Requirements of the AssureMOSS Scheme

Architecture

ID	ARCH.1
Requirement	All 3rd party libraries are identified and known to be needed.
Scope	Component
Rationale	The ToE does not include more libraries and dependencies than it is necessary for providing the functionalities and goals it aims to achieve. This requirement ensures that the attack surface is not broader than necessary.
Evidence	List of libraries and a justification for their usage (e.g., a model of functionalities and mapped used libraries).
Related indicators	Number of third-party libraries
AssureMOSS support	WP3 static analysis tools can provide a list of the libraries
References to standards	MSTG-ARCH-1

ID	ARCH.2
Requirement	Security controls are enforced on remote endpoints.
Scope	Global
Rationale	In case of the ToE has client- and server-side components, it is not sufficient to enforce security controls on the client side (e.g., password policy, input sanitization), but it is necessary to implement these features on the server side. This requirement also ensures that user input is sanitized before reaching the inner components of the ToE to provide protection against injection attacks.
Evidence	Source code, fuzzing test results
AssureMOSS support	WP2 tools help to create system models (architecture, data flow) that can be continuously derived from the source code. These then can provide the list of endpoints, which helps the human assessor in checking the security controls.
References to standards	MSTG-ARCH-2

ID	ARCH.3
Requirement	Cryptographically sign and verify automatic updates.
Scope	Global
Rationale	In case the ToE supports automatic update mechanism, cryptographic signage and verification of the update files is required. Recommended algorithms according to NIST [34]: DSA, RSA and ECDSA.
Evidence	Source code, firmware update file

Data Storage and Privacy

ID	DATA.1
Requirement	No sensitive data should be stored outside of the secure container.
Scope	Global
Rationale	Security sensitive data (e.g., keys, certificates) and privacy sensitive data (e.g., PII) should only be stored in a container that has restricted access and encrypted storage. Access from other modules is configured to ensure least privileges and minimal access.
Evidence	Sensitive data tagged, database schemes, database configuration (encryption, access roles), data flow diagram
References to standards	MSTG-STORAGE-1 MSTG-STORAGE-2 CWE-922 – Insecure Storage of Sensitive Information CWE-522 – Insufficiently Protected Credentials

	CWE-359 – “Exposure of Private Information (‘Privacy Violation’)” CWE-312 – Cleartext Storage of Sensitive Information CWE-311 – Missing Encryption of Sensitive Data CWE-200 – Information Exposure

ID	DATA.2
Requirement	No sensitive data is written to logs.
Scope	Component
Rationale	Security sensitive data (e.g., keys, certificates) and privacy sensitive data (e.g., PII) should not be written to log files (even in case of crashes). Access to the system or application logs could provide valuable information to the attacker.
Evidence	Log samples, source code parts of logging implementation, data flow diagram
AssureMOSS support	WP4 monitoring of runtime execution can provide log samples.
References to standards	MSTG-STORAGE-3 CWE-359 – “Exposure of Private Information (‘Privacy Violation’)” CWE-200 – Information Exposure CWE-532 – Information Exposure Through Log Files CWE-534 – Information Exposure Through Debug Log Files CWE-117 – Improper Output Neutralization for Logs

ID	DATA.3
Requirement	No sensitive data is shared with third parties unless it is a necessary part of the architecture.
Scope	Global
Rationale	Security sensitive data (e.g., keys, certificates) and privacy sensitive data (e.g., PII) should not be leaving the system to third party servers. The ToE should be responsible for the data it processes and manage the secure storage, and communication of them. If it is necessary to share such data (e.g., audit logging), confidentiality and integrity of the sensitive data should be guaranteed during transit.
Evidence	Network log samples, source code parts of handling sensitive data in transit, list of endpoints, data flow diagram.
AssureMOSS support	WP4 monitoring of runtime execution can provide network log samples. In its models (architecture, data flow), WP2 can provide pointers to monitoring/tracing capabilities, places where inspections have revealed sensitive data in the source code (e.g., plaintext credentials), and a list of used datastores and their connections to other components (such as services).
References to standards	MSTG-STORAGE-4 CWE-359 – “Exposure of Private Information (‘Privacy Violation’)” CWE-200 – Information Exposure

