Mobile Application Security Verification Standard

OWASP MASTG v1.5.0 (commit: 3b92785
OWASP MASVS v1.4.2 (commit: 2a8b582)

Architecture, Design and Threat Modeling Requirements

ID	MASVS-ID	Detailed Verification Requirement	L1	L2	R	Common	Android	iOS	Status
1.1	MSTG-ARCH-1	All app components are identified and known to be needed.	■	■					Pass
1.2	MSTG-ARCH-2	Security controls are never enforced only on the client side, but on the respective remote endpoints.	■	■					Pass
1.3	MSTG-ARCH-3	A high-level architecture for the mobile app and all connected remote services has been defined and security has been addressed in that architecture.	■	■					Pass
1.4	MSTG-ARCH-4	Data considered sensitive in the context of the mobile app is clearly identified.	■	■					Pass
1.5	MSTG-ARCH-5	All app components are defined in terms of the business functions and/or security functions they provide.		■					Pass
1.6	MSTG-ARCH-6	A threat model for the mobile app and the associated remote services has been produced that identifies potential threats and countermeasures.		■					N/A
1.7	MSTG-ARCH-7	All security controls have a centralized implementation.		■					Pass
1.8	MSTG-ARCH-8	There is an explicit policy for how cryptographic keys (if any) are managed, and the lifecycle of cryptographic keys is enforced. Ideally, follow a key management standard such as NIST SP 800-57.		■					N/A
1.9	MSTG-ARCH-9	A mechanism for enforcing updates of the mobile app exists.		■					N/A
1.10	MSTG-ARCH-10	Security is addressed within all parts of the software development lifecycle.		■					Pass
1.11	MSTG-ARCH-11	A responsible disclosure policy is in place and effectively applied.		■					N/A
1.12	MSTG-ARCH-12	The app should comply with privacy laws and regulations.	■	■					N/A

Data Storage and Privacy Requirements

ID	MASVS-ID	Detailed Verification Requirement	L1	L2	R	Common	Android	iOS	Status
2.1	MSTG-STORAGE-1	System credential storage facilities need to be used to store sensitive data, such as PII, user credentials or cryptographic keys.	■	■					Pass
2.2	MSTG-STORAGE-2	No sensitive data should be stored outside of the app container or system credential storage facilities.	■	■					Pass
2.3	MSTG-STORAGE-3	No sensitive data is written to application logs.	■	■					Pass
2.4	MSTG-STORAGE-4	No sensitive data is shared with third parties unless it is a necessary part of the architecture.	■	■					Pass
2.5	MSTG-STORAGE-5	The keyboard cache is disabled on text inputs that process sensitive data.	■	■					Pass
2.6	MSTG-STORAGE-6	No sensitive data is exposed via IPC mechanisms.	■	■					Pass
2.7	MSTG-STORAGE-7	No sensitive data, such as passwords or pins, is exposed through the user interface.	■	■					Pass
2.8	MSTG-STORAGE-8	No sensitive data is included in backups generated by the mobile operating system.		■					Pass
2.9	MSTG-STORAGE-9	The app removes sensitive data from views when moved to the background.		■					Pass
2.10	MSTG-STORAGE-10	The app does not hold sensitive data in memory longer than necessary, and memory is cleared explicitly after use.		■					Pass
2.11	MSTG-STORAGE-11	The app enforces a minimum device access-security policy, such as requiring the user to set a device passcode.		■					Fail
2.12	MSTG-STORAGE-12	The app educates the user about the types of personally identifiable information processed, as well as security best practices the user should follow in using the app.	■	■					Fail
2.13	MSTG-STORAGE-13	No sensitive data should be stored locally on the mobile device. Instead, data should be retrieved from a remote endpoint when needed and only be kept in memory.		■					Pass
2.14	MSTG-STORAGE-14	If sensitive data is still required to be stored locally, it should be encrypted using a key derived from hardware backed storage which requires authentication.		■					N/A
2.15	MSTG-STORAGE-15	The app's local storage should be wiped after an excessive number of failed authentication attempts.		■					Pass

Cryptography Requirements

ID	MASVS-ID	Detailed Verification Requirement	L1	L2	R	Common	Android	iOS	Status
3.1	MSTG-CRYPTO-1	The app does not rely on symmetric cryptography with hardcoded keys as a sole method of encryption.	■	■					N/A
3.2	MSTG-CRYPTO-2	The app uses proven implementations of cryptographic primitives.	■	■					N/A
3.3	MSTG-CRYPTO-3	The app uses cryptographic primitives that are appropriate for the particular use-case, configured with parameters that adhere to industry best practices.	■	■					N/A
3.4	MSTG-CRYPTO-4	The app does not use cryptographic protocols or algorithms that are widely considered deprecated for security purposes.	■	■					N/A
3.5	MSTG-CRYPTO-5	The app doesn't re-use the same cryptographic key for multiple purposes.	■	■					N/A
3.6	MSTG-CRYPTO-6	All random values are generated using a sufficiently secure random number generator.	■	■					N/A

