Automated Mobile Application Security Testing with Mobile Security Framework

Ajin Abraham

About Me

- Security Engineering @ IMMUNIO
  - Next Gen Runtime Application Self Protection (RASP)
- Author of OWASP Xenotix XSS Exploit Framework, Mobile Security Framework.
- Teach Security via https://opsecx.com
- Blog about Security: http://opensecurity.in

ROMA MMXVI APPSEC EUROPE

The Takeaways

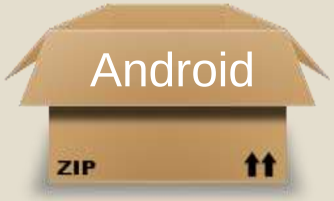
- A FREE and Open Source Security Tool for Mobile App Security Assessment.
- Mobile App Pentesters/Mobile Malware Analysts - How to make your job easier with MobSF.
- Developers – Build secure mobile Apps identifying vulnerabilities at all stages of development. (SDLC Integration)
- Web Pentesters – REST API Fuzzer capable of detecting vulnerabilities like SSRF, XXE, IDOR etc.

Agenda

- What is MobSF?
- MobSF Architecture
  - Static Analyzer
  - Dynamic Analyzer
  - Web API Fuzzer
- Static Analysis
  - Static Analysis & some Statistics
  - Top Indian and European Bank Mobile Apps
  - Top Indian and European Wallet Mobile Apps
  - Observations
- Dynamic Analysis
  - Dynamic SSL Testing
  - Exported Activity Tester
  - Challenges in Dynamic Analysis
  - Dynamic Analysis on Custom VM/ Rooted Android Device.
- Web API Fuzzer
  - Vulnerabilities API Fuzzer detects.
  - Explains the API Fuzzer Logic.
- Conclusion

What is MobSF?

Mobile Security Framework is an open source mobile application (Android/iOS) automated pentesting framework capable of performing end to end security testing of mobile Apps.

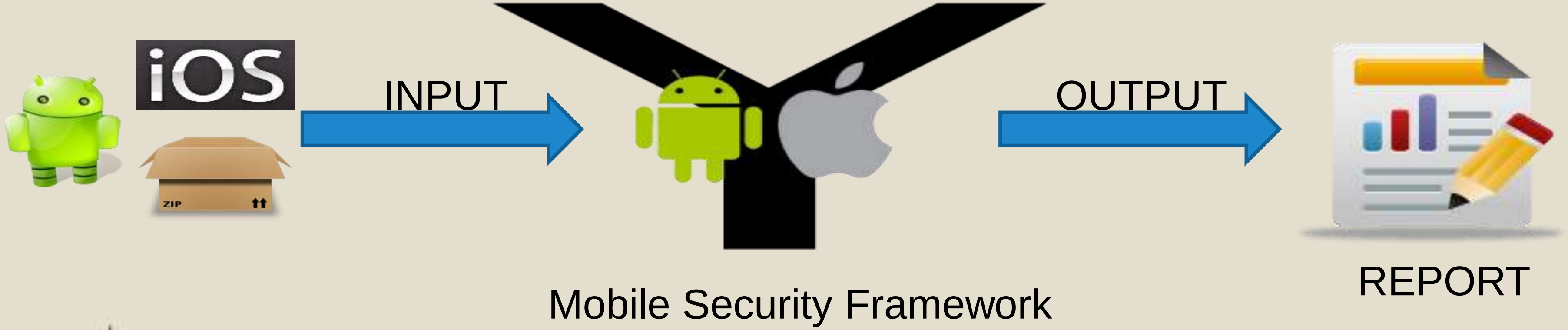


Hosted in your environment. Your application and data is never send to the cloud.

