CREALOGIX

CLX.MAP & Mobile Security

Agenda

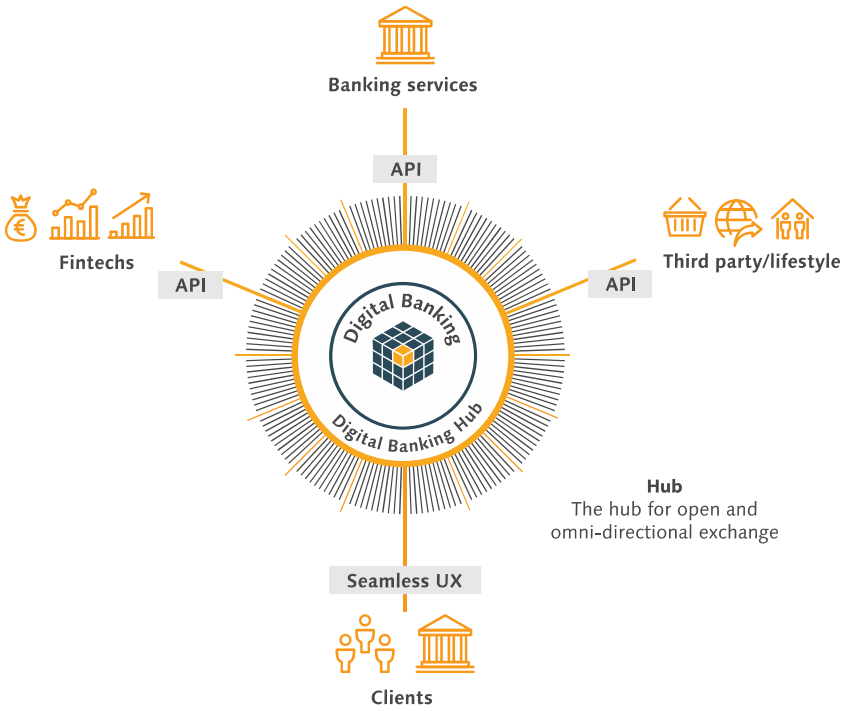
1. Digital Banking
2. Mobile Banking Apps
3. CLX.MAP
4. Mobile Security
5. App Hardening
6. Is my App Secure?

Digital Banking

Digital Banking Trends

Digital Banking Hub
- Customer orientation
- Omni-directional communication
- Orchestration of services and applications
- API based integration of 3rd party solutions

Mobile Banking
- Mobile only users
- Single universal Mobile Banking App
  - Extensible for 3rd party applications
  - Flexible customization



Mobile Banking Apps

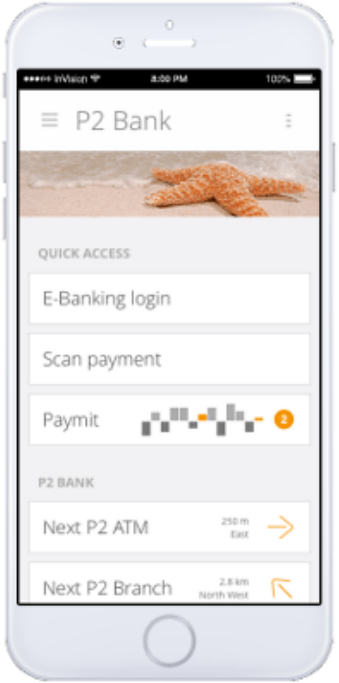
What can a Mobile Banking App do?

Support multiple (existing) applications

Provide security features

Customization, Branding

Support updates



How can you build a Mobile Banking App?

