ETSI TS 1DD DDD Vm.t.e (yyyy-mm)

Testing (MTS);

Security Testing;

IoT Security Functional Modules

Part #: Part element of title;

Sub-part #: Sub-part element of title

Release #

<

**TECHNICAL SPECIFICATION**

Reference

DTS/MTS-TST10SecTest\_IoTmodule

Keywords

security, functional testing, IoT

***ETSI***

650 Route des Lucioles

F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B

Association à but non lucratif enregistrée à la

Sous-préfecture de Grasse (06) N° w061004871

***Important notice***

The present document can be downloaded from:  
<http://www.etsi.org/standards-search>

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the prevailing version of an ETSI deliverable is the one made publicly available in PDF format at [www.etsi.org/deliver](http://www.etsi.org/deliver).

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at <https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx>

If you find errors in the present document, please send your comment to one of the following services:  
<https://portal.etsi.org/People/CommiteeSupportStaff.aspx>

If you find a security vulnerability in the present document, please report it through our

Coordinated Vulnerability Disclosure Program:

<https://www.etsi.org/standards/coordinated-vulnerability-disclosure>

***Notice of disclaimer & limitation of liability***

The information provided in the present deliverable is directed solely to professionals who have the appropriate degree of experience to understand and interpret its content in accordance with generally accepted engineering or

other professional standard and applicable regulations.

No recommendation as to products and services or vendors is made or should be implied.

No representation or warranty is made that this deliverable is technically accurate or sufficient or conforms to any law and/or governmental rule and/or regulation and further, no representation or warranty is made of merchantability or fitness for any particular purpose or against infringement of intellectual property rights.

In no event shall ETSI be held liable for loss of profits or any other incidental or consequential damages.

Any software contained in this deliverable is provided "AS IS" with no warranties, express or implied, including but not limited to, the warranties of merchantability, fitness for a particular purpose and non-infringement of intellectual property rights and ETSI shall not be held liable in any event for any damages whatsoever (including, without limitation, damages for loss of profits, business interruption, loss of information, or any other pecuniary loss) arising out of or related to the use of or inability to use the software.

***Copyright Notification***

No part may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm except as authorized by written permission of ETSI.  
The content of the PDF version shall not be modified without the written authorization of ETSI.  
The copyright and the foregoing restriction extend to reproduction in all media.

© ETSI yyyy.

All rights reserved.

Contents

Intellectual Property Rights 4

Foreword 4

Modal verbs terminology 4

Executive summary 4

Introduction 4

1 Scope 5

2 References 5

2.1 Normative references 5

2.2 Informative references 5

3 Definition of terms, symbols and abbreviations 5

3.1 Terms 5

3.2 Symbols 6

3.3 Abbreviations 6

4 Specification of the IoT Modules 6

4.1 ISO/IEC 30141 Reference Architecture 6

4.2 IoTAC System Architecture 9

4.2.1 Front End Access Management 9

4.2.2 IoT Security Gateway 10

4.2.3 Run-time monitoring system 11

4.2.4 Attack Detection 12

4.2.5 Honeypots 13

4.2.6 Multi-module Security Assessment 13

5 Relevant Security Test Methods 15

5.1 Functional and Security Testing 15

5.1.1 Functional Testing 15

5.1.2 Static Application Security Testing (SAST) 16

5.1.3 Dynamic Application Security Testing (DAST) 17

5.2 TDL-TO as a specification technique 18

5.3 Methodology for translation 19

6 Detailed List of Test Purposes 20

Annex A (normative or informative): Title of annex 21

Annex (informative): Bibliography 22

Annex (informative): Change History 23

History 24

# Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server ([https://ipr.etsi.org](https://ipr.etsi.org/)).

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

**DECT™**, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™**and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

# Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Method for Testing and Specification (MTS).

# Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the [ETSI Drafting Rules](https://portal.etsi.org/Services/editHelp!/Howtostart/ETSIDraftingRules.aspx) (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

# Executive summary

The goal of the present document is to assemble security related functional modules within an IoT architecture, that support Security by Design and trustworthiness to retrieve relevant security testing methods and specific detailed test purposes using TDL-TO for generic IoT architectures applicable in multiple industrial domains.

# Introduction

This document presents an introduction and guide to security testing for generic IoT architectures that can be used in a variety of industries. It belongs to multipart technical specification addressing the most relevant testing aspects of the IoT architecture:

* Functional Testing
* Static Application Security Testing (SAST), and
* Dynamic Application Security Testing (DAST).

While the functional testing part presents a comprehensive set of test purposes, the SAST and DAST sections focus on evaluating relevant testing techniques and providing examples that are specific to IoT.

# 1 Scope

This document provides general security considerations and guidelines for generic IoT architectures. To provide an outline of possible implementations, this document presents example Test Purposes (TPs).

# 2 References

## 2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non‑specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at <https://docbox.etsi.org/Reference>.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

[1] ISO/IEC 30141:2018 Internet of Things (IoT) – Reference Architecture

## 2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non‑specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1]ISO/IEC 30141:2018 Internet of Things (IoT) – Reference Architecture

NOTE: Available at <https://www.iso.org/standard/65695.html>

[i.2]ETSI TR 101 583 v1.1.1 (2015-03) MTS; Security Testing; Basic Terminology

# 3 Definition of terms, symbols and abbreviations

## 3.1 Terms

For the purposes of the present document, the terms apply:

**black-box testing:** testing activity conducted without knowledge of the internal structure of the system under test

**white-box testing:** testing based on an analysis of the internal structure of the component or system under test

**system under test:** real open system in which the implementation under test resides

## 3.2 Symbols

Void.

## 3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

DAST Dynamic Application Security Testing

SAST Static Application Security Testing

SDLC System/Software Development Lifecycle

TP Test Purpose

# 4 Specification of the IoT Modules

## 4.1 ISO/IEC 30141 Reference Architecture

ISO/IEC IoT Reference Architecture (RA) provides a high-level reference framework applicable for a wide range of IoT domains [1]. Unfortunately, the standard itself does not provide sufficient detail on security-related aspects. This problem was identified during the writing phase of the IoTAC proposal, where we used the standard ISO/IEC IoT RA Functional View for our initial analysis [i.1]. Hence, we determined that adding additional security components to ISO/IEC 30141 RA would be necessary to address this security-related shortcoming. In Figure 1, newly introduced components within the IoTAC project are shown in purple.

Graphical user interface

Description automatically generated

Figure : Extended ISO/IEC 30141 Reference Model (high-level view)

**The Sensing and Controlling Domain (SCD)** is the central domain of the ISO/IEC 30141 RA, containing most of its innovative functionalities. The IoTAC project introduces the following components:

* **IoTAC Security Gateway** serves as a secure gateway for the Internet of Things in an enterprise network. The gateway protects data by receiving, verifying, and distributing sensor messages, as well as relaying control commands to actuators. In particular, it is expected that the IoTAC Secure Gateway should perform the following functions: (i) receiving, scanning, and distributing messages of sensors and other devices; (ii) registering security events of system and network, (iii) detecting attempts of intrusion within the internal enterprise network; (iv) ensuring cybersecurity of the device itself and providing methods for controlling connected devices, etc.
* **AI-based Attack Detection** uses machine learning to detect malicious activities (or abnormalities) for individual traffic packets or for a bunch of packets. A further advantage is that the machine learning method is trained while the module is running online, eliminating the need to collect and store data offline. Attack Detection component is linked to IoT Security Gateway, and if malicious activity is identified, it sends Threat Notification messages.
* **AI-based Network-wide Attack Assessment (NW-AA)** starts with an individual security assessment of some (or all) of the individual devices in IoT network and attempts to provide a system-wide assessment of the security. Detecting infected IoT devices is done by analysing attack decisions made on an individual device basis. With the NW-AA training, all parameters and weights for the necessary connections are kept up to date. Training with AI-based NW-AA requires access to attack decisions collected from locally deployed attack detector systems and the current parameters.
* **Honeypots.** The purpose of honeypots is to induce potential attackers to focus their attention on specific environments that are separate from the overall system. We hope that fewer attacks will be made against the real system. Also, it will allow the attackers to be monitored, facilitating deeper understanding of their behaviour and more timely adoption of security countermeasures. Honeypots powered by IoTAC's IoT-enabled technologies will feature innovative anomaly detection algorithms. It is planned to develop both lightweight and advanced anomaly detection techniques supporting the early detection of IoT device behaviour changes, identifying potential intrusions, and root causes for specific attacks.
* **FEAM Gateway module** is a part of the overall Front-end Access Control Management. The FEAM Gateway module serves as a link between the protected device/system and the FEAM Management module. It is responsible for controlling access to the Protected system, so logically belongs to the SCD.

