Tools for Protecting Your Users' Data

MAURICE GAVIN

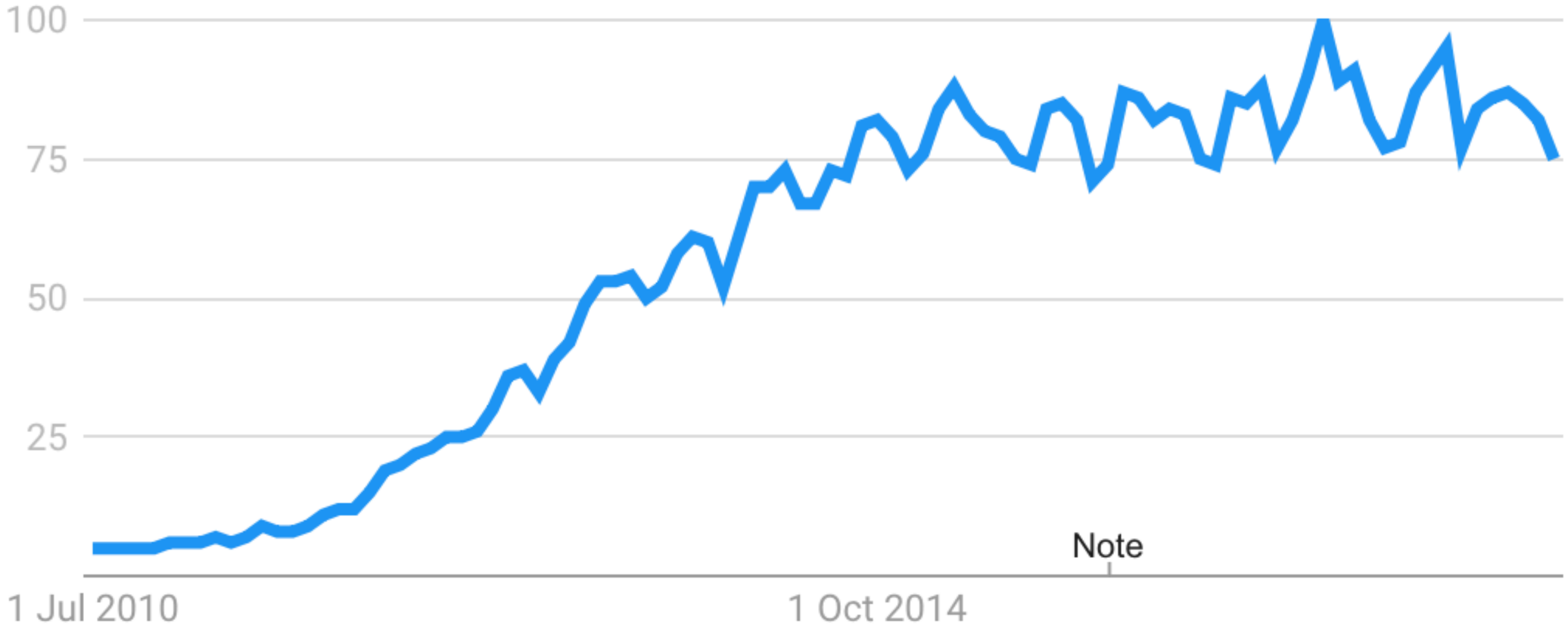
Maurice

Gavin

Android Engineer at toothpic

Introduction

Interest in “Big Data” 2010 -2019



source: Google Trends, https://g.co/trends/pU2mc

Data Breaches - Yahoo (2013-14)

3 billion user accounts

Wiped **$350 million** off Yahoo's sale price to Verizon

2018

March

Cambridge Analytica

May

GDPR

May 2019

WhatsApp vulnerability

1.5 Billion users

Used to target human rights lawyers

https://nvd.nist.gov/vuln/detail/CVE-2019-3568

What we'll focus on

Types of Data

Identify Risks

Improve Security

Types of Data

Types of Data you might collect

PII - Personally Identifiable Information

- Name, email, date of birth, address

PHI - Personal Health Information

- Medical history, clinical images, feedback from medical professional

Types of Data you might collect

Crash Reports

- Stack traces, application logs

Analytics

- App usage statistics, funnel drop off

Identify Risks

- Automated Tools
- Developer Knowledge
- Penetration Testing

Automated Tools

Static and Dynamic Analysis

Static analysis

- Detect issues in source code
- e.g. hardcoded passwords, insecure configuration, etc.