ROMA MMXVI APPSEC EUROPE

MobSF Architecture

Static Analyzer



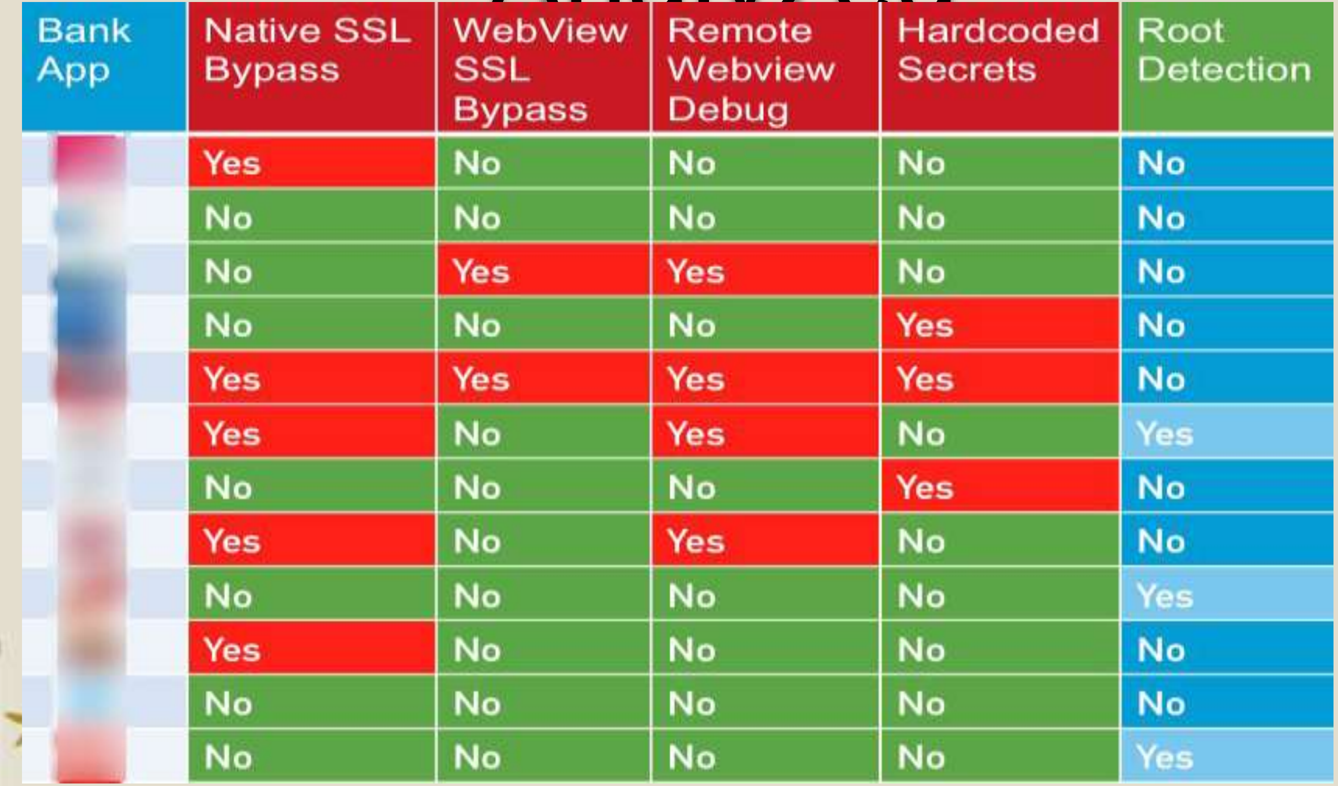
Demo

Static Analysis & Report Generation

Static Analysis & Some Statistics

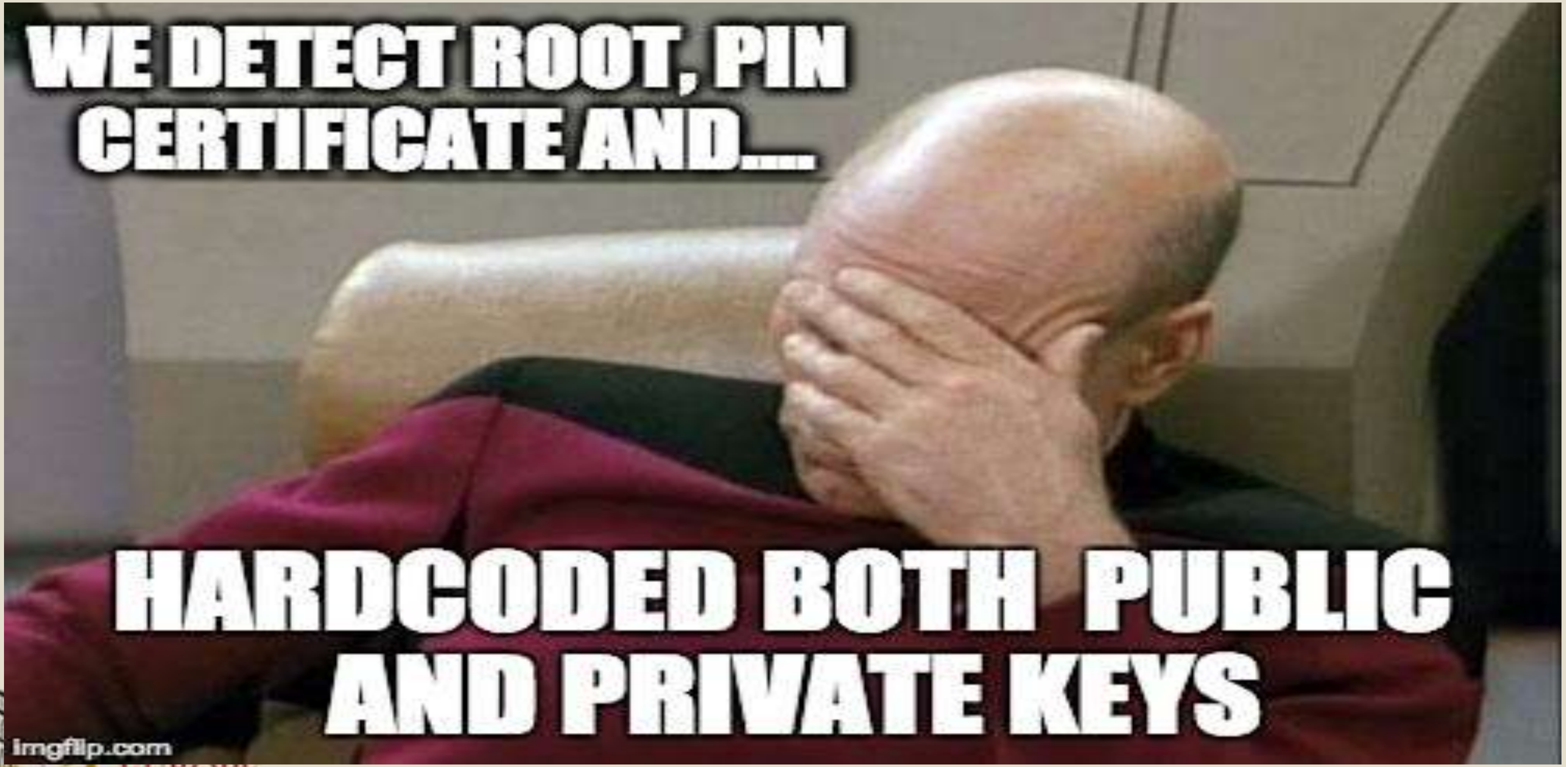
- Static Analysis on Top Financial Apps - Criteria
  - SSL bypass in Native Code
  - SSL bypass in WebView
  - Weak Crypto
  - Remote Web View Debugging
  - Hardcoded Secrets

