Security Assessment Report

KZen - iOS application PT

Presented to

Yaron Hakon
Application Security Expert & CEO, AppSec Labs

June 2019

Table of contents:

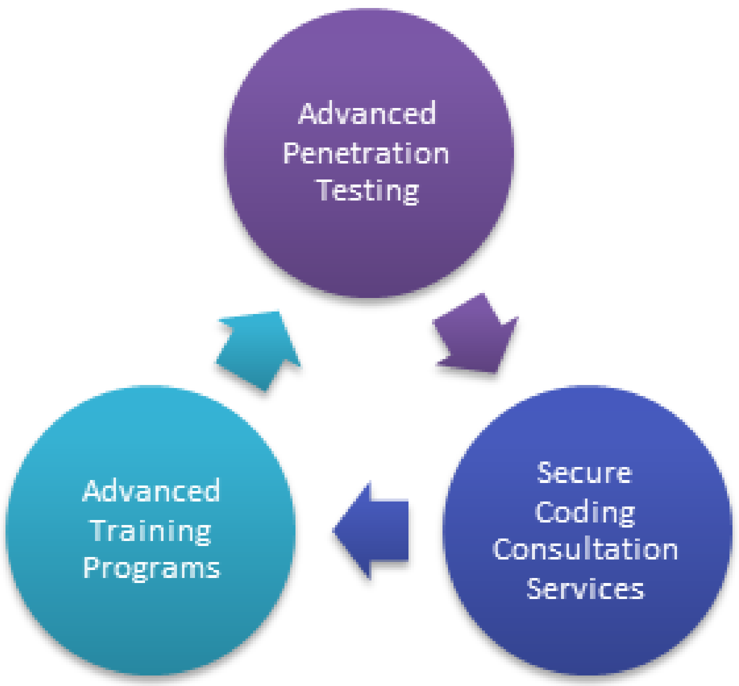
Chapter A – Introduction to AppSec

AppSec Labs provides cutting edge professionalism and excellence in the field of **application security**.

Our expertise in security allows us to provide the correct solutions for our clients in each of their fields, from hi-tech and software, through biomed and education as well as banking and finance, government, national security, communications, e-commerce and IoT. This diverse customer-base enables us to maintain our position at the forefront of technology.

Providing customers, a full range of application security services, including:

- ✓ **R&D** – security consulting throughout the R&D and design stages
- ✓ **SDLC** – secure development lifecycle implementation
- ✓ **Training** – our academy offers hands-on courses in application hacking and secure coding
- ✓ **Security tests** – application penetration testing for web, desktop, cloud, mobile and IoT applications


Advanced Penetration Testing
Secure Coding Consultation Services
Advanced Training Programs

Chapter B – Executive Summary

General

AppSec Labs was requested by KZen to perform security assessment for the iOS Application during February 2019. AppSec Labs hereby confirms that the tests have been completed and the results were delivered to KZen.
The following document summarizes the current security state of the iOS Application.

Testing methodology

The penetration testing cycle was performed using automated and manual tools, in order to cover a wide range of applicative vulnerabilities as recommended by the OWASP and WASC methodologies.

A gray box approach was utilized during the tests. Gray box testing is a combination of both black and white box testing and refers to testing a system while having at least some knowledge of the internals of a system. The tests included access to the following resources:

- Interviews with developers
- Source code full access (mobile and server)

For a list of performed tests please refer to Appendix A.

Testing Scope

The following areas were covered and included in the testing scope for iOS Application security assessment:

- All tests were performed on the iOS application, version 2019.6.3.72806, and corresponding backend server located at pentest.pingpong-service.com domain.
- Test environment: dedicated pent-test environment
- Users: self-enrolled

Limitations

The following limitations/disclaimers were taken into consideration while performing the security analysis:

- The goal of the penetration test is to provide an effective security evaluation of the system, considering the testing scope, details and limitations. The goal is to perform a best effort to identify and provide a list of unique security issues that can be used to exploit and jeopardize the system. The report does not necessarily cover all instances of each vulnerability, and the suggested mitigations should be implemented throughout the entire application, and not only for the provided examples.

Conclusions

The system has been found to meet AppSec Labs' security criteria as recommended by the OWASP and WASC methodologies.
Based on the results of testing and verification process completed on June 2019, and in accordance with the testing scope, details and limitations as stated in this document, AppSec Labs confirms that there are no open critical-risk, high-risk, or medium-risk vulnerabilities identified at the time of report submission.