Dynamic analysis

- Detect vulnerabilities at runtime
- e.g. sensitive data written in device logs or remote code execution
- Don’t need the source code

Tool - MobSF

https://github.com/MobSF/Mobile-Security-Framework-MobSF

ISSUE	SEVERITY
The App uses an insecure Random Number Generator.	high
The App logs information. Sensitive information should never be logged.	info
This App uses Java Hash Code. It's a weak hash function and should never be used in Secure Crypto Implementation.	high
Files may contain hardcoded sensitive informations like usernames, passwords, keys etc.	high
App uses SQLite Database and execute raw SQL query. Untrusted user input in raw SQL queries can cause SQL Injection. Also sensitive information should be encrypted and written to the database.	high
This App may have root detection capabilities.	secure

Tool - DataTheorem

https://www.datatheorem.com/

A new P1 issue for "Toothpic" (com.toothpic.app/ANDROID) has been found: "Google Play Blocker: App Compiled with Outdated Target SDK" (008328)

Feb 21 2:55 am

DependencyCheck

Scan 3rd party SDKs and Open Source Libraries for known CVEs

https://github.com/jeremylong/DependencyCheck

DEPENDENCY-CHECK

Dependency-Check is an open source tool performing a best effort analysis of 3rd party dependencies; false positives and false negatives may exist in the analysis performed by the tool. Use of the tool and the reporting provided constitutes acceptance for use in an AS IS condition, and there are NO warranties, implied or otherwise, with regard to the analysis or its use. Any use of the tool and the reporting provided is at the user's risk. In no event shall the copyright holder or OWASP be held liable for any damages whatsoever arising out of or in connection with the use of this tool, the analysis performed, or the resulting report.

How to read the report | **Suppressing false positives** | **Getting Help: google group** | **github issues**

Project: project ':app'

Scan Information (show all):

- *dependency-check version*: 3.1.1
- *Report Generated On*: Feb 19, 2018 at 18:11:19 +00:00
- *Dependencies Scanned*: 77 (76 unique)
- *Vulnerable Dependencies*: 3
- *Vulnerabilities Found*: 4
- *Vulnerabilities Suppressed*: 0
- ...

Display: Showing Vulnerable Dependencies (click to show all)

Dependency	CPE	Coordinates	Highest Severity	CVE Count	CPE Confidence	Evidence Count
stetho-okhttp3-1.3.1.jar	cpe:/a:facebook:facebook:1.3.1	com.facebook.stetho:stetho-okhttp3:1.3.1 ✓	High	1	Low	24
stetho-1.3.1.jar	cpe:/a:facebook:facebook:1.3.1	com.facebook.stetho:stetho:1.3.1 ✓	High	1	Low	24
jackson-databind-2.8.2.jar	cpe:/a:fasterxml:jackson-databind:2.8.2 cpe:/a:fasterxml:jackson:2.8.2	com.fasterxml.jackson.core:jackson-databind:2.8.2 ✓	High	2	Highest	40

Automate It!

Integrate it with your Continuous Integration environment.

Run reports each time you submit a pull request, run tests or create a release.

Developer Knowledge

Where I started

Android Security documentation:
https://developer.android.com/topic/security/best-practices

Levelling up

OWASP Mobile Security Testing Guide (MSTG)
https://github.com/OWASP/owasp-mstg

Examples of topics covered:

- Tampering and Reverse Engineering on Android
- Android Anti-Reversing Defenses

Staying up to date

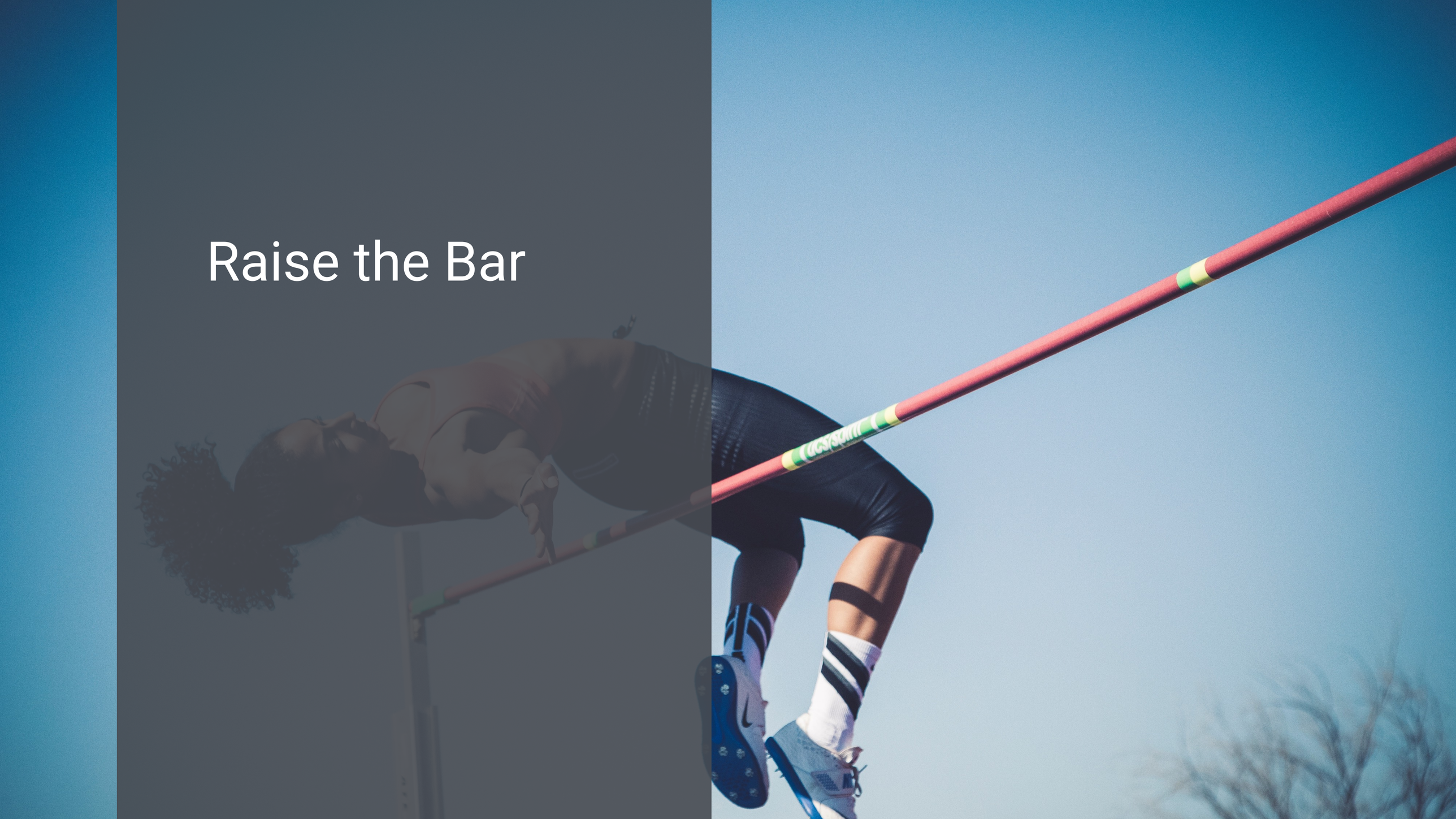
Monthly Android Security bulletin
https://source.android.com/security/bulletin/

