

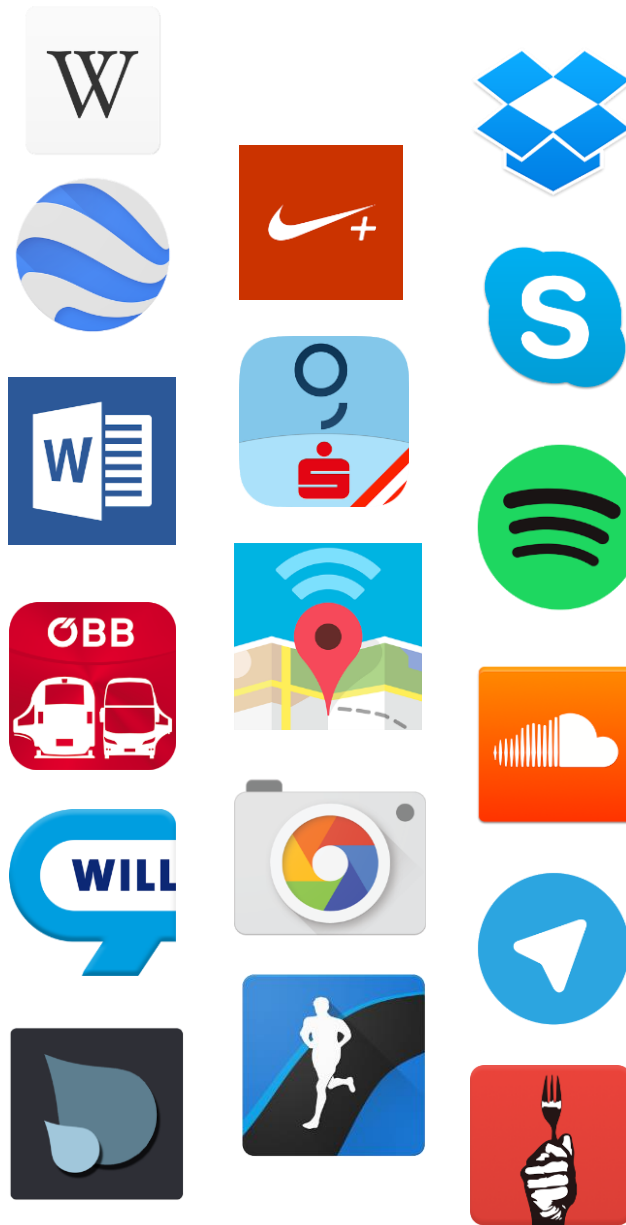
# Assignment 1

*Mobile Security 2021*

Johannes Feichtner  
[johannes.feichtner@iaik.tugraz.at](mailto:johannes.feichtner@iaik.tugraz.at)



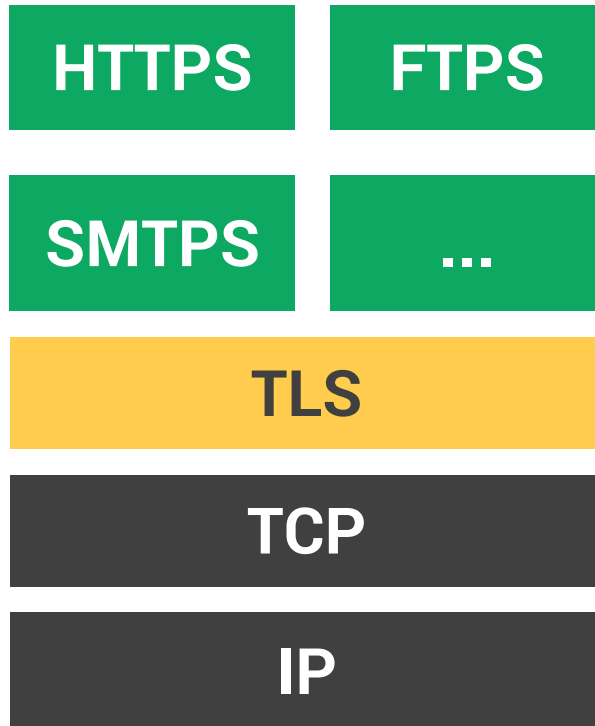
...wants  
privacy



- *Am I talking to who I think I do?*
- *Does anyone tamper my data?*
- *Who else can see my conversation?*

