OWASP
Open Web Application
Security Project

Needle

Finding Issues within iOS Applications

Marco Lancini

Whoami

Me: Marco Lancini

- Security Consultant at MWR InfoSecurity
- @lancinimarco

MWR InfoSecurity

- Research-led Security Consultancy
- Offices in the UK, USA, Singapore, South Africa, Germany, Poland…

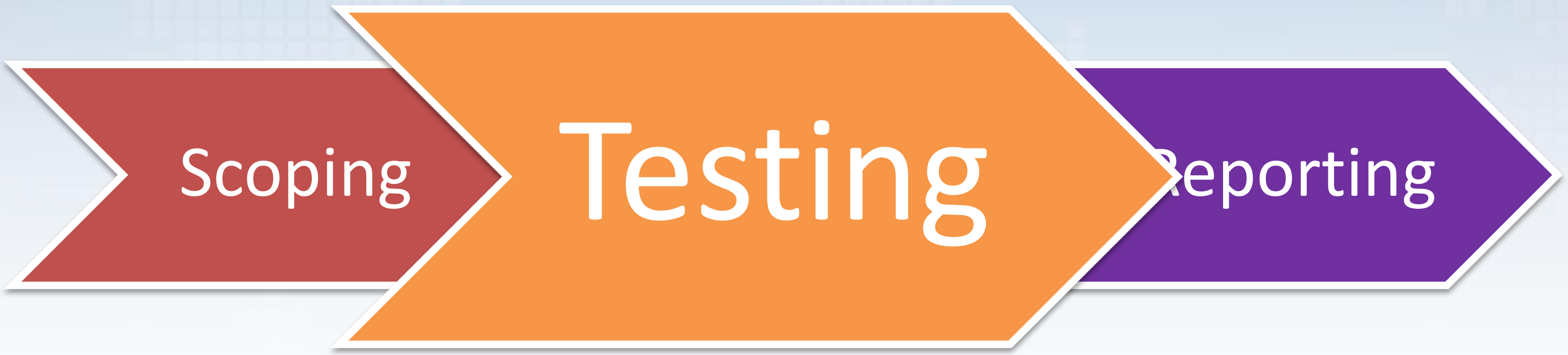
What is this talk about?

- Current State of Mobile (in)Security
- iOS Pentesting (the current state)
- Needle (idea, architecture, features, etc.)
- Demo
- Roadmap

