Supported by OWASP

OWASP MOBILE SECURITY TESTING GUIDE 101

Jeroen Willemsen – Open Security Summit

About me

Jeroen Willemsen
@commjoenie
jeroen.willemsen@owasp.org
“Security architect”
“Full-stack developer”
“Mobile security”

Xebia Security

@OWASP_MSTG

OWASP Open Web Application Security Project

Agenda

- Introduction into the MASVS
- Introduction into the MSTG
- Some examples

Mobile security

1. Can you do a Cross Site Scripting (XSS) attack in a native app?
    1. What if there is no webview?

2. Can you do a Cross-Site Request Forgery (CSRF) attack in a native app without a webview?

Mobile security?

- So CSRF and XSS do not easily apply.

- But path-traversals do…

OWASP
Open Web Application Security Project

Mobile security?

- So CSRF and XSS do not easily apply.
- But path-traversals do…
- And then there is… Data leakage
  - through logging,
  - through insecure storage,
  - Through IPC.
- What about weak authentication mechanisms?
- What about reverse engineering?

How do we fix this?

Mobile Application Security Verification Standard
https://github.com/OWASP/owasp-masvs

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

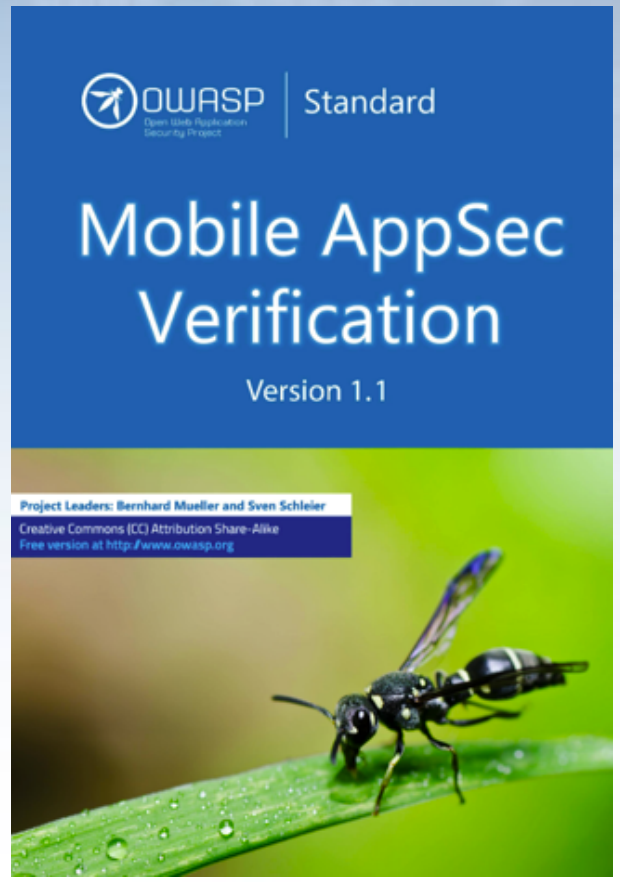
Mobile Appsec Checklist

OWASP
Open Web Application Security Project

OWASP Mobile AppSec Verification Standard

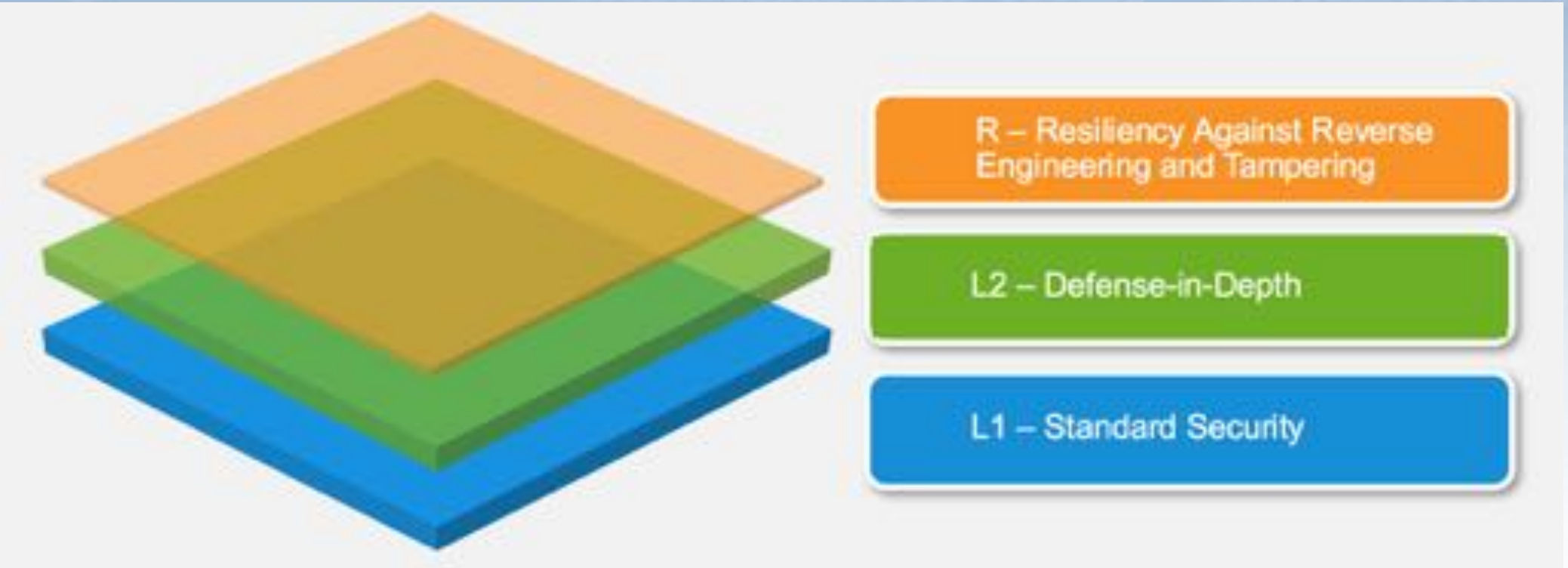
- Started as a fork of the OWASP ASVS
- Formalizes best practices and other security requirements
- Mobile-specific, high-level, OS-agnostic

- Why?
  - Shift left: give security requirements a-priori.
  - Give a clear goal during implementation
  - Give a clear goal during penetration testing



OWASP
Open Web Application Security Project

OWASP Mobile AppSec Verification Standard



- Architecture & design
- Data storage & privacy
- Cryptography
- Authentication & Session management
- Network Communication
- Platform Interaction
- Code quality & build settings
- Resilience requirements

OWASP Mobile AppSec Verification Standard

V2: Data Storage and Privacy Requirements

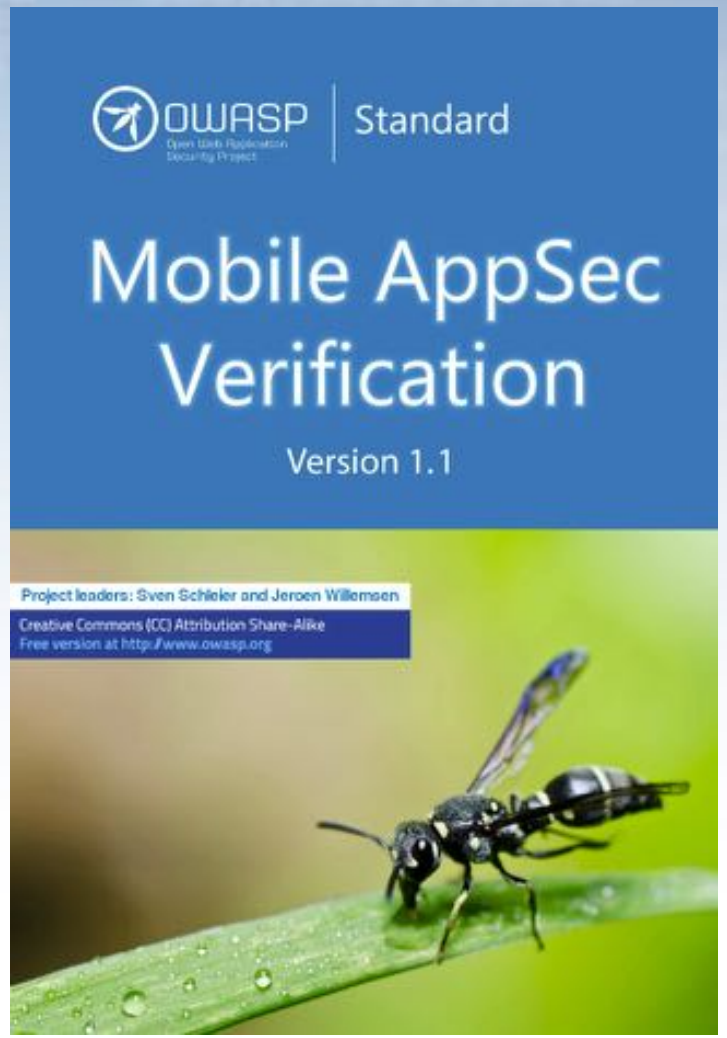
#	Description	L1	L2
2.1	System credential storage facilities are used appropriately to store sensitive data, such as user credentials or cryptographic keys.	✓	✓
2.2	No sensitive data is written to application logs.	✓	✓
2.3	No sensitive data is shared with third parties unless it is a necessary part of the architecture.	✓	✓
2.4	The keyboard cache is disabled on text inputs that process sensitive data.	✓	✓
2.5	The clipboard is deactivated on text fields that may contain sensitive data.	✓	✓
2.6	No sensitive data is exposed via IPC mechanisms.	✓	✓
2.7	No sensitive data, such as passwords or pins, is exposed through the user interface.	✓	✓
2.8	No sensitive data is included in backups generated by the mobile operating system.		✓