Top Indian Bank Apps Analyzed

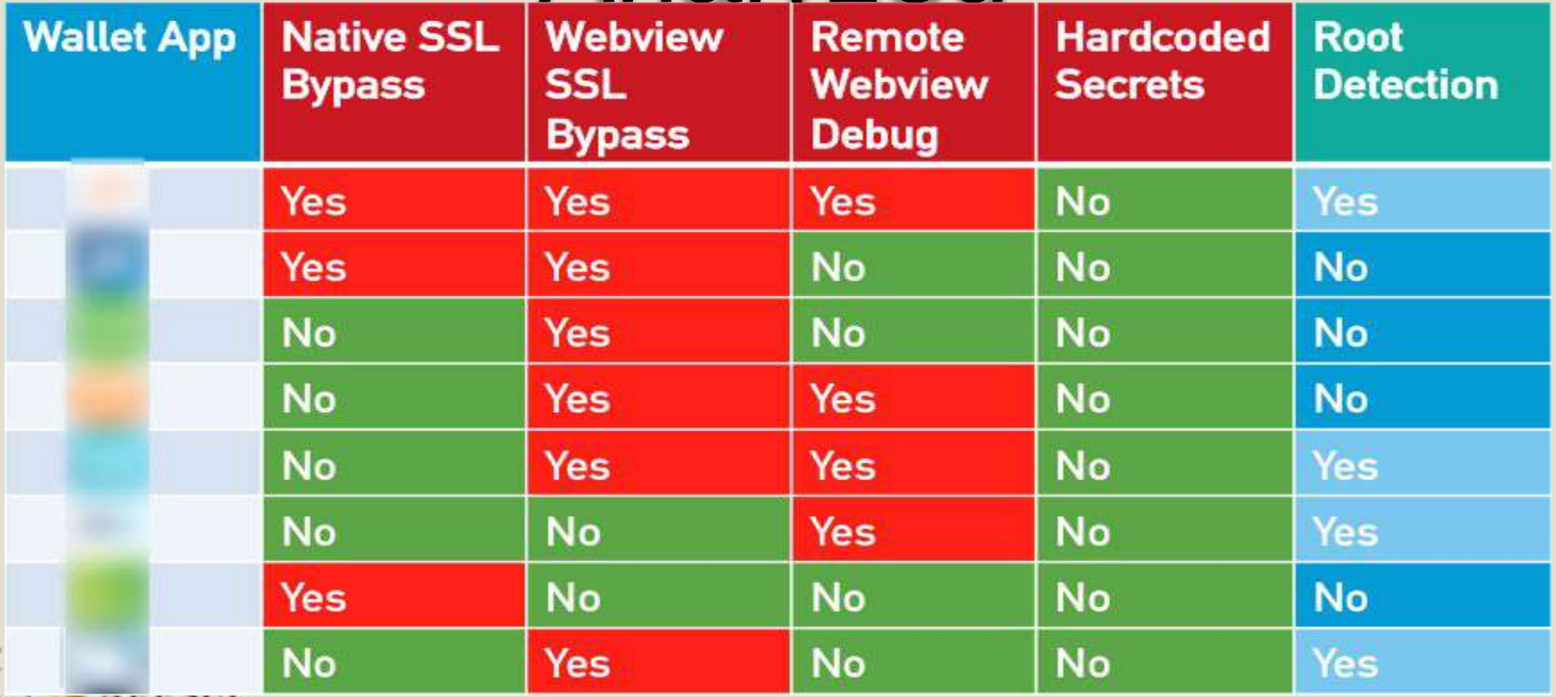


Bank App	Native SSL Bypass	WebView SSL Bypass	Remote Webview Debug	Hardcoded Secrets	Root Detection
[redacted]	Yes	No	No	No	No
[redacted]	No	No	No	No	No
[redacted]	No	Yes	Yes	No	No
[redacted]	No	No	No	Yes	No
[redacted]	Yes	Yes	Yes	Yes	No
[redacted]	Yes	No	Yes	No	Yes
[redacted]	No	No	No	Yes	No
[redacted]	Yes	No	Yes	No	No
[redacted]	No	No	No	No	Yes
[redacted]	Yes	No	No	No	No
[redacted]	No	No	No	No	No
[redacted]	No	No	No	No	Yes

Face palm

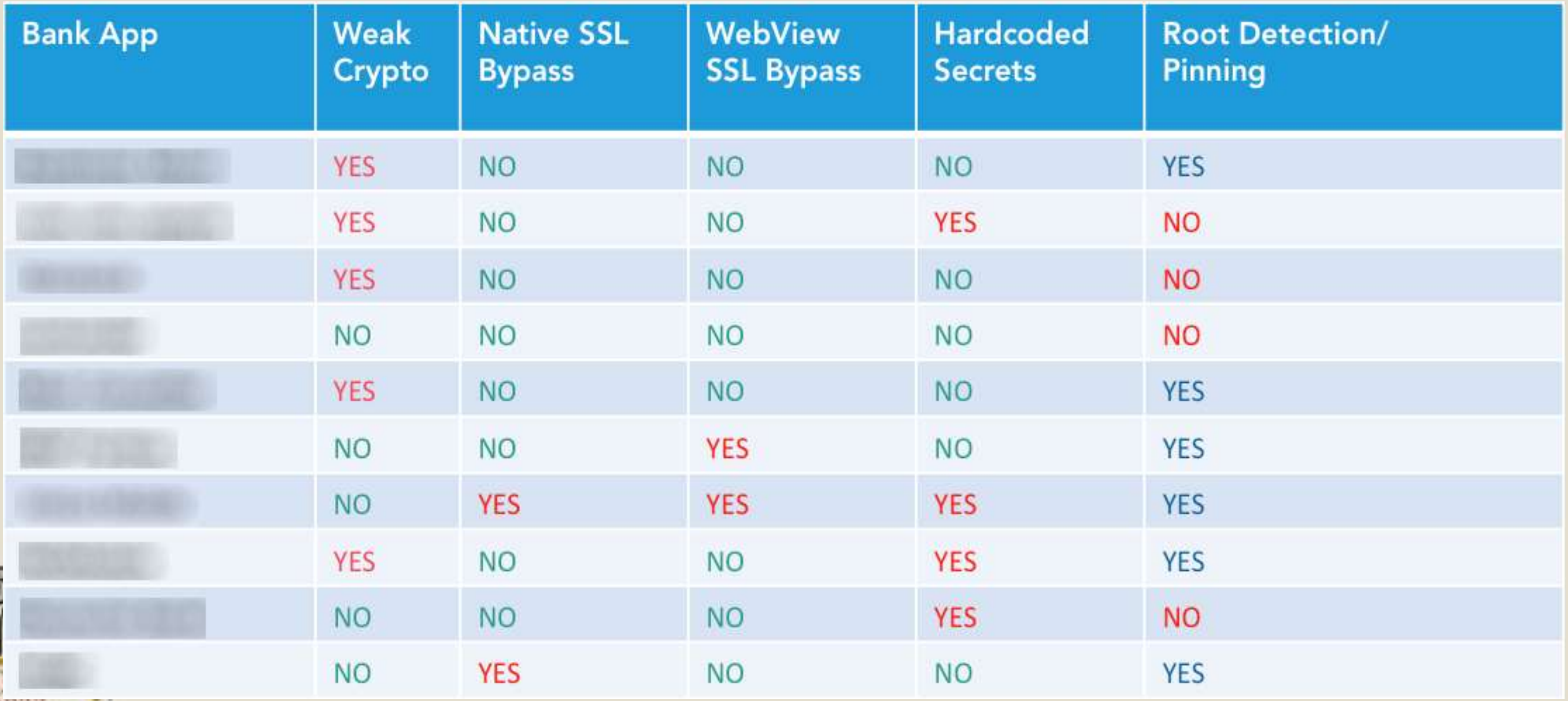

WE DETECT ROOT, PIN CERTIFICATE AND....
HARDCODED BOTH PUBLIC AND PRIVATE KEYS
imgflip.com

Top Indian Wallet Apps Analyzed



Wallet App	Native SSL Bypass	Webview SSL Bypass	Remote Webview Debug	Hardcoded Secrets	Root Detection
	Yes	Yes	Yes	No	Yes
	Yes	Yes	No	No	No
	No	Yes	No	No	No
	No	Yes	Yes	No	No
	No	Yes	Yes	No	Yes
	No	No	Yes	No	Yes
	Yes	No	No	No	No
	No	Yes	No	No	Yes