**The Resource & Interchange Domain (RAID)** includes all the functions required to access the IoT system resources.

* **Front-end Access Management (FEAM).** The innovative FEAM component is delegated capability-based access control system, that meets the requirements of the Zero Trust concept, as it uses smart cards for sensitive data storage, digital signatures and certificates, multi-factor authentication, provides fine-grained privileged access management and implements the least privilege principle on session level. A novel feature of FEAM is the separation both in time and space the delegation of access privileges from Authentication and Authorization.

**The Operation and Management Domain (OMD).** The OMD contains functional components responsible for the overall management of the IoT system. According to ISO/IEC 30141 RA there are two major functional components, the operational support systems (OSS) and the business support systems (BSS).

* **Run-time Monitoring System (RMS)** can be positioned as specialization of the OSS component. It enables a real time service that collects security-related data from monitored IoT system components or applications and stores them for further processing. Analytics algorithms analyse the collected data to detect abnormal patterns. The system features monitoring probes responsible for the data collection and publishing to the monitoring platform.

**The Application and Service Domain (ASD)** represents the collection of functions implementing application and service logic that realizes specific business functionalities for the service providers in the ASD. The Application Support subgroup provides the execution infrastructure and various kinds of data stores, historical data repositories, etc. During the system analysis phase, we have identified the need to introduce following common IoTAC components:

* **Data Bus** is a communications channel through which all real time data is routed. IoTAC's platform supports publish-subscribe functionality, where users may push their own data or subscribe to receive data tailored to their needs. As part of IoTAC, the Data Bus will be used to exchange real-time data from the different components to each other or to report abnormal behaviour.
* **The Security Configuration (SecCM) Repository** contains data about all IoT System assets and security related data that is being monitored by the IoTAC platform.
* **Observational Repository** is a repository that allows permanent storage of data from the IoTAC platform that is monitored or processed. By using these data, the attack detection component will be able to train itself and come up with a set of security templates that will be used in the future to identify potential security issues on the target IoT system.
* **Attack Detection Repository** hosts both the offline-trained version of AD model for parameter storage as well as the online-trained version to be used for performance evaluation. Performance evaluations should be carried out periodically to ensure that AD with online training achieves high performance when compared to AD with offline training.

## 4.2 IoTAC System Architecture

The IoTAC Systems Level Architecture has been designed and specified based on the ISO/IEC 30141 Reference Model presented in previous section.

Diagram

Description automatically generated

Figure : IoTAC Domain-based Reference Model (detailed view)

### 4.2.1 Front End Access Management

**The Front-end Access Management Module** is a capability-based access control system, where the function of authorizing transactions and authenticating users is delegated to the front-end, the secure element of the user. FEAM module groups several heterogeneous functionalities, that are represented as individual components (i.e., User Secure Application, Management Module, FEAM Gateway Module, and FEAM SDK).