ID	DATA.4
Requirement	The software does not hold sensitive data in memory longer than necessary, and memory is cleared explicitly after use.
Scope	Component
Rationale	If the attacker is able to access the machine which runs the microservice, the application should ensure that secrets are handled in a way, that makes for the attacker eavesdropping significantly harder even in case of monitoring the memory.
Evidence	Source code parts related to handling sensitive data, memory dumps
AssureMOSS support	WP4 monitoring of runtime execution can provide memory dumps.
References to standards	MSTG-STORAGE-10 CWE-359 - "Exposure of Private Information ('Privacy Violation')" CWE-200 - Information Exposure CWE-524 - Information Exposure Through Caching

Cryptography

ID	CRYP.1
Requirement	The software does not rely on symmetric cryptography with hardcoded keys as a sole method of encryption also does not reuse keys for multiple purposes.
Scope	Component

Rationale	The use of symmetric cryptography should be avoided in case the ToE uses hardcoded keys. Also, the ToE does not reuse keys for different purposes (e.g., encryption and signature).
Evidence	Cryptographic algorithms used by the ToE, Source code parts using cryptographic libraries, referring to keys.
References to standards	MSTG-CRYPTO-1

ID	**CRYP.2**
Requirement	The software uses cryptographic primitives that are appropriate for the use-case and currently advised.
Scope	Component
Rationale	Cryptographic algorithms are chosen for their usage based on best practices from NIST [34] for a particular challenge. E.g., usage of CRC for integrity protection is not valid. The software does not use cryptographic protocols or algorithms that are widely considered depreciated for security purposes. The use of own implementation of cryptographic libraries is highly discouraged. The following algorithms are recommended: Confidentiality algorithms: AES-GCM-256 or ChaCha20-Poly1305 Integrity algorithms: SHA-256, SHA-384, SHA-512, Blake2 Digital signature algorithms: RSA (3072 bits and higher), ECDSA based on NIST P-384 Key establishment algorithms: RSA (3072 bits and higher), DH (3072 bits or higher), ECDH based on NIST P-384
Evidence	Cryptographic algorithms used by the ToE, Source code parts using cryptographic libraries, list of 3rd party libraries
References to standards	MSTG-CRYPTO-3

ID	**CRYP.3**
Requirement	The software uses cryptographic primitives that are configured with parameters that adhere to industry best practices.
Scope	Component
Rationale	The configuration of cryptographic algorithms is up to date according to NIST [34] . E.g., usage of 3072-bit RSA, 256-bit AES-GCM.
Related indicators	Security best practices (audited libraries)
Evidence	Cryptographic algorithms used by the ToE, Source code parts using cryptographic libraries, configuration files
AssureMOSS support	WP4 can analyze configuration files.
References to standards	MSTG-CRYPTO-3 WSTG-CRYP-04

ID	**CRYP.4**
Requirement	Random values are generated using a sufficiently secure random number generator.
Scope	Component
Rationale	In case a secure random number is required for a particular functionality, the usage of general random number generators is not sufficient to prevent attacks. CPRNGs provided by the operating system, or a certified hardware component should be used for e.g., key generation.
Evidence	Source code parts using random libraries.
AssureMOSS support	QualityGate [35]
References to standards	MSTG-CRYPTO-6 CWE-337 - Predictable Seed in PRNG CWE-338 - Use of Cryptographically Weak Pseudo Random Number Generator (PRNG)

