

Web and MOBILE application security TESTING ORDER FORM

Document Date: 04.04.2021

Document Protection Level: CONFIDENTIAL

Service Provider: SecTeam OÜ (reg.no 12434689)

Contact Person: Rünno Reinu, info@secteam.eu, +372 520 5558

Web page: <https://www.secteam.eu>

Background

The following form is intended for any organisation ordering web or mobile application security testing. This set of information clarifies the full scope of the testing project and is an essential input for penetration testing service providers to prepare the tender.

Completing this form does not oblige for any contractual agreements, but if needed, could be an addendum of a contract.

Order Form

|  |  |  |
| --- | --- | --- |
| **Specification** | **Comments / Options** | **Value** (filled in by the Customer) |
| Company Name | Name of the customer organisation |       |
| Web or Mobile Application Name | Name of the target system |       |
| Application Version | Version of the target system |       |
| URL | Full URL of the target system |       |
| IP | Corresponding IP-address of the target system |       |
| Target Category | Check all the target categoriesNote, in practise API testing is done similarly to web application testing (ASVS). | [x]  Web application[ ]  API[ ]  Mobile app – Android[ ]  Mobile app – iOS |
| Standard Name | For web app: OWASP ASVSFor mobile app: OWASP MASVS |  |
| Standard Version | Latest ASVS version: v.4.0.2Latest MASVS version: v.1.2 |  |
| Testing Level | ASVS levels:* Level 1 (Opportunistic)
* Level 2 (Standard)
* Level 3 (Advanced)

MASVS levels:* MASVS-L1 (Standard Security)
* MASVS-L1+R
* MASVS-L2 (Defense-in-Depth)
* MASVS-L2+R
 |  |
| Scope of the (M)ASVS Standard Requirements | Options:* All requirements
* All requirements, except V1 requirements related to software development life cycle process and documentation
* All requirements, except (please specify)
 | (Specify here if needed) |
| Scope of the Target | Specify which parts of the target system needs to be verified | (Specify here if needed) |
| Access to Resources | Check all the resources pentester will have access to | [ ]  Source code[ ]  API documentations, including endpoints descriptions, query samples and permissions matrix (if API is present)[ ]  Technical documentation, for example architectural documents, administrator or user manuals[ ]  Application administrator account[ ]  API access token[ ]  Application log files[ ]  Web server log files [ ]  Application and server confirmation files[ ]  Backend access to database |
| Testing Method | * Access to all resources: White Box (recommended)
* Partial access to resources: Grey Box
* No special access to resources: Black Box (not recommended)

NB! Black Box method is available only on the testing Level 1. |  |
| Testing Environment | Choose the environment where the security testing will take placeNB! Testing in production takes place solely on the responsibility of the Customer | (Specify here if needed) |
| Complexity or Volume of the Testing Project | Enter any information describing the target system complexity and indicating the potential amount of work hours, for example:* Available access to a demo environment
* High-level design document
* Main components of the system
* Programming language
* Quantity of the source code or functionalities
* User manual
 |       |
| Reporting Language | Final report language options:* English
* Estonian
 |  |

Additional Explanations

*Feel free to skip all the text below if you are a frequent and experienced customer.*

## Time Indications

Every service provider is evaluating their testing hours based on the order form data and internal capacity. The very rough average for a testing time frame may be:

* 2 - 2,5 weeks (80 – 100 hours) for Level 1
* 3 - 4 weeks (120 – 160 hours) for Level 2

Please note, any contractual bureaucracy and organising the initial access to the testing environment consumes additional time and the effective dates of the contract itself should be at least twice the time mentioned above.

## Defining the Targets

* A mobile app on the iOS and Android devices are two different testing targets, even though they may share some common code base.
* Mobile app and its API may be defined as one unified solution but are typically tested as separate targets - mobile app based on OWASP MASVS and the API based on OWASP ASVS (however, not all ASVS requirements apply). Generally, a similar principle applies to web applications and their API-s.

## Know the Standard

We strongly recommend getting familiar with the OWASP ASVS (Application Security Verification Standard) and MASVS (Mobile Application Security Verification Standard) standards, at least the first methodology sections of them:

* Link to OWASP ASVS homepage: <https://owasp.org/www-project-application-security-verification-standard/>
* Link to the ASVS standard v.4.0.2: <https://github.com/OWASP/ASVS/blob/master/4.0/OWASP%20Application%20Security%20Verification%20Standard%204.0.2-en.pdf>
* Link to OWASP MASVS homepage: <https://owasp.org/www-project-mobile-security-testing-guide/>
* Link to the MASVS standard v.1.2: <https://github.com/OWASP/owasp-masvs/releases/download/v1.2/OWASP_MASVS-v1.2-en.pdf>

## Standard Version and Scope Limitations

While it is reasonable to carry out testing based on the latest standard version, there may be individual cases where the customer is asking for an older version. It can be delivered, but some newest technology nuances may escape the verification checks.

Newer versions are approaching to the target not only as a stand-alone application but are also focusing on the designing and development lifecycle phases (V1 requirements group). This verification is typically done with interviews, system documentation and software development process analysis. From the customer perspective, they sometimes ask to skip the “soft” verification part and examine only the technical solution itself.