/\*Component diagram specification

Collaborative component diagram:

<https://online.visual-paradigm.com/w/iealczom/diagrams/#diagram:workspace=iealczom&proj=2&id=82>

Diagram

Description automatically generated

Figure : Front End Access Management Component Diagram

Table : User Secure Application Component

|  |  |
| --- | --- |
| Front End Access Management /User Secure Application | |
| Name | Description |
| Functionality | The User secure application authenticates the User, stores User Credentials, authorizes Commands based on the stored Credentials, prepares, and signs the access Tokens. |
| Relevant Use Cases | FEAC\_UC-001 User registration (Priority: High)  FEAC\_UC-002 Download of Credentials (Priority: High)  FEAC\_UC-003 Management of the Admin module (Priority: High)  FEAC\_UC-006 Service management (Priority: High) |
| Functional Requirements | FEAC\_FR-004 Multi factor user authentication (Priority: Must)  FEAC\_FR-007 Use on-card key generation mechanism (Priority: Must)  FEAC\_FR-017 Over the air delivery of new Credentials to User secure application (Priority: Should)  FEAC\_FR-023 Execution of user authentication and transaction authorisation on the User secure application (Priority: Must)  FEAC\_FR-024 Preparation of signed JWT by User secure application for representation of authorized Operations (Priority: Must) |

### 4.2.2 IoT Security Gateway

**Kaspersky IoT Secure Gateway (KISG) 1000** is a hardware gateway with routing features designed to protect IoT networks. KISG 1000 has system monitoring and IoT cyber protection function. It collects data and allows you to manage connected devices using the MQTT protocol over TLS. The gateway runs on the Advantech UTX-3117 hardware platform managed by Kaspersky Security Center. The complex of the two products protects IoT infrastructure at the gateway level, helping to monitor it and manage gateway events from a single center. For more details on KSP Gateway and Kaspersky OS please check. KSP Gateway module groups several heterogeneous functionalities, that are represented as individual components (i.e., Event Storage, Device Detection & Classification (DDC), MQTT Broker, Secure Config Manager (SCM), and Network Attack Protection (NAP)). The functionalities, linked use cases and functional requirements are described in Table x-y.

Diagram

Description automatically generated

Figure : IoT Security Gateway

|  |  |
| --- | --- |
| **IoTAC Security Gateway /Secure Config Management** | |
| **Name** | **Description** |
| **Functionality** | Facilitates data exchange between components of Kaspersky IoT Secure Gateway and serves as the point for receiving information about the security state of other components. Used to store the configuration parameters of all system components. |
| **Relevant Use Cases** | SG\_UC\_2: Checking Device Status (Priority: Medium)  SG\_UC\_3: Receiving security events (Priority: High) |
| **Functional Requirements** | SG\_FR\_1: User interface to configure & control (Priority: Must)  SG\_FR\_7: Syslog sender (Priority: Must) |

### 4.2.3 Run-time monitoring system

**Runtime Monitoring System (RMS)** will provide the specifications and relevant framework implementation to enable a real time service which will collect security related data from monitored IoT system components or applications and store them for further processing. The collected data will be used to drive analytics algorithms that detect patterns of abnormal behaviour.  The system will feature lightweight monitoring probes that will be responsible for the data collection and publishing to the monitoring platform. The RMS will provide appropriate configuration and management mechanism over the monitoring probes as well as appropriate data models and data transformation engines that will enable the discoverability and reusability of the collected data. The probe management will be facilitated by an internal probe registry that will maintain the probe information along with their status and will enable the probe creation, reconfiguration, and discovery. This module includes six core components, as follows: Probe Management & Configuration, Probe Registry, MPPE Registry, Automatic Reconfiguration, Data Routing, and Multipurpose Processing Engine. The functionalities, linked use cases, and functional requirements are described below. The interfaces between RMS and other IoTAC modules and Common Components (i.e., Data Bus, Observational Repository, and SecCM Repository) are shown in ..

