INFINITYS
a company of ACS

DIGITAL SIGNAGE SOLUTIONS | www.infinitys.it

INFINITYS SPA
Sede legale ed amministrativa
Via Luigi Negrelli, 6
39100 Bolzano (I)

Le nostre sedi
Bolzano . Bressanone . Trento . Verona
Vicenza . Venezia . Bologna . Milano

Tel. +39 0471 063 333
info@infinitys.it . info@pec.infinitys.it

ANIMA Ops and Security

Authors
INFINITYS SpA

Creation Date
01.09.2025

State
Final

Last Update
12.01.2026

Version
4.1

Part. IVA & Cod. fisc. & Reg. Impr.: IT03311260214
SDI - Cod. Destinatario: T04ZHR3 (2ª cifra è uno zero)
Società a socio unico INFINITYS SPA - direzione e coordinamento da parte della ACS Data Systems SPA, iscritta al Registro delle Imprese BZ 00701430217

Index

1 [Devices Side] Security

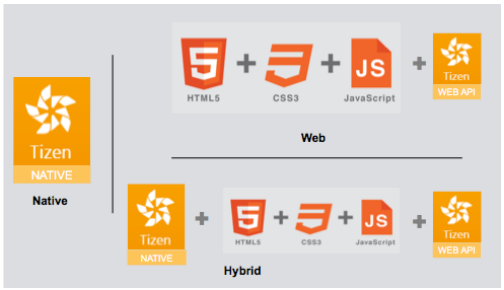
On the device side the first distinction to be made is about the nature of the device itself, we will say the HW/OS supported. Listed:

- SAMSUNG® Tizen OS.
- GNU/Linux (Ubuntu distro).
- MICROSOFT® Windows.
- Google® Android.

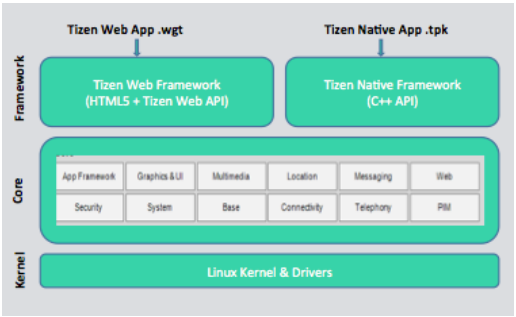
1.1 SAMSUNG® Tizen OS

Applications in Tizen can be written with native code using C/C++ or HTML5/JavaScript/CSS. Like other mobile platforms, Tizen supports three kinds of applications.

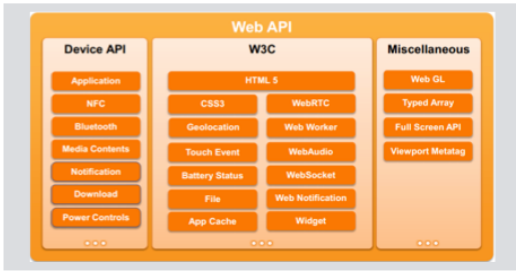
- Native Applications: written in C/C++.
- Web Applications: written on HTML5/JavaScript/CSS. This is the type of application chosen by INFINITYS for the development of ANIMA.
- Hybrid Applications: having web components as well as a native component.



The Tizen architecture consists of three layers. At the bottom we have the Linux Kernel & Drivers. On top of that, we have the Tizen Core layer which acts as an interface between Application Framework layer and the Kernel layer. It facilitates access to device hardware and other features. The Application Framework in the Core layer contains all the middleware, hardware-related services and provides the set of APIs needed for developing Native, Web or Hybrid Apps.



The Framework includes Tizen Native Framework that facilitates the running of Native and Hybrid Applications and the Tizen Web Framework which provides the Web Runtime (WRT) where the Web Applications run. The Web Apps make use of Web API which consist of HTML5 API's as well as a set of Device APIs provided by Tizen which is protected by Content Security Policy (CSP) and Privileges.



The Web API also includes miscellaneous APIs like WebGL, Viewport Metatag, Typed Array etc.

Like permissions in Android, Tizen relies on privileges to enforce a least privilege model. Additionally, Tizen requires application signing which ensures nobody can replace/update the installed application except for the original author and each application is sandboxed by SMACK (Simplified Mandatory Access Control Kernel).