CURRENT STATE OF MOBILE (IN)SECURITY

OWASP
Open Web Application Security Project

My Life as a Pentester



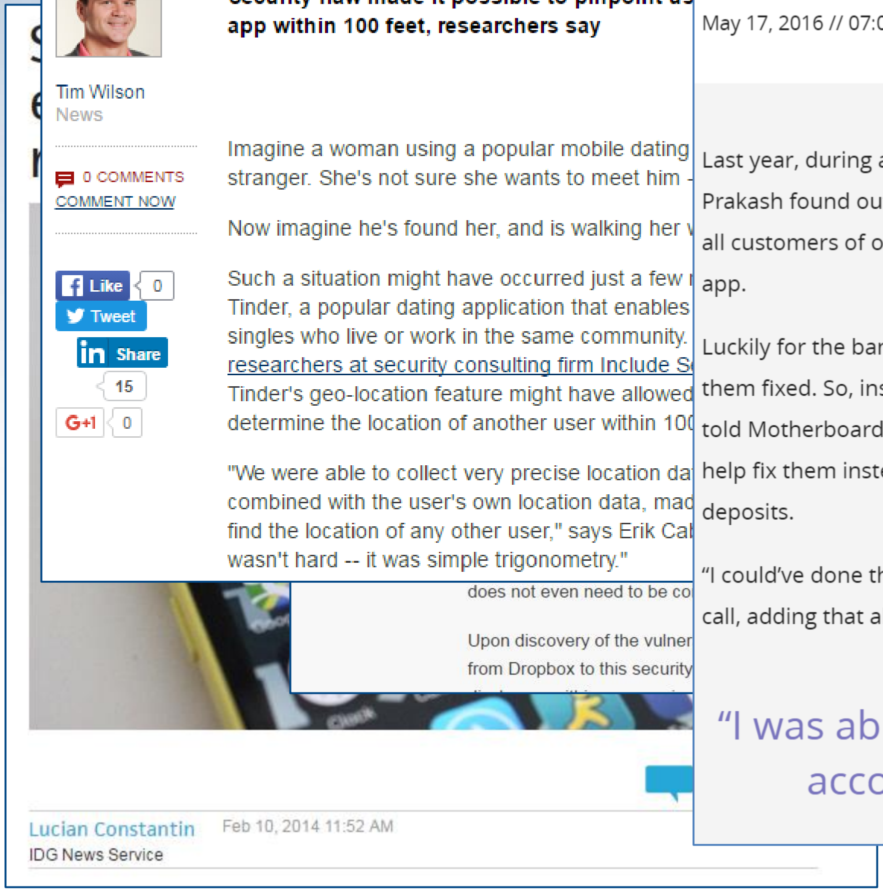
Mobile app lifecycle

- Idea
- Execution
- Public Release

OWASP
Open Web Application Security Project

Mobile app lifecycle

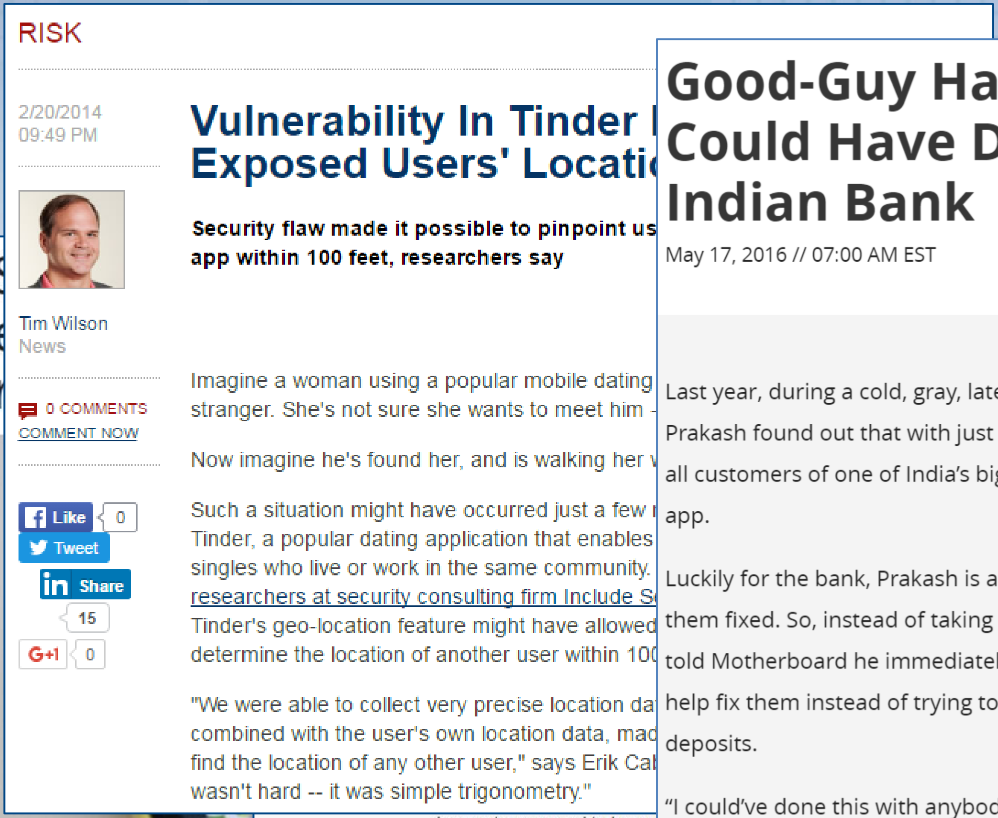
Some real life examples…



does not even need to be co

Upon discovery of the vulner
from Dropbox to this security

Lucian Constantin Feb 10, 2014 11:52 AM
IDG News Service



RISK

2/20/2014
09:49 PM

Tim Wilson
News

0 COMMENTS
COMMENT NOW

Vulnerability In Tinder I Exposed Users' Locati

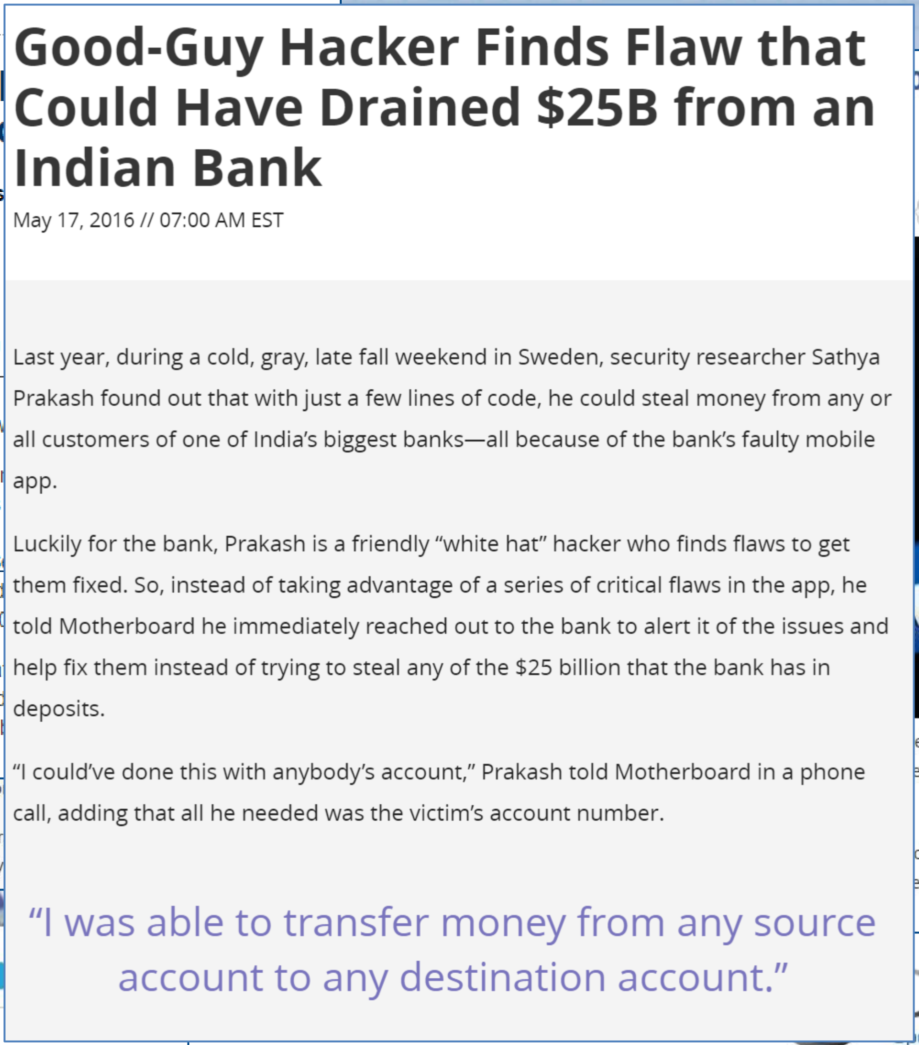
Security flaw made it possible to pinpoint us app within 100 feet, researchers say

Imagine a woman using a popular mobile dating stranger. She's not sure she wants to meet him -

Now imagine he's found her, and is walking her v

Such a situation might have occurred just a few r Tinder, a popular dating application that enables singles who live or work in the same community. researchers at security consulting firm Include S Tinder's geo-location feature might have allowed determine the location of another user within 100

"We were able to collect very precise location da combined with the user's own location data, mad find the location of any other user," says Erik Ca wasn't hard -- it was simple trigonometry."



Good-Guy Hacker Finds Flaw that Could Have Drained $25B from an Indian Bank

May 17, 2016 // 07:00 AM EST

Last year, during a cold, gray, late fall weekend in Sweden, security researcher Sathya Prakash found out that with just a few lines of code, he could steal money from any or all customers of one of India's biggest banks—all because of the bank's faulty mobile app.

Luckily for the bank, Prakash is a friendly "white hat" hacker who finds flaws to get them fixed. So, instead of taking advantage of a series of critical flaws in the app, he told Motherboard he immediately reached out to the bank to alert it of the issues and help fix them instead of trying to steal any of the $25 billion that the bank has in deposits.

"I could've done this with anybody's account," Prakash told Motherboard in a phone call, adding that all he needed was the victim's account number.

"I was able to transfer money from any source account to any destination account."

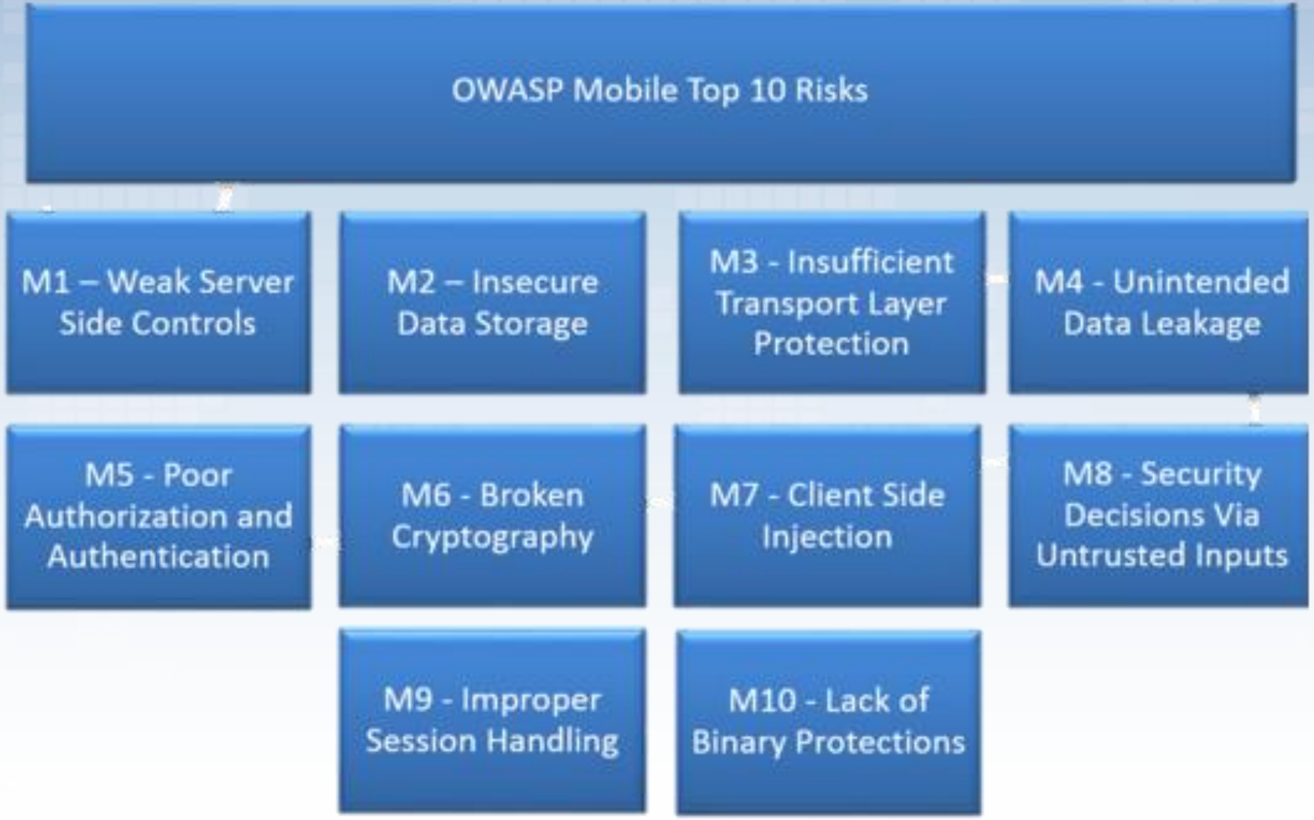
steal access

ebook has fixed multiple
eal access tokens and

ck Secure, has who
essenger

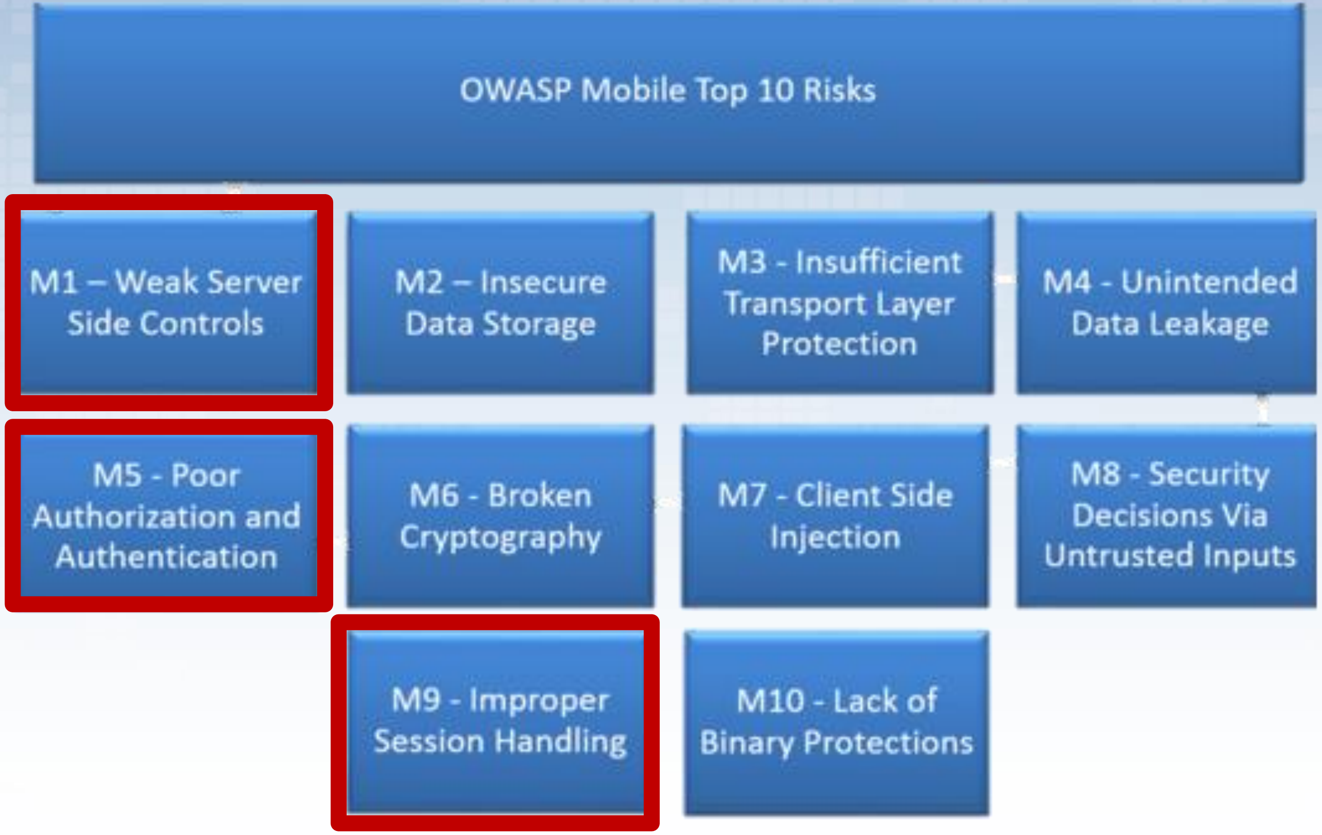
Where to focus

OWASP Mobile Top 10 (2014)