Native Apps

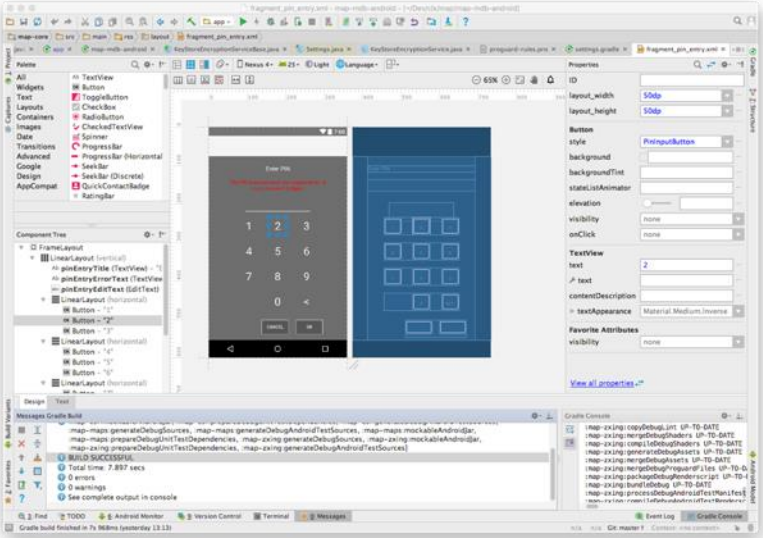
- Platform conformity
- Direct HW support
- Performance

Web Apps

- Easier maintenance
- Existing web content

Hybrid/Multi-platform frameworks

- Cordova/Xamarin/React Native etc
- Extensive code base and plugins (Pro & Con)



CLX.MAP

What are the advantages?

Infrastructure for configuring, building and deploying apps

Native container for various (web based) apps

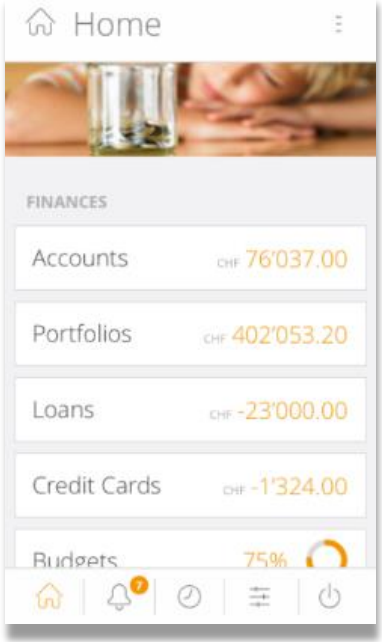
- ✓ Access through a single app

HTML5

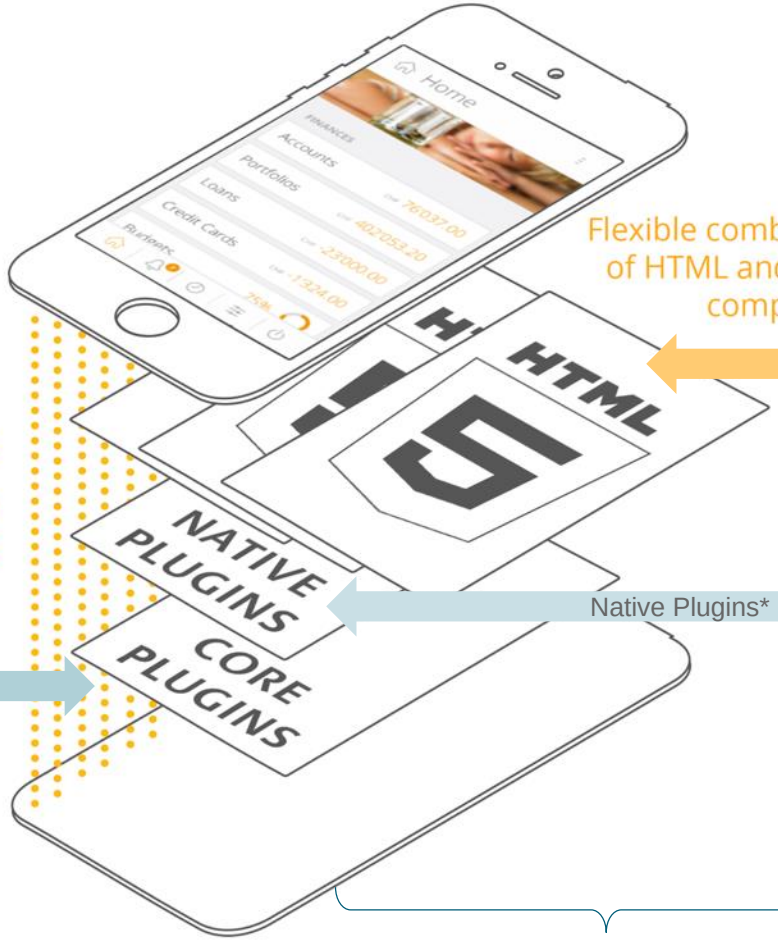
- ✓ Simplified development & update process