Top EU Bank Apps Analyzed



Bank App	Weak Crypto	Native SSL Bypass	WebView SSL Bypass	Hardcoded Secrets	Root Detection/ Pinning
[redacted]	YES	NO	NO	NO	YES
[redacted]	YES	NO	NO	YES	NO
[redacted]	YES	NO	NO	NO	NO
[redacted]	NO	NO	NO	NO	NO
[redacted]	YES	NO	NO	NO	YES
[redacted]	NO	NO	YES	NO	YES
[redacted]	NO	YES	YES	YES	YES
[redacted]	YES	NO	NO	YES	YES
[redacted]	NO	NO	NO	YES	NO
[redacted]	NO	YES	NO	NO	YES

Top EU Wallet Apps Analyzed

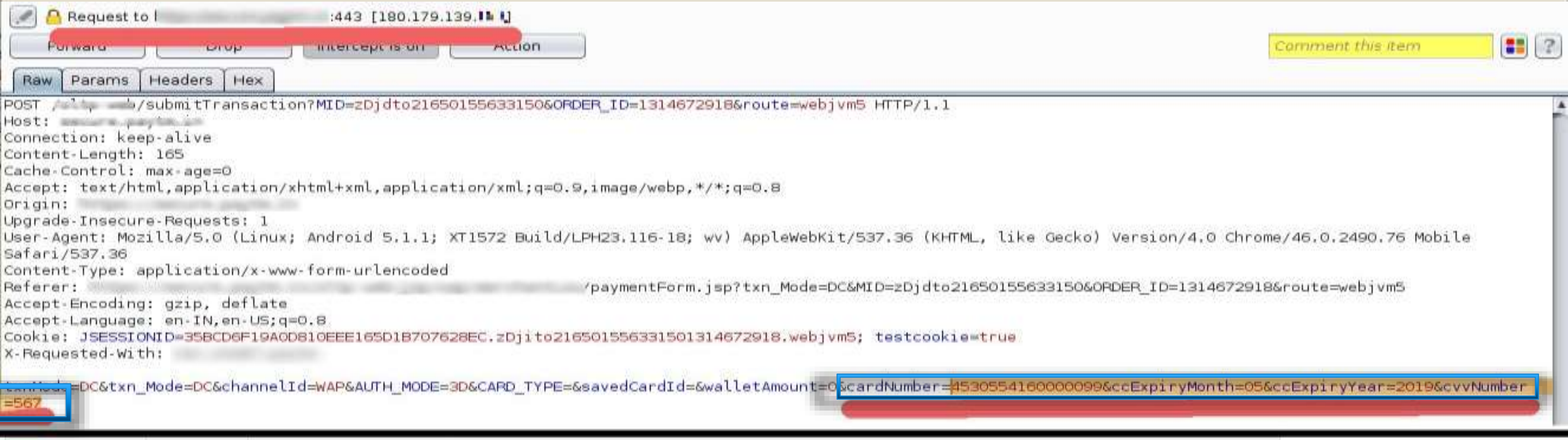
Wallet App	Weak Crypto	Native SSL Bypass	WebView SSL Bypass	Hardcoded Secrets	Root Detection/ Pinning
[redacted]	YES	NO	NO	YES	YES
[redacted]	YES	NO	NO	YES	NO
[redacted]	NO	NO	NO	NO	YES
[redacted]	NO	NO	YES	NO	YES
[redacted]	NO	NO	NO	NO	YES
[redacted]	YES	NO	NO	NO	YES
[redacted]	YES	NO	NO	NO	YES

ROMA MMXVI APPSEC EUROPE

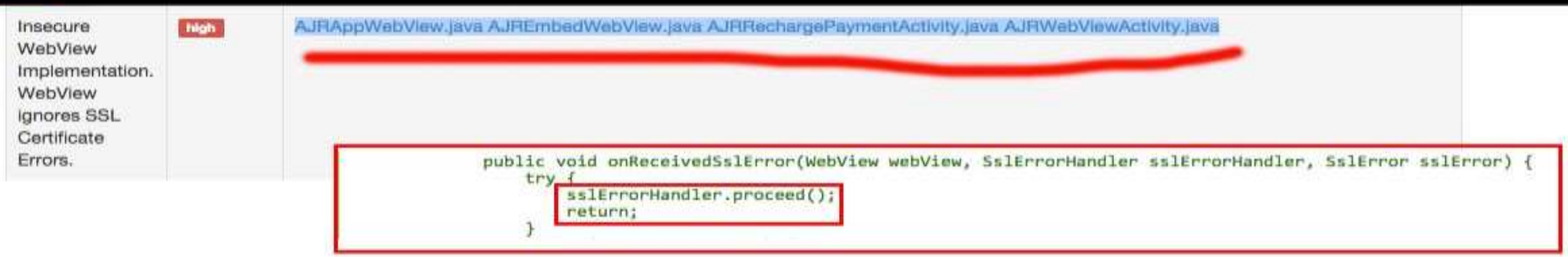
Observations

- State of Mobile App Security, Not evolved as Web Security.
- Most common issue are:
  - Indian Apps: SSL Bypass in (Both Native Code and WebView)
  - European Apps : Weak Crypto and Hardcoded Secrets
- SSL Error bypassed in WebViews are really really bad.