Current status MASVS

Project Lead	Lead Author	Contributors and Reviewers
Sven Schleier & Jeroen Willemsen	Bernhard Mueller	Alexander Antukh, Mesheryakov Aleksey, Bachevsky Artem, Jeroen Beckers, Vladislav Chelnokov, Ben Cheney, Stephen Corbiaux, Manuel Delgado, Ratchenko Denis, Ryan Dewhurst, Tereshin Dmitry, Christian Dong, Oprya Egor, Ben Gardiner, Rocco Gränitz, Henry Hu, Sjoerd Langkemper, Vinícius Henrique Marangoni, Martin Marsicano, Roberto Martelloni, Gall Maxim, Rio Okada, Abhinav Sejpal, Stefaan Seys, Yogesh Shamrma, Prabhant Singh, Nikhil Soni, Anant Shrivastava, Francesco Stillavato, Romuald SZKUDLAREK, Abdessamad Temmar, Koki Takeyama, Chelnokov Vladislav

Your turn!

- https://github.com/OWASP/owasp-masvs
- https://mobile-security.gitbook.io/masvs/



- ✓ Download it
- ✓ Read it
- ✓ Use it
- ✓ Give Feedback! Create an issue or a PR
- ✓ Tweet about it (@OWASP_MSTG)

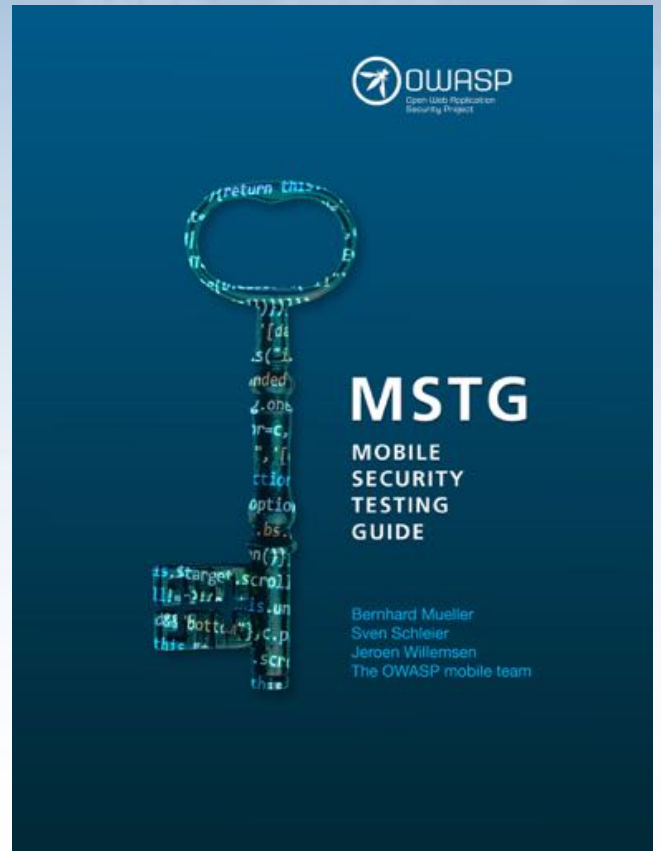
Agenda

- Introduction into the MASVS
- **Introduction into the MSTG**
- Some examples

OWASP Mobile Security Testing Guide (MSTG)

- Manual for testing security maturity of iOS and Android (mostly) native apps.
- Maps on MASVS requirements.

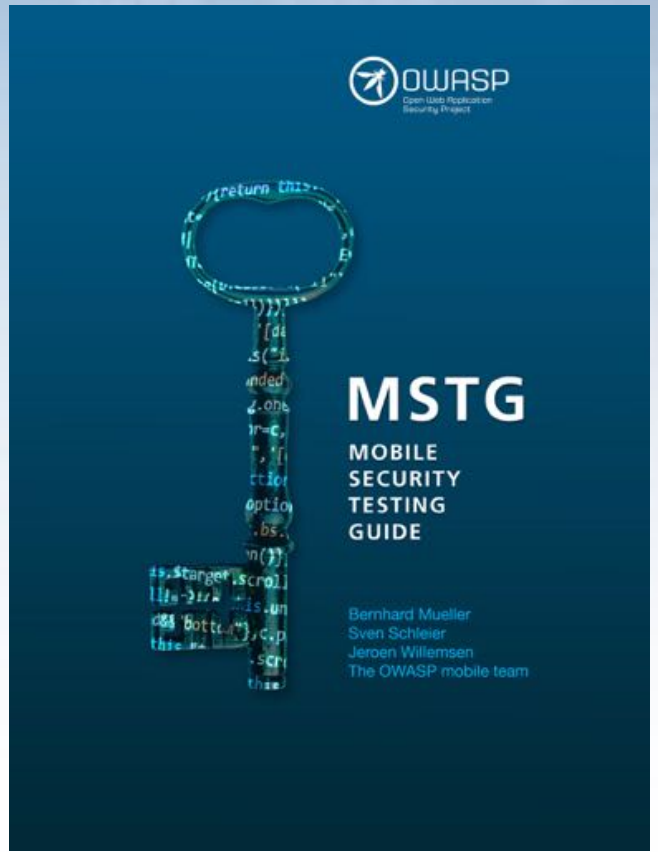
- Why?
  - Educate developers and penetration testers.
  - Provide a baseline for automated checks



OWASP
Open Web Application Security Project

OWASP Mobile Security Testing Guide (MSTG)

- General testing guide
- Android Testing guide
- iOS Testing guide



OWASP
Open Web Application Security Project

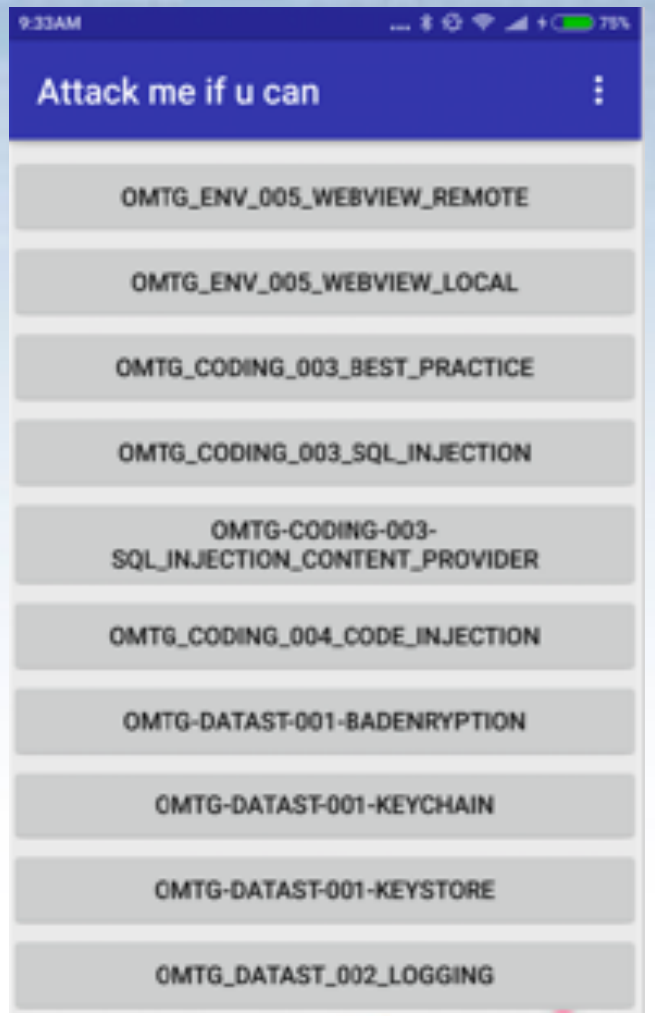
OWASP Mobile Security Testing Guide (MSTG)

- General testing guide
- Android Testing guide
- iOS Testing guide
- Crackme’s & Challenges

Kudos to Bernhard Mueller @bernhardm for his hard work!