Authentication and Session Management Requirements

ID	MASVS-ID	Detailed Verification Requirement	L1	L2	R	Common	Android	iOS	Status
4.1	MSTG-AUTH-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.	■	■					Pass
4.2	MSTG-AUTH-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.	■	■					Pass
4.3	MSTG-AUTH-3	If stateless token-based authentication is used, the server provides a token that has been signed using a secure algorithm.	■	■					Pass
4.4	MSTG-AUTH-4	The remote endpoint terminates the existing session when the user logs out.	■	■					Pass
4.5	MSTG-AUTH-5	A password policy exists and is enforced at the remote endpoint.	■	■					Pass
4.6	MSTG-AUTH-6	The remote endpoint implements a mechanism to protect against the submission of credentials an excessive number of times.	■	■					Pass
4.7	MSTG-AUTH-7	Sessions are invalidated at the remote endpoint after a predefined period of inactivity and access tokens expire.	■	■					Pass
4.8	MSTG-AUTH-8	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.		■					N/A
4.9	MSTG-AUTH-9	A second factor of authentication exists at the remote endpoint and the 2FA requirement is consistently enforced.		■					Fail
4.10	MSTG-AUTH-10	Sensitive transactions require step-up authentication.		■					N/A
4.11	MSTG-AUTH-11	The app informs the user of all sensitive activities with their account. Users are able to view a list of devices, view contextual information (IP address, location, etc.), and to block specific devices.		■					N/A
4.12	MSTG-AUTH-12	Authorization models should be defined and enforced at the remote endpoint.	■	■					Pass

Network Communication Requirements

ID	MASVS-ID	Detailed Verification Requirement	L1	L2	R	Common	Android	iOS	Status
5.1	MSTG-NETWORK-1	Data is encrypted on the network using TLS. The secure channel is used consistently throughout the app.	■	■					Pass
5.2	MSTG-NETWORK-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.	■	■					Pass
5.3	MSTG-NETWORK-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.	■	■					Pass
5.4	MSTG-NETWORK-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.		■					N/A
5.5	MSTG-NETWORK-5	The app doesn't rely on a single insecure communication channel (email or SMS) for critical operations, such as enrollments and account recovery.		■					Pass
5.6	MSTG-NETWORK-6	The app only depends on up-to-date connectivity and security libraries.		■					Pass

Platform Interaction Requirements

ID	MASVS-ID	Detailed Verification Requirement	L1	L2	R	Common	Android	iOS	Status
6.1	MSTG-PLATFORM-1	The app only requests the minimum set of permissions necessary.	■	■					Pass
6.2	MSTG-PLATFORM-2	All inputs from external sources and the user are validated and if necessary sanitized. This includes data received via the UI, IPC mechanisms such as intents, custom URLs, and network sources.	■	■					Pass
6.3	MSTG-PLATFORM-3	The app does not export sensitive functionality via custom URL schemes, unless these mechanisms are properly protected.	■	■					Pass
6.4	MSTG-PLATFORM-4	The app does not export sensitive functionality through IPC facilities, unless these mechanisms are properly protected.	■	■					N/A
6.5	MSTG-PLATFORM-5	JavaScript is disabled in WebViews unless explicitly required.	■	■					N/A
6.6	MSTG-PLATFORM-6	WebViews are configured to allow only the minimum set of protocol handlers required (ideally, only https is supported). Potentially dangerous handlers, such as file, tel and app-id, are disabled.	■	■					N/A
6.7	MSTG-PLATFORM-7	If native methods of the app are exposed to a WebView, verify that the WebView only renders JavaScript contained within the app package.	■	■					N/A
6.8	MSTG-PLATFORM-8	Object deserialization, if any, is implemented using safe serialization APIs.	■	■					Pass
6.9	MSTG-PLATFORM-9	The app protects itself against screen overlay attacks. (Android only)		■					N/A
6.10	MSTG-PLATFORM-10	A WebView's cache, storage, and loaded resources (JavaScript, etc.) should be cleared before the WebView is destroyed.		■					N/A
6.11	MSTG-PLATFORM-11	Verify that the app prevents usage of custom third-party keyboards whenever sensitive data is entered (iOS only).		■					N/A