# Transport Layer Security

Problem: „Secure Identity“



**Authentication / Availability**

*Am I talking to who I think I do?*

**Data integrity**

*Does anyone tamper with my data?*

**Confidentiality**

*Who else can see my conversation?*

Problem: Key Exchange

# Man-in-the-middle

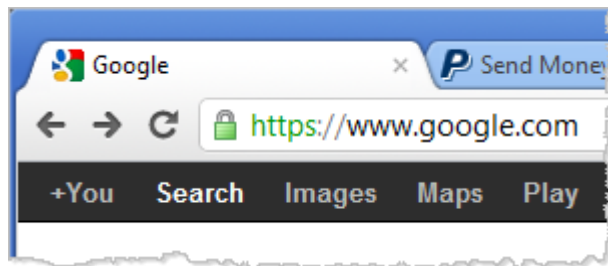


## Active attacker

*Secretly relay (and possibly modify) traffic between client and server*



google.com



www.google.com

## Client

*Ideally does not notice anything  
(from an attacker's perspective)*

# Practical Defenses

- Only accept specific server certificates
  - Authenticate certificates using a 2<sup>nd</sup> channel: DNSSEC (DANE)
  - **Certificate Pinning**
    - Send public key hashes in HTTP header

## Public-Key-Pins:

```
pin-sha256="GRAH5Ex+kB4cCQi5gMU82urf+6kEgbVtzfCSkw55AGk=";  
pin-sha256="lERGk61FITjzyKHcJ89xpc6aDwtRkOPAU0jdnUqzW2s=";  
max-age=15768000; includeSubDomains
```

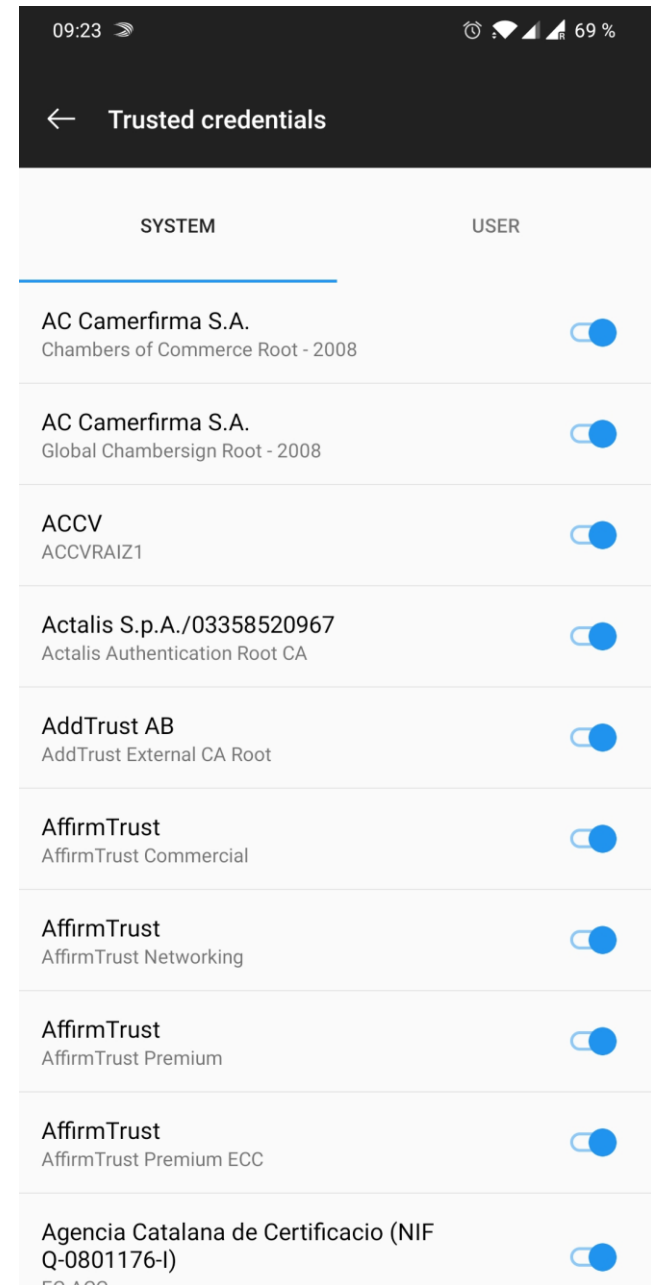
- **Bundle public key hashes with application**
- Apply mutual PKI authentication
  - Server and client mutually validate their certificates → VPN