OWASP Mobile Security Testing Guide (MSTG)

- General testing guide
- Android Testing guide
- iOS Testing guide
- Crackme’s & Challenges
- MSTG playground (External)



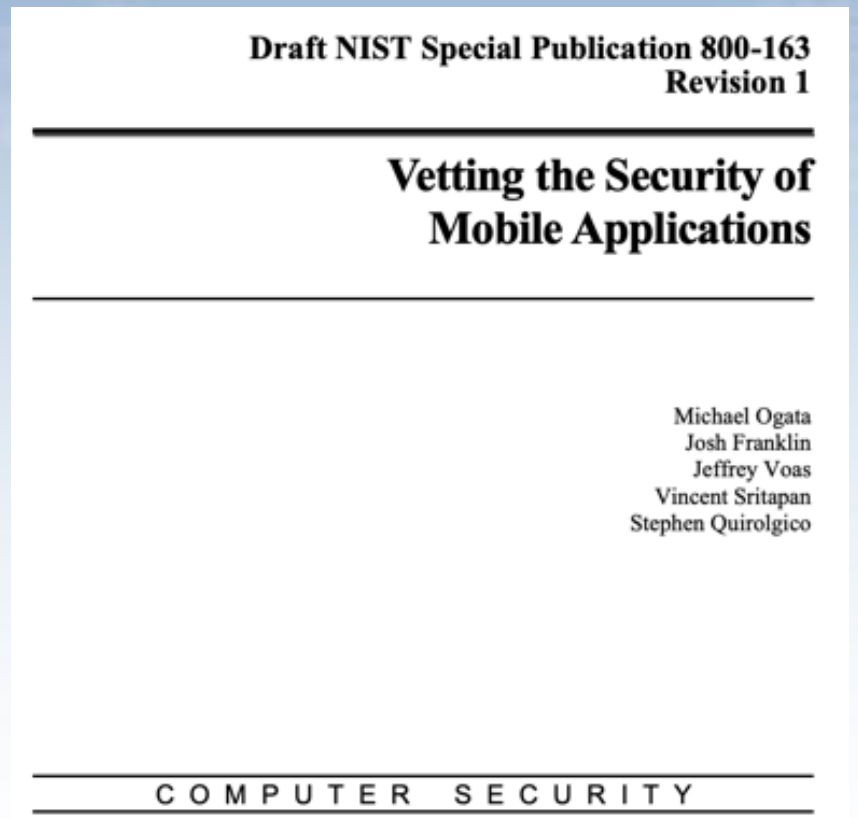
Current status MSTG

Authors	Co-Authors	Top Contributors	Reviewers	Editors
Bernhard Mueller Jeroen Willemsen (@jeroenwillemsen) Sven Schleier (@sushi2k)	Carlos Holguera Romuald Szkudlarek	Jeroen Beckers Pawel Rzepa Francesco Stillavato Andreas Happe Alexander Anthuk Henry Hoggard Wen Bin Kong Abdessamad Temmar Bolot Kerimbaev Cláudio André Slawomir Kosowski	Sjoerd Langkemper Anant Shrivastava Jeroen Beckers	Heaven Hodges Caitlin Andrews Nick Epson Anita Diamond Anna Szkudlarek

The full list of contributors is available on GitHub:
https://github.com/OWASP/owasp-mstg/graphs/contributors

OWASP
Open Web Application Security Project

MSTG Project status



NIST
National Institute of
Standards and Technology
U.S. Department of Commerce

OWASP
Open Web Application
Security Project

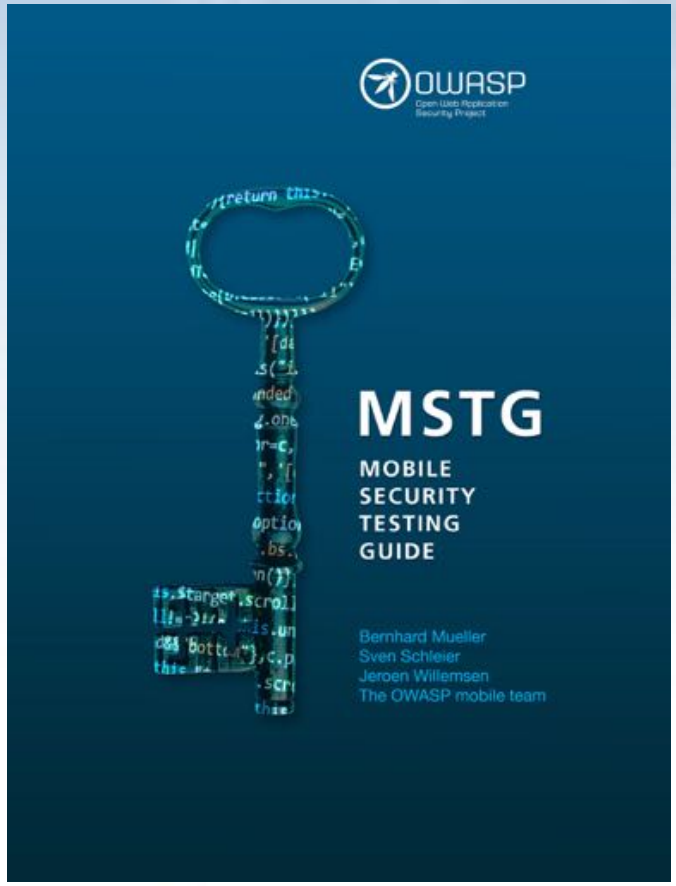
MSTG Project status – allways more work

- Update to iOS 12/13 & Android Pie/Q
- Restructure MSTG
- Add missing testcases
- Automate MSTG playground & merge with crackmes

Your turn!

- https://github.com/OWASP/owasp-mstg
  https://mobile-security.gitbook.io/mstg/

- ✓ Download it
- ✓ Read it
- ✓ Use it
- ✓ Give Feedback (file an issue)
- ✓ **Fix issues: send in your Pull Requests!**
- ✓ Tweet about it (@OWASP_MSTG)



Agenda

- Introduction into the MASVS
- Introduction into the MSTG
- **Some examples**

Network Communication Requirements

#	Description	L1	L2
5.1	Data is encrypted on the network using TLS. The secure channel is used consistently throughout the app.	✓	✓
5.2	The TLS settings are in line with current best practices, or as close as possible if the mobile operating system does not support the recommended standards.	✓	✓
5.3	The app verifies the X.509 certificate of the remote endpoint when the secure channel is established. Only certificates signed by a trusted CA are accepted.	✓	✓
5.4	The app either uses its own certificate store, or pins the endpoint certificate or public key, and subsequently does not establish connections with endpoints that offer a different certificate or key, even if signed by a trusted CA.		✓
5.5	The app doesn't rely on a single insecure communication channel (email or SMS) for critical operations, such as enrollments and account recovery.		✓
5.6	The app only depends on up-to-date connectivity and security libraries.		✓

Network Communication Requirements

#	Description	L1	L2
5.1	Data is encrypted on the network using TLS. The secure channel is used consistently throughout the app.	✓	✓
5.2	The TLS settings are in line with current best practices, or as close as possible if the mobile operating system does not support the recommended standards.	✓	✓
5.3	The app verifies the X.509 certificate of the remote endpoint when the secure channel is established. Only certificates signed by a trusted CA are accepted.	✓	✓

- The MSTG will guide you on how to review the code & do dynamic analysis of
  - The usage of TLS in general
  - The settings of the TLS connection in general
  - Certificate validation (general, iOS and Android specific)