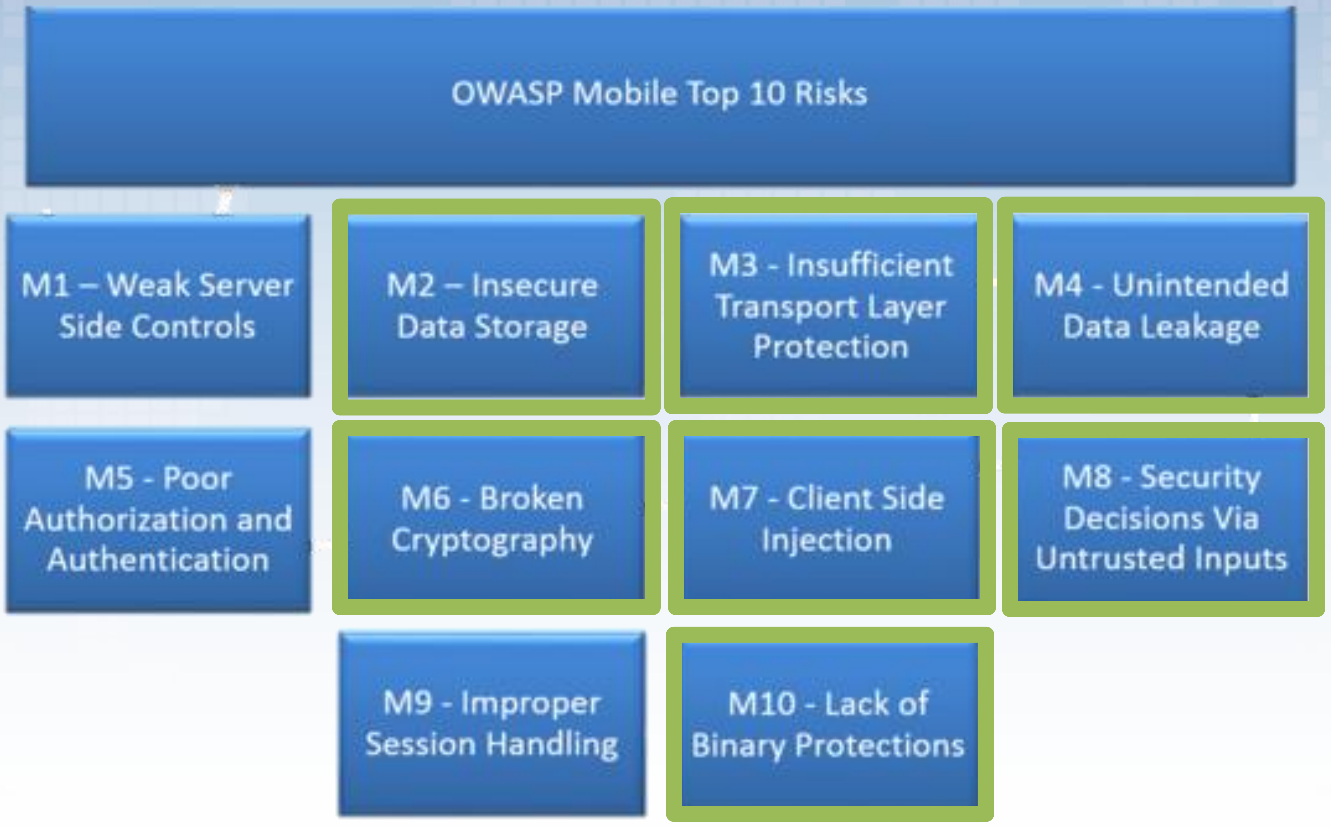

OWASP Mobile Top 10 Risks
M1 – Weak Server Side Controls
M2 – Insecure Data Storage
M3 - Insufficient Transport Layer Protection
M4 - Unintended Data Leakage
M5 - Poor Authorization and Authentication
M6 - Broken Cryptography
M7 - Client Side Injection
M8 - Security Decisions Via Untrusted Inputs
M9 - Improper Session Handling
M10 - Lack of Binary Protections

OWASP
Open Web Application Security Project

Server Side Security


OWASP Mobile Top 10 Risks
M1 – Weak Server Side Controls
M2 – Insecure Data Storage
M3 - Insufficient Transport Layer Protection
M4 - Unintended Data Leakage
M5 - Poor Authorization and Authentication
M6 - Broken Cryptography
M7 - Client Side Injection
M8 - Security Decisions Via Untrusted Inputs
M9 - Improper Session Handling
M10 - Lack of Binary Protections

Client Side Security


OWASP Mobile Top 10 Risks
M1 – Weak Server Side Controls
M2 – Insecure Data Storage
M3 - Insufficient Transport Layer Protection
M4 - Unintended Data Leakage
M5 - Poor Authorization and Authentication
M6 - Broken Cryptography
M7 - Client Side Injection
M8 - Security Decisions Via Untrusted Inputs
M9 - Improper Session Handling
M10 - Lack of Binary Protections

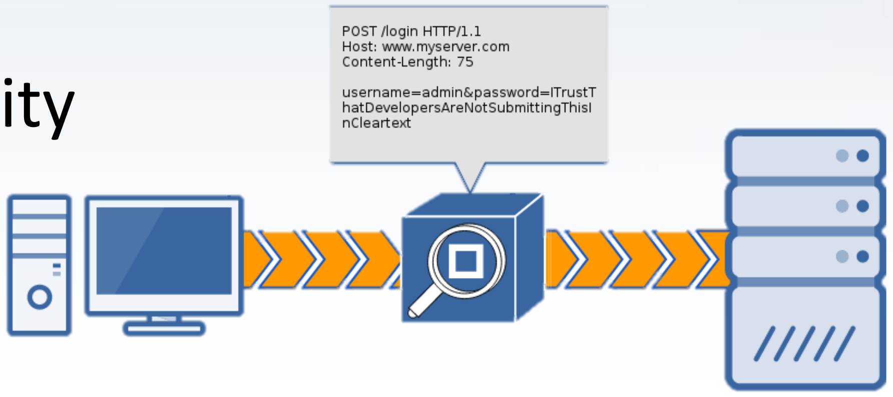
Attacker's Perspective

- Physical access
  - Stolen device
  - Unattended device
  - Shared environment
- Malware
  - JB devices
  - Non-JB devices
- Exploitation
  - Outdated software
  - 0day

OWASP
Open Web Application Security Project

Attacker's Perspective

- Network communications
  - Man-in-the-Middle (MitM)
  - Clear text / Weak encryption
  - Client-side attacks
- The web server
  - Web application security

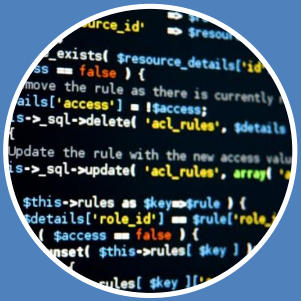


```
POST /login HTTP/1.1
Host: www.myserver.com
Content-Length: 75

username=admin&password=ITrustT
hatDevelopersAreNotSubmittingThisI
nCleartext
```

(the current state)