## Testing Levels

Standards explain the testing levels more in-depth. The correct path for deciding the testing level incorporates risk assessment and threat modelling. However, here are some hints.

### ASVS

Level 1 (Opportunistic) – the very minimum for all applications and suitable only for systems processing low-value non-sensitive data. L1 is the only level allowing black box testing and is focusing on the “low-hanging fruits”.

Level 2 (Standard) – is suitable for most applications processing sensitive business data, personal data, [a special category of personal data](https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/lawful-basis-for-processing/special-category-data/), covering industries like gaming, financial and health-care and so on.

Level 3 (Advanced) – is the highest level of assurance for industries like health and safety, critical infrastructure, military. The compromise of the application would lead to a significant threat to the whole organisation and beyond.

### MASVS

MASVS-L1: Standard Security – covers security best practices. Suitable for all mobile apps as a baseline level. Also, for those processing some level of sensitive data, including personal data at some degree. Stick with it, if there is no ground for a higher level.

MASVS-L2: Defense-in-Depth – for apps with high risks and processing more sensitive data and compromise may outcome high negative impact. For example, apps processing personal data usable for identity theft, [a special category of personal data](https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/lawful-basis-for-processing/special-category-data/), payment data.

MASVS-R: Resiliency Against Reverse Engineering and Tampering – this level is not a separate one but could be added to levels 1 and 2 to verify additional controls against client-side malicious tampering and modification of the app. R-level measures make the reverse-engineering and cracking the app more expensive and sophisticated. Please do not forget there is no 100% secure solution. Corresponding testing levels would be the following:

MASVS-L1+R – for example, apps containing the organisation’s intellectual property, and there is a clear need to protect it, gaming apps to avoid cheating and unfair competition.

MASVS-L2+R – the highest level of MASVS standard. Suitable for apps like online banking, authentication systems.

## Testing Method

Should you choose the white box, grey box or black box testing? Short answer – white box. The reasoning behind it is simple. The attack-and-defend “game” is completely unfair. A testing project has a limited time frame while the customer wants to have the maximum assurance and find all possible weaknesses. Opposite to attackers, who usually have no time limits and need to identify only single entry-point. The maximum testing outcome is plausible to achieve only with the full transparency and access to any supporting resources.

Anyhow, the method could be:

* If testers have access to all relevant resources, then method is White Box
* If testers have partial access to resources, then method is Grey Box
* If testers have no special access to resources, the method is Black Box

Depending on the testing target, the relevant resources may be:

* Source code
* API documentations, including endpoints descriptions and query samples (if API is present)
* Technical documentation, for example architectural documents, administrator or user manuals
* Application administrator account
* API access token
* Application log files
* Web server log files
* Application and server configuration files
* Backend access to database

Worth to mention about automated scanners, that only half of the ASVS requirements can be verified with fully automated tools and scanner, the second half requires human assistance as all the applications, their setup or business logic is somehow unique.

## Pentesting Team

Every customer is free to set the professional requirements to the service provider. If no starting point is present, our friendly suggestion is to start with the following simple roles and competences:

* Project lead – a person with general knowledge about information security management and practices. Usually, a sufficient quality sign would be certifications like CISM, CISSP, CISA, CEH or similar
* Penetration tester – a person with particular application security testing skills. Suitable quality labels are certifications like GWAPT, CWAPT, OSEE or similar.

## Reporting

The final outcome of a testing project is generally delivered as a comprehensive testing report containing:

* High-level executive summary
* Findings with detailed technical descriptions
* Risk ratings of the findings
* Recommendations to fix the issues
* Overview about compliance with the standard requirements

## Preparing for the Testing Project

Being ready for the testing project involves several activities the customer should organise internally:

* Timing – while security verification check could be done during different phases, the most reasonable and used timing is right AFTER the development is ready, smoke and acceptance tests delivered, BUT BEFORE installing the application to the production environment. This way, the project budget is utilised most effectively, and the same functionalities and components don’t need rechecks several times. Please leave extra time in the calendar to fix the found issues and maybe also to reverify those.
* Prepare the testing environment – please do not test in the production environment. If you do, it is taking place solely on the customer’s responsibility! Pentester’s intention is not to make the service unavailable, but it cannot be foreseen how the test target behaves to modified queries. Also, testers shouldn’t see and don’t want to see the personal data of users. Any other environment will do, and it should be *as similar as possible to the production*. Avoid any production data but add some sample data covering all functionalities. If the environment is somehow critical for you, please make a fresh backup.
* Access limitation – to avoid unwanted visitors to the testing environment, feel free to add extra access controls, like IP whitelisting or VPN. Do not implement too aggressive access limits, which are not there in the production, for example, Citrix solutions restricting the testing in the test environment but are not used in the production.
* Access roles – testers should have at least one application administrator account, which can create additional users in different roles and permissions. This eases the testing of various access escalation attacks.
* Guidance and document package – finally, prepare clear guidance on how to access the testing environment, user accounts, technical documentation, logs, configuration files and other relevant data. It is wise to encrypt this data before sending out to the pentesters.

Enjoy the ride!

or ask more: info@secteam.eu