>ApplicationData.Current.TemporaryFolder</code> - URI - <code>ms-appdata:///temporary/</code></td>
</tr>
<tr>
<td></td>
<td><code>C:\Data\Users\DefApps\APPDATA\Local\Packages\%packageName%\TempState</code></td>
</tr>
<tr>
<td>Cache data store</td>
<td><code>ApplicationData.Current.LocalCacheFolder</code></td>
</tr>
<tr>
<td></td>
<td><code>C:\Data\Users\DefApps\APPDATA\Local\Packages\%packageName%\LocalCache</code></td>
</tr>
</tbody>
</table>
## Storage locations and physical paths

<table>
<thead>
<tr>
<th>Locations</th>
<th>Windows Runtime Apps</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>URI: <code>ms-appx://</code> or <code>ms-appx-web://</code></td>
</tr>
<tr>
<td></td>
<td><code>C:\Data\SharedData\PhoneTools\AppxLayouts\{GUID}\</code></td>
</tr>
<tr>
<td>SD Card</td>
<td>KnownFolders.RemovableDevices</td>
</tr>
<tr>
<td>Cache data store</td>
<td>ApplicationData.Current.LocalCacheFolder</td>
</tr>
<tr>
<td></td>
<td><code>C:\Data\Users\DefApps\APPDATA\Local\Packages\%packageName%\LocalCache</code></td>
</tr>
</tbody>
</table>

Local and Roaming Setting save data in `C:\Data\Users\DefApps\APPDATA\Local\Packages\%packageName%\Settings\settings.dat` which is a Windows NT registry file (REGF) - and NOT encrypted.
## Storage locations and physical paths

<table>
<thead>
<tr>
<th>Locations</th>
<th>Silverlight Apps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application local folder</td>
<td>C:\Data\Users\DefApps\APPDATA{GUID}\Local</td>
</tr>
<tr>
<td>Application Settings</td>
<td>IsolatedStorageSettings.ApplicationSettings</td>
</tr>
<tr>
<td></td>
<td>C:\Data\Users\DefApps\APPDATA{GUID}\Local\ApplicationSetting</td>
</tr>
<tr>
<td></td>
<td>C:\Data\Programs{GUID}\Install</td>
</tr>
<tr>
<td>Cached data</td>
<td>C:\Data\Users\DefApps\APPDATA{GUID}\INetCache</td>
</tr>
<tr>
<td>Cookies</td>
<td>C:\Data\Users\DefApps\APPDATA{GUID}\INetCookies</td>
</tr>
<tr>
<td>SD Card</td>
<td>(read only)</td>
</tr>
</tbody>
</table>
private async void DoLogin()
{
    bool? isChecked = this.checkBoxRicordami.IsChecked;
    if (((isChecked.GetValueOrDefault() ? 0 : (isChecked.HasValue ? 1 : 0)) != 0)
        this.saveCredentials();

    // [...]

private void saveCredentials()
{
    if (!((this.textBlockUsername.Text != "") || !((this.textBlockPassword.Password != "")))
        return;
    this.storageSettingsRememberMe.Remove("Username");
    this.storageSettingsRememberMe.Remove("Password");
    this.storageSettingsRememberMe.Remove("isChecked");
    this.storageSettingsRememberMe.Add("Username", this.textBlockUsername.Text);
    this.storageSettingsRememberMe.Add("Password", this.textBlockPassword.Password);
    this.storageSettingsRememberMe.Add("isChecked", true);
    this.storageSettingsRememberMe.Save();
}
```
private void GetUserCompleted(object sender, EventArgs e)
{
    if (e == null)
    {
        // ...
    }
    else
    {
        NetUserCompletedEventArgs completedEventArgs = (NetUserCompletedEventArgs) e;
        byte[] numArray1 = Crypto.encryptString(completedEventArgs.user.username);
        byte[] numArray2 = Crypto.encryptString(completedEventArgs.user.password);
        this.isolatedStorageSettings.StoreValueForKey("Username", (object) numArray1);
        this.isolatedStorageSettings.StoreValueForKey("Password", (object) numArray2);
        CurrentAppConfig.Instance.User = completedEventArgs.user;
        this.storeCurrentUserStoresPreferences(completedEventArgs.user);
    }
}
```

```
public class Crypto
{
    public static byte[] encryptString(string input)
    {
        return Encoding.UTF8.GetBytes(input);
    }
}
```
encoding is just a data representation, not encryption (at all)
Hunting for insecure data storage