Cautionary note

The vulnerability assessment that AppSec Labs performed was based on past experiences, currently available information, and known threats as of the date of testing. Given the constantly evolving nature of information security threats and vulnerabilities, there can be no assurance that any assessment will identify all possible vulnerabilities, or propose exhaustive and operationally viable recommendations to mitigate those exposures. The statements relevant to the security of the iOS Application *in this letter reflect the conditions found at the completion of testing. In accepting our report, KZen has acknowledged the validity of the above cautionary statement.*

Chapter C – Threat Level Methodology

Threat level of the vulnerabilities

The severity of the vulnerabilities detected during the tests was determined using OWASP and WASC methodologies. The following describes the impact of each threat level:

Critical

- A security breach that exposes a major security risk with a direct exploit (not needing user involvement). If exploited, the security threat might cause major damage to the application and/or have major impact on the company. The likelihood of such attack to occur is high, considering the architecture/business-logic/complexity of the exploit.

High

- The weakness identified has the potential to directly compromise the confidentiality, integrity and / or availability of the system or data, but the likelihood to occur is not high, considering the architecture/business-logic/complexity of the exploit. The possible damage to the application or the company is high, but not a total disaster.
- In applications involving sensitive data, the risk might be considered high in case the weakness by itself is against common regulations (e.g. PCI).

Medium

- A medium security issue that imposes some affect/damage to the application. Often it cannot be used directly, but can assist an attacker to launch further attacks.

Low

- No direct threat exists. It is a risk much more rather than a threat and does not cause damage by itself. The vulnerability may be leveraged with other vulnerabilities in order to launch further attacks.
- The risk reveals technical information which might assist an attacker in launching future attacks.

Appendix A - List of Attacks and Tests

The following table is a generic list of tests performed by AppSec Labs during the original security testing cycle (and **NOT** a list of vulnerabilities detected in the system). The list includes known attacks valid at the time of the production of this report.

Category	Test Name
Information Gathering	Search engine discovery / reconnaissance
	Web application fingerprint
	Review Webpage Comments and Metadata for Information Leakage
	Application entry points Identification
	Execution paths mapping
	Web application framework fingerprinting
	Web application fingerprinting
	Application architecture mapping
	Information Disclosure by error codes
	SSL Weakness - SSL/TLS Testing (SSL Version, Algorithms, Key length, Digital Cert. Validity)
Configuration and Deploy Management Testing	Application Configuration management weakness
	File extensions handling - sensitive information
	Old, Backup and Unreferenced Files - Sensitive Information
	Unauthorized Admin Interfaces access
	HTTP Methods enabled, XST permitted, HTTP Verb
	Http strict transport security
	RIA cross domain policy
	Role definitions enumeration
	Vulnerable user registration process
	Vulnerable account provisioning process
	Permissions of Guest/Low Permission Accounts
	Account suspension/resumption process
Authentication Testing	Credentials Transported over Unencrypted Channel
	User enumeration
	Account lockout
	Authentication bypass
	"Remember password" functionality
	Browser caching
	Weak password policy
	Weak password security mechanisms
	Weak password change or reset flow
	Race conditions
	Weak multiple factors authentication
	Weak CAPTCHA implementation
	Weaker authentication in alternative channel
Authorization Testing	Directory traversal/file inclusion
	Authorization schema bypass

Session Management Testing	Privilege escalation
	Insecure direct object references
	Session management bypass
	Cookies are set without 'HTTP Only', 'Secure', and no time validity
	Session fixation
	Exposed session variables
	Cross site request forgery (CSRF)
	Logout management
	Session timeout
	Session puzzling
Data Validation Testing	Reflected cross site scripting
	Stored cross site scripting
	HTTP verb tampering
	HTTP Parameter pollution / manipulation
	SQL injection
	LDAP injection
	ORM injection
	XML injection
	SSI injection
	Xpath Injection
	IMAP/SMTP injection
	Code injection
	Local/remote file inclusion
	Command injection
	Buffer overflow
	Heap overflow
	Stack overflow
	Format string manipulation
	Incubated vulnerabilities
	HTTP splitting/smuggling
Error Handling	Analysis of Error Codes
	Analysis of Stack Traces
Cryptography	Weak SSL/TLS ciphers, insufficient transport layer protection
	Padding oracle
	Sensitive information sent via unencrypted channels
Business Logic Testing	Business logic data validation
	Ability to Forge Requests
	Integrity checks
	Process timing
	Replay attack
	Circumvention of Work Flows
	Abuse of Functionality
	File upload vulnerabilities
Client Side Testing	DOM based Cross Site Scripting
	Javascript Execution
	Html/css injection
	Client side url redirect
	Client side resource manipulation

	Cross origin resource sharing
	Cross site flashing
	Clickjacking / UI rendering
	Web sockets
	Web messaging
	Local storage / session storage sensitive information
AJAX Testing	AJAX weakness
Denial of Service Testing	SQL Wildcard vulnerability
	Locking customer accounts
	Buffer overflows
	User specified object allocation
	User Input as a Loop Counter
	Writing User Provided Data to Disk
	Failure to Release Resources
	Storing too Much Data in Session
Web Services Testing	WS information gathering
	WSDL weakness
	Weak xml structure
	XML content-level
	WS HTTP GET parameters/REST
	WS Naughty SOAP attachments
	WS replay testing