Native Plugins

- ✓ Access to device (e.g. camera)
- ✓ Improved security through hardening
- ✓ Integration of native content



Architecture



Flexible combination of HTML and native components

Mobile application framework

Webview/CSP*

HTML5/Java Script Engine
For customized user interface

Native Plugins*

Payment Scanner
PushTAN
QR Code
PhotoPayment
Notifications
Maps
…

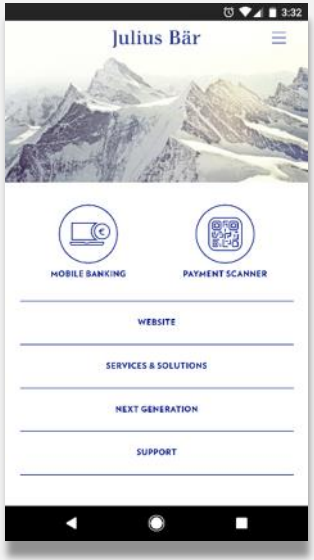
Core Plugins*

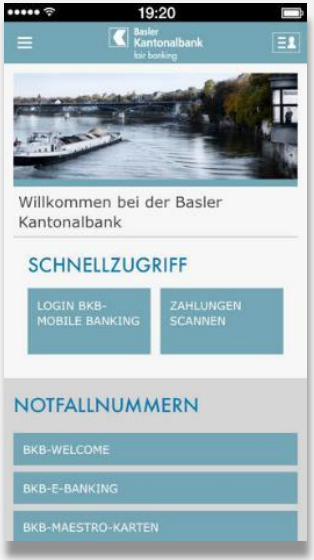
Network
Security/Hardening
Cryptography
Localisation

* provided by CREALOGIX, 3rd parties or by the bank

Feature Overview

Customization Examples





Mobile Security

What are the mobile security threats?

Vulnerable Apps:

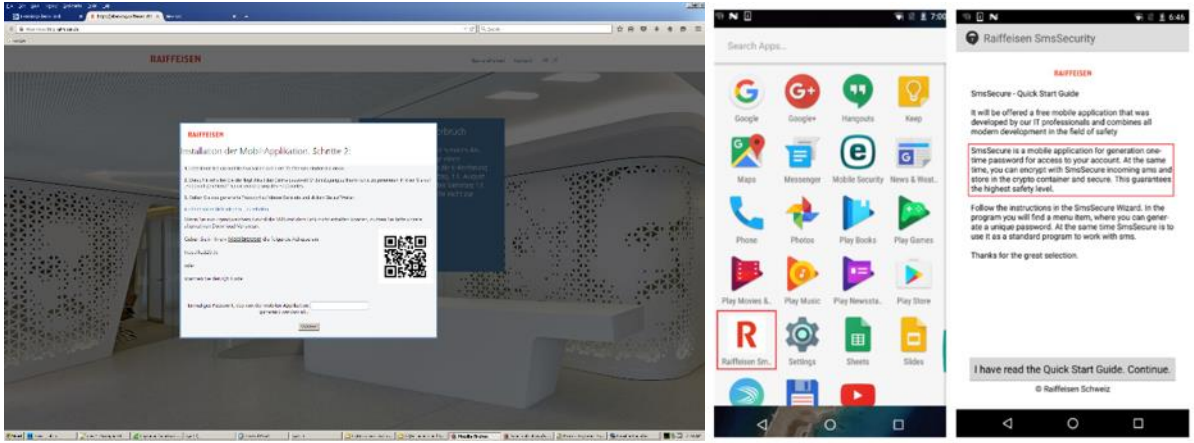
- M1 Platform misuse
- M2 Insecure data storage
- M3 Insecure data communication