Real-world Exploitation



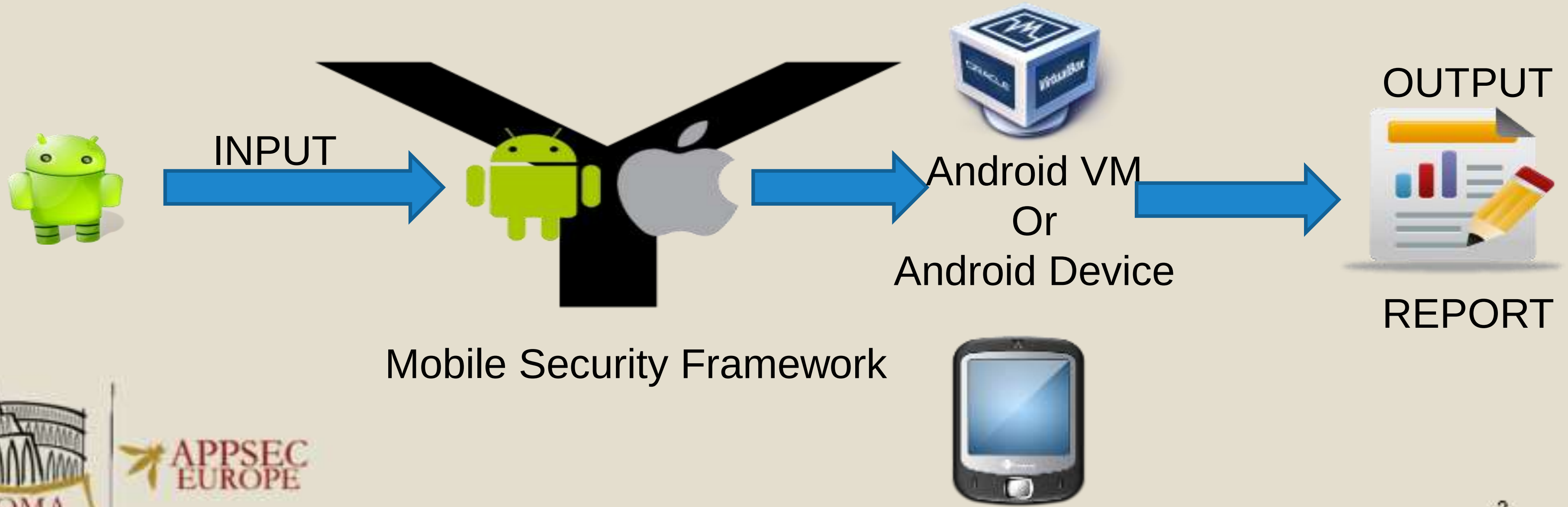
```
POST [redacted]/submitTransaction?MID=zDjdto21650155633150&ORDER_ID=1314672918&route=webjvm5 HTTP/1.1
Host: [redacted]
Connection: keep-alive
Content-Length: 165
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Origin: [redacted]
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Linux; Android 5.1.1; XT1572 Build/LPH23.116-18; wv) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/46.0.2490.76 Mobile
Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer: [redacted]/paymentForm.jsp?txn_Mode=DC&MID=zDjdto21650155633150&ORDER_ID=1314672918&route=webjvm5
Accept-Encoding: gzip, deflate
Accept-Language: en-IN,en-US;q=0.8
Cookie: JSESSIONID=35BCD6F19A0D810EEE165D1B707628EC.zDjito216501556331501314672918.webjvm5; testcookie=true
X-Requested-With: [redacted]

txnMode=DC&txn_Mode=DC&channelId=WAP&AUTH_MODE=3D&CARD_TYPE=&savedCardId=&walletAmount=0&cardNumber=4530554160000099&ccExpiryMonth=05&ccExpiryYear=2019&cvvNumber
=567
```



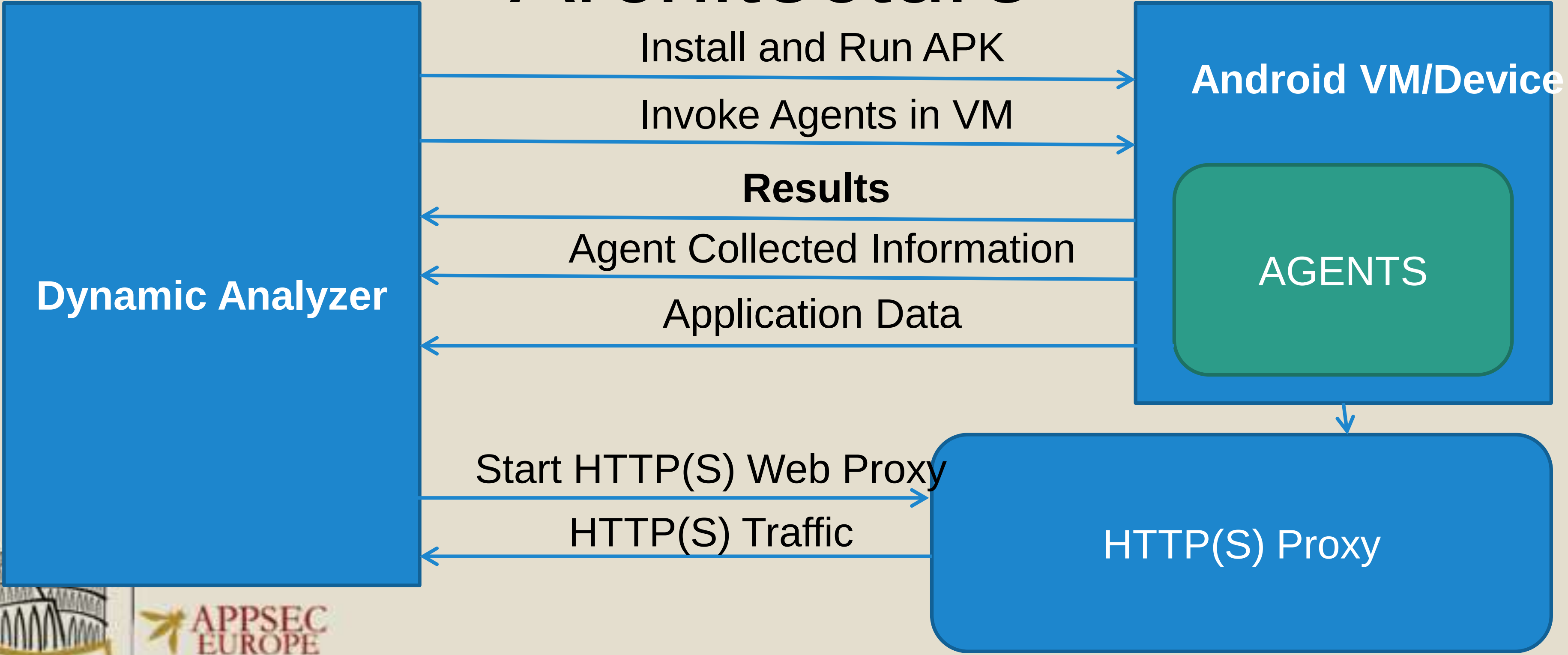
Insecure WebView Implementation. WebView ignores SSL Certificate Errors.	high	AJRAppWebView.java AJREmbedWebView.java AJRRechargePaymentActivity.java AJRWebViewActivity.java

```
public void onReceivedSslError(WebView webView, SslErrorHandler sslErrorHandler, SslError sslError) {
    try {
        sslErrorHandler.proceed();
        return;
    }
```