Android Security 2018 Year in Review
https://source.android.com/security/reports/Google_Android_Security_2018_Report_Final.pdf

Penetration Testing

Improve Security

Raise the Bar



Application Sandbox

Application Sandbox

- Other processes on the system can't access your code or data
- Internal storage visible to your app only
- External storage visible to all apps with `READ_EXTERNAL_STORAGE` permission

Code Obfuscation

Code Obfuscation

Original Source Code Before Rename Obfuscation

```
private void
CalculatePayroll(SpecialList
employeeGroup) {
   while(employeeGroup.HasMore()) {
      employee =
employeeGroup.GetNext(true);
      employee.UpdateSalary();
      DistributeCheck(employee);
   }
}
```

Reverse-Engineered Source Code After Rename Obfuscation

```
private void a(a b) {
   while (b.a()) {
      a = b.a(true);
      a.a();
      a(a);
   }
}
```

source: https://www.preemptive.com/obfuscation

Code Obfuscation Tools

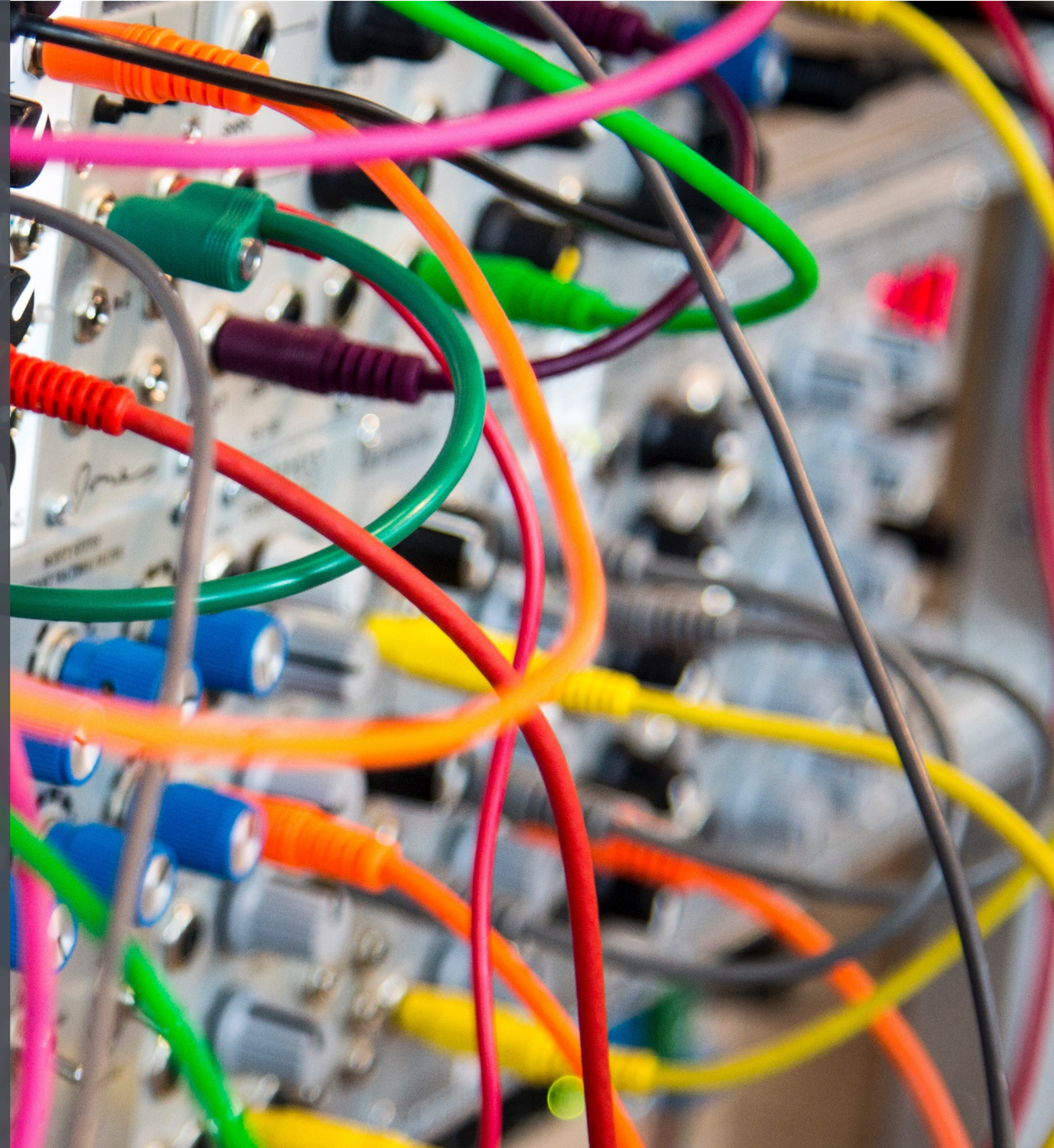
ProGuard/R8 (https://developer.android.com/studio/build/shrink-code)