Authentication and Session Management

ID	**AUTH.1**

Requirement	If the software provides users access a remote service, some form of authentication, such as username/password authentication, is performed at the remote endpoint.
Scope	Global
Rationale	The services provided by the software should be enabled for the client only after authentication, which should be done with at least the combination of username+password or by using additional factors (e.g., time-sensitive authenticator code) or by using PKI-based certificates.
Evidence	Network traffic capture, source code parts of authentication. Architecture Models, Data Flow Diagrams.
AssureMOSS support	WP4 provides network traffic capture. WP2 tools help to create system models (architecture, data flow) that can be continuously derived from the source code. These then can provide the authentication and authorization features found on the remote endpoints in an inspection.
References to standards	MSTG-AUTH-1 WSTG-ATHN-10 CWE-287 - Improper Authentication

ID	**AUTH.2**
Requirement	If stateful session management is used, the remote endpoint uses randomly generated session identifiers to authenticate client requests without sending the user's credentials.
Scope	Global
Rationale	The following criteria must be met in case of using stateful session management: Session IDs are randomly generated on the server side. The IDs can't be guessed easily (use proper length and entropy). Session IDs are always exchanged over secure connections (e.g. HTTPS). The server verifies the session whenever a user tries to access privileged application elements, (a session ID must be valid and must correspond to the proper authorization level). Authentication shouldn't be implemented from scratch but built on top of proven frameworks.
Evidence	Source code parts of session management
References to standards	MSTG-AUTH-2 CWE-287 - Improper Authentication MS-SS-10

ID	**AUTH.3**
Requirement	If stateless token-based authentication is used, the server provides a token that has been signed using a secure algorithm.
Scope	Global
Rationale	Use standard solutions, e.g., JWT tokens for stateless session management with the following points met: the token expiry times should be as short as possible HMAC is checked for all incoming requests containing a token The private signing key or HMAC secret key should remain on the server and should never be shared with the client No sensitive data, such as personal identifiable information, is embedded in the JWT Replay attacks are addressed with the jti (JWT ID) claim Tokens are stored securely
Evidence	Source code parts of session management
References to standards	MSTG-AUTH-3 CWE-287 - Improper Authentication MS-SS-1

ID	**AUTH.4**
Requirement	The remote endpoint terminates the existing session when the user logs out.
Scope	Global
Rationale	In case of stateful session management, the server removes the active session from the database in case the user logs out of the service. Hereafter, the user has to reauthenticate to be able to access the services.
Evidence	Source code parts of session management, network logs, database snapshots

References to standards	MSTG-AUTH-4 WSTG-ATHN-03 CWE-287 - Improper Authentication

ID	AUTH.5
Requirement	A password policy exists and is enforced at the remote endpoint.
Scope	Global
Rationale	If authentication is implemented with username+password, a password policy has to be enforced on the server side based on best practices: length is at least 8 characters, should contain numbers, small and capital letters and special characters.
Evidence	Source code of password policy, logs of fuzzing tests
References to standards	MSTG-AUTH-5 WSTG-ATHN-07 CWE-287 - Improper Authentication

ID	AUTH.6
Requirement	The remote endpoint implements a mechanism to protect against the submission of credentials an excessive number of times.
Scope	Global
Rationale	There should be a throttling mechanism implemented on server side to limit the number of requests from the same direction and using the same user's session tokens. After the limit is reached, the service should drop exceeding packets and/or ban the user for a limited time to mitigate flooding/DoS attacks.
Evidence	Source code, dynamic tests (scripts result)
References to standards	MSTG-AUTH-6 CWE-287 - Improper Authentication MS-SS-8

ID	AUTH.7
Requirement	Sessions are invalidated by access token expiration.
Scope	Global
Rationale	After the token expiration date is passed, the service shall not accept the token for performing any operation on the behalf of the associated user before reauthentication.
Evidence	Network traffic capture, source code
References to standards	MSTG-AUTH-7 WSTG-SESS-06 WSTG-SESS-07 CWE-287 - Improper Authentication