Network Communication Requirements

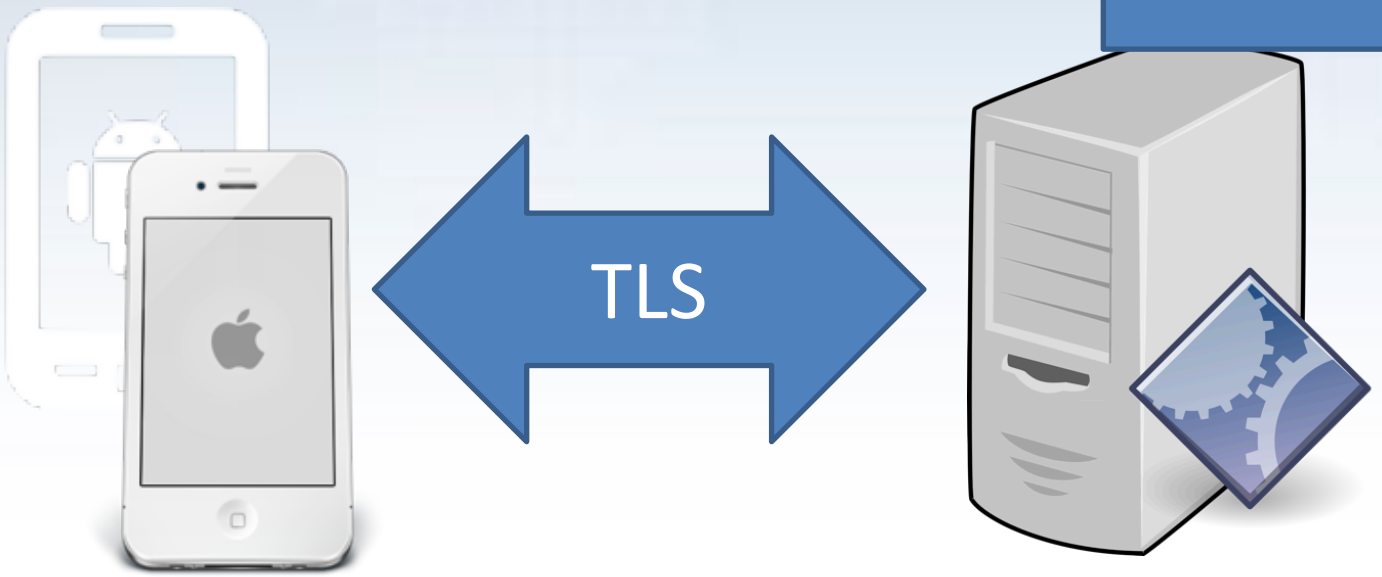
5.4	The app either uses its own certificate store, or pins the endpoint certificate or public key, and subsequently does not establish connections with endpoints that offer a different certificate or key, even if signed by a trusted CA.		✓

SSL pinning

Root CA

Intermediate

Leaf cert



Version	
Certificate Serial Number	
Certificate Algorithm Identifier for Certificate Issuer's Signature	
Issuer	
Validity Period	
Subject	
Subject Public-Key Information	Algorithm Identifier
	Public-key Value
Issuer Unique Identifier	
Subject Unique Identifier	
Extensions	

Certification Authority's Digital Signature

OWASP
Open Web Application Security Project

Network Communication Requirements

- The MSTG helps in finding ways to do pinning
  - In Android (OKHttp, WebView, networkSecurityConfig, using TrustManagers,
  - In iOS (NSURLConnection, TrustKit, AFNetworking, Alamofire)
  - Hybrid/multiplatform: Apache Cordova, Xamarin, Phonegap.

- But what about verifying it? Or bypassing it?

SSL Pinning – verify whether it is on

- Android:
  - Below Android 7: install your Burp/mitmproxy/Zap CA on the device,
  - Android 7 and above: rework networksecurityconfig.xml
  - Try to MiTM the application.

- iOS:
  - Install install your Burp/mitmproxy/Zap CA on the device
  - Try to MiTM the application.

SSL Pinning – bypassing it

- iOS: SSL Killswitch V2
- iOS: Frida & Objection
- Android: Xposed
- Android: Frida & Objection

SSL Pinning – SSL killswitch V2

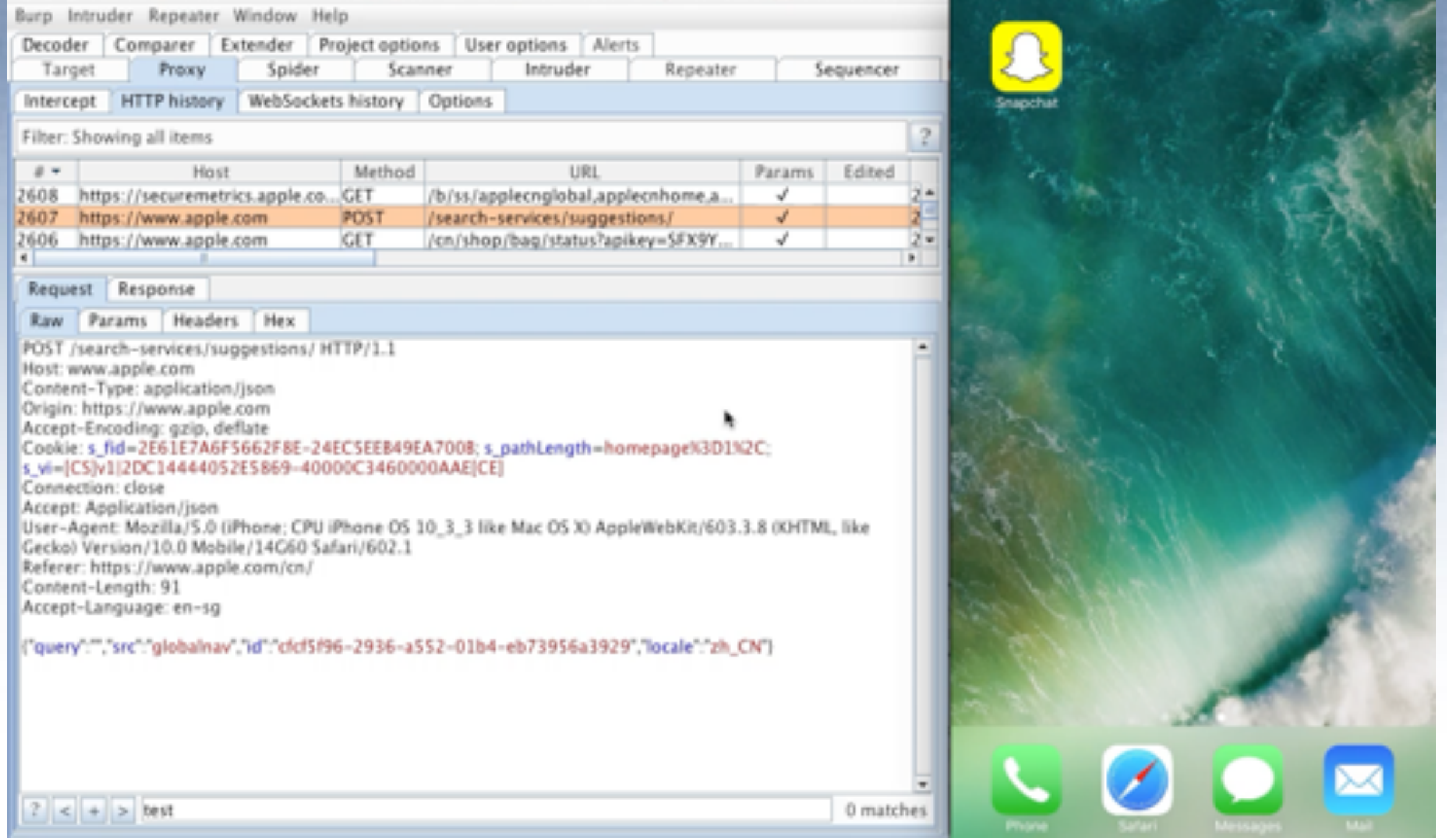
Two easy ways to break most pinners:

1. Jailbreak → use Cydia & SSL Killswitch V2

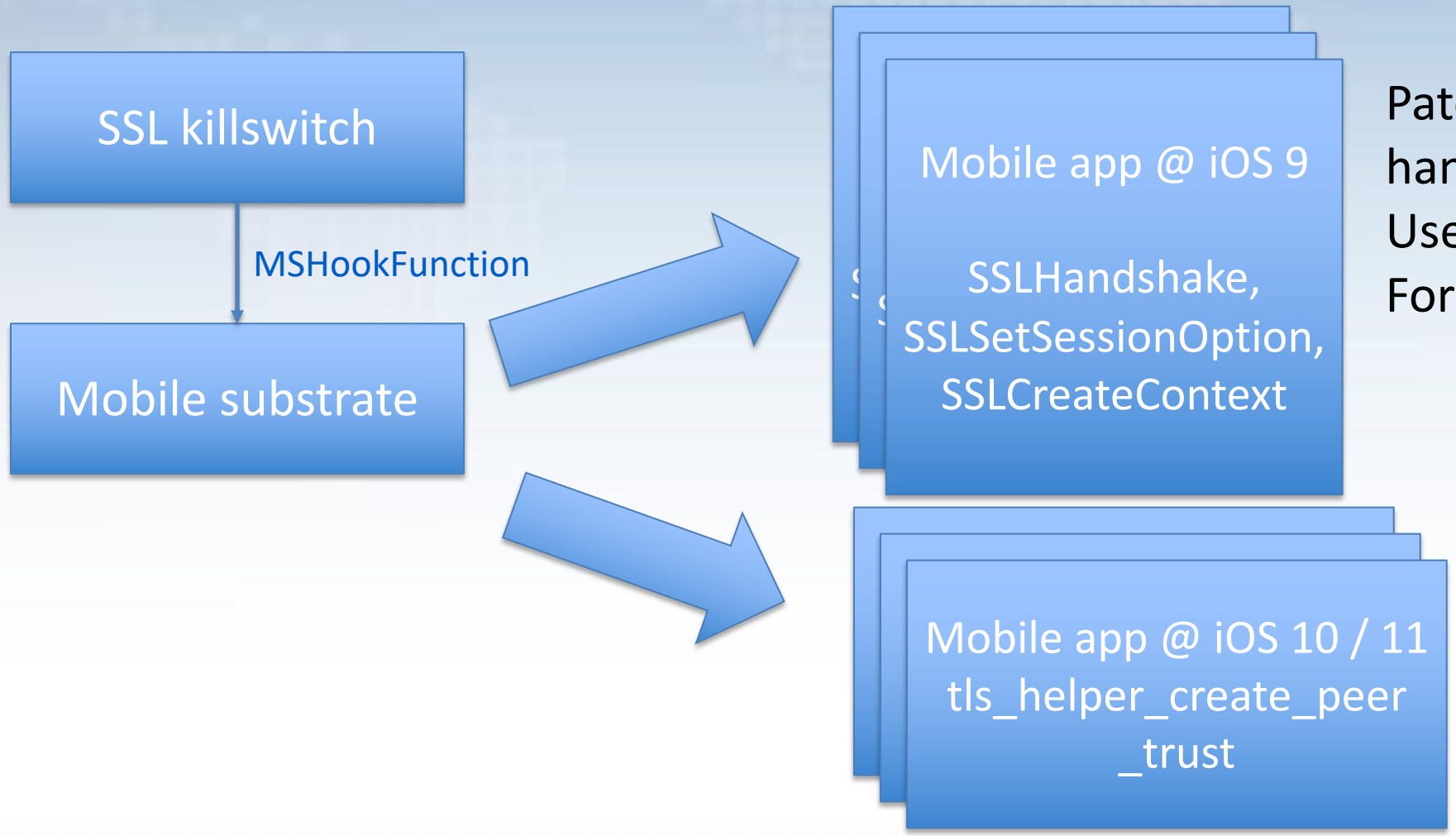
2. Do dynamic instrumentation on a non-jailbroken device

FRIDA

See https://github.com/OWASP/owasp-mstg/blob/master/Document/0x04f-Testing-Network-Communication.md
and https://github.com/OWASP/owasp-mstg/blob/master/Document/0x06g-Testing-Network-Communication.md


Burp Intruder Repeater Window Help
Decoder Comparer Extender Project options User options Alerts
Target Proxy Spider Scanner Intruder Repeater Sequencer
Intercept HTTP history WebSockets history Options
Filter: Showing all items
Host Method URL Params Edited
2608 https://securemetrics.apple.co... GET /b/ss/applecnglobal,applecnhome,a... ✓
2607 https://www.apple.com POST /search-services/suggestions/ ✓
2606 https://www.apple.com GET /cn/shop/bag/status?apikey=SFX9Y... ✓
Request Response
Raw Params Headers Hex
POST /search-services/suggestions/ HTTP/1.1
Host: www.apple.com
Content-Type: application/json
Origin: https://www.apple.com
Accept-Encoding: gzip, deflate
Cookie: s_fid=2E61E7A6F5662F8E-24EC5EEB49EA7008; s_pathLength=homepage%3D1%2C;
s_vi=[CS]v1|2DC14444052E5869-40000C3460000AAE[CE]
Connection: close
Accept: Application/json
User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 10_3_3 like Mac OS X) AppleWebKit/603.3.8 (KHTML, like
Gecko) Version/10.0 Mobile/14G60 Safari/602.1
Referer: https://www.apple.com/cn/
Content-Length: 91
Accept-Language: en-sg
{"query":"","src":"globalnav","id":"cfcf5f96-2936-a552-01b4-eb73956a3929","locale":"zh_CN"}
test
0 matches
Snapchat
Phone
Safari
Messages
Mail

SSL Pinning – SSL killswitch V2



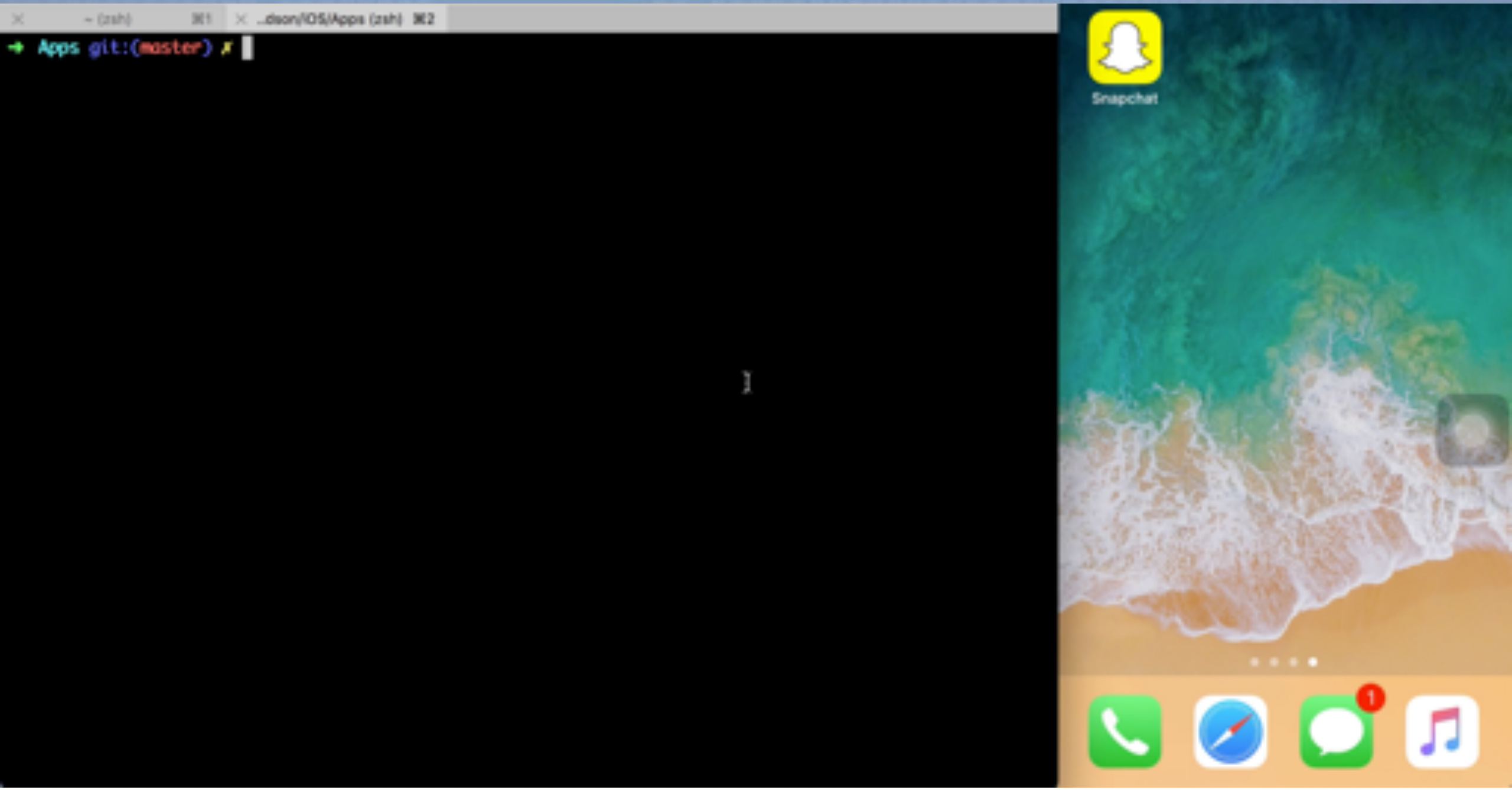
Patch underlying SSL handshake implementation
Used by NSURLConnection
For all apps…

What if you don’t want to jailbreak?