- Free
- Code optimisation, protects against static analysis

DexGuard (https://www.guardsquare.com/en/products/dexguard)

- Enterprise product with an enterprise price
- Much better protection through string and class encryption

Android Network Security Config

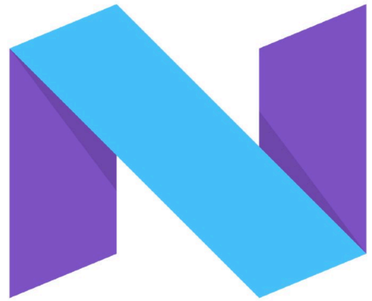


Android Network Security Config

Configuration for…

- Custom TrustStores
- Debug only settings
- Block clear text traffic
- Certificate Pinning

Available for 7.0+



Back ported to Android 4.2

https://github.com/commonsguy/cwac-netsecurity

Android Network Security Config

```
<application
    android:name="com.toothpic.app.ToothpicApplication"
    android:networkSecurityConfig="@xml/network_security_config"
    android:allowBackup="false"
    android:icon="@mipmap/app_icon"
    android:label="@string/app_name"
    android:theme="@style/AppTheme.Toothpic">
```

AndroidManifest.xml

Android Network Security Config

```
<application
    android:name="com.toothpic.app.ToothpicApplication"
    android:networkSecurityConfig="@xml/network_security_config"
    android:allowBackup="false"
    android:icon="@mipmap/app_icon"
    android:label="@string/app_name"
    android:theme="@style/AppTheme.Toothpic">
```

AndroidManifest.xml

Certificate Authority

Pros

- Determine if certificate we've never seen before is valid
- Required for generic clients (e.g. web browsers)

Cons

- Rogue CAs

network_security_config.xml

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <base-config>
        <trust-anchors>
            <certificates src="system"/>
            <certificates src="user"/>
        </trust-anchors>
    </base-config>
</network-security-config>
```

Trusting Custom CAs

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <base-config>
        <trust-anchors>
            <certificates src="@raw/trusted_roots"/>
        </trust-anchors>
    </base-config>
</network-security-config>
```

App-wide config

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <base-config>
        <trust-anchors>
            <certificates src="@raw/trusted_roots"/>
        </trust-anchors>
    </base-config>
</network-security-config>
```

Per-Domain Config

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config>
        <domain includeSubdomains="true">toothpic.com</domain>
        <trust-anchors>
            <certificates src="@raw/trusted_roots"/>
        </trust-anchors>
    </domain-config>
</network-security-config>
```

Custom CA for Debug

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <debug-overrides>
        <trust-anchors>
            <certificates src="@raw/debug_cas"/>
        </trust-anchors>
    </debug-overrides>
</network-security-config>
```

Certificate Pinning

Pros

- Trust **specific certificates** instead of CA (i.e. many certificates)
- Protects against rogue CAs

Cons

- Problematic to update, requires user to update app

Certificate Pinning

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config>
        <domain includeSubdomains="true">mobiusconf.com</domain>
        <pin-set expiration=“2019-05-23">
            <pin digest=“SHA-256”>7HIpactkIAq2Y4…oQYcRhJ3Y=</pin>
        </pin-set>
    </domain-config>
</network-security-config>
```

Certificate Pinning

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config>
        <domain includeSubdomains="true">mobiusconf.com</domain>
        <pin-set expiration="“2019-05-23">
            <pin digest=“SHA-256”>7HIpactkIAq2Y4…oQYcRhJ3Y=</pin>
            <!-- backup pin -->
            <pin digest=“SHA-256”>fwza0LRMXouZRC…f3O/DM1oE=</pin>
        </pin-set>
    </domain-config>
</network-security-config>
```

Block clear text traffic

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config cleartextTrafficPermitted="false">
        <domain includeSubdomains=“true">secure.toothpic.com</domain>
    </domain-config>
</network-security-config>
```