ID	AUTH.8
Requirement	Sessions are invalidated at the remote endpoint after a predefined period of inactivity.
Scope	Global
Rationale	Sessions should be expiring automatically in case the session is not maintained by the user for a predefined period of time. After the inactivity period is reached (the amount should be determined by the developer according to what is suitable for the use-case), the user should be reauthenticated to get a new valid token.
Evidence	Network traffic capture, source code
References to standards	MSTG-AUTH-7 CWE-287 - Improper Authentication

Communication

ID	COMM.1
Requirement	Outgoing data is encrypted on the network using TLS.
Scope	Global
Rationale	The service has to communicate with 3rd party endpoints and users on an encrypted channel. The TLS settings should be in line with best practices: at least TLS 1.2 is used

	with NIST-approved [34] cipher suites: AES GCM, AES CCM, Camellia CBC, ARIA GCM, RC2 CBC, CHaCha20-Poly1305 and integrity protection.
Evidence	Network traffic capture, source code
AssureMOSS support	WP2 tools help to create system models (architecture, data flow) that can be continuously derived from the source code. These then can provide a list of client, endpoints and channels used between them, as well as properties such as whether a secure (encrypted) channel is used or not.
References to standards	MSTG-NETWORK-1 MSTG-NETWORK-2 WSTG-CONF-01 WSTG-CRYP-01 CWE-326 - Inadequate Encryption Strength MS-SS-4

ID	**COMM.2**
Requirement	The secure channel is used consistently throughout the software.
Scope	Component
Rationale	This requirement ensures that in case an information flow was sent over a secure channel, then the same information should not be propagated via an insecure channel among the remote components of the microservice or third parties. Local communication in a closed subnet (security group) is permitted to be in plain.
Evidence	Network traffic logs, architecture model, data flow diagram, list of endpoints
AssureMOSS support	WP2 tools help to create system models (architecture, data flow) that can be continuously derived from the source code. These then can provide a list of endpoints and channels used, as well as properties such as whether a secure (encrypted) channel is used or not.
References to standards	MSTG-NETWORK-1 WSTG-CONF-01 CWE-319 - Cleartext Transmission of Sensitive Information

ID	**COMM.3**
Requirement	The software verifies the X.509 certificate of the remote endpoint when the secure channel is established. Only certificates signed by a trusted CA are accepted.
Scope	Global
Rationale	Testing with expired and invalid certificates should be rejected by the endpoints on both client and server sides.
Evidence	Network traffic capture, source code
AssureMOSS support	Tagged source code
References to standards	MSTG-NETWORK-3 CWE-295 - Improper Certificate Validation

Code Quality and Build Settings

ID	**QUAL.1**
Requirement	All inputs received from the endpoints are validated and if necessary sanitized.
Scope	Component
Rationale	Input sanitization based on policies relevant to the content should be implemented to avoid e.g., injection, code execution and denial of service attacks.
Evidence	Source code, network traffic logs, fuzzing tests
Related indicators	Audited libraries for input sanitization used
AssureMOSS support	Tagged source code
References to standards	MSTG-PLATFORM-2 WSTG-INPV-01...19

ID	**QUAL.2**
Requirement	Object deserialization, if any, is implemented using safe serialization APIs.
Scope	Component

Rationale	Objects coming from other services and user input should be deserialized with safe serialization libraries.
Evidence	List of serialization libraries used, source code
Related indicators	Audited libraries for serialization used
AssureMOSS support	Tagged source code
References to standards	MSTG-PLATFORM-8

ID	**QUAL.3**
Requirement	All third-party components used by the software, such as libraries and frameworks, are identified, and checked for known vulnerabilities.
Scope	Component
Rationale	The third-party components should be checked for known vulnerabilities in relevant CVE databases and advisories.
Evidence	List of used libraries and frameworks with versions extracted.
Related indicators	Number of vulnerable components, CVE scores
AssureMOSS support	Output of WP3 and WP4 tools about used libraries, and known vulnerabilities
References to standards	MSTG-CODE-5

