Test security and privacy of your mobile application (iOS & Android), detect OWASP Mobile Top 10 and other weaknesses.

Summary of Mobile Application Security Test

APP NAME	APP ID	APP VERSION
SmartTemp	com.stap.SMT775	47
DEVICE TYPE	**TEST STARTED**	**TEST FINISHED**
iOS	November 6th 2019, 12:52	November 6th 2019, 13:16

DAST was not performed because the uploaded iOS application is not compiled for Simulator.

OWASP Mobile Top 10

The automated audit revealed the following security flaws and weaknesses that may impact the application:

WARNINGS	LOW RISK	MEDIUM RISK	HIGH RISK
4	0	1	0

Zero false-positive SLA and advanced manual testing of application is only available in ImmuniWeb® MobileSuite.

PREDICTABLE RANDOM NUMBER GENERATOR [M5] [CWE-338] [SAST]

MEDIUM

Description:
The mobile application uses a predictable Random Number Generator (RNG).
Under certain conditions this weakness may jeopardize mobile application data encryption or other protection based on randomization. For example, if encryption tokens are generated inside of the application and an attacker can provide application with a predictable token to validate and then execute a sensitive activity within the application or its backend.

Example of insecure code:

```
FILE *fp = fopen("/dev/random", "r");

if (!fp) {
    perror("randgetter");
    exit(-1);
}

uint64_t value = 0;
int i;
for (i=0; i<sizeof(value); i++) {
    value <<= 8;
    value |= fgetc(fp);
}

fclose(fp);
```

Example of secure code:

```
uint8_t randomBytes[16];
int result = SecRandomCopyBytes(kSecRandomDefault, 16, randomBytes);
if(result == 0) {
    NSMutableString *uuidStringReplacement = [[NSMutableString alloc]
initWithCapacity:16*2];
    for(NSInteger index = 0; index < 16; index++)
    {
        [uuidStringReplacement appendFormat: @"%02x", randomBytes[index]];
    }
    NSLog(@"uuidStringReplacement is %@", uuidStringReplacement);
} else {
    NSLog(@"SecRandomCopyBytes failed for some reason");
}
```

Details:
File: ios/Payload/ios.app/FridaGadget.dylib
- Binary match usage of 'random' function/method.
- Binary match usage of 'srand' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftCore.dylib
- Binary match usage of 'random' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftFoundation.dylib
- Binary match usage of 'random' function/method.

File: ios/SwiftSupport/iphoneos/libswiftCore.dylib
- Binary match usage of 'random' function/method.

File: ios/SwiftSupport/iphoneos/libswiftFoundation.dylib
- Binary match usage of 'random' function/method.

CVSSv3 Base Score:
4.8 (AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:L/A:N)

Reference:
- https://developer.apple.com/library/content/documentation/Security/Conceptual/cryptoservices/GeneralPurposeCrypto/GeneralPurposeCrypto.html
- https://developer.apple.com/library/content/documentation/Security/Conceptual/cryptoservices/RandomNumberGenerationAPIs/RandomNumberGenerationAPIs.html

WEAK HASHING ALGORITHMS [M5] [CWE-916] [SAST]

WARNING

Description:
The mobile application uses weak hashing algorithms. Weak hashing algorithms (e.g. MD2, MD4, MD5 or SHA-1) can be vulnerable to collisions and other security weaknesses, and should not be used when reliable hashing of data is required.

Details:
File: ios/Payload/ios.app/FridaGadget.dylib
- Binary match usage of 'CC_SHA1' function/method.

CVSSv3 Base Score:
5.5 (AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N)

Reference:
- https://developer.apple.com/library/content/documentation/Security/Conceptual/cryptoservices/GeneralPurposeCrypto/GeneralPurposeCrypto.html

USAGE OF BANNED API FUNCTIONS [M10] [CWE-477] [SAST]

WARNING

Description:
The mobile application uses some of the banned API functions. API functions are usually banned for compelling security and privacy reasons and shall not be used.

Details:
File: ios/Payload/ios.app/FridaGadget.dylib
- Binary match usage of 'alloca' function/method.
- Binary match usage of 'memcpy' function/method.
- Binary match usage of 'printf' function/method.
- Binary match usage of 'sprintf' function/method.
- Binary match usage of 'strcat' function/method.
- Binary match usage of 'strcpy' function/method.
- Binary match usage of 'strncpy' function/method.
- Binary match usage of 'vsnprintf' function/method.
- Binary match usage of 'gets' function/method.
- Binary match usage of 'scanf' function/method.
- Binary match usage of 'sscanf' function/method.
- Binary match usage of 'strlen' function/method.
- Binary match usage of 'wcslen' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftObjectiveC.dylib
- Binary match usage of 'alloca' function/method.
- Binary match usage of 'memcpy' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftCore.dylib
- Binary match usage of 'alloca' function/method.
- Binary match usage of 'printf' function/method.
- Binary match usage of 'sprintf' function/method.
- Binary match usage of 'gets' function/method.
- Binary match usage of 'memcpy' function/method.
- Binary match usage of 'strcpy' function/method.
- Binary match usage of 'strlen' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftCoreGraphics.dylib
- Binary match usage of 'alloca' function/method.
- Binary match usage of 'memcpy' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftUIKit.dylib
- Binary match usage of 'memcpy' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftMetal.dylib
- Binary match usage of 'alloca' function/method.
- Binary match usage of 'memcpy' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftDispatch.dylib
- Binary match usage of 'memcpy' function/method.
- Binary match usage of 'alloca' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftos.dylib
- Binary match usage of 'memcpy' function/method.
- Binary match usage of 'alloca' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftDarwin.dylib
- Binary match usage of 'alloca' function/method.
- Binary match usage of 'printf' function/method.
- Binary match usage of 'vsnprintf' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftFoundation.dylib
- Binary match usage of 'alloca' function/method.