IOS PENTESTING

Assessment Scenarios



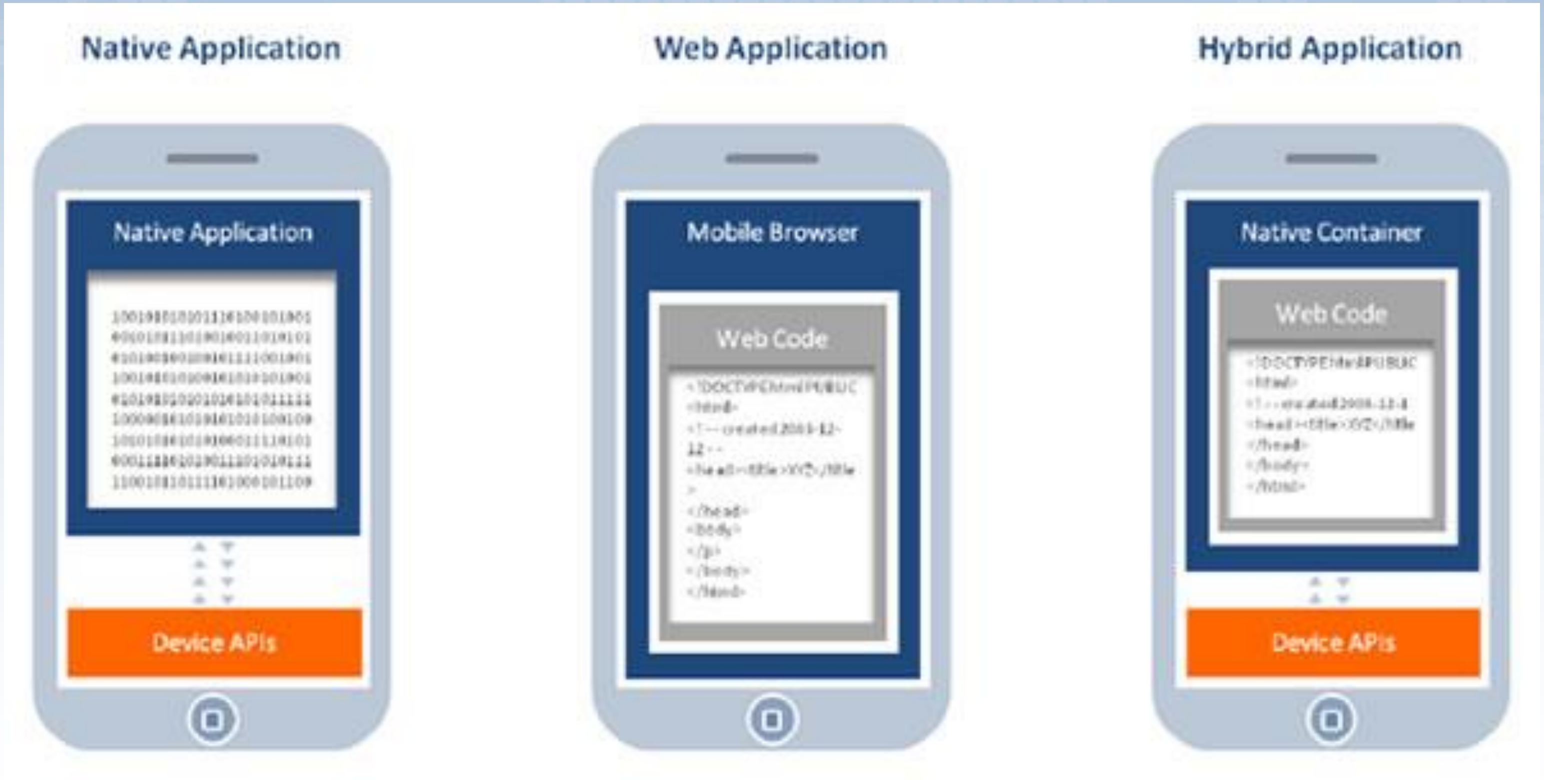
Source Code Review

Mobile App Test

Device Review

Mobile Device Management

Types of Applications



Analysing iOS Applications

- Run the app on a jailbroken device
- MiTM all the network communications
- Inspect the app via instrumentation
- Manipulate the runtime
- Review the codebase

Techniques / 1

Static Analysis

- Reverse engineer the binary
- Perform code review

Data Security

- Look for insecure storage
- Assess data sources (keychain, plist files, cookies)
- Check presence of caching

Techniques / 2

Runtime Analysis

- Bypass integrity checks
- [Patch the binary]
- Instrument the app (hooking)

Transport Security

- Proxy the traffic
- [Bypass TLS pinning]
- Asses WebViews / exploit JS bridges

OWASP
Open Web Application Security Project

iOS Testing Environment

Testing Tools

- Jailbroken device
  - Weaken the sandbox
  - Emulate attackers' perspective
- Alternate Market (Cydia)
  - Common unix tools (BigBoss)
  - OpenSSH
- Hooking framework
  - Cycript/Frida/Theos
- Intercepting proxy (Burp)

JailbreakMe

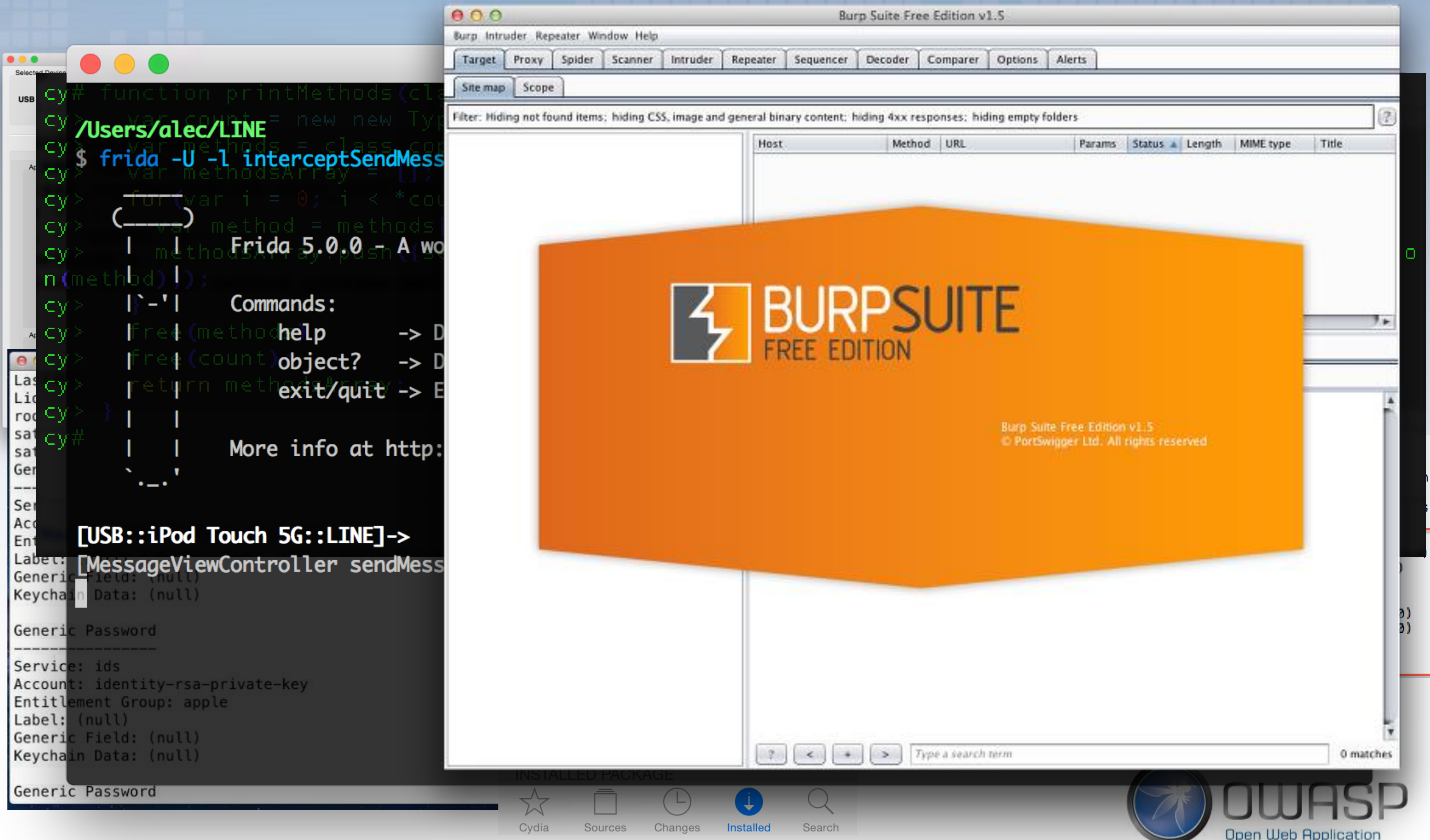
by comex (et al.)

JailbreakMe

Jailbreak to get tweaks and apps Apple won't allow in the App Store. Free, legal, safe.

You should sync with iTunes before using this tool.

Testing Tools



```
cy# function printMethods(cla
cy> var count = new new Typ
cy> var methods = class_cop
cy> var methodsArray = [];
cy> for(var i = 0; i < *cou
cy> var method = methods
cy> methodsArray.push({se
n(method)});
cy> }
cy> free(methods);
cy> free(count);
cy> return methodsArray;
cy> }
cy#
```

```
/Users/alec/LINE
$ frida -U -l interceptSendMess
     _____
    (_____)
     |   |    Frida 5.0.0 - A wo
     |   |
     |`-'|    Commands:
     |   |        help        -> D
     |   |        object?     -> D
     |   |        exit/quit   -> E
     |   |
     |   |    More info at http:
      `._.'

[USB::iPod Touch 5G::LINE]->
[MessageViewController sendMess
```

```
Las
Lic
roo
sat
sat
Gen
---
Ser
Acc
Ent
Label:
Generic Field: (null)
Keychain Data: (null)

Generic Password
----------------
Service: ids
Account: identity-rsa-private-key
Entitlement Group: apple
Label: (null)
Generic Field: (null)
Keychain Data: (null)

Generic Password
```

Burp Suite Free Edition v1.5

Filter: Hiding not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders

Host	Method	URL	Params	Status	Length	MIME type	Title

OWASP
Open Web Application
Security Project

Common problems

- Need to rely on a multitude of different tools
  - each one developed for a specific need
  - each one with its own mode of operation (and syntax)
- Issues
  - steep learning curve
  - time wasted in configuring many different tools
  - a “*drozer for iOS*” was missing

OWASP
Open Web Application
Security Project

(a new format)

INTRODUCING: NEEDLE

What is Needle?