<table>
<thead>
<tr>
<th>Locations</th>
<th>Classes, Methods and Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local folders</td>
<td><strong>StorageFile</strong></td>
</tr>
<tr>
<td></td>
<td>- OpenReadAsync() - OpenAsync()</td>
</tr>
<tr>
<td></td>
<td>- GetFileFromApplicationUriAsync() - GetFileFromPathAsync()</td>
</tr>
<tr>
<td></td>
<td><strong>StorageFolder</strong></td>
</tr>
<tr>
<td></td>
<td>- GetFilesAsync() - GetFileAsync() - CreateFileAsync()</td>
</tr>
<tr>
<td></td>
<td>- IsolatedStorageFile.CreateFile()</td>
</tr>
<tr>
<td>Application or Roaming Settings</td>
<td><strong>IsolatedStorageSettings.ApplicationSettings</strong> – property</td>
</tr>
<tr>
<td></td>
<td><strong>ApplicationData.LocalSettings</strong> – property</td>
</tr>
<tr>
<td></td>
<td><strong>ApplicationData.RoamingSettings</strong> - property</td>
</tr>
<tr>
<td>SD Card (WP 8.1 only)</td>
<td><strong>KnownFolders.RemovableDevices</strong> returns a <strong>StorageFolder</strong> object that can be sequentially used to read/write data from the SD card</td>
</tr>
<tr>
<td>Local database</td>
<td><strong>Identify objects that inherit from System.Data.Linq.DataContext.</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Verify the existence of reserved data stored in the local .sdf file</strong></td>
</tr>
</tbody>
</table>
Secure coding tips

- Mind my mantras 😊
- The **Data Protection API** (DPAPI) should be used to encrypt data
- Account credentials should be protected using the **PasswordVault** class
- Never hardcode encryption keys
- Never place encryption keys in unsafe device areas
- Do not use custom encryption algorithms
Data leakage

- (unintended) data leakage is addressed by M4 MTT for 2014
- Involuntary data exposure caused by OS or frameworks side-effects
  - System caching
  - Application backgrounding
  - System logging
  - Telemetry frameworks which expose private data
- A privileged access to target device file system - or connected network - is required to properly exploit these issues
Video

Local exploiting of data leakage
The PIN-protected box who leaked its token
it was a dark and stormy night..

.. and I was analyzing my favorite contacts backup app..
Oh (my) contacts.. WHERE are you going today?

Intercept == on.. Yes, I was lucky!
private void preparecontacts()
{
    int cellcnt = 0;
    int emailcnt = 0;

    // [...]
    if (str == "")
        this.UploadFile();
    else if (new DateTime(Convert.ToInt32(str.Substring(0, 4)), Convert.ToInt32(str.Substring(4, 2)), Convert.ToInt32(str.Substring(6, 2))).AddDays(1.0) < DateTime.Now)
        this.UploadFile();
}

private async void UploadFile()
{
    StreamReader reader1 = new StreamReader((Stream)new IsolatedStorageFileStream(this.sFile1, FileMode.Open, this.myFile));
    byte[] byteArray1 = Encoding.UTF8.GetBytes(reader1.ReadToEnd());
    reader1.Close();
    MemoryStream fileStream1 = new MemoryStream(byteArray1);
    string fileUploadUrl = "http://cb.whatasolution.com/FileUpload";
    HttpClient client = new HttpClient();
    fileStream1.Position = 0L;
    MultipartFormDataContent content = new MultipartFormDataContent();
    content.Add((HttpContent)new StreamContent((Stream)fileStream1), "file", "CB_" + DateTime.Now.ToString("yyyyMMddHHmmssfff") + ".vcf");
    try
    {
        await client.PostAsync(fileUploadUrl, (HttpContent)content).ContinueWith((Action<Task<HttpResponseMessage>>) (postTask =>
        {
            try
            {
                postTask.Result.EnsureSuccessStatusCode();
            }
            // [...]
        })
    }
    catch
    {
        Well, a “disaster recovery” backup rules..
## Hunting for potential data leakage