Block clear text traffic

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config cleartextTrafficPermitted="false">
        <domain includeSubdomains=“true">secure.toothpic.com</domain>
    </domain-config>
</network-security-config>
```

Android Network Security Config

Read more at

https://developer.android.com/training/articles/security-config

Try the code lab

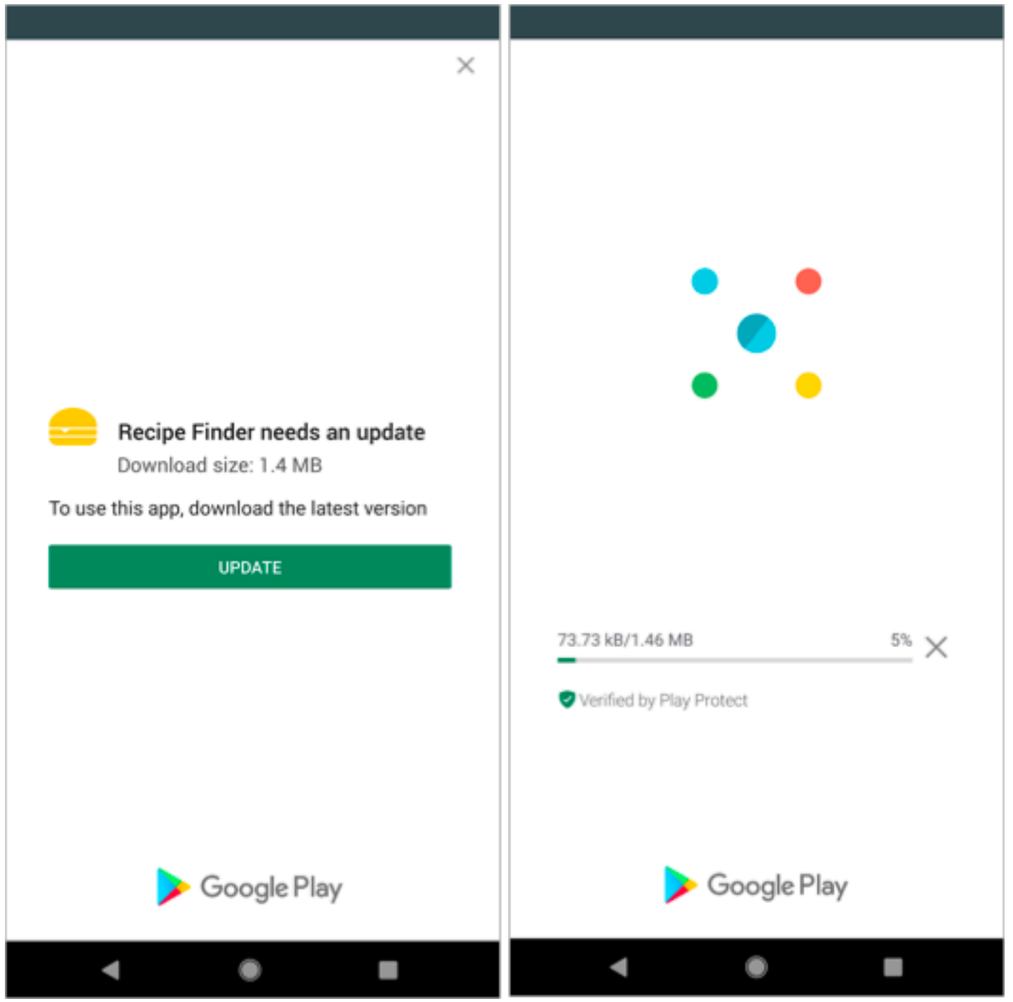
https://codelabs.developers.google.com/codelabs/android-network-security-config

Force Update

Force Update

Having the ability to force users to update can help mitigate impact of security bugs if they do arise.

Not everyone enables background updates.

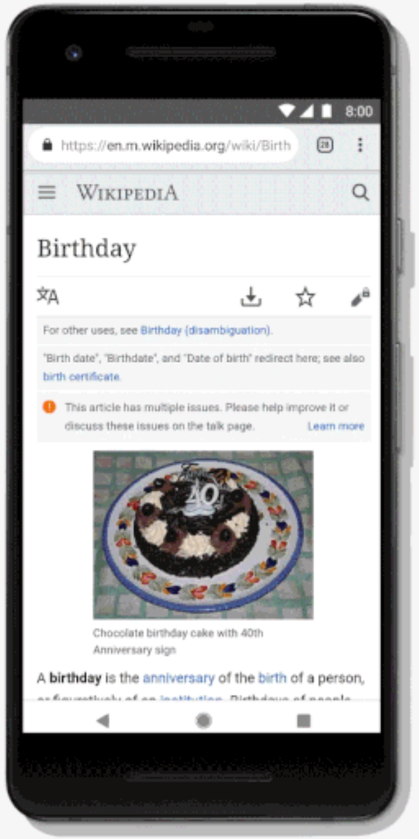


Force Update

In-app updates is a Play Core library feature that introduces a new request flow to prompt active users to update your app.

2 modes:
Immediate and Flexible

Immediate in-app update



Flexible in-app update

Implementation

```
// Creates instance of the manager.
AppUpdateManager appUpdateManager =
AppUpdateManagerFactory.create(context);