Dynamic Analyzer



Dynamic Analyzer - Architecture



DEMO

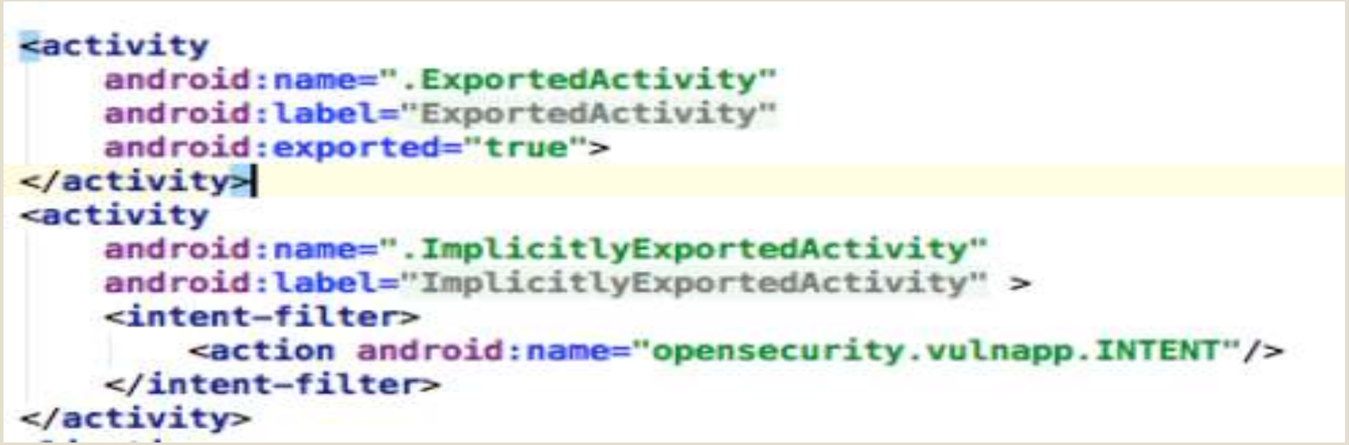
(LockX)

Dynamic SSL Testing

- Dynamically verify if SSL connections are securely implemented.
- Disable JustTrustMe and Remove MobSF Root CA.
- If you can still access the decrypted HTTPS Web Traffic then that means the app is bypassing SSL errors.

Exported Activity Tester

- Android Exported Activities.



```
<activity
    android:name=".ExportedActivity"
    android:label="ExportedActivity"
    android:exported="true">
</activity>
<activity
    android:name=".ImplicitlyExportedActivity"
    android:label="ImplicitlyExportedActivity" >
    <intent-filter>
        <action android:name="opensecurity.vulnapp.INTENT"/>
    </intent-filter>
</activity>
```

DEMO

Challenges in Dynamic Analysis

- Some Android Apps are built with security in mind.
  - Anti VM Detection
  - Anti Root Detection
  - Anti MITM with Certificate Pinning.
  - Missing Libraries/Dependencies
  - Some Apps / Malwares have sophisticated methods to detect Virtual Machines.