Diagram

Description automatically generated with medium confidence

Figure : Run-time Monitoring System

### 4.2.4 Attack Detection

The objective of this module is to detect malicious activities (or anomalies) for individual traffic packets or a bucket of packets in wireless networks while training the core machine learning method during the online operation of the module, which eliminates data collection with offline experiments. The ***RNN-based Attack Detector*** module is designed to detect botnet attacks and has the following three functionalities illustrated in Figure x:

* **Metrics Extraction** component which calculates the predefined metrics for the current traffic packets at each call. In our design, there are three different metrics: (i) the total size of the last 500 transmitted packets; (ii) the average inter-transmission times of the packets over the last 500 packets, and (iii) the total number of packets that are transmitted in a time window with a duration of 100 seconds.
* **AA-Dense Attack Detection** component which decides on how likely the current packet (or bucket) is malicious based on the recent statistics of the network traffic.
* **AA-Dense RNN Training** which trains the RNN model of the attack detector and selects the best threshold based on the past traffic statistics. The AA-Dense RNN Training is a component that will be called for online sequential training of AD for regular time intervals.

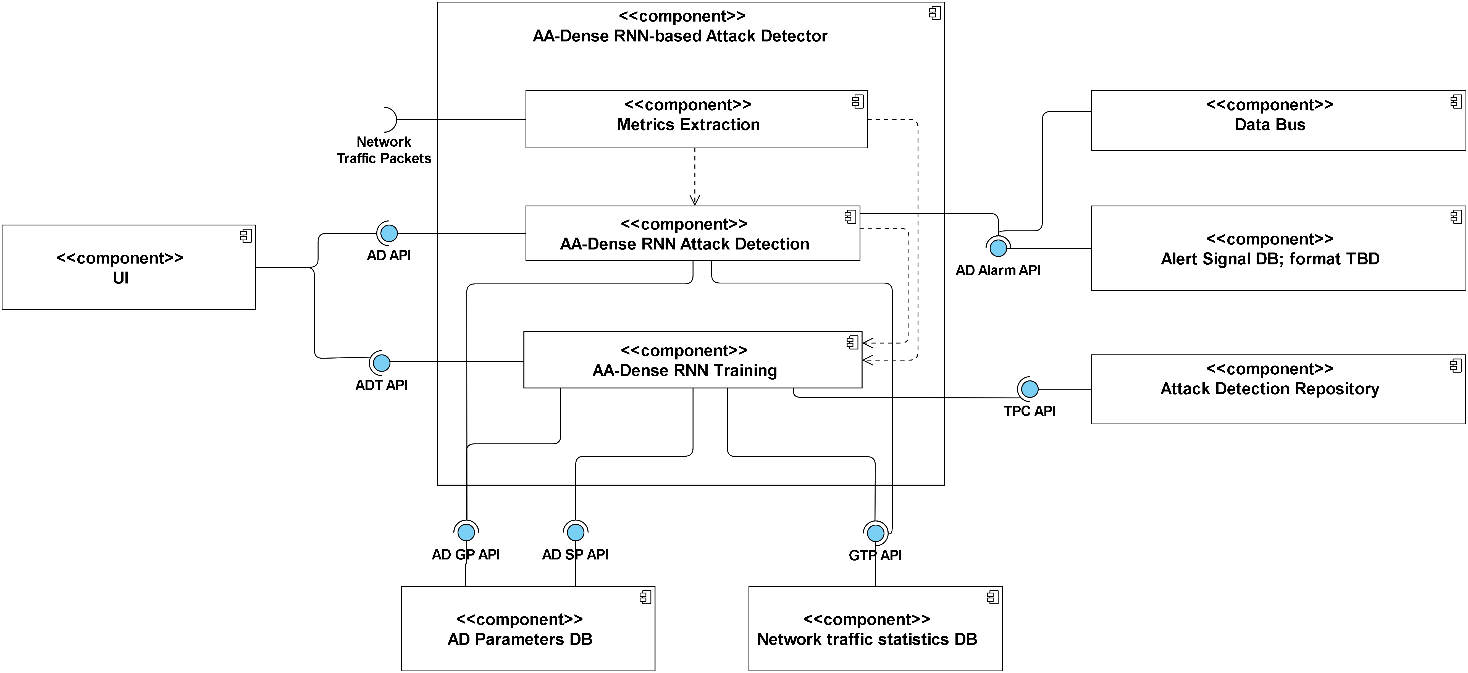


Figure : Attack Detection

### 4.2.5 Honeypots

***Honeypots*** are software designed components to detect novel and emerging attack patterns against specific systems. Hence, a vulnerable version of the protected system will be created, that typically represents an easy way to attack the system. With special preparation, a honeypot serves as a trap that firstly attracts the interest of the attacker and secondly allows the creator to study and find the occurring attacks against the system. Generally, a Honeypot is deployed as a part of the network, although it needs to be properly isolated and protected, but should still contain spoofed information and functionalities that are valuable for the attacker.

We are building a Honeypot that can share attack information via an API and learn from network peers, e.g., other honeypots, AI detectors, etc. It can be configured using a configuration file and emits *Threat information* via an API to inform network peers about ongoing threats. In the first part of the project, we have studied the most popular attack patterns executed against IoT devices. To tackle these issues, we will implement one component for each of the four attack classes, as shown in Figure x. Furthermore, an *Advance Detection* component will implement a shared model that can detect network wide attacks of the covered attack methods.

Diagram

Description automatically generated

Figure : Honeypots