// Returns an intent object that you use to check for an update.
Task<AppUpdateInfo> appUpdateInfoTask =
appUpdateManager.getAppUpdateInfo();
```

Implementation

```
// Checks that the platform will allow the type of update.
appUpdateInfoTask.addOnSuccessListener(appUpdateInfo -> {
    if (appUpdateInfo.updateAvailability() == UPDATE_AVAILABLE
            && appUpdateInfo.isUpdateTypeAllowed(IMMEDIATE)) {
                // Request the update.
    }
});
```

Implementation

```
// Checks that the platform will allow the type of update.
appUpdateInfoTask.addOnSuccessListener(appUpdateInfo -> {
    if (appUpdateInfo.updateAvailability() == UPDATE_AVAILABLE
            && appUpdateInfo.isUpdateTypeAllowed(FLEXIBLE)) {
                // Request the update.
    }
});
```

Implementation

```
appUpdateManager.startUpdateFlowForResult(
    // Pass the intent that is returned by 'getAppUpdateInfo()'.
    appUpdateInfo,
    // Or 'AppUpdateType.FLEXIBLE' for flexible updates.
    AppUpdateType.IMMEDIATE,
    // The current activity making the update request.
    this,
    // Include a request code to later monitor this update request.
    MY_REQUEST_CODE);
```

Force Update

In-App Updates

https://developer.android.com/guide/app-bundle/in-app-updates

SafetyNet

SafetyNet APIs

Attestation API

Safe Browsing API

reCAPTCHA API

Verify Apps API

SafetyNet APIs

Attestation API

Safe Browsing API

reCAPTCHA API

Verify Apps API

SafetyNet Attestation API

Checks for:

- Device rooting
- Compatibility Test Suite (CTS) match

Compatibility Test Suite

Anyone can build hardware that runs the Android.

Compatibility Test Suite

Anyone can build hardware that runs the Android.

To qualify as "Android compatible" device must pass CTS.

Implementation Example: Payment

Android App

Toothpic API

Implementation Example: Payment

Android App

Send metadata associated with request

e.g. payment_id

amount

Toothpic API

Implementation Example: Payment

Android App

Send metadata associated with request

e.g. payment_id

amount

Toothpic API

Implementation Example: Payment

Android App

Toothpic API

Implementation Example: Payment

Android App

SafetyNet API

Implementation Example: Payment

```
client.attest(nonce, apiKey)
```

Android App

SafetyNet API

Implementation Example: Payment

```
client.attest(nonce, apiKey)
```

Android App

SafetyNet API

Implementation Example: Payment