The main principles of this Security Model are:

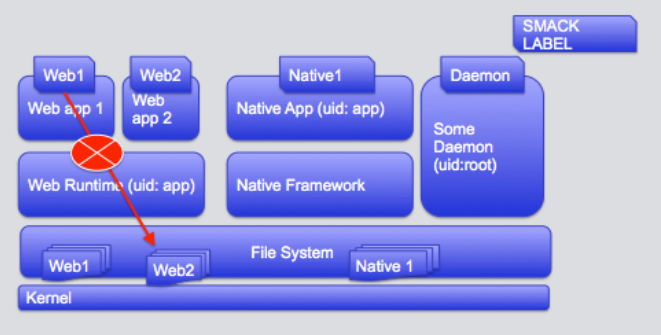
- **Non root applications:**

  - All applications run under same non-root user ID.
  - Most of the middleware and daemons will run as non-root.
- **Application Sandboxing:**
  - All applications are sandboxed by SMACK.
  - An application is allowed to read/write files only on its home directory and in shared media directory (/opt/usr/media)
  - Each application is unable to send IPC and sockets messages, r/w other application files.
- **Content Security Framework (CSF):**
  - Set of APIs/hooks used to create security-related services.
  - These are intended for AV Solutions.
  - Two types of engines: Scan Engine and Site Engine.
  - Scan Engine scans Data and Application for malicious behavior.
  - Site Engine scans URLs and blocks malicious URLs.
- **Application Signing:**
  - Application can be signed by Authors as well as Distributors.
- **Permission Model/Least privilege:**
  - All applications will have manifest/config file describing privileges.
  - Native apps use manifest.xml.
  - Web apps use config.xml.
  - Manifest files describe SMACK labels and rule as well.
- **Content Security Policy for Web Apps:**
  - For Web Applications, Policy or Content Security Policy is defined in the config.xml file.
- **Encrypt HTML/JS/CSS stored in Device:**
  - Encrypts at Install time and decrypts at runtime.

Tizen's sandbox is called Simplified Mandatory Access Control Kernel (SMACK). The basic rule of application sandbox is "**what's mine is mine; what's yours is yours.**" SMACK allows you to add controlled exception to this basic rule. SMACK is a kernel level Linux security module that determines how processes interact with each other.

In Tizen, every application has its own SMACK label. These labels identify the application and provide access controls. SMACK Terms:

- Subject: Any Running Process (Have Smack Label).
- Object: File, IPC, Sockets, Process.
- Access: Read (r), Write (w), Execute (e), Append (a), Lock (l), Transmute (t).



In SMACK, the subject can only access an object if the labels match or if there exists permission that grant access to the requested resource. A subject is an active entity while an object is a passive entity, which includes files, directories, IPC, sockets and processes.

SMACK ensures that applications are sandboxed and one application cannot access the files and data of other application and vice versa. However SMACK allows controlled exceptions to this. The interesting thing about SMACK

rules is that Tizen got about 41,000 SMACK rules in Tizen version 2.2.1. The number of SMACK rules is so huge that, there is a high chance that developers may mess up. So, in Tizen 3 onwards they will introduce Cynara and a Smack Three domain Model.

For Tizen applications, privileges are like permissions for Android applications. To use different APIs, appropriate privileges should be defined. The following are the used permissions, as of the date of this document, by the ANIMA APP:

```
<tizen:privilege name="http://tizen.org/privilege/tv.audio"/>
<tizen:privilege name="http://tizen.org/privilege/tv.window"/>
<tizen:privilege name="http://tizen.org/privilege/unlimitedstorage"/>
<tizen:privilege name="http://tizen.org/privilege/download"/>
<tizen:privilege name="http://tizen.org/privilege/internet"/>
<tizen:privilege name="http://tizen.org/privilege/filesystem.read"/>
<tizen:privilege name="http://tizen.org/privilege/filesystem.write"/>
<tizen:privilege name="http://tizen.org/privilege/content.read"/>
<tizen:privilege name="http://tizen.org/privilege/content.write"/>
<tizen:privilege name="http://tizen.org/privilege/tv.display"/>
<tizen:privilege name="http://tizen.org/privilege/tv.inputdevice"/>
<tizen:privilege name="http://developer.samsung.com/privilege/network.public"/>
<tizen:privilege name="http://developer.samsung.com/privilege/b2bcontrol"/>
<tizen:privilege name="http://developer.samsung.com/privilege/b2bsyncplay"/>
```

Tizen Web Runtime (WRT) is based on WebKit2 on top of which, the web apps run. WebKit2 is a new API layer over WebKit. It supports split process model like the Google® Chrome tabs. In Chrome, each tab is a separate independent process, similarly in Tizen WRT, every web application runs as a separate process. So, if anything goes wrong in one process, it should not affect other running processes. In WRT, every web app runs inside a sandbox. One of the main reasons is that WebKit is known to have a lot of vulnerabilities. A lot of them are still being reported and coming up. To prevent the impact of these, Tizen make use of the split process model and the application sandbox.