Typical attacks:

- Malware
- MITM
- Fake Apps
- Phishing/Account takeover

Especially on:

- Jailbroken/rooted devices
- Outdated OSs



Source: Peter Stancik: Nicht nur Tescos Bankkunden sind im Visier von Retefe (10.11.2016)

4 Levels of Security Controls

4 Transaction
Workflow and message level

3 Data
Data leakage prevention in transit & storage

2 App
Architectural & Hardening features

1 Device / Platform
OS security features

Security Controls – Device / Platform Level

- OS/Platform
  - App and developer verification (incl. code signing)
  - App runtime protection
  - Security updates (Android long-term support?)
- Device
  - Device locking & encryption
  - Secure Enclave & keychain
  - Fingerprint recognition service

Security Controls – App Level

- **Architecture**
  - Check and use the platform features secure M1
  - Device Binding
  - User authentication
  - Harden against re-engineering
- **Processes**
  - SDLC (Secure Development Life Cycle)
  - Release & support processes
- **User awareness**
  - App tour
  - Phishing warnings
  - Notifications

Security Controls – Data Level

M2

Data Leakage on device

- Logs / crash reports
- Backups
- Screenshot protection
- Copy/paste prevention
- File Encryption
  - External storage
  - Cache files
  - Shared Preferences

M1 & M3

IPC misuse

- Unverified extra data (parameters)
- Use multi-app solutions with care

Insecure HTTPS

- CA Truststore
- Verify certificate chain
- Use a secure HTTPS stack

HSTS & HPKP not supported

- Not supported on many platforms
- Implement effective certificate pinning

Security Controls – Transaction Level

2-Factor Authentication

- Authentication on login
- Payment Confirmation

Secure 2nd channel

- PushTAN or FotoTAN preferred to mTAN
- Message level encryption

Server-side workflow features

- Whitelist beneficiaries
- Limit transactions
- Fraud detection

App Hardening

App Hardening Tools

Hardening tools

- Standard options: ProGuard, compiler options
- Commercial tools: DexGuard, Arxan, Promon

Functionality

- ✓ Anti-debug protection
- ✓ Jailbreak/rooting detection
- ✓ Screenshot prevention
- ✓ Copy/Paste prevention
- ✓ Runtime integrity checks
- ✓ Data & code obfuscation
- ✓ Whitebox cryptography
- ✓ Native modules

Is my App secure?

	Deutsche Bank		Commerzbank		Norisbank		Comdirect	
	[11]	[10]	[6]	[7]	[21]	[22]	[4]	[5]
Enforces Out-of-Band	□		□		□		■	
Analyzed Version	2.6.0	2.1.7	4.0.1	7.1.7	2.6.0	2.1.7	2.1.5	6.0.6
Release Date	Jul 7	Jun 6	Sep 21	Sep 12	12 Jul	Jun 8	Feb 3	Mar 7
Denies Backup	□	□	□	■	□	□	□	□
Anti-Rooting	□	□	□	■	□	□	□	▧
Anti-Repackaging	□	□	□	■	□	□	□	■
Obfuscation	□	▧	■	■	□	▧	▧	▧
Fingerprinting	–	ID, IMEI	–	ID, IMEI	–	ID, IMEI	–	ID
TLS Pinning	■	–	□	–	■	–	□	–