ID	**QUAL.4**
Requirement	The software catches and handles possible exceptions.
Scope	Component
Rationale	Exception handling is present (if the programming language provides it) and implemented in a way that the software is able to gracefully handle possible exceptions
Evidence	Fuzzing tests, source code, crash logs
Related indicators	Test coverage
AssureMOSS support	WP4 runtime monitoring
References to standards	MSTG-CODE-6 WSTG-ERRH-01

ID	**QUAL.5**
Requirement	In unmanaged code, memory is allocated, freed and used securely.
Scope	Component
Rationale	In case of programming languages and code parts, where memory management is the programmer's responsibility, the developer shall allocate, free and use memory in secure ways to exclude the possibility of memory leaking, opportunity for DoS attacks and buffer overflows caused by malicious user input.
Evidence	Memory traces from runtime monitoring, source code
Related indicators	Test coverage
AssureMOSS support	WP4 runtime monitoring
References to standards	MSTG-CODE-8

ID	**QUAL.6**
Requirement	An integrity check shall be performed and automatically monitored to detect image manipulations and reported to the monitoring services at start-up and runtime of virtual machine or container images.
Scope	Global
Rationale	Continuous validation of the running images should be provided by the orchestration service in place.
Evidence	Source code, configuration
Related indicators	Test cases
AssureMOSS support	WP4 monitoring tools
References to standards	EUCS-PSS-04.3

Environment

ID	ENVI.1
Requirement	The developer shall test proposed changes before deployment.
Scope	Global, Component
Rationale	Unit and integration tests should ensure that the cornerstone parts of the services are implemented correctly.
Evidence	Source code, configuration
Related indicators	Number of automated tests

ID	ENVI.2
Requirement	The developer shall harden orchestrator components under its responsibility according to accepted industry standards and best practices
Scope	Component
Rationale	Orchestrator configuration (e.g., Kubernetes) should be checked to for CIS and OWASP best practices.
Evidence	CIS benchmark results
Related indicators	CIS benchmark results
AssureMOSS support	WP4 runs CIS benchmark for Kubernetes configuration.
References to standards	EUCS-OPS-21.1

ID	ENVI.3
Requirement	The developer shall harden container configurations under its responsibility according to accepted industry standards and best practices
Scope	Component
Rationale	Container configuration (e.g., Docker) should be checked to for CIS and OWASP best practices.
Evidence	CIS benchmark results
Related indicators	CIS benchmark results
AssureMOSS support	WP4 runs CIS benchmark for Docker configuration.
References to standards	EUCS-OPS-21.1

ID	ENVI.4
Requirement	The developer shall harden guest OS components under its responsibility according to accepted industry standards and best practices
Scope	Component
Rationale	The guest OS of the service should be selected and configured to provide the minimal required environment necessary for the service and be configured securely based on CIS benchmark tests to host the service.
Evidence	CIS benchmark results
Related indicators	CIS benchmark results
AssureMOSS support	WP4 runs CIS benchmark for guest OS
References to standards	EUCS-OPS-21.1

ID	ENVI.5
Requirement	Resilience level should be approved by human evaluator
Scope	Global
Rationale	The Resilience level form the Resilience Tool should be above a certain limit value of 1 to successfully certify the ToE.
Evidence	Resilience level
Related indicators	Resilience level
AssureMOSS support	WP5 Resilience Tool calculates and monitors the Resilience level of the ToE.

ID	ENVI.6
Requirement	Network policies are defined to regulate inter-container communication based on minimal access.
Scope	Global

Rationale	No open ports for undefined services should be present on the OS-es and published by the containers. Connections should be restricted among the microservice components according to usage, the direction and source of the communication should also be taken into account to determine permissions.
Evidence	Network policy descriptions in configuration files.
AssureMOSS support	WP4 checks network policy configuration