### 4.2.6 Multi-module Security Assessment

The objective of this module, namely ***Multi-module Security Assessment (MSA)*** via Adversarial Random Neural Network (ARNN), is to provide a system-wide assessment of the security of all devices that are connected to each other and hence which must operate as a system. MSA via ARNN starts with the individual security assessment of some (or all) of the individual devices in IoT network and attempts to provide a system-wide assessment of the security. It is comprised of two main modules which include submodules:

* **Network-Wide Attack Assessment (NW-AA)** detects the infected IoT devices by assessing the attack decisions made for individual devices via Attack Detection component. NW-AA module consists of two submodules which are ARNN Infected Device Detection (IDD) and ARNN Training. IDD component, at each call, uses the connection weights and the parameters (which have been computed in training stage) of the algorithm from the NW-AA Parameters DB via NW-AA GP (Get Parameters) API and gets the attack decisions of local detectors as an input from the Alert Signal DB via AD Alarm API. ARNN Training, at each call, first gets the collected attack decisions of local detectors from Alert Signal DB via AD Alarm API and the current parameters from NW-AA Parameters DB via NW-AA GP API; then, updates the parameters in NW-AA Parameters DB via NW-AA SP (Set Parameters) API.
* **System-Wide Vulnerability Assessment (SW-VA)** makes a decision on the vulnerability high-level of a software system, considering the communication between system’s components and also the provided vulnerability predictions by the local vulnerability detectors. SW-VA consists of two submodules “ARNN Vulnerability Assessment (VA)” and “Components Interconnection Analyzer (CIA)”. The CIA gets the analysed software project and identifies the interconnections between its different software components. These interconnections along with the Vulnerability Prediction results per component that exist in “Component Vulnerability Prediction DB” constitute the input to the ARNN Vulnerability Assessment, which produces the overall vulnerability assessment of the analysed project. More specifically, the ARNN VA assesses the vulnerability level of the software components based on their interconnections.

*Diagram, schematic

Description automatically generated*

Figure : Multi-module Security Assessment

# 5 Relevant Security Test Methods

## 5.1 Functional and Security Testing

The approach for testing and evaluation of IoTAC run-time components is focused on detection of functional errors and security vulnerabilities. The following three phases are defined:

* **Functional (Security) Testing** – to verify the functionality of a component according to the functional requirements of T2.2
* **Static Application Security Testing (SAST)** is a “white box testing approach” for proactive prevention, early detection, and identification of security issues (e.g., grep for dangerous patterns)
* **Dynamic Application Security Testing (DAST)** is a “black box testing” for simulation of live attacks.