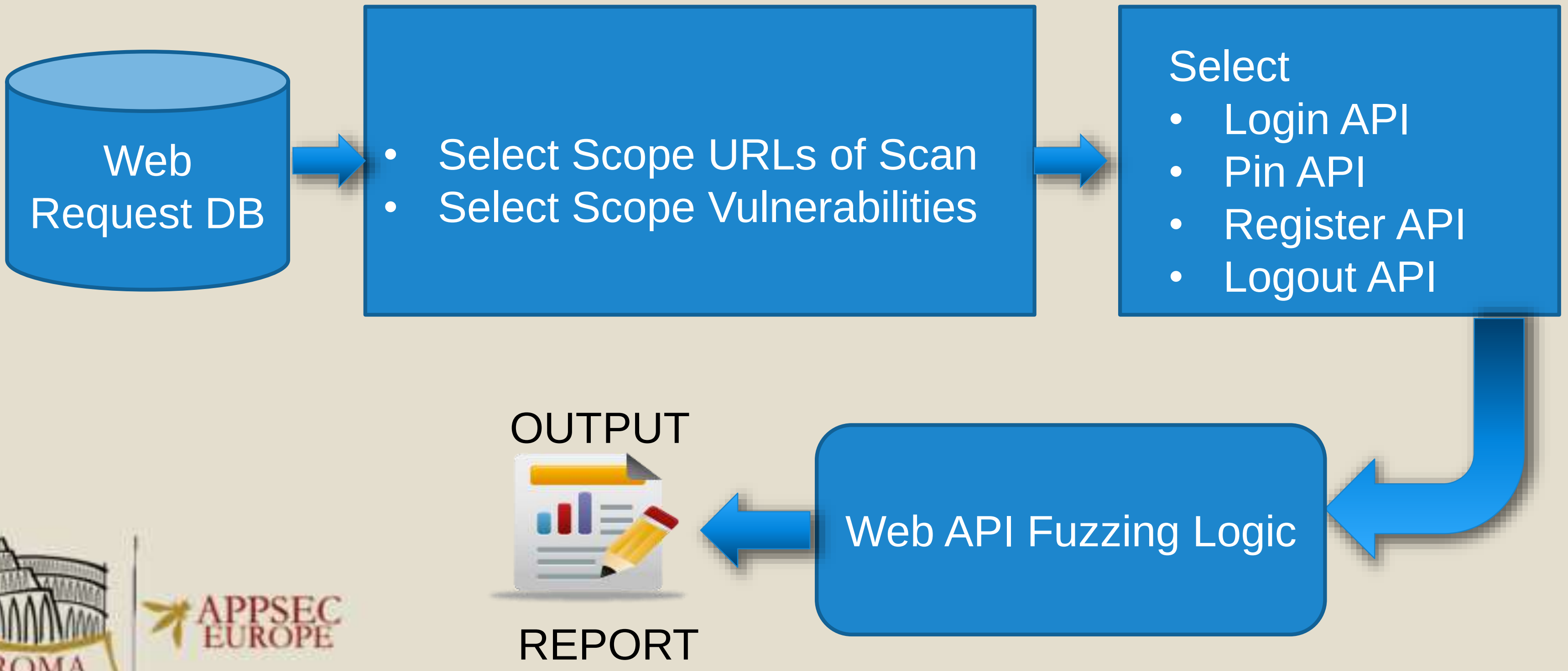
How to deal with these Challenges

- API overriding with Xposed Framework
  - Anti VM Detection Bypass –> Android Blue Pill
  - Anti Root Detection Bypass -> RootCloak
  - Anti MITM Certificate Pinning Bypass -> JustTrustMe
- APK smali Patching, Custom Xposed Module
- For sophisticated apps and malware, Use a real device for dynamic analysis.

Dynamic Analysis on Device

- MobSFy Script – Convert your VM/ Device to support MobSF Dynamic Analysis
- Documentation here: https://github.com/ajinabraham/Mobile-Security-Framework-MobSF/wiki/2.-Configure-MobSF-Dynamic-Analysis-Environment-in-your-Android-Device-or-VM

Web API Fuzzer



Fuzzing REST APIs

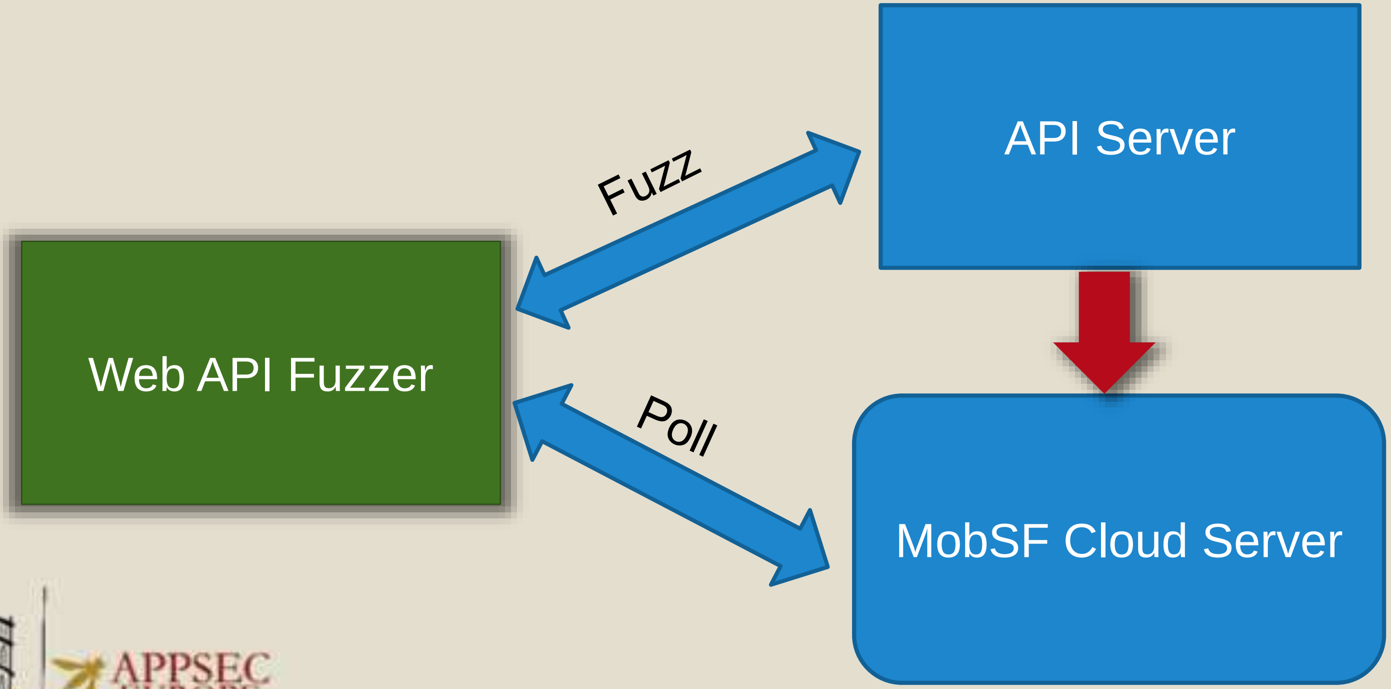
- Why most web scanners suck at API Testing?
- We have knowledge about the application and generic API routes (Login, Logout, Register).
- So we use more of Whitebox approach than Blackbox approach.
- Detects vulnerabilities like IDOR, SSRF and XXE.

What We Detect

- XXE
- SSRF
- IDOR
- Directory Traversal or Path Traversal
- Logical and Session Related
- API Rate Limiting