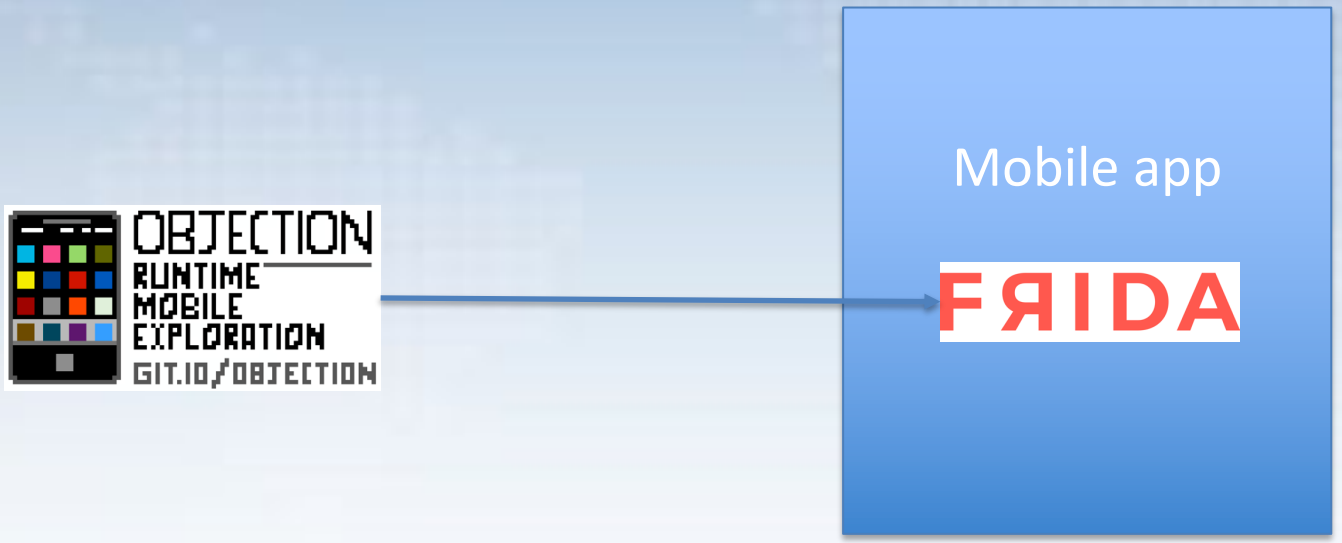
- Jailbroken devices require maintenance
- Jailbreaks are getting harder to find
- What about jailbreak protection of the app?
- Let’s patch the app itself!

FRIDA

OWASP
Open Web Application Security Project


~ (zsh) ⌘1
...dson/iOS/Apps (zsh) ⌘2
Apps git:(master) ✗
Snapchat
1

SSL Pinning – Objection

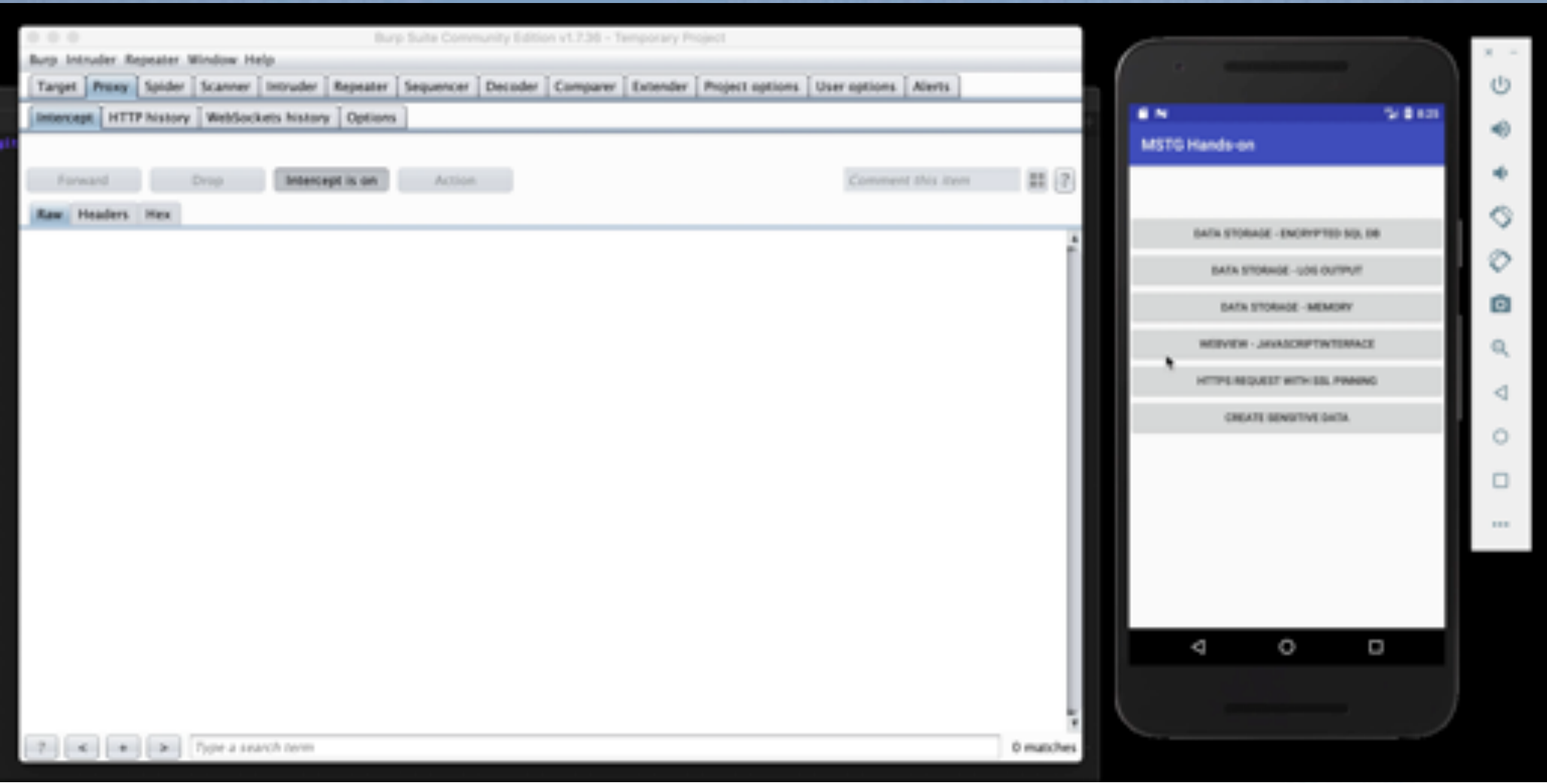


Patch underlying SSL handshake implementation
Used by NSURLConnection
For ***one*** app.

1. Frida server in Gadget waits
2. Objection connects to server with explore REPL
3. Objection calls script that patches underlying SSL handshake implementation