The overall approach will be performed in the Continuous Integration (CI) of the DevSecOps lifecycle as illustrated in .

**A picture containing website

Description automatically generated**

Figure : IoTAC DevSecOps lifecycle

### 5.1.1 Functional Testing

Functional testing will be performed on three levels:

* **Unit tests** – to verify the functionality of a component
  + at *least three tests* per component
  + for each unit test we have defined *Functional Test Case Specification*
* **Integration tests** – at least *four test case scenarios* for the interoperability tests
* **System tests** – black box functional tests.

For each test case, the following steps are defined:

* **Step 1: Test Case Specification** 
  + Based on the component specification and requirements elicitation documented in D2.3 and D2.2 (natural language)
* **Step 2: Test Implementation**
  + Technical realization of the test cases (e.g., JUnit)
* **Step 3: (Automated) Test Execution**
  + Test cases are executed against the target
  + Automated test execution within the CI pipeline
* **Step 4: Test Evaluation and Reporting**
  + Results of the test execution are compared with the expected results
  + Failures will be reported

**Step 1: Test Case Specification**

Functional test case specification template is presented in Figure 10.

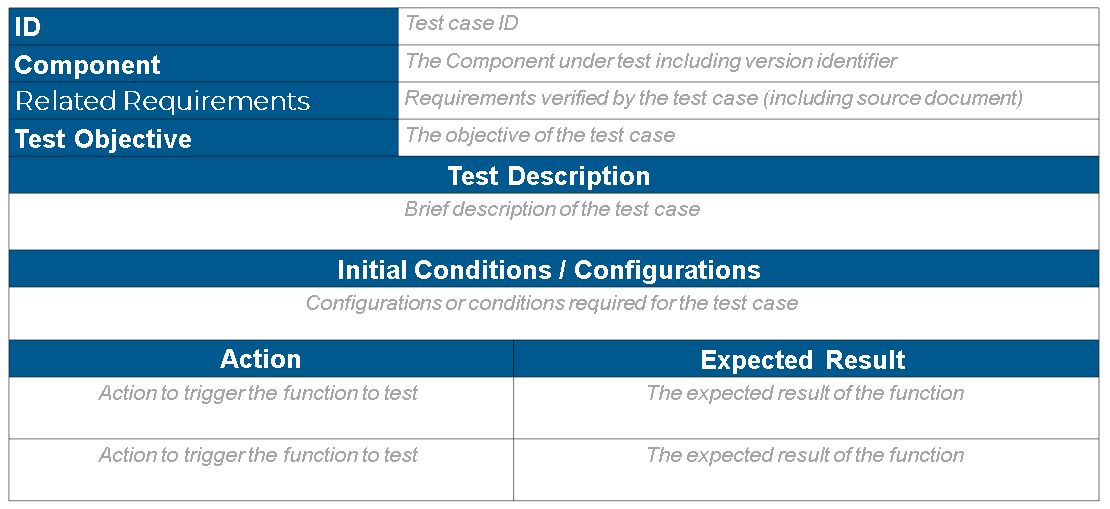


Figure : Functional Test Case Specification Template

**Step 2: Test Implementation**

Various technologies will be considered for the realization of test cases such as JUnit, NUnit, PyUnit, etc.

**Step 3: Text Execution**

The text execution pipeline is illustrated in

Graphical user interface, application

Description automatically generated with medium confidence

Figure : Test Execution Pipeline

**Step 4: Evaluation and Reporting**

### 5.1.2 Static Application Security Testing (SAST)

* Analyze application source code
* Technologies to identify dangerous patterns within the code
* Part of source code quality control techniques

Advantages:

* Scalability - running analyses across multiple software repeatedly
* Automatic identification of well-known security flaws
* Precision in highlighting security flaws and affected code areas to developers.