<table>
<thead>
<tr>
<th>Conditions</th>
<th>Classes, Methods or Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Backgrounding and Closing</td>
<td>Handler for the Application.Suspending event, typically the OnSuspending() method in App.xaml.cs</td>
</tr>
<tr>
<td></td>
<td>Handler for the Application.Deactivated event, typically the Application_Deactivated() method in App.xaml.cs</td>
</tr>
<tr>
<td></td>
<td>Handler for the Application.Closing event, typically the Application_Closing() method in App.xaml.cs</td>
</tr>
<tr>
<td></td>
<td>Handler for the Application.UnhandledException event, typically the Application_UnhandledException() method in App.xaml.cs</td>
</tr>
<tr>
<td>Use of Telemetry Frameworks</td>
<td>HockeyApp, BugSense, etc.</td>
</tr>
</tbody>
</table>
## Secure coding tips

<table>
<thead>
<tr>
<th>Actions</th>
<th>Classes, Methods or Properties</th>
</tr>
</thead>
</table>
| Remove cached data on app closing, suspension or deactivation | **server-side**  
WebBrowserExtensions.ClearInternetCacheAsync()  
WebBrowser.ClearInternetCacheAsync()  
WebView - no programmatic way |
| Remove stored cookies                        | **client-side**  
WebView - use HttpCookieManager.GetCookies() + HttpCookieManager.DeleteCookie()  
WebBrowser.ClearCookiesAsync()  
WebBrowserExtensions.ClearCookie() |
|                                              | **Cache-Control: no-store**     |
Authorization and Authentication Issues
Authorization and authentication issues

- Security decisions without *server-side engagement*
  - M5 - Poor Authorization and Authentication (MTT 2014)
  - May also involve M7 – Security Decisions via Untrusted Inputs

- Common issues
  - Offline authentication
  - Issues related to password complexity (e.g., 4 digits PIN)
  - Client-side generation of (predictable) authorization tokens
  - Authorization issues on *premium* functionalities or data access
Hunting for insecure tokens forgery

<table>
<thead>
<tr>
<th>Identification Data</th>
<th>Namespaces</th>
<th>Classes, Methods or Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device Name</td>
<td>Microsoft.Phone.Info</td>
<td>DeviceStatus.DeviceName</td>
</tr>
<tr>
<td>Hardware Identification</td>
<td>Microsoft.Phone.Info</td>
<td>DeviceExtendedProperties.GetValue(&quot;DeviceName&quot;)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DeviceExtendedProperties.GetValue(&quot;DeviceUniqueId&quot;)</td>
</tr>
<tr>
<td>Hashing Functions</td>
<td>Windows.Security.Cryptography</td>
<td>SHA1Managed, SHA256Managed, SHA384Managed and SHA512Managed classes</td>
</tr>
<tr>
<td></td>
<td>Windows.Security.Cryptography.Core</td>
<td>(or any other 3rd party libraries implementing these functions)</td>
</tr>
<tr>
<td>Geo Location Coordinates</td>
<td>Windows.Devices.Geolocation</td>
<td>Geolocator / Geoposition / Geocoordinate</td>
</tr>
<tr>
<td></td>
<td>System.Device.Location</td>
<td>GeoCoordinateWatcher / GeoPosition / GeoCoordinate</td>
</tr>
</tbody>
</table>
Weak custom code encryption

```csharp
protected override void OnNavigatedTo(NavigationEventArgs e)
{
    base.OnNavigatedTo(e);
    using (IsolatedStorageFile storeForApplication = IsolatedStorageFile.GetUserStoreForApplication())
    {
        this.fileExists = storeForApplication.FileExists("wp contacts backup.zip");
        if (!this.fileExists)
        {
            this.infoTextBlock.Text = "No backup file exists! Please create one before trying to download it.";
        }
        else
        {
            try
            {
                this.server = new HttpServer(2, 65536);
                this.server.Start(new IPEndPoint(IPAddress.Parse("0.0.0.0"), 5656));
                this.server.TextReceived += new EventHandler<HttpDataReceivedEventArgs>(this.server_TextReceived);
                this.infoTextBlock.Text = "http://" + this.server.LocalEndpoint.ToString();
            }
            catch
            {
                this.infoTextBlock.Text = "Unable to start WEB Server. Please check your connectivity settings.";
            }
        }
    }
}
```