How we Detect

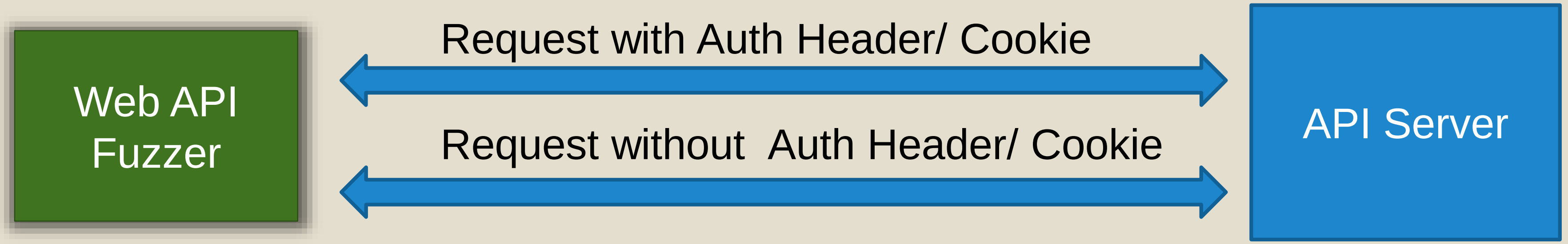
SSRF & XXE



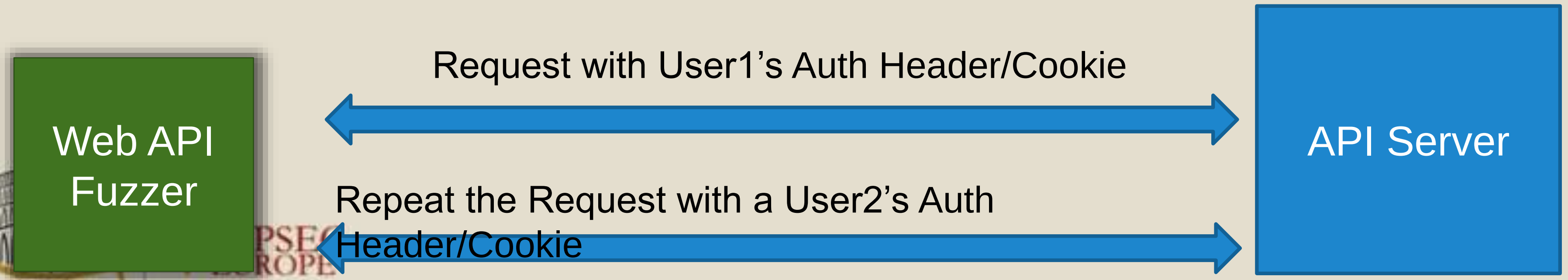
Cloud Server: APITester/cloud/cloud_server.py

Insecure Direct Object Reference (IDOR)

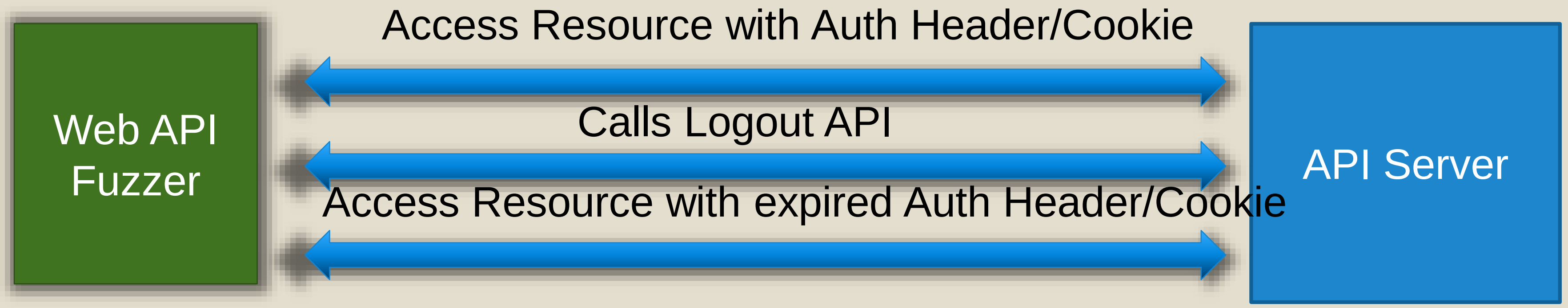
- Without Credentials.


Web API Fuzzer
Request with Auth Header/ Cookie
Request without Auth Header/ Cookie
API Server

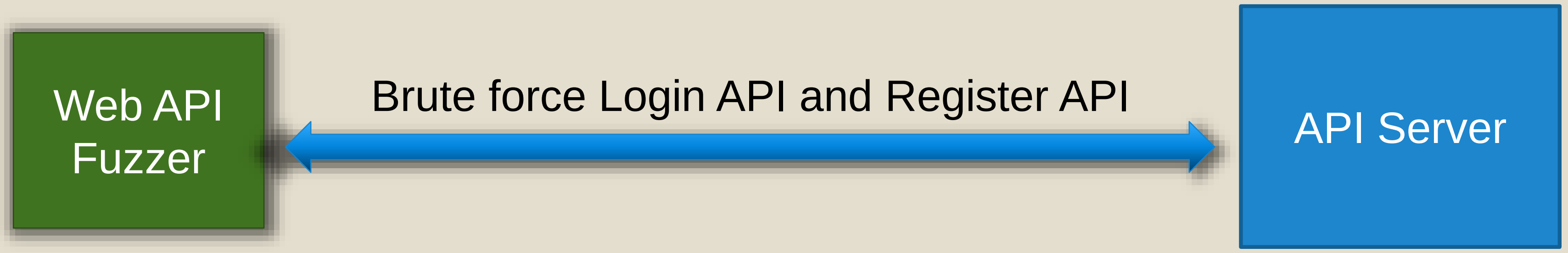
- With multiple user credentials (needs two login attempts)


Web API Fuzzer
Request with User1's Auth Header/Cookie
Repeat the Request with a User2's Auth Header/Cookie
API Server

Session Related Checks


Web API Fuzzer
Access Resource with Auth Header/Cookie
Calls Logout API
Access Resource with expired Auth Header/Cookie
API Server

Rate Limiter



Other Checks

- Security Headers and Info Gathering
- Directory/ Path Traversal

DEMO

ROMA MMXVI APPSEC EUROPE

What's Coming Soon?

- Windows App Security Analyzer.
- iOS App Dynamic Analysis with Device.
- API Fuzzer to support detection of SQLi and RCE.
- Export Proxy logs to BurpSuite/IronWASP/ZAP

Useful Links

- Source: https://github.com/ajinabraham/Mobile-Security-Framework
- Issues: https://github.com/ajinabraham/Mobile-Security-Framework/issues
- Documentation: https://github.com/ajinabraham/Mobile-Security-Framework-MobSF/wiki
- Video Course: https://opsecx.com/index.php/product/automated-mobile-application-security-assessment-with-mobsf/

QA

Thanks & Credits

- Sachinraj Shetty
- Kamaiah Nadavala
- Bharadwaj Machiraju
- Yashin Mehboobe
- Anto Joseph
- Tim Brown
- Thomas Abraham
- Graphics/Image Owners

@ajinabraham

ajin25@gmail.com

ROMA MMXVI APPSEC EUROPE