SAST approach is based on 2 phases:

* **Coding phase (IDE Approach)**
* Provide real-time feedback to developers
* Fix issues before they commit the code
* Recommendations on how to fix discovered issues
* **Building phase (CI Approach)**
* Developers integrate code into a shared repository.
* Each integration is verified by automated build and automated tests.

### 5.1.3 Dynamic Application Security Testing (DAST)

* Technologies to detect security vulnerability in an application in its running state
* Executing simulated attacks

**Security Requirements**

Sources of requirements:

* Task 2.2
* OWASP Application Security Verification Standard (ASVS) & IoT Security Verification Standard (ISVS) requirements are considered
* Using defined test cases from ETSI TS 103 701 v1.1.1 (2021-08) (The test scenarios target a baseline effort according to ETSI EN 603 645, which addresses a baseline security level for protecting IoT products against the most common cybersecurity threats)

ETSI TS 103 701 v1.1.1 (2021-08) Cyber; Cyber Security for Consumer Internet of Things: Conformance Assessment of Baseline Requirements

ETSI EN 603 645 Cyber Security Testing and Evaluation Service

**Techniques to be used**

Test Techniques to be used:

* **Vulnerabilities scans**
* **Port scans**
* **Fuzzing**
* **Denial of Service (DoS) attacks**
* **Code injection attacks**
* **Brute force attacks**
* …

Security Test Case Specification Template is illustrated in Figure 12.

Graphical user interface, text, application, email

Description automatically generated

Figure : Security Test Case Specification Template

DAST Test Case Execution pipeline is illustrated in Figure 13.

A picture containing timeline

Description automatically generated

Figure : DAST Test Case Execution

**Test Evaluation and Reporting**

## 5.2 TDL-TO as a specification technique

/\*Provide the definition of TDL and motivation

The Test Description Language (TDL) is a new domain-specific language for specifying test descriptions and presenting execution results. TDL is developed and standardized by the Technical Committee Methods for Testing and Specification (TC MTS) at the European Telecommunication Standards Institute (ETSI). It consists of a standardized meta-model that defines relevant concepts, the relationships between them, and the associated semantics.

/\*Introduce TDL-TO

Using TDL, test objectives can be specified in a simplified and generic manner using informal text. With the Structured Test Objective (TDL-TO) extension, you can specify (semi-) structured test objectives in a more formal manner and use concrete syntax notation to present those concepts.

/\* Introduce Concept of Test Purposes

/\* Explain domain specification with TDL-TO

Domain specification concepts add the notion of abstract events and entities as illustrated in Figure 14.

Text

Description automatically generated with medium confidence

Figure : Domain Specification with TDL-TO

## 5.3 Methodology for translation

/\*Methodology for translation of functional tests

* Step 1: Analysis of Functional Test Case Specification Templates (FTCST)
* Step 2: Definition of common test configurations

|  |  |
| --- | --- |
|  | TDL-TO |
| 1 | **TP Id** |
| 2 | **Package name** |
| 3 | **Reference** |
| 4 | **Test purpose/Test Objective** |
| 5 | **Common configuration file** |
| 6 | **Expected behaviour block/If** |
| 7 | **Expected behaviour block/Then** |

/\*DAST Methodology for translation of security tests

* Step 1: Analysis of Functional Test Case Specification Templates (FTCST)
* Step 2: Definition of common test configurations

# 6 Detailed List of Test Purposes

/\* This Section contains list of all Test Purposes

Annex A (normative or informative):  
Title of annex

Annex (informative):  
Bibliography

Annex (informative):  
Change History

| Date | Version | Information about changes |
| --- | --- | --- |
| <Month year> | <#> | <Changes made are listed in this cell> |
|  |  |  |
|  |  |  |
|  |  |  |

# History

|  |  |  |
| --- | --- | --- |
| **Document history** | | |
| <Version> | <Date> | <Milestone> |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

*Latest changes made on 2022-06-23*