- A tool for auditing iOS Application Security
- An open source, modular framework
  - streamline the entire process
  - acts as a central hub

needle

OWASP
Open Web Application
Security Project

What it’s *not*

- Not a “drozer” for iOS
  - does not require an agent installed on the device (for now)
  - does require a jailbroken device
- Not a vuln scanner
  - knowledge (and intuition) of the tester is still required

Open Web Application Security Project

Motivation

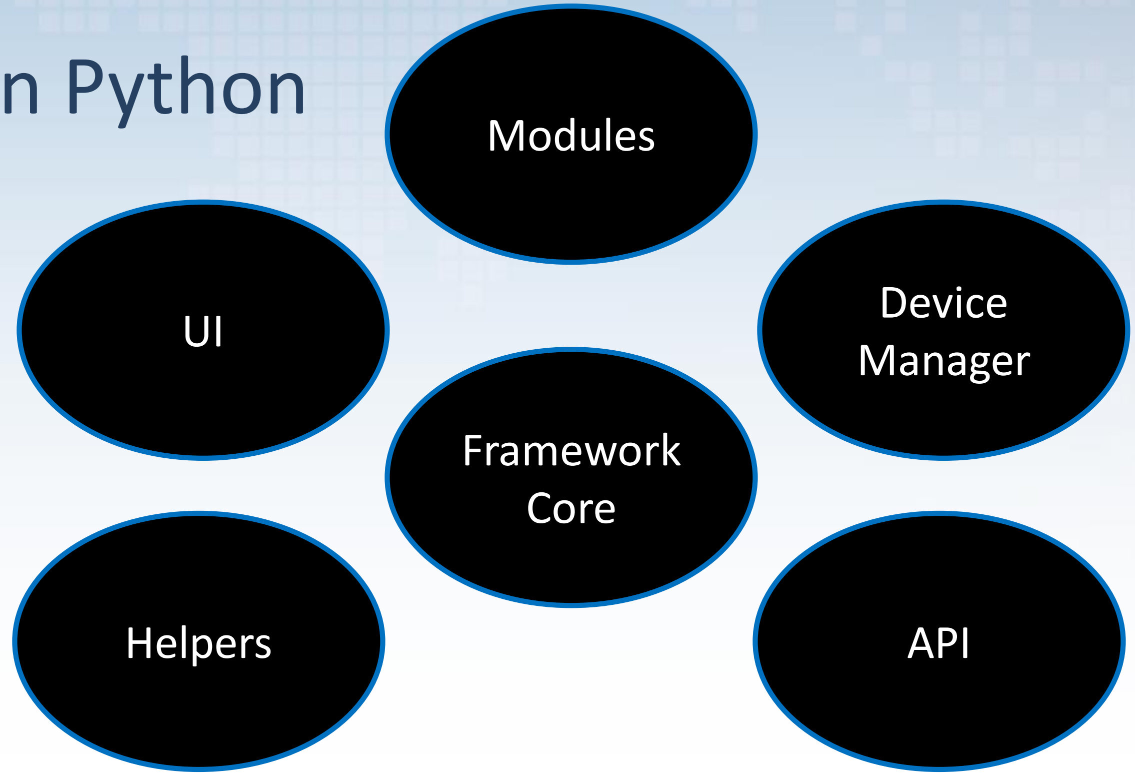
- Beginners: easy to use
- Professionals: save time during assessments
- Developers: quickly test their products

OWASP
Open Web Application Security Project

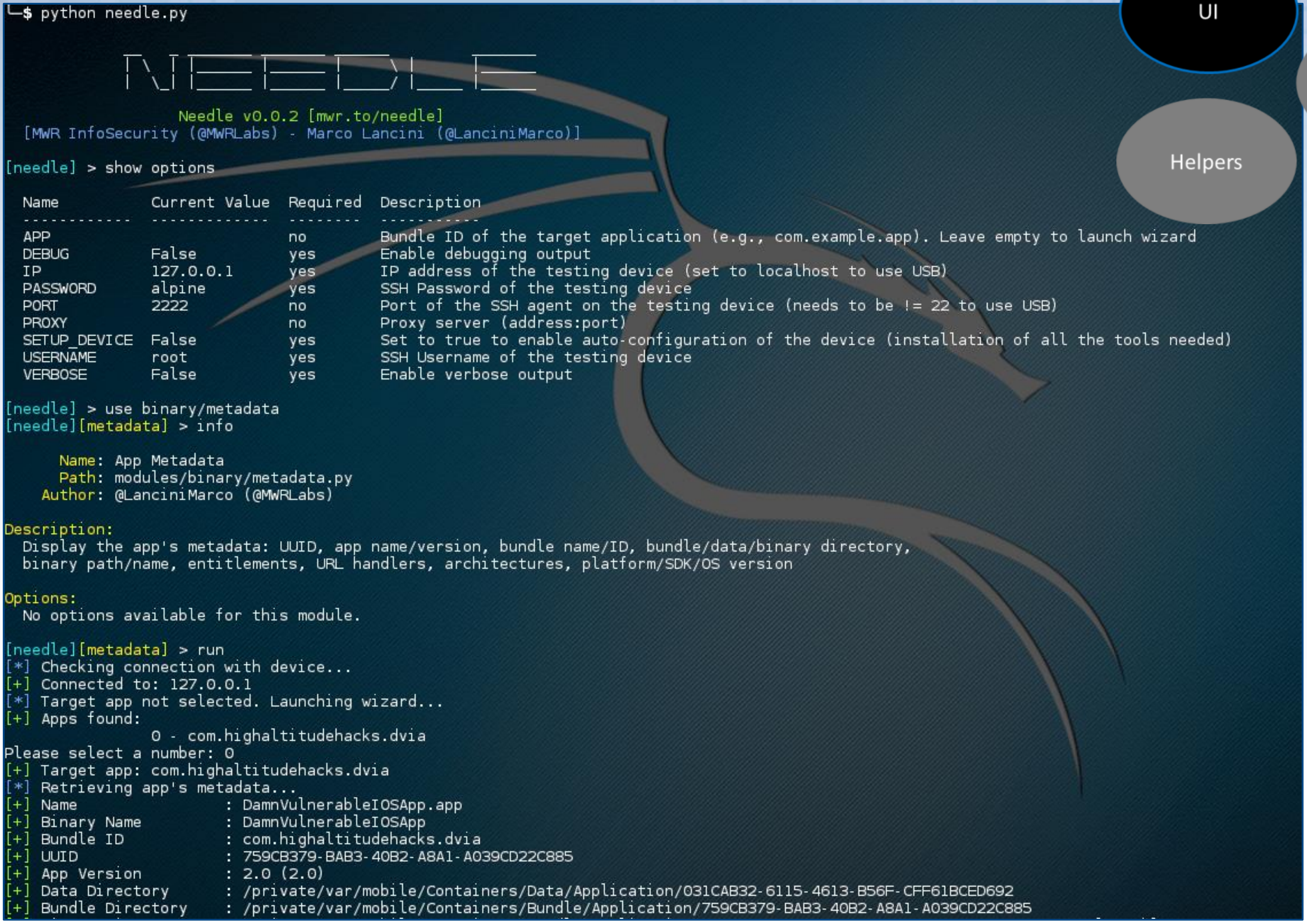
The Architecture

Architecture

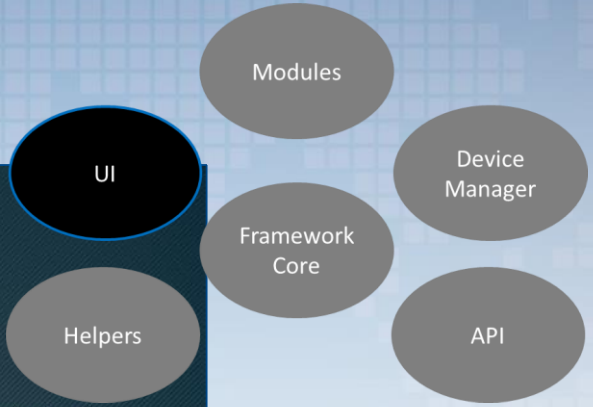
- Decoupled components
- Entirely written in Python