Code Quality and Build Setting Requirements

ID	MASVS-ID	Detailed Verification Requirement	L1	L2	R	Common	Android	iOS	Status
7.1	MSTG-CODE-1	The app is signed and provisioned with a valid certificate, of which the private key is properly protected.	■	■					Fail
7.2	MSTG-CODE-2	The app has been built in release mode, with settings appropriate for a release build (e.g. non-debuggable).	■	■					Fail
7.3	MSTG-CODE-3	Debugging symbols have been removed from native binaries.	■	■					N/A
7.4	MSTG-CODE-4	Debugging code and developer assistance code (e.g. test code, backdoors, hidden settings) have been removed. The app does not log verbose errors or debugging messages.	■	■					Fail
7.5	MSTG-CODE-5	All third party components used by the mobile app, such as libraries and frameworks, are identified, and checked for known vulnerabilities.	■	■					Pass
7.6	MSTG-CODE-6	The app catches and handles possible exceptions.	■	■					Pass
7.7	MSTG-CODE-7	Error handling logic in security controls denies access by default.	■	■					N/A
7.8	MSTG-CODE-8	In unmanaged code, memory is allocated, freed and used securely.	■	■					Pass
7.9	MSTG-CODE-9	Free security features offered by the toolchain, such as byte-code minification, stack protection, PIE support and automatic reference counting, are activated.	■	■					Pass

Resilience Requirements

ID	MASVS-ID	Detailed Verification Requirement	L1	L2	R	Common	Android	iOS	Status
8.1	MSTG-RESILIENCE-1	The app detects, and responds to, the presence of a rooted or jailbroken device either by alerting the user or terminating the app.			■				N/A
8.2	MSTG-RESILIENCE-2	The app prevents debugging and/or detects, and responds to, a debugger being attached. All available debugging protocols must be covered.			■				N/A
8.3	MSTG-RESILIENCE-3	The app detects, and responds to, tampering with executable files and critical data within its own sandbox.			■				N/A
8.4	MSTG-RESILIENCE-4	The app detects, and responds to, the presence of widely used reverse engineering tools and frameworks on the device.			■				N/A
8.5	MSTG-RESILIENCE-5	The app detects, and responds to, being run in an emulator.			■				N/A
8.6	MSTG-RESILIENCE-6	The app detects, and responds to, tampering the code and data in its own memory space.			■				N/A
8.7	MSTG-RESILIENCE-7	The app implements multiple mechanisms in each defense category (8.1 to 8.6). Note that resiliency scales with the amount, diversity of the originality of the mechanisms used.			■				N/A
8.8	MSTG-RESILIENCE-8	The detection mechanisms trigger responses of different types, including delayed and stealthy responses.			■				N/A
8.9	MSTG-RESILIENCE-9	Obfuscation is applied to programmatic defenses, which in turn impede de-obfuscation via dynamic analysis.			■				N/A
8.10	MSTG-RESILIENCE-10	The app implements a 'device binding' functionality using a device fingerprint derived from multiple properties unique to the device.			■				Fail
8.11	MSTG-RESILIENCE-11	All executable files and libraries belonging to the app are either encrypted on the file level and/ or important code and data segments inside the executables are encrypted or packed. Trivial static analysis does not reveal important code or data.			■				Fail
8.12	MSTG-RESILIENCE-12	If the goal of obfuscation is to protect sensitive computations, an obfuscation scheme is used that is both appropriate for the application task and robust against manual and automated de-obfuscation methods, considering currently published research. The effectiveness of the obfuscation scheme must be verified through manual testing. Note that hardware-based			■				N/A
8.13	MSTG-RESILIENCE-13	As a defense in depth, next to having solid hardening of the communicating parties, application level payload encryption can be applied to further impede eavesdropping.			■				Pass

Mobile Application Security Verification Standard

OWASP MASTG v1.5.0 (commit: 3b9278f)
OWASP MASVS v1.4.2 (commit: 2a8b582)

About the Project

The OWASP Mobile Application Security (MAS) flagship project led by Carlos Holguera and Sven Schleier defines the industry standard for mobile application security.

https://owasp.org/mas/

The OWASP MASVS (Mobile Application Security Verification Standard) is a standard that establishes the security requirements for mobile app security.

https://mas.owasp.org/MASTG/0x01-Foreword/

The OWASP MASTG (Mobile Application Security Testing Guide) is a comprehensive manual for mobile app security testing and reverse engineering. It describes technical processes for verifying the controls listed in the MASVS.

https://mas.owasp.org/MASVS/0x01-Foreword/

Feedback

If you have any comments or suggestions, please post them on our GitHub Discussions.

https://github.com/OWASP/owasp-mastg/discussions/categories/ideas

Licence

https://github.com/OWASP/owasp-mastg/blob/master/License.md