# The Problem

# Eve and Mallory mobiles

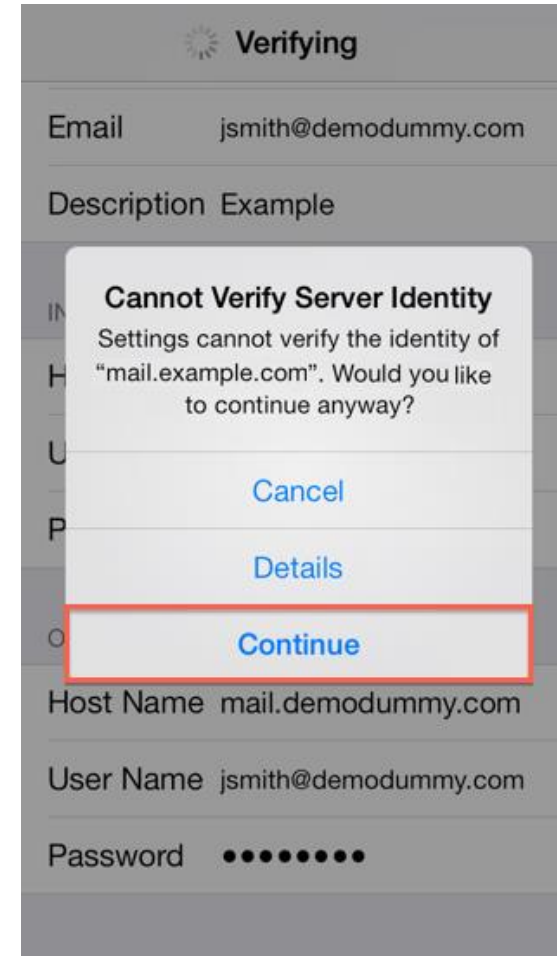
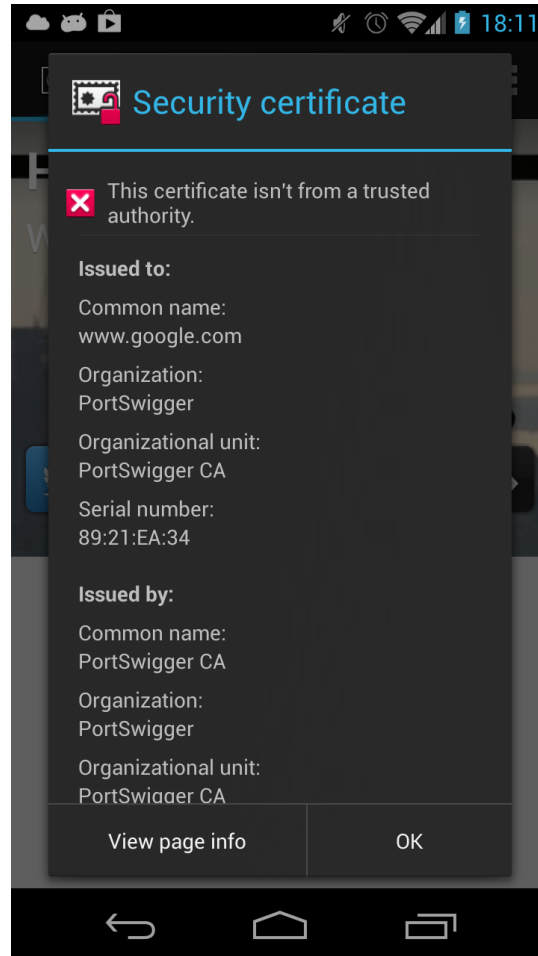
## General challenges

- Servers have to be trusted by a CA (~130 pre-installed)
- Certificates have to be...
  - ... renewed regularly
  - ... could be revoked (Check using OCSP, CRL)
  - ... validated correctly
- Detect / prevent:
  - Man-in-the-middle (MITM)
  - Fraudulent certificates, issued by compromised CAs



# Fact Check

Default behavior: Correct implementation of TLS on major OS





# Current situation

Q: “Does someone know how to accept a self-signed certificate on Android?

A code sample would be perfect.”

A: “Use the AcceptAllTrustManager”.

Q: “All I need to do is download some basic text-based and image files from a web server that has a self-signed SSL certificate...getting the SSL to work is a nightmare...”