```
client.attest(nonce, apiKey)
```

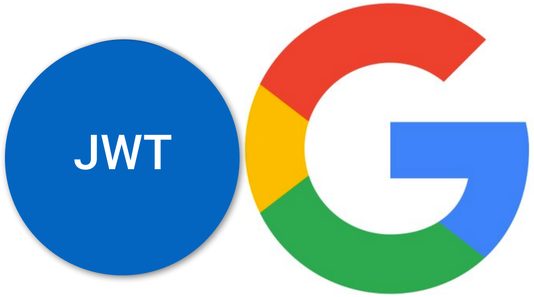
Android App

SafetyNet API

Implementation Example: Payment

Android App

SafetyNet generates a JSON Web Token

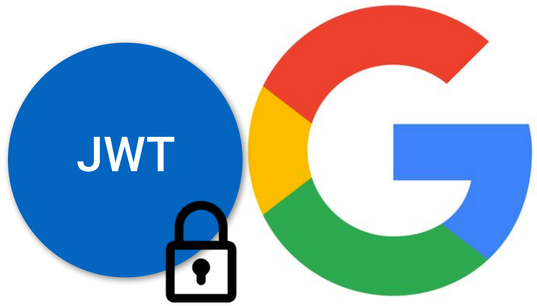


SafetyNet API

Implementation Example: Payment

Android App

SafetyNet generates a JSON Web Token and signs it with Google private key



SafetyNet API

Implementation Example: Payment

Android App

SafetyNet API

Implementation Example: Payment

Android App

And that’s it, right?

Implementation Example: Payment

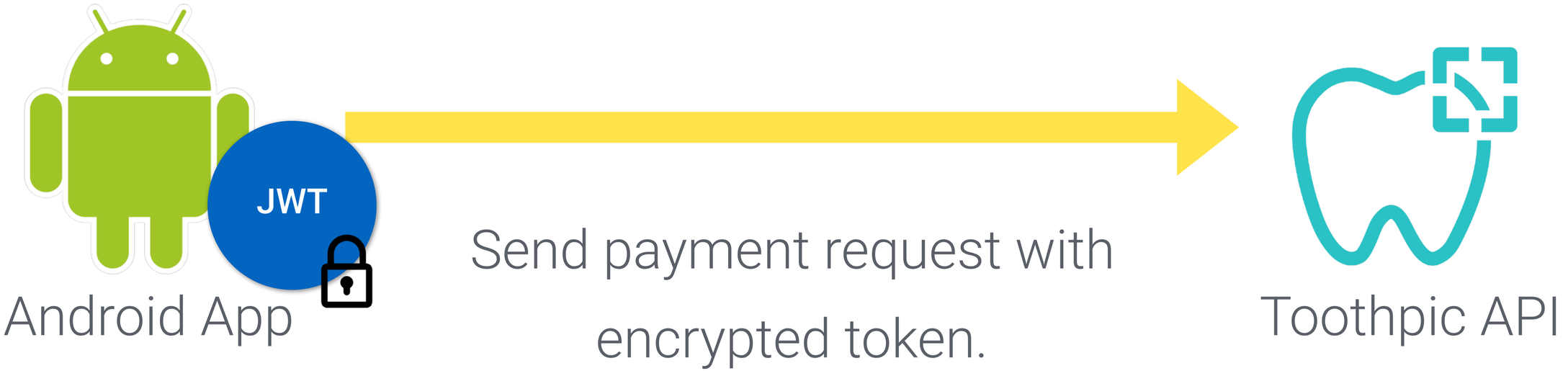
Android App

And that’s it, right?

Wrong!

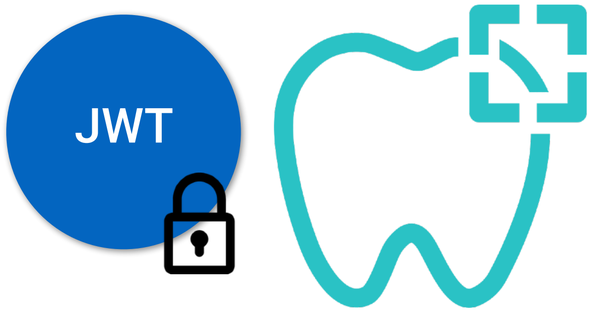
We can’t trust the client.

Implementation Example: Payment



Implementation Example: Payment

Android App

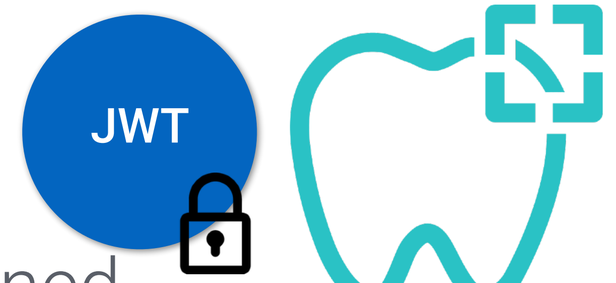


Toothpic API

Implementation Example: Payment

Android App

Server verifies that it was signed by Google certificate chain



Toothpic API

Implementation Example: Payment

Android App

Server verifies that it was signed by Google certificate chain and decrypts it

Toothpic API

Implementation Example: Payment

Android App

nonce
timestamp
apkCertificateDigestSha256
ctsProfileMatch
basicIntegrity

Toothpic API

Possible attestation results

Device Status	ctsProfileMatch	basicIntegrity
Certified, genuine device that passes CTS	✓	✓
Certified device with unlocked bootloader	✗	✓
Device with custom ROM (not rooted)	✗	✓
Emulator	✗	✗
Signs of system integrity compromise, such as rooting	✗	✗
Signs of other active attacks, such as API hooking	✗	✗

source: https://developer.android.com/training/safetynet/attestation#possible-results

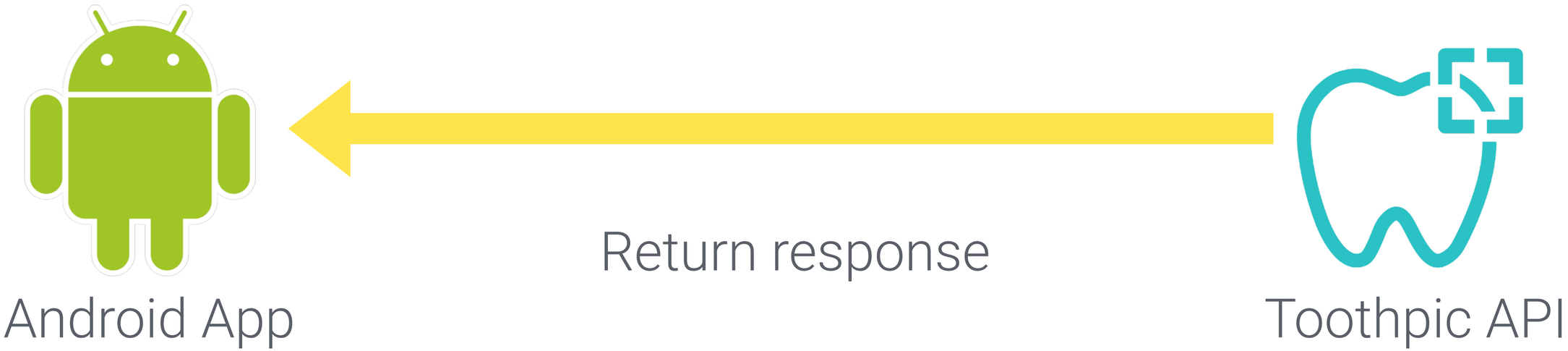
Implementation Example: Payment

Android App

nonce