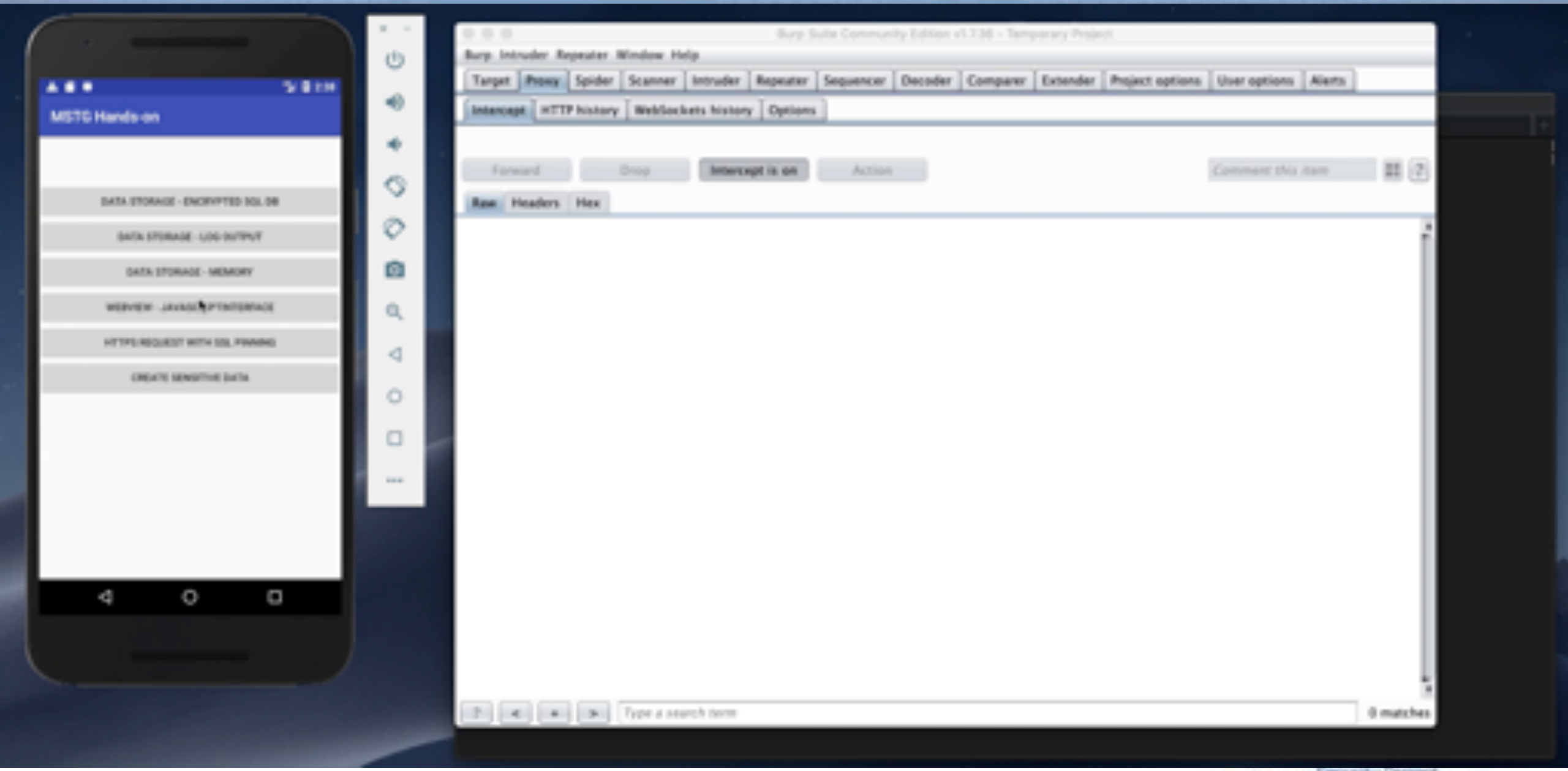
SSL Pinning in Android - Objection

Let’s do similar runtime patching in Android…


Burp Suite Community Edition v1.7.36 - Temporary Project
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
Intercept HTTP history WebSockets history Options
Forward Drop Intercept is on Action
Comment this item
Raw Headers Hex
Type a search term
0 matches
MSTG Hands-on
DATA STORAGE - ENCRYPTED SQL DB
DATA STORAGE - LOG OUTPUT
DATA STORAGE - MEMORY
WEBVIEW - JAVASCRIPTINTERFACE
HTTPS REQUEST WITH SSL PINNING
CREATE SENSITIVE DATA

SSL Pinning in Android - Xposed

Let’s pick it up from the rooted device again…


MSTG Hands-on
DATA STORAGE - ENCRYPTED SQLDB
DATA STORAGE - LOG OUTPUT
DATA STORAGE - MEMORY
WEBVIEW - JAVASCRIPTINTERFACE
HTTPS REQUEST WITH SSL PINNING
CREATE SENSITIVE DATA
Burp Suite Community Edition v1.7.36 - Temporary Project
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
Intercept HTTP history WebSockets history Options
Forward Drop Intercept is on Action
Comment this item
Raw Headers Hex
Type a search term
0 matches

Authentication requirements

#	Description	L1	L2
4.1	If the app provides users access to a remote service, some form of authentication, such as username/password authentication, is performed at the remote endpoint.	✓	✓
4.2	If stateful session management is used, the remote endpoint uses randomly generated session identifiers to authenticate client requests without sending the user's credentials.	✓	✓
4.3	If stateless token-based authentication is used, the server provides a token that has been signed using a secure algorithm.	✓	✓
4.4	The remote endpoint terminates the existing session when the user logs out.	✓	✓
4.5	A password policy exists and is enforced at the remote endpoint.	✓	✓
4.6	The remote endpoint implements a mechanism to protect against the submission of credentials an excessive number of times.	✓	✓
4.7	Biometric authentication, if any, is not event-bound (i.e. using an API that simply returns "true" or "false"). Instead, it is based on unlocking the keychain/keystore.		✓
4.8	Sessions are invalidated at the remote endpoint after a predefined period of inactivity and access tokens expire.		✓
4.9	A second factor of authentication exists at the remote endpoint and the 2FA requirement is consistently enforced.		✓
4.10	Sensitive transactions require step-up authentication.		✓
4.11	The app informs the user of all login activities with their account. Users are able view a list of devices used to access the account, and to block specific devices.		✓

TouchID the wrong way: using LAContext

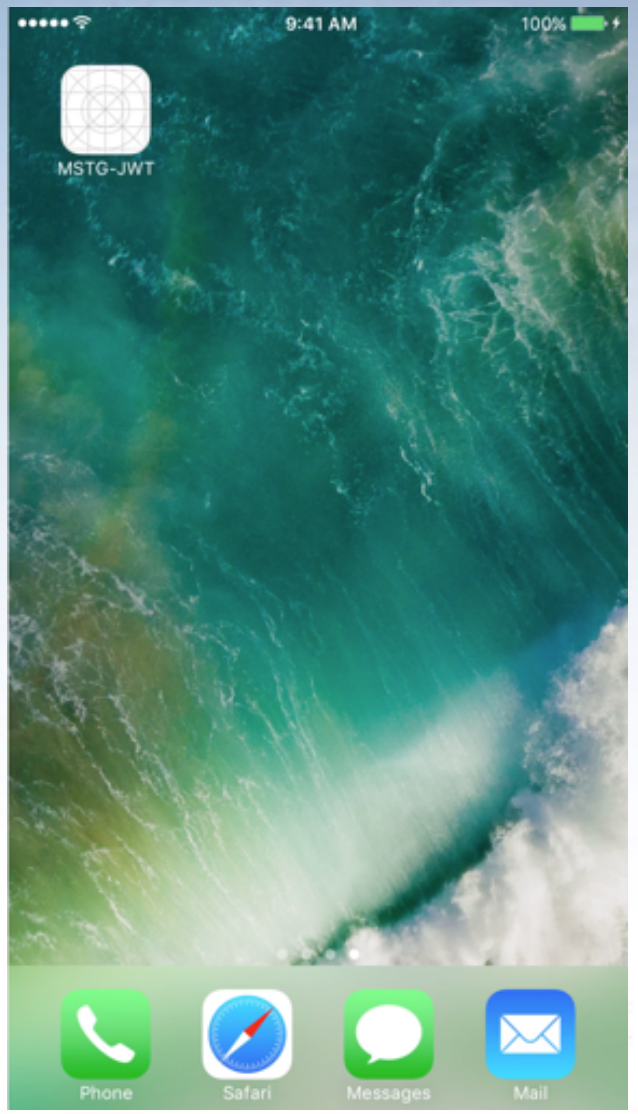
There are 2 ways to use TouchID:

1. Protect an entry in the keychain and unlock it via TouchID

2. Use the LocalAuthenticationContext :

```
LocalAuthenticationContext.evaluatePolicy(.deviceOwnerAuthenticationWithBiometrics, localizedReason: reasonString) {
success, evaluateError in {
If success {
    successmethods()
} else {
    ....
}
```

What if we call the successmethods() directly?