contacts backup file is stored in app's sandbox
protected override void OnNavigatedTo(NavigationEventArgs e)
{
    base.OnNavigatedTo(e);
    using (IsolatedStorageFile storeForApplication = IsolatedStorageFile.GetUserStoreForApplication())
    {
        bool fileExists = storeForApplication.FileExists("wp contacts backup.zip");
        if (!fileExists)
        {
            this.infoTextBlock.Text = "No backup file exists! Please create one before trying to download it.";
        }
        else
        {
            try
            {
                this.server = new HttpServer(2, 65536);
                this.server.Start(new IPEndPoint(IPAddress.Parse("0.0.0.0"), 5656));
                this.server.TextReceived += new EventHandler<HttpDataReceivedEventArgs>(this.server_TextReceived);
                this.infoTextBlock.Text = "http://" + this.server.LocalEndpoint.ToString();
            }
            catch
            {
                this.infoTextBlock.Text = "Unable to start WEB Server. Please check your connectivity settings.";
            }
        }
    }
}
Introducing IPC with Windows Phone

- Windows Phone provides limited support to Inter Process Communication (IPC)
  - WP 7.x does not support IPC
  - WP 8.x provides file and URI association

- A third undocumented IPC exists
  - Shell_PostMessageToast (ShellChromeAPI.dll) allows performing Cross-Application Navigation Forgery attacks
  - A malicious app can send a *toast message* that, once tapped, allows to open an arbitrary XAML page of an arbitrary app – XAML page code behind can be fed with malicious input
app://{GUID}/_default#/AssemblyName;component/Page.xaml?par=val1&par2=val2

Native Toast Launcher
Target Uri
app://5B04B775-356B-4AA0-AAF8-6491FFEAS605/_default

Native Toast Launcher
Target Uri
app://5B04B775-356B-4AA0-AAF8-6491FFEAS605/_default

IMPOSTAZIONI

informazioni

informazioni sul telefono
Nome: daath
Modello: SAMSUNG GT-i8750
Software: Windows Phone 8.0
Risoluzione dello schermo: 720x1280

altre informazioni

contenuti online
Condizioni per l’utilizzo
Informativa sulla privacy
© 2012 Microsoft Corporation

ripristina il telefono
Video

Attacking a weak authentication mechanism
Bypassing DropBox security passcode
Behind the bypass

protected override void OnNavigatedTo(NavigationEventArgs e)
{
    // [...]
    this.ViewModel.Init(Enum.Parse(typeof(LockPageType), this.NavigationContext.QueryString["type"]));
}

public void Init(LockPageType type)
{
    this.NbrTry = 0;
    this.Type = type;

    if (this.Type == LockPageType.CHANGEPIN)
        this._createstep = CreationStep.ENTEROLDPASSCODE;

    this.ManageType();
}

namespace Dropbox.Core.ViewModels.Lock
{
    public enum LockPageType
    {
        UNLOCK, // 0
        CREATEPIN, // 1
        CHANGEPIN, // 2
        DISABLEPIN, // 3
    }
}
Behind the bypass