Source: Vincent Haupert, Tilo Mueller: On App-based Matrix Code Authentication in Online Banking (12.10.2016)

```java
// banking App - downloaded March 2017

// requesting OAuth access token with code or refresh_token

public static void requestOauthToken(String grantCode, InternalRequestOauthTokenCallbackImpl callback) {
    if (!localStorage.isHawkInitialized() || localStorage.getRefreshToken() == null) {
        Logger.logInfo("Request token is called");
        ServiceFactory.INSTANCE.executePostRequestWithClientCertificate(oauthServerConfiguration.getServerUrl()
            + oauthServerConfiguration.getTokenEndpoint(), callback, grantCode, false);
        return;
    }
    Logger.logInfo("Refresh token is called");
    ServiceFactory.INSTANCE.executePostRequestWithClientCertificate(oauthServerConfiguration.getServerUrl()
        + oauthServerConfiguration.getTokenEndpoint(), callback, grantCode, true);
}

// posting OAuth token requests with client certificate

public C1039e executePostRequestWithClientCertificate(String url, C0503f callback, String grantCode, boolean isRefresh) {
    C1065w requestBody;
    if (isRefresh) {
        requestBody = new C1053o().m4143a("grant_type", "refresh_token").m4143a("scope",
            OauthLibrary.getOauthServerConfiguration().getScopes()[0]).m4143a("refresh_token", grantCode).m4144a();
    } else {
        requestBody = new C1053o().m4143a("grant_type", "authorization_code").m4143a("redirect_uri",
            OauthLibrary.getOauthServerConfiguration().getRedirectUrl()).m4143a("code", grantCode).m4144a();
    }
    C1039e call = this.okHttpClientWithSocketFactory.m4183a(new C1063a().m4225a(url).m4224a(requestBody).m4231b());
    call.m4076a(callback);
    return call;
}

// client certificate configuration

private void m2309u() {
    FeatureHandler.init(this);
    C0829i.m3140a((C0245b) this, (int) R.id.mainContainerLayout, false);
    SessionManager.INSTANCE.init(this);
    OauthLibrary.init(new OauthServerConfiguration("/mobile/mobile", "/mobile/token/endpoint",
        "https://mobile.**********.ch", "https://localhost:8443/***client", 123456, C0496a.f1457a),
        new OauthCertificateConfiguration("=K8_**********p2V", "***Client.p12", "***Client"), this, false);
}
```

“Not to be missed” mobile security controls

Data Leakage

- Logs / crash reports
- Backups
- External storage
- Screenshot protection
- Copy/paste prevention
- File Encryption
- Cache files
- Shared Preferences

IPC misuse

- Use multi-app solutions with care
- Secure inter-app communication

Insecure HTTPS

- CA Trust, certificate chain
- Use a secure HTTPS stack
- Implement effective certificate pinning

Hardening

- Anti-debug protection
- Jailbreak/rooting detection
- Screenshot prevention
- Copy/Paste prevention
- Runtime integrity checks
- Data & code obfuscation
- Whitebox cryptography
- Native modules

I’ve done all that, can I sleep easy?

Of course not …

- Hardening tools do not cover everything
  - Is the hardening configuration correct?
- Was the solution PEN tested on all devices & platforms?
  - Legacy devices and OS versions need additional effort
- For example:
  - Certificate pinning
  - AndroidKeystore

Certificate Pinning

Certificate pinning in hybrid Apps

- HPKP not supported on iOS/Safari or Android/Webview
- Certificate pinning in manifest (>= Android N)

App must still implement certificate pinning

- Intercept web requests
- native HTTPS implementation
- Verify responses
- Handle redirects and POST requests correctly

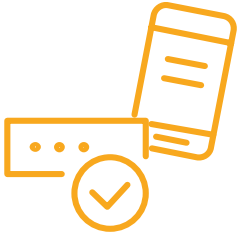
Android Keystore

Secure Keystore

- Secure Enclave or SW AndroidKeystore?
- Android Kitkat & Marshmallow keystore issues?
  - Modified device security level (PIN to PATTERN)
- Handle updated fingerprint set?

App must check for and use hardware keystore correctly

- Check device is secured
  - i.e. PIN/PWD device lock & device encrypted
- Use key attributes to detect updated fingerprint sets
- Consider fingerprint fallback authentication
- Handle devices without secure enclave



CREALOGIX

Thank you!