timestamp
apkCertificateDigestSha256
ctsProfileMatch
basicIntegrity

Toothpic API

Implementation Example: Payment



Implementation Example: Payment

SafetyNet Attestation API

Android App

Toothpic API

SafetyNet API

SafetyNet Attestation

Read more at

https://developer.android.com/training/safetynet/attestation

Clone the samples

https://github.com/googlesamples/android-play-safetynet

Leaking Data

Be Careful of… Logging

You should treat your application logs as public.

- Other apps on the device may have `READ_LOGS` permission.
- Could be exported to 3rd party services such as Crashlytics as a result of Exception handling or application crash.

Be Careful of… Analytics

Grindr sends HIV status to third parties, and some personal data unencrypted

Devin Coldewey @techcrunch / Apr 2, 2018

Comment

source: https://techcrunch.com/2018/04/02/grindr-sends-hiv-status-to-third-parties-and-some-personal-data-unencrypted/

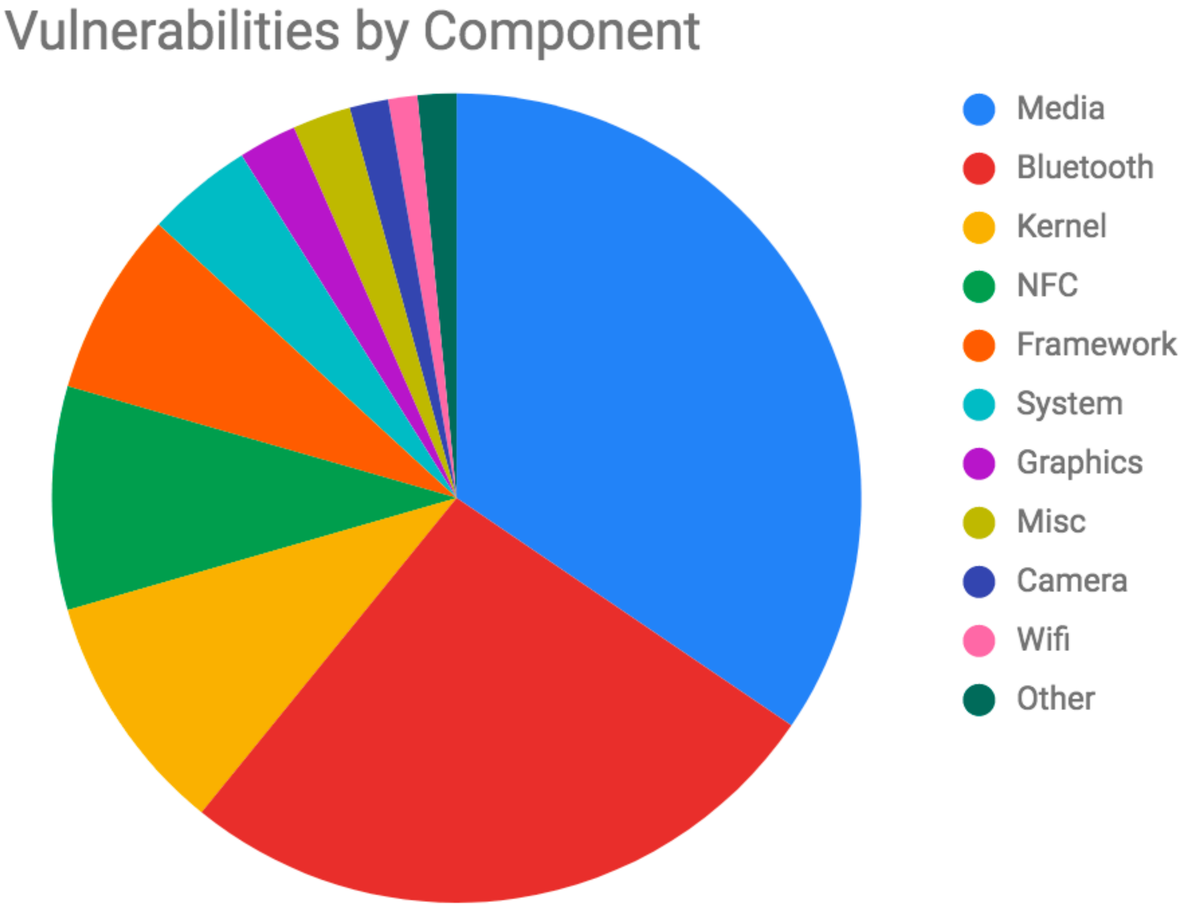
Only collect what you need

You can’t leak data you don’t have.

New in Q

New in Q

- **TLS 1.3** enabled by default
- Platform hardending
- Significant improvements to **Biometrics API**
- **Jetpack security** library alpha



New in Q

- **Scoped storage**
- More user control over **location permissions**
- New restrictions on launching Activities from background
- New restrictions on accessing device serial and IMEI
- Access to some Wi-Fi and Bluetooth scanning methods requires location permission

Types of Data	Identify Risks	Improve Security
PII	Automated Tools	“Raise the Bar”
PHI	Dev. Knowledge	Android Sandbox
Crash Reports	Penetration Test	Obfuscation
Analytics		Net. Security Config
		Force Update
		SafetyNet
		Leaking Data
		New in Q

Thank You!

@mauricegavin