UI

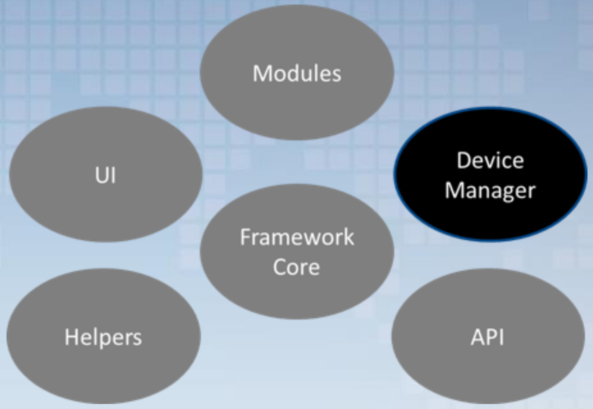


```
└─$ python needle.py

              NEEDLE

                 Needle v0.0.2 [mwr.to/needle]
  [MWR InfoSecurity (@MWRLabs) - Marco Lancini (@LanciniMarco)]

[needle] > show options

  Name          Current Value  Required  Description
  ------------  -------------  --------  -----------
  APP                          no        Bundle ID of the target application (e.g., com.example.app). Leave empty to launch wizard
  DEBUG         False          yes       Enable debugging output
  IP            127.0.0.1      yes       IP address of the testing device (set to localhost to use USB)
  PASSWORD      alpine         yes       SSH Password of the testing device
  PORT          2222           no        Port of the SSH agent on the testing device (needs to be != 22 to use USB)
  PROXY                        no        Proxy server (address:port)
  SETUP_DEVICE  False          yes       Set to true to enable auto-configuration of the device (installation of all the tools needed)
  USERNAME      root           yes       SSH Username of the testing device
  VERBOSE       False          yes       Enable verbose output

[needle] > use binary/metadata
[needle][metadata] > info

      Name: App Metadata
      Path: modules/binary/metadata.py
    Author: @LanciniMarco (@MWRLabs)

Description:
  Display the app's metadata: UUID, app name/version, bundle name/ID, bundle/data/binary directory,
  binary path/name, entitlements, URL handlers, architectures, platform/SDK/OS version

Options:
  No options available for this module.

[needle][metadata] > run
[*] Checking connection with device...
[+] Connected to: 127.0.0.1
[*] Target app not selected. Launching wizard...
[+] Apps found:
                0 - com.highaltitudehacks.dvia
Please select a number: 0
[+] Target app: com.highaltitudehacks.dvia
[*] Retrieving app's metadata...
[+] Name                : DamnVulnerableIOSApp.app
[+] Binary Name         : DamnVulnerableIOSApp
[+] Bundle ID           : com.highaltitudehacks.dvia
[+] UUID                : 759CB379-BAB3-40B2-A8A1-A039CD22C885
[+] App Version         : 2.0 (2.0)
[+] Data Directory      : /private/var/mobile/Containers/Data/Application/031CAB32-6115-4613-B56F-CFF61BCED692
[+] Bundle Directory    : /private/var/mobile/Containers/Bundle/Application/759CB379-BAB3-40B2-A8A1-A039CD22C885
```



Open Web Application Security Project

Device Manager



- Manage connections with the iDevice
  - SSH over Wi-Fi
  - SSH over USB
- Device setup, port forwarding, cleanup…
- Basic commands
  - shell, push/pull
- App management
  - metadata, open, decrypt, data protection…

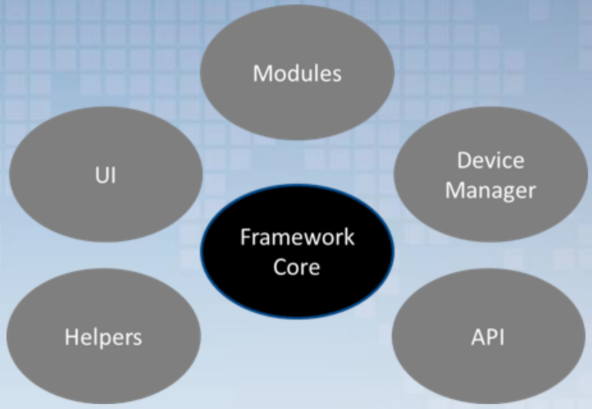
demo

Basic Usage

DEMO

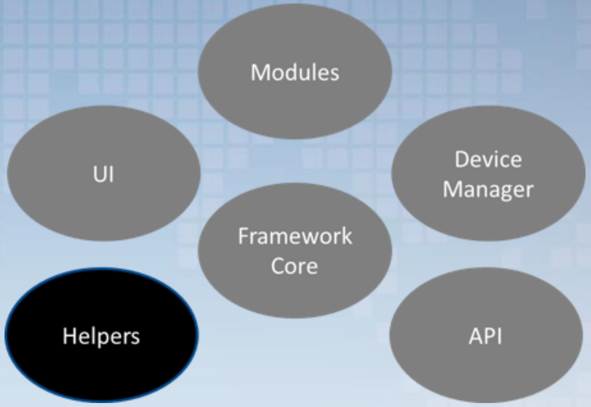
OWASP
Open Web Application
Security Project

Framework Core



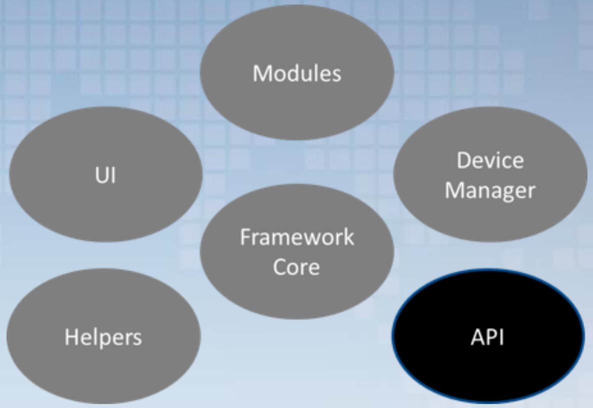
- Initialize and manage all the other components
- Load/execute modules/jobs
- Maintain status
  - global options, loaded modules, running jobs, device status…
  - pointers to instantiated objects
  - constants

Helpers



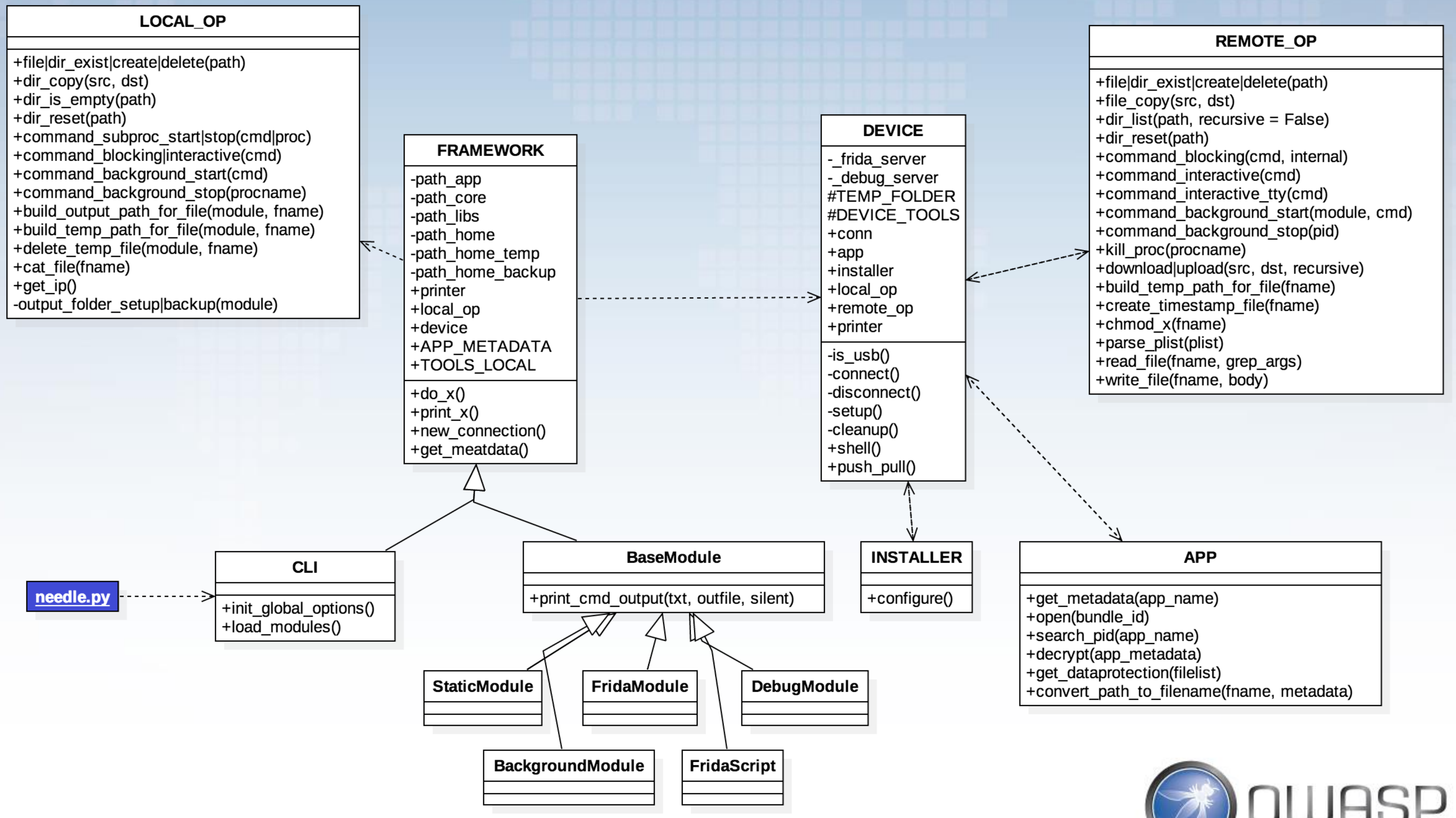
- Common functionalities offered both to the Core and APIs
- Sanitization, logging, printing…