...component/Pages/Lock/LockPage.xaml?type=1

t.this.Type = LockPageType.CREATEPIN = 1

public void ManageType()
{
    switch (this.Type)
    {
    case LockPageType.CREATEPIN:
        switch (this._createstep)
        {
        case CreationStep.ENTERPASSCODE:
            this.LegendText = AppResources.ProtectionEnterPin;
            break;
        case CreationStep.VERIFYPASSCODE:
            this.LegendText = AppResources.ProtectionVerifyPin;
            break;
        }
    
    So we can overwrite the previous passcode and..
Hunting for insecure IPC

<table>
<thead>
<tr>
<th>Actions</th>
<th>Platform</th>
<th>Namespaces</th>
<th>Classes, Methods and Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>WP 8.1</td>
<td>Windows.UI.Xaml.Application</td>
<td>OnFileActivated() method</td>
</tr>
</tbody>
</table>
Secure coding tips

- Avoid client-side generation of tokens
- Avoid using `DeviceExtendedProperties.GetValue("DeviceUniqueId")`
  - The returned identification is unique only *per device*
- `HostInformation.PublisherHostId` property is unique *per device* and *per publisher*
  - A malicious app should be published by the author of the targeted one to steal the ID
- Positive validate all your input and authorize every actions
  - Carefully audit each `OnNavigatedTo()` methods!
CLIENT SIDE INJECTIONS
Client-side injection

- Feeding an interpreter with untrusted data
  - Similar to the server-side ones, but the interpreter resides at the “app-side”

- Different interpreters exist – and so for the related injection
  - Offline authentication
  - Local database querying systems
  - XML parsers
  - HTML and JavaScript engines
  - File handling routines

- Attacks impact depends on the data handled by the interpreter
all input is evil

trust no one and map the sources for malicious data
Hunting for untrusted data sources

- Input from network
- Bluetooth and NFC
- Inter Processor Communication (IPC) mechanisms
- Files accessed from SD card – which is a shared storage area
- User typed input – via UI, speech to text, camera (QR code), USB data, ..
## Hunting for injections

<table>
<thead>
<tr>
<th>Interpreters</th>
<th>Namespaces</th>
<th>Classes, Methods and Properties</th>
</tr>
</thead>
</table>
| HTML/JavaScript | Microsoft.Phone.Controls       | WebBrowser  
|               | Windows.UI.Xaml.Controls     | WebView  
|               |                                | NavigateToString()  
|               |                                | InvokeScript()  
|               |                                | IsScriptEnabled = true (property)  
|               |                                | InvokeScriptAsync()  
|               |                                | NavigateToLocalStreamUri()  
|               |                                | NavigateWithHttpRequestMessage()  
| XML | System.Xml.Linq            | XDocument.Load() |
|               | System.Xml                   | XmlReaderSettings.DtdProcessing = DtdProcessing.Parse |
| XAML | System.Windows.Markup         | XamlReader.Load() |
# Hunting for injections

<table>
<thead>
<tr>
<th>Interpreters</th>
<th>Namespaces</th>
<th>Classes, Methods and Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL</td>
<td>SQLitePCL</td>
<td><code>SQLiteConnection.Prepare()</code></td>
</tr>
<tr>
<td></td>
<td>SQLite-Net-WP8</td>
<td>Query() / Query&lt;T&gt;() / QueryAsync&lt;T&gt;() / Execute() / ExecuteAsync()</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>ExecuteScalar&amp;lt;T&amp;gt;()</code> / <code>ExecuteScalarAsync&amp;lt;T&amp;gt;()</code> / DeferredQuery() / DeferredQuery&lt;T&gt;()</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>FindWithQuery&amp;lt;T&amp;gt;()</code> / CreateCommand()</td>
</tr>
<tr>
<td></td>
<td>CSharp-SQLite</td>
<td><code>IDbCommand.CommandText</code> (property)</td>
</tr>
<tr>
<td></td>
<td>SQLiteWinRT</td>
<td><code>Database.ExecuteStatementAsync()</code> / <code>Database.PrepareStatementAsync()</code></td>
</tr>
</tbody>
</table>
## File handling

