Security Risks

Xamarin University

What are we protecting against?

- Indirect Attacks
- Direct Attacks
- Theft / Physical Attacks

Personal Data
Sensitive Data

Xamarin University

40%

- Cases where is was possible to recover passwords from stolen devices

Xamarin Evolve 2014

Xamarin Evolve 2014

"Security in IT is like locking your house or car – it doesn't stop the bad guys, but if it's good enough they may move on to an easier target."

Paul Herbka
VP at IT security firm South Seas Corp

Application Data Storage

Xamarin University

AES Encryption

- Advanced Encryption Standard
- Symmetric Encryption
- Requires a key

Xamarin University

System.Security.Cryptography

- Part of the .NET Framework
- Easy to use
- Keys can be auto generated for ease of use
- Default properties are safe to use
- Supports *Secret-Key* encryption and *Public-Key* encryption

You don't need to be an expert

Xamarin University

Secret Key Encryption

- Also known as symmetric encryption
- Uses a single secret key to encrypt and decrypt data
- Key MUST be kept secret
- Algorithms are typically very fast

Xamarin University

AES Encryption / Decryption

```
using System.Security.Cryptography;

public static string EncryptStringAES (string plainText,
                                       string sharedSecret)
{
}

public static string DecryptStringAES (string encryptedText,
                                       string sharedSecret)
{
}
```

Xamarin University

RijndaelManaged class

- Gives access to encryption algorithms
- Use with **Rfc2898DeriveBytes()** to create sufficiently strong encryption keys

```
// generate the key from shared secret and salt
// with repeating hashing to create a longer key
var key = new Rfc2898DeriveBytes(sharedSecret, Salt);
aesAlgorithm = new RijndaelManaged();
aesAlgorithm.Key = key.GetBytes(aesAlgorithm.KeySize / 8);
```

Xamarin University

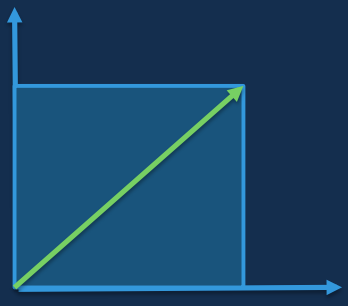
It's better with Salt

- Random Data
- Added to the end of Keys
- Defends against *Dictionary Attacks*

Xamarin University

Initialization Vector

- Commonly referred to as "IV"
- Random Value
- Ensures that the encrypted data is always different
- Provided by our Encryption object (RijndaelManaged)



Xamarin University

ICryptoTransform Interface

- Used to manage Encryption block size
- Represents the cryptographic transform (encryption / decryption)

- `CreateEncryptor()` – if you want to *encrypt* data
- `CreateDecryptor()` – if you want to *decrypt* data

Xamarin University

Demo

Xamarin University

Accessing Web Services with OAuth

Xamarin University

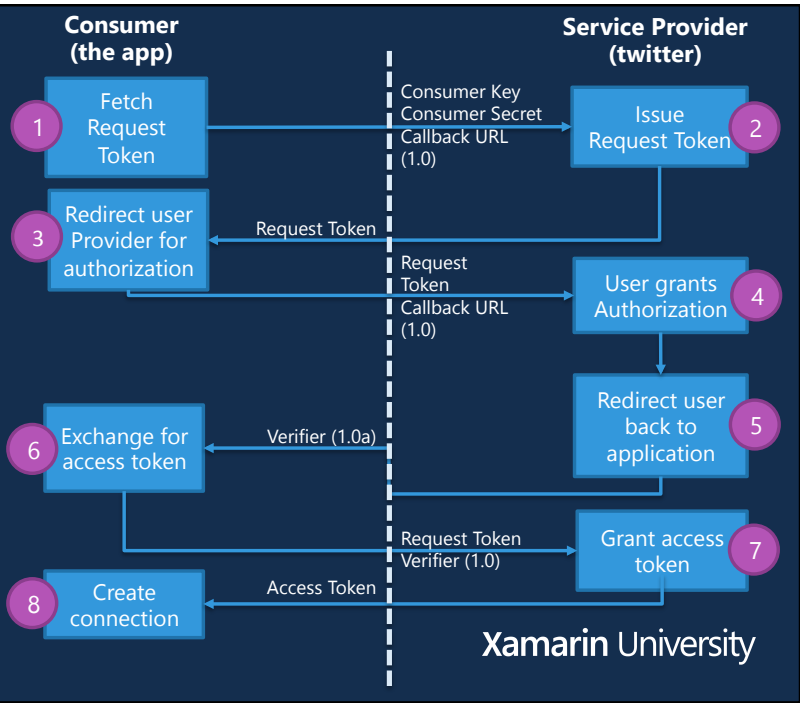
OAuth

- Allow 3rd parties access to a resource without giving them login credentials
- Designed to work over HTTP
- Uses Access Tokens instead of direct login information

Xamarin University

OAuth 1. 0 Flow

- 3 *Actors*, the User, the Consumer and the Service Provider
- The Consumer gains access to the Provider
- The Consumer never sees the Login information



Xamarin.Auth

- Cross-platform API for Authentication
- Xamarin.iOS & Xamarin.Android
- Supports OAuth 1.0 & 2.0
- Open Source & Extensible

Xamarin University

OAuth 2.0

```
using Xamarin.Auth;
...
var auth = new OAuth2Authenticator (
          clientId: "App ID from https://developers.facebook.com/apps",
          scope: "",
          authorizeUrl: new Uri ("https://m.facebook.com/dialog/oauth/"),
          redirectUrl: new Uri
             ("http://www.facebook.com/connect/login_success.html"));
```

Xamarin Evolve 2014

Demo

Xamarin University

Summary

- Protect your user's data with symmetric encryption
- **System.Security.Cryptography** provides the tools we need for encryption
- **Xamarin.Auth** makes accessing OAuth enabled services much easier

Xamarin University

Xamarin Evolve 2014

Securing Data in your Mobile Apps

Adrian Stevens
adrian.stevens@xamarin.com

Xamarin University