API

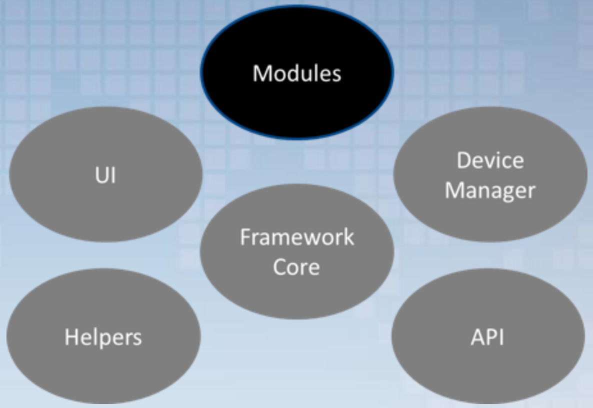


- The framework core exposes APIs to interact with the local and remote OS
- These wraps common functionalities
  - file and data access
  - command execution
  - networking
- Speed-up creation of new modules

API

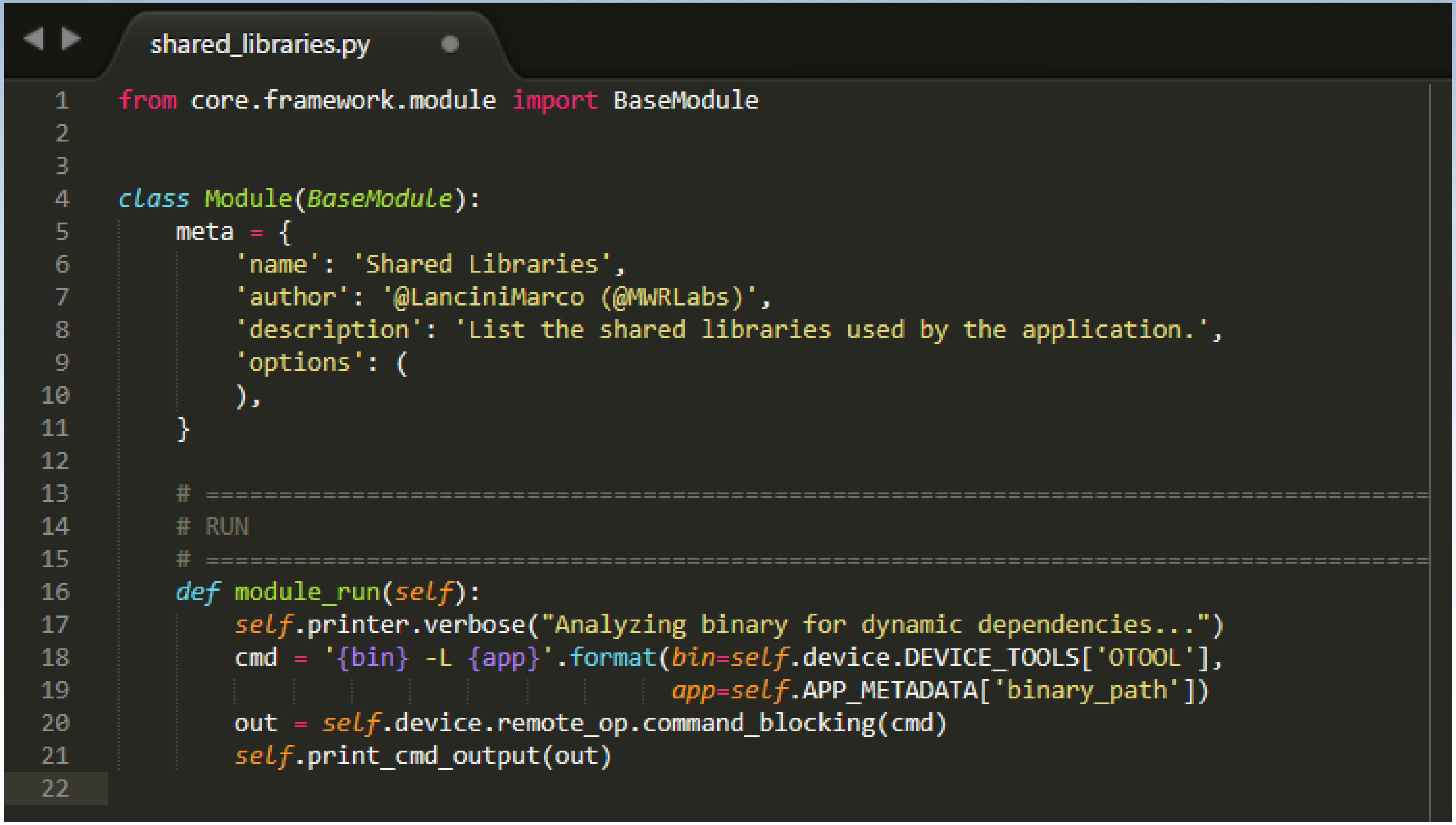

LOCAL_OP
+file|dir_exist|create|delete(path)
+dir_copy(src, dst)
+dir_is_empty(path)
+dir_reset(path)
+command_subproc_start|stop(cmd|proc)
+command_blocking|interactive(cmd)
+command_background_start(cmd)
+command_background_stop(procname)
+build_output_path_for_file(module, fname)
+build_temp_path_for_file(module, fname)
+delete_temp_file(module, fname)
+cat_file(fname)
+get_ip()
-output_folder_setup|backup(module)
FRAMEWORK
-path_app
-path_core
-path_libs
-path_home
-path_home_temp
-path_home_backup
+printer
+local_op
+device
+APP_METADATA
+TOOLS_LOCAL
+do_x()
+print_x()
+new_connection()
+get_meatdata()
DEVICE
-_frida_server
-_debug_server
#TEMP_FOLDER
#DEVICE_TOOLS
+conn
+app
+installer
+local_op
+remote_op
+printer
-is_usb()
-connect()
-disconnect()
-setup()
-cleanup()
+shell()
+push_pull()
REMOTE_OP
+file|dir_exist|create|delete(path)
+file_copy(src, dst)
+dir_list(path, recursive = False)
+dir_reset(path)
+command_blocking(cmd, internal)
+command_interactive(cmd)
+command_interactive_tty(cmd)
+command_background_start(module, cmd)
+command_background_stop(pid)
+kill_proc(procname)
+download|upload(src, dst, recursive)
+build_temp_path_for_file(fname)
+create_timestamp_file(fname)
+chmod_x(fname)
+parse_plist(plist)
+read_file(fname, grep_args)
+write_file(fname, body)
needle.py
CLI
+init_global_options()
+load_modules()
BaseModule
+print_cmd_output(txt, outfile, silent)
INSTALLER
+configure()
APP
+get_metadata(app_name)
+open(bundle_id)
+search_pid(app_name)
+decrypt(app_metadata)
+get_dataprotection(filelist)
+convert_path_to_filename(fname, metadata)
StaticModule
FridaModule
DebugModule
BackgroundModule
FridaScript

Modules



- Heart of Needle’s functionalities
- Collection of python scripts

Modules / Sample



shared_libraries.py

```python
from core.framework.module import BaseModule


class Module(BaseModule):
    meta = {
        'name': 'Shared Libraries',
        'author': '@LanciniMarco (@MWRLabs)',
        'description': 'List the shared libraries used by the application.',
        'options': (
        ),
    }

    # ==========================================================================
    # RUN
    # ==========================================================================
    def module_run(self):
        self.printer.verbose("Analyzing binary for dynamic dependencies...")
        cmd = '{bin} -L {app}'.format(bin=self.device.DEVICE_TOOLS['OTOOL'],
                                      app=self.APP_METADATA['binary_path'])
        out = self.device.remote_op.command_blocking(cmd)
        self.print_cmd_output(out)

```

demo

Modules

DEMO

OWASP
Open Web Application
Security Project

Currently Supported Modules

Binary

- App Metadata
- Compilation Checks
- Shared Libraries
- Strings
- Class Dump
- Install IPA
- Pull IPA

Storage

- Binary Cookies
- Cache.db Files
- Plist Files
- SQL Files
- Dump Keychain
- Screenshot Caching
- Keyboard Autocomplete Caching

OWASP
Open Web Application Security Project

Currently Supported Modules

Dynamic	Hooking
• Jailbreak Detection • URI Handler • Heap Dump • Monitor File changes • Monitor OS Pasteboard • Syslog Monitor • Syslog Watch	• Cycript shell • Frida shell • Frida trace • Frida launcher • Enumerate Classes (script) • Enumerate Methods (script) • Enumerate All Methods (script)