<table>
<thead>
<tr>
<th>Interpreters</th>
<th>Namespaces</th>
<th>Classes, Methods and Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>StorageFolder</td>
<td>CreateFileAsync()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RenameAsync()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GetFolderFromPathAsync()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GetFolderAsync()</td>
</tr>
<tr>
<td></td>
<td>StorageFile</td>
<td>CopyAsync()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GetFileFromApplicationUriAsync()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GetFileFromPathAsync()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RenameAsync()</td>
</tr>
<tr>
<td></td>
<td>IsolatedStorageFile</td>
<td>OpenFile()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CopyFile()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CreateDirectory() - CreateFile()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DeleteDirectory() - DeleteFile()</td>
</tr>
</tbody>
</table>

Hunting for injections
private void ButtonView_Click(object sender, RoutedEventArgs e)
{
    this.ButtonView.IsEnabled = false;
    this.iptexttemp = this.TextIP.Text.Trim() + "xxxxxxxxx";
    this.WebBrowser1.NavigateToString("<body bgcolor=black>
" +
"<form action='https://www.REMOTE-SITE.com/path/resource.asp' method=post>
" +
"<input name='iname' value='" + this.TextAdmin.Text.Trim() + "' type='hidden'>" +
"<input name='pword' value='" + this.PasswordBox1.Password.Trim() +
" type='hidden'><input name='ip' value='" + this.TextIP.Text.Trim() +
" type='hidden'><input name='port' value='" + this.TextPort.Text.Trim() +
" type='hidden'><input name='versi' value='ori' type='hidden'></form>
" +
"<script>document.forms[0].submit();</script></body>");
}

app renders user-controlled data without any validation
Video

Cross-Site Scripting (XSS) attack via IPC
Stealing Vodafone Business app credential via XSS
Vulnerable NavigateToString() method

```csharp
protected override void OnNavigatedTo(NavigationEventArgs e)
{
    base.OnNavigatedTo(e);
    if (this.NavigationContext.QueryString.ContainsKey("Question"))
    {
        this.QuestionArrived = this.NavigationContext.QueryString["Question"];
        this.textBlockQuestion.Text = this.QuestionArrived;
    }
    if (this.NavigationContext.QueryString.ContainsKey("Answer"))
    {
        this.AnswerArrived = this.NavigationContext.QueryString["Answer"];
        this.webView.NavigateToString("<html><head>
        function getSize(){ " +
        "var h = document.getElementById('content').offsetHeight; var s = "rendered_height=" + h; window.external.notify(s);} " +
        "</script> <style type="text/css">body {font-family: "segoe"; font-size:19;} </style></head><body><div id="content" " +
        Uri.UnescapeDataString(this.AnswerArrived) + ""/>
    }
```
Secure coding tips

- Implement proper input *positive* validation
- Prevent XSS – **WebView** and **WebBrowser** controls
  - Validate parameters passed to InvokeScript(), Navigate(), NavigateToString(), etc.
- Avoid SQL Injections
  - Use LINQ to SQL :-P
  - Adopt parameterized query
- Validating file names/paths handled by methods defined for
  - StorageFolder
  - StorageFile
  - IsolatedStorageFile
Final considerations

Well, I don’t like writing conclusion, but I have to. A 30-pages-long whitepaper – yeah, it was pretty hard - will be released in 15 minutes - thanks Dhillon. It simply represents the first public catalog of insecure usage of APIs provided by Windows Phone SDK, and covers both Silverlight and Windows Runtime technologies. Substantial part of my work – the Windows Runtime one - will be valid for Universal Apps, too – so your favorite Windows 10 app can be safely developed as well. That’s it. Thanks to my wife Silvia, Stefano and my awesome Tiger Team := {Giovanni, Alberto, Eros, Francesco, Primo, Filippo, Matteo} for supporting me during the research! Btw, thanks for your attention and I hope you enjoyed the talk. See ya. Ciao Mamma!