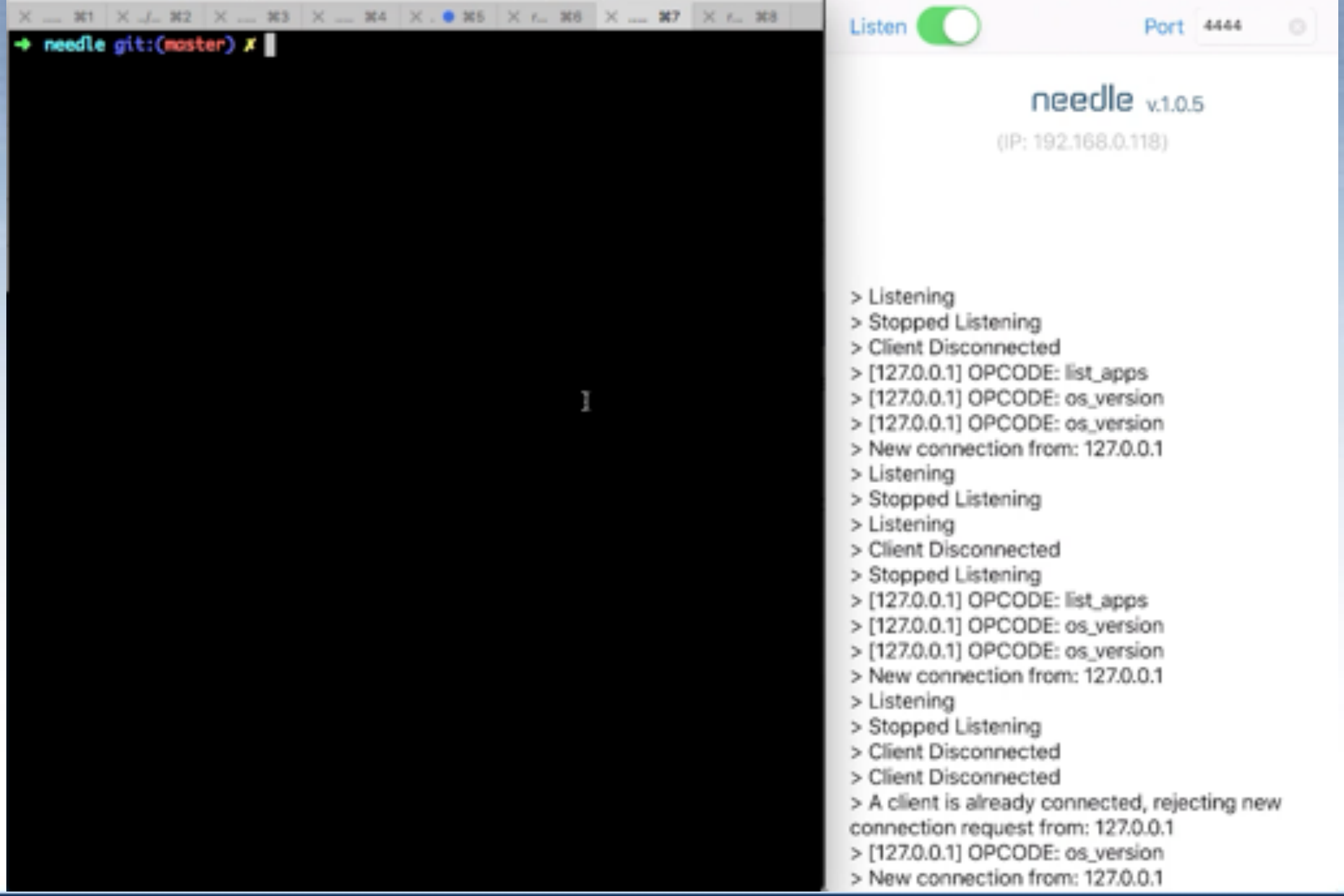


Bypassing Touch-ID

- With needle
- With 

- Both cases: use Frida to hook onto *`evaluatePolicy:localizedReason:reply`*
  - Ensures that when *evaluatePolicy* is calls that the reply its success is set to true (E.g.: call success methods)

See https://github.com/OWASP/owasp-mstg/blob/master/Document/0x06f-Testing-Local- Authentication.md


➜ needle git:(master) ✗
Listen
Port
4444
needle v.1.0.5
(IP: 192.168.0.118)
> Listening
> Stopped Listening
> Client Disconnected
> [127.0.0.1] OPCODE: list_apps
> [127.0.0.1] OPCODE: os_version
> [127.0.0.1] OPCODE: os_version
> New connection from: 127.0.0.1
> Listening
> Stopped Listening
> Listening
> Client Disconnected
> Stopped Listening
> [127.0.0.1] OPCODE: list_apps
> [127.0.0.1] OPCODE: os_version
> [127.0.0.1] OPCODE: os_version
> New connection from: 127.0.0.1
> Listening
> Stopped Listening
> Client Disconnected
> Client Disconnected
> A client is already connected, rejecting new connection request from: 127.0.0.1
> [127.0.0.1] OPCODE: os_version
> New connection from: 127.0.0.1

Data storage & privacy requirements

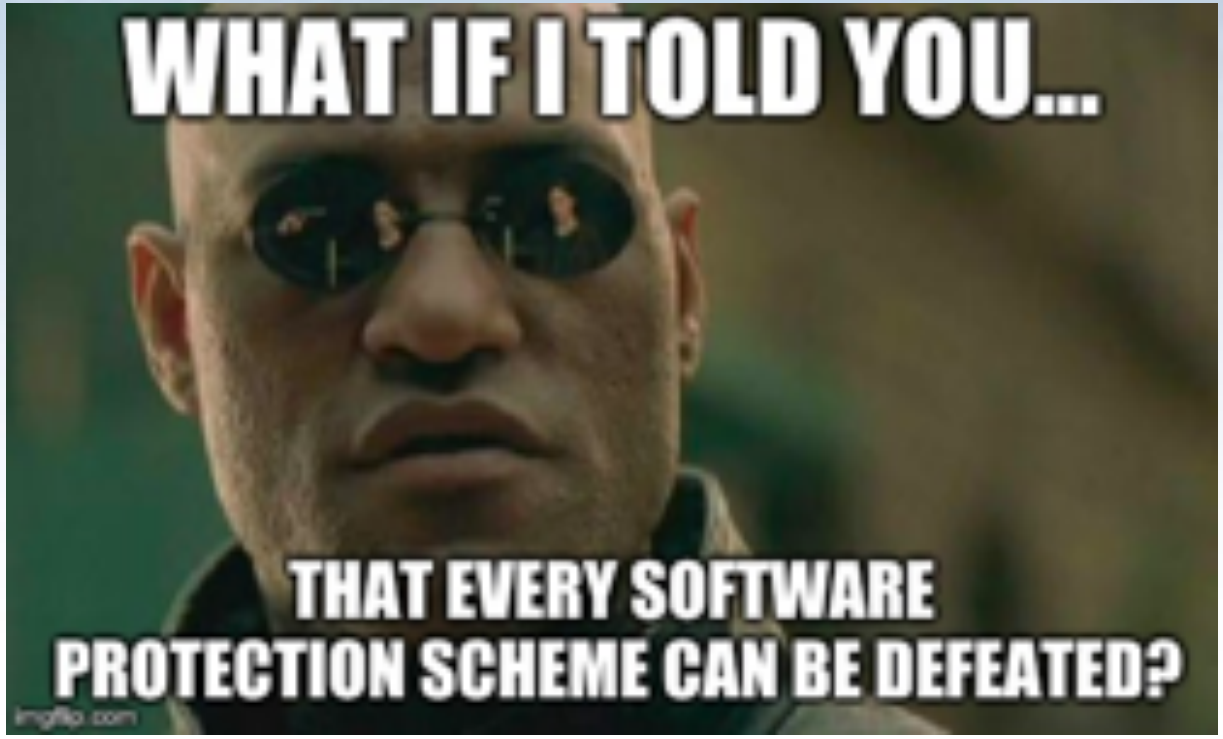
#	
2.1	System credential storag data, such as PII, user c
2.2	No sensitive data should credential storage faciliti
2.3	No sensitive data is writt
2.4	No sensitive data is sha the architecture.
2.5	The keyboard cache is c
2.6	No sensitive data is exp
2.7	No sensitive data, such interface.
2.8	No sensitive data is inclu system.
2.9	The app removes sensit
2.10	The app does not hold s memory is cleared expli
2.11	The app enforces a mini the user to set a device
2.12	The app educates the us information processed, a follow in using the app.

OWASP
Open Web Application Security Project

There is much more!

- Reverse Engineering

  - ✓ Root / Jailbreak Detection
  - ✓ Anti-Debugging
  - ✓ Detecting Reverse Engineering Tools
  - ✓ Emulator Detection / Anti-Emulation
  - ✓ File and Memory Integrity Checks
  - ✓ Device Binding
  - ✓ Obfuscation



There is much more!

- Reverse Engineering
- Analysis & best practices for
  - Storage
  - Cryptography
  - Local Authentication
  - Network Communication
  - Platform interaction
  - Code quality & build settings


I WANT YOU
TO HELP US TO
THE NEXT LEVEL!

QUESTIONS?

@OWASP_MSTG

jeroen.willemsen@owasp.org

OWASP
Open Web Application Security Project

THANK YOU!

@OWASP_MSTG

jeroen.willemsen@owasp.org

Supported by OWASP

MOBILE SECURITY TESTING GUIDE ONBOARDING

Jeroen Willemsen – Open Security Summit

Agenda

- Introduction into the current state of the MSTG.
  - Issues
  - Milestones
  - Project Page
- Release process.
- Contribution guidelines.
- Outline of the activities planned for this week.
- How to get started
- Notes for contributors & reviewers

How to get started

1. Fork the repo you want to work on:
   - https://github.com/OWASP/owasp-mstg
   - https://github.com/OWASP/owasp-masvs
2. Setup local git at your system (preferrably with ssh keys)
3. Clone the repo to your system
4. Add the upstream repo (MASVS/MSTG) to your repo configuration
5. Create a branch, start your work, commit and push when ready
6. Pull request and ask our attention to speed it up ☺ .
7. Review feedback? Parse it as soon as you can, so you can move forward and add your stuff.

Notes for contributors

- For any tool: focus on the installation, basics and guide towards it's own (online) help

- For every feature of a platform: focus on its working, best practices, pitfalls and insecurities

Notes for reviewers

- Really bad PR? Ask to get in touch and work together
- Ok-ish PR with big errors: comments
- Small issues: try to comment
- In parallel: PR for your own fixes, but keep it to a ### level per PR to cause less conflicts

FINAL NOTES:

- ALL EVENING sessions are in villa 708!