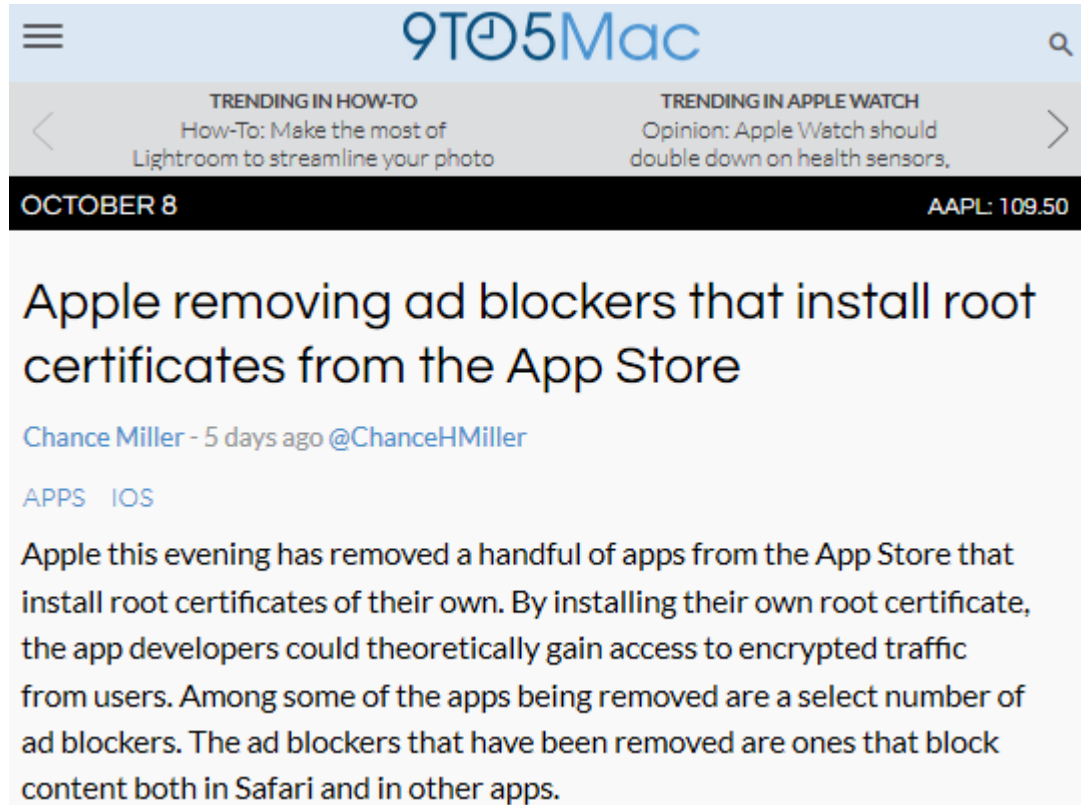
A: “I found two great examples of how to accept self-signed SSL certificates, one each for `HttpsURLConnection` and `HttpClient`.”

[Source: Stackoverflow]

## Mobile Apps

- Can overwrite certificate validation routines (system default: correct check)
- Might use self-signed certificates → require custom TrustManager / profiles
- Have to implement pinning on their own if wanted

# MITM hitting iOS



The screenshot shows the 9to5Mac website interface. At the top, there's a navigation bar with the site logo, a search icon, and a menu icon. Below this, there are two trending article sections: 'TRENDING IN HOW-TO' with the article 'How-To: Make the most of Lightroom to streamline your photo' and 'TRENDING IN APPLE WATCH' with the article 'Opinion: Apple Watch should double down on health sensors,'. A black bar below the trending sections displays 'OCTOBER 8' on the left and 'AAPL: 109.50' on the right. The main article title is 'Apple removing ad blockers that install root certificates from the App Store'. The author is 'Chance Miller - 5 days ago @ChanceHMiller'. The article is categorized under 'APPS' and 'IOS'. The main text of the article states: 'Apple this evening has removed a handful of apps from the App Store that install root certificates of their own. By installing their own root certificate, the app developers could theoretically gain access to encrypted traffic from users. Among some of the apps being removed are a select number of ad blockers. The ad blockers that have been removed are ones that block content both in Safari and in other apps.'

Source: <http://goo.gl/Ucikbg>



The image shows the top portion of an article from ars TECHNICA. The site's logo is in the top left, with 'SUBSCRIBE' and 'SIGN IN' buttons in the top right. The article title is 'Dozens of popular iOS apps vulnerable to intercept of TLS-protected data'. Below the title, it says '76 apps in Apple's App Store still don't use best practices to protect user data.' and 'SEAN GALLAGHER - 2/7/2017, 12:38 AM'. The main image is a corkboard with a yellow sticky note pinned to it. The sticky note has a hand-drawn padlock on it. Below the image, there is a paragraph of text: 'While developing a tool for evaluating mobile application security, researchers at Sudo Security Group Inc. found out something unexpected. Seventy-six popular applications in Apple's iOS App Store, they discovered, had implemented encrypted communications with their back-end services in such a way that user information could be intercepted by a man-in-the-middle attack. The applications could be fooled by a forged certificate sent back by a proxy, allowing their Transport

Source: <http://goo.gl/wABxVg>

# MITM hitting Android

## Application vulnerabilities

+ Libraries!