- Binary match usage of 'memcpy' function/method.
- Binary match usage of 'strlen' function/method.

File: ios/SwiftSupport/iphoneos/libswiftObjectiveC.dylib
- Binary match usage of 'alloca' function/method.
- Binary match usage of 'memcpy' function/method.

File: ios/SwiftSupport/iphoneos/libswiftCore.dylib
- Binary match usage of 'alloca' function/method.
- Binary match usage of 'printf' function/method.
- Binary match usage of 'sprintf' function/method.
- Binary match usage of 'gets' function/method.
- Binary match usage of 'memcpy' function/method.
- Binary match usage of 'strcpy' function/method.
- Binary match usage of 'strlen' function/method.

File: ios/SwiftSupport/iphoneos/libswiftCoreGraphics.dylib
- Binary match usage of 'alloca' function/method.
- Binary match usage of 'memcpy' function/method.

File: ios/SwiftSupport/iphoneos/libswiftUIKit.dylib
- Binary match usage of 'memcpy' function/method.

File: ios/SwiftSupport/iphoneos/libswiftMetal.dylib
- Binary match usage of 'alloca' function/method.
- Binary match usage of 'memcpy' function/method.

File: ios/SwiftSupport/iphoneos/libswiftDispatch.dylib
- Binary match usage of 'memcpy' function/method.
- Binary match usage of 'alloca' function/method.

File: ios/SwiftSupport/iphoneos/libswiftos.dylib
- Binary match usage of 'memcpy' function/method.
- Binary match usage of 'alloca' function/method.

File: ios/SwiftSupport/iphoneos/libswiftDarwin.dylib
- Binary match usage of 'alloca' function/method.
- Binary match usage of 'printf' function/method.
- Binary match usage of 'vsnprintf' function/method.

File: ios/SwiftSupport/iphoneos/libswiftFoundation.dylib
- Binary match usage of 'alloca' function/method.
- Binary match usage of 'memcpy' function/method.
- Binary match usage of 'strlen' function/method.

Reference:
- https://msdn.microsoft.com/en-us/library/bb288454.aspx
- https://developer.apple.com/library/content/documentation/Security/Conceptual/SecureCodingGuide/Articles/BufferOverflows.html
- https://developer.apple.com/library/content/documentation/Security/Conceptual/SecureCodingGuide/SecurityDevelopmentChecklists/SecurityDevelopmentChecklists.html#//apple_ref/doc/uid/TP40002415-CH1-SW1

USAGE OF MALLOC() FUNCTION [M10] [CWE-477] [SAST]

WARNING

Description:
The mobile application uses malloc() function to allocate new memory instead of more secure calloc(), thus endangering application privacy under certain circumstances (e.g. if freed memory can be accessed by an

attacker).

Details:
File: ios/Payload/ios.app/FridaGadget.dylib
- Binary match usage of 'malloc' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftCore.dylib
- Binary match usage of 'malloc' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftCoreGraphics.dylib
- Binary match usage of 'malloc' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftMetal.dylib
- Binary match usage of 'malloc' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftDispatch.dylib
- Binary match usage of 'malloc' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftos.dylib
- Binary match usage of 'malloc' function/method.

File: ios/Payload/ios.app/Frameworks/libswiftFoundation.dylib
- Binary match usage of 'malloc' function/method.

File: ios/SwiftSupport/iphoneos/libswiftCore.dylib
- Binary match usage of 'malloc' function/method.

File: ios/SwiftSupport/iphoneos/libswiftCoreGraphics.dylib
- Binary match usage of 'malloc' function/method.

File: ios/SwiftSupport/iphoneos/libswiftMetal.dylib
- Binary match usage of 'malloc' function/method.

File: ios/SwiftSupport/iphoneos/libswiftDispatch.dylib
- Binary match usage of 'malloc' function/method.

File: ios/SwiftSupport/iphoneos/libswiftos.dylib
- Binary match usage of 'malloc' function/method.

File: ios/SwiftSupport/iphoneos/libswiftFoundation.dylib
- Binary match usage of 'malloc' function/method.

Reference:
- https://developer.apple.com/library/content/documentation/Performance/Conceptual/ManagingMemory/Articles/MemoryAlloc.html

MISSING ANTI-EMULATION [SAST]

WARNING

Description:
The mobile application does not use any anti-emulation or anti-debugger techniques (e.g. detecting rooted devices or checking if contacts are authentic).
This can significantly facilitate application debugging and reverse-engineering processes.

Mobile Application Behaviour

Mobile Application Functionality

The mobile application uses the following functionality that can endanger user's privacy under certain circumstances:

Camera

The mobile application can use phone’s camera for taking pictures or videos.

Location

The mobile application has an access to user geographical location.