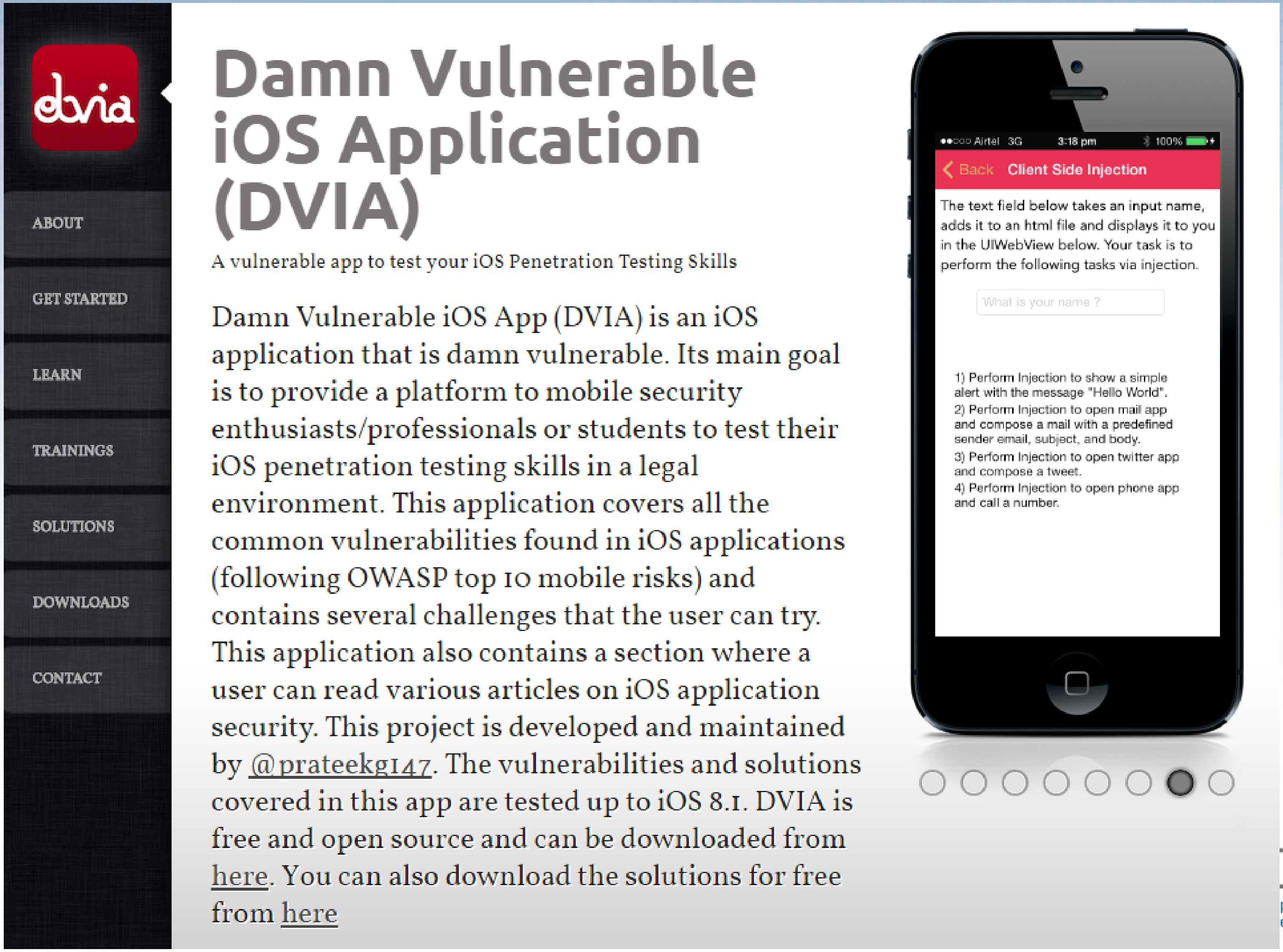
Currently Supported Modules

Comms	Static
• List Installed Certificates • Export Installed Certificates • Import Installed Certificates • Delete Installed Certificates • Install MitmProxy CA Certificate • Intercepting Proxy	• Code Checks

OWASP
Open Web Application
Security Project

ACTION TIME

DVIA



Damn Vulnerable iOS Application (DVIA)

A vulnerable app to test your iOS Penetration Testing Skills

Damn Vulnerable iOS App (DVIA) is an iOS application that is damn vulnerable. Its main goal is to provide a platform to mobile security enthusiasts/professionals or students to test their iOS penetration testing skills in a legal environment. This application covers all the common vulnerabilities found in iOS applications (following OWASP top 10 mobile risks) and contains several challenges that the user can try. This application also contains a section where a user can read various articles on iOS application security. This project is developed and maintained by @prateekg147. The vulnerabilities and solutions covered in this app are tested up to iOS 8.1. DVIA is free and open source and can be downloaded from here. You can also download the solutions for free from here

The text field below takes an input name, adds it to an html file and displays it to you in the UIWebView below. Your task is to perform the following tasks via injection.

1) Perform Injection to show a simple alert with the message "Hello World".
2) Perform Injection to open mail app and compose a mail with a predefined sender email, subject, and body.
3) Perform Injection to open twitter app and compose a tweet.
4) Perform Injection to open phone app and call a number.

demo

Binary Analysis

DEMO

Open Web Application Security Project

demo

Storage

DEMO

Open Web Application Security Project

demo

Dynamic Analysis

DEMO

OWASP
Open Web Application
Security Project

demo

Hooking

DEMO

Open Web Application Security Project

demo

Network Comms

DEMO

Open Web Application Security Project

demo

Static Analysis

DEMO

Open Web Application Security Project

ROADMAP

Roadmap

Agent to deploy on device

- Replace all the dependencies

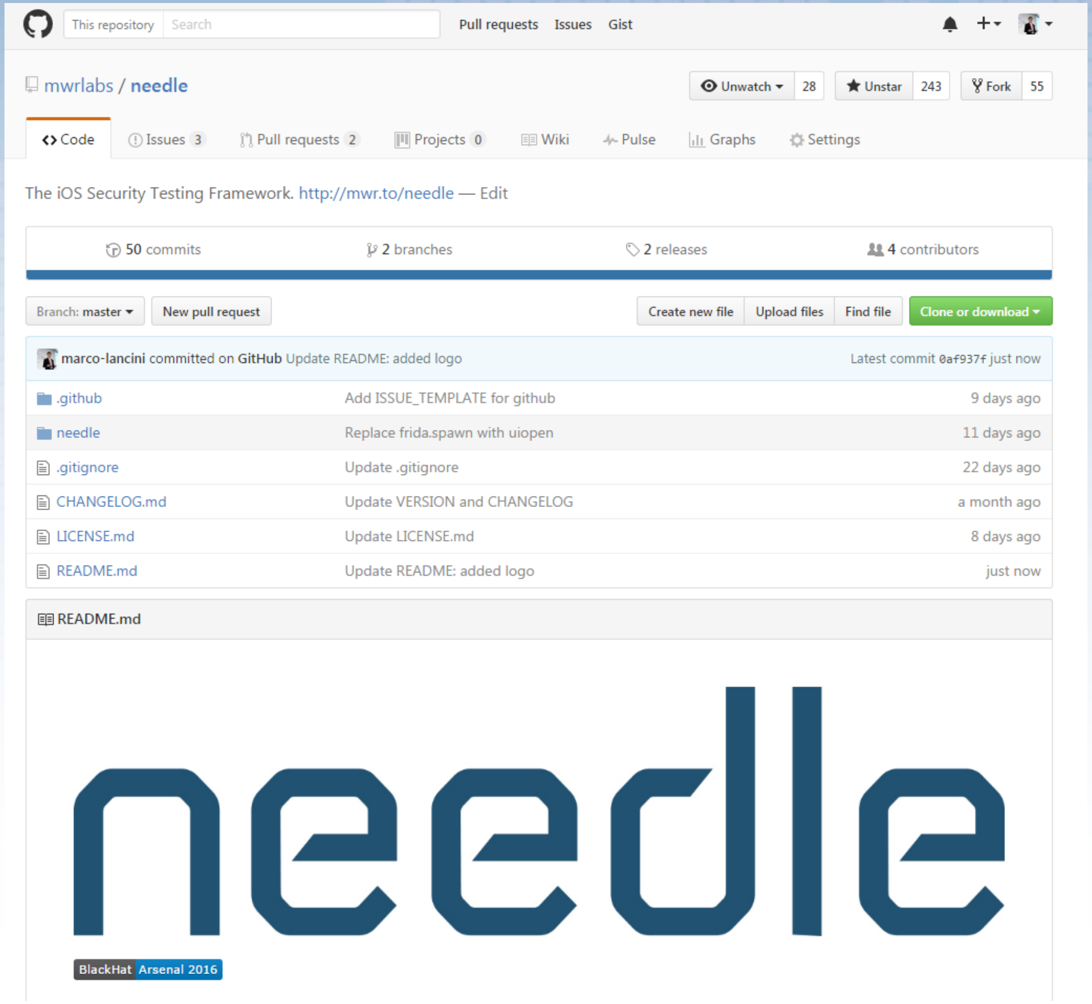
Support for non-jailbroken devices

New modules

- Substrate integration
- WebView scanner
- Hook Swift methods
- URI handlers fuzzer
- Pinning detection/bypass
- Obfuscation detection

… community based

Wanna help?



mwrlabs / needle

The iOS Security Testing Framework. http://mwr.to/needle — Edit

50 commits · 2 branches · 2 releases · 4 contributors

marco-lancini committed on GitHub Update README: added logo — Latest commit 0af937f just now

.github	Add ISSUE_TEMPLATE for github	9 days ago
needle	Replace frida.spawn with uiopen	11 days ago
.gitignore	Update .gitignore	22 days ago
CHANGELOG.md	Update VERSION and CHANGELOG	a month ago
LICENSE.md	Update LICENSE.md	8 days ago
README.md	Update README: added logo	just now

README.md

needle

BlackHat Arsenal 2016

Want to know more?

mwr.to/needle

@mwrneedle

OWASP
Open Web Application
Security Project