- Trusting all certificates
  - Custom-implement `javax.net.ssl.TrustManager` → `checkServerTrusted(...)` { }
- Allow all hostnames
  - Check certificate signature but allow `attacker.com` for `google.com`
- Ignore TLS errors in WebKit
  - Configure browser component to suppress errors
- Implement your own „sophisticated“ validation
  - E.g. check certificate common name against hard-coded `xy.com`

# Certificate Pinning

## Problem:

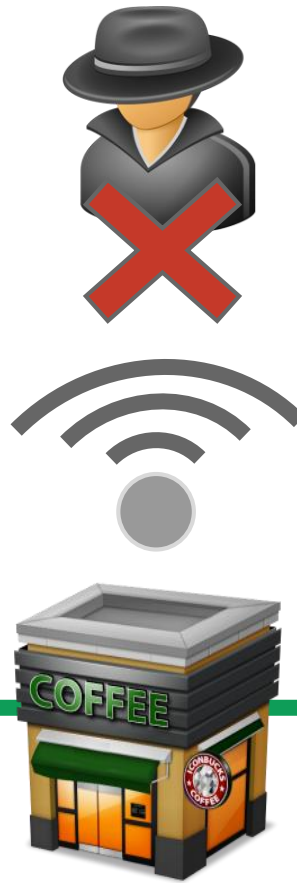
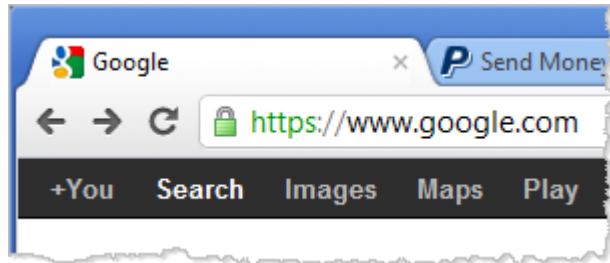
What if the Certificate Authority (CA), having issued the certificate, has been compromised?

## Status quo:

- Some libraries available but no OS provided support for Pinning
  - Remarkable exception: Android  $\geq 7$
- Different approaches to implement this feature
  - What to pin? Only end-user certificate? CA also?
- Implementation requires overwriting the entire certificate validation!

# Your Task

# What we want...



## Passive attacker

*It's all encrypted! Random bits...*

## Active attacker

*MITM failed? Fake cert not eaten?*

<https://www.google.com>

<https://www.apple.com>

<https://www.microsoft.com>

End-to-end encrypted communication channel

# Task 1

**Analyse** a set of min. 8 applications for either Android, iOS, or Windows Phone

- Find out if they are susceptible to MITM / detected by app
- Whether they make use of Certificate Pinning

## Roadmap

1. Select and install arbitrary apps on your phone
2. Get used to the topic of MITM / Pinning and learn an attack tool
3. Probe the chosen apps and summarize your results

**Grading of Task 1:** Your result report + conducted spot tests

Major impact on grade: Task 2 **but** positive finish only if you solve Task 1 **and** 2!

# On the dark side...



## MITM attack tools

- Burp Suite (free edition)
- Fiddler
- mitmproxy.org
- Ettercap
- .. or any other – ideally with trust profile for iOS / proxy listener for Android

## Tip: Benefit from this task in the long term!

- Get familiar with at least one of these tools
- Understand MITM and Pinning & verify the concepts on real-world apps



# Submission

Submit until 10.04.2021:


- No strict format but PDF recommended
- List of analysed apps and versions
- Used attack tool?
- **Describe how** you analysed each of the applications
  - Text, screenshots, excerpts from PCAP dumps etc.
- Find out whether:
  - MITM detected by app → User warned?
  - Hostname forging possible? Certificate Pinning used? → Reverse-engineer the app!
  - Eventually, if retrievable:
    - Other nice things like hard-coded credentials / PINs
  - Yes/no answers are NOT sufficient! Provide evidence („why“) for your conclusions

# Submission cont.

Submit until 10.04.2021:

- ZIP file with all traffic dumps / exported files from MITM tool
- Filename: *[your matr. number\_lastname\_firstname].zip*
  - E.g. *01234567\_feichtner\_johannes.zip*
- Upload to <https://seafile.iaik.tugraz.at/u/d/212eaa8a8cd146c780cf/>

Dateien hochladen zu **acn21t1**

freigegeben von: Johannes Feichtner 

+ Dateien hinzufügen

+ Ordner hinzufügen

⊘ Alle abbrechen

1. Datei Drag & Drop wird von Chrome, Safari 5.0+, Firefox 4.0+ und IE 10.0+ unterstützt.

2. Ordner Drag & Drop wird von Chrome unterstützt.

# Questions?