In addition to Tizen OS, major vendors provide additional protection with their own proprietary solutions. As an example, Samsung large format displays are shipped with Samsung Knox Security which provides, among the others:

- A tamper resistant hardware
- Dedicated execution environment for applications that handle sensitive data
- All sensitive data is stored in secure storage encrypted with device's unique hardware key.
- Use internationally recognized and standardized cryptographic technologies
- A variety of authentication methods including PIN, pattern, and password, and combine them to deliver strong user authentication
- Proactively deploy solutions that detect and prevent illegal tampering attempts on our products to maintain their safety and integrity
- A strict security development process throughout the entire product lifecycle
- Apply the latest security updates and patches to combat attacks from the ever changing malware and hacking landscape

2 [Backend Side] Asset Security

ANIMA SaaS offering is based on many assets:

- *Infrastructure*: this asset group includes all connectivity assets, network level devices and any other devices on which base, application server(s) grant service over the internet.
- *Application Server*: this asset group includes web server used both to manage HTTPS protocol (pages) and the execution of ANIMA business logic.
- *Database Server*: this asset group includes all server(s) managing data persistence layer, both relational (RDBMS) and media content (*.mp4, *.jpeg…).
- *File Server*: this asset group includes all server(s) managing static assets like media contents (*.mp4, *.jpeg…).

2.1 Authentication & Authorization

ANIMA authentication system is based on OpenIddict, running on INFINITYS Cloud Servers and managed by INFINITYS. OpenIddict is an OAuth 2.0/OpenID Connect framework for .NET Core.

It is used to:

- protect ANIMA resources.
- authenticate ANIMA users using a local account store.
- provide session management.
- manage and authenticate ANIMA clients such as displays.
- issue identity and access tokens to ANIMA clients.
- validate tokens.

Administration and access to assets are managed using many fine-grained policies:

- *2 factor authentications:* INFINITYS directory is a Microsoft 365 Business directory, where 2FA is defined as mandatory for all administrator-level accounts.
- *Segregation of Duties:* inside INFINITYS, *Infrastructure* administrators are persons, belonging to the “Cloud Team”, while *Application Server* and *Database Server* are administered by the “DEV Team”. Access to RDMBS is based on application-users that are different from database owner user. The responsibility of this owner belongs to Cloud Team.

Regarding access to the web app of devices and content, say the part of the solution that the content managers specified by the customer must have access, INFINITYS suggests using the federation feature between directories. Specifically, the suggestion is to use the integration functionality between ANIMA and the client's user directory. In this way:

- content managers can use their corporate account for access, reducing the complexity of use.
- Credential management policies (password policies) are the same as those used at the enterprise-wide level.
- Any credential revocation at the enterprise level also impacts, immediately, the ANIMA system.

If integration with the corporate directory is not possible, ANIMA supports its own authentication module, with local identity, which provides the following features:

- The password length must be 12 chars or more.
- password complexity:
  - 1 or more uppercase char.
  - 1 or more lowercase char.
  - 1 or more special char.
  - 1 or more number.
- password change is enforced at least every 60 days.
- ANIMA enforces new passwords not to correspond to the last 5 passwords used.
- login trace: after a successful login, user sees where (IP) and when the last successful login has been done.
- failure trace: latest 10 unsuccessful login attempts are kept in logs, including IP source information.

2.2 Penetration Test(s) and Vulnerability Assessment

At infrastructure level, INFINITYS executes, on a monthly base "gray box" tests, identifying target systems and goals with the help of pre-shared background and system information.

In the last couple of years, INFINITYS work has created a structured base of tests ("test suite") using FOSS tools, in particular Metasploit Framework https://www.metasploit.com/ using Kali Linux distro https://www.kali.org/. On every test repetition:

- a subset of colleagues is defined as the „*attackers team*";
- first of all „*attackers team*" creates the test environment beginning from a „black box" environment, implementing it with highly relevant insider-level information;
- „*attackers team*" executes tests;
- Assessment results is shared between „*attackers*" team with INFINITYS Top Executives, who shares this content with Cloud Team;
- Immediately, Cloud Team works on „*patching*" and/or changing software/hardware configuration for „*closing*" detected vulnerabilities;

2.3 Software/Firmware Assets Management

ANIMA SaaS service delivery is based on different assets:

- *Server Operating Systems:* every machine involved in service delivery is virtual, based on Windows Server OS and GNU/Linux OS. Following producer(s) guidelines both are regularly updated on a monthly schedule. Moreover, INFINITYS certified technicians daily follow security bulletins, i.e.:
  - https://cve.mitre.org/
  - https://portal.msrc.microsoft.com/en-us/security-guidance
  - https://access.redhat.com/security/security-updates/#/

  identifying relevant issues/vulnerabilities and applying patches on the shortest available timeframe, if needed. Active monitoring is done with the help of advanced detection and response platforms such as SentinelOne by INFINITYS Security Operation Center, active 24/7.
- *Infrastructure devices OS/firmware:* for infrastructure devices (physical router/switches, disks array…), INFINITYS certified technicians follow producer(s) guideline and maintenance best practices, applying any security patch on the shortest available timeframe.

3 Development Operations

The software development process for Infinitys follows the Agile software development methodology. It advocates adaptive planning, evolutionary development, prompt delivery, and continual improvement, and it encourages rapid and flexible responses to change. The agile principles followed by INFINITYS are:

- Customer satisfaction by early and continuous delivery of valuable software.
- Deliver working software frequently.
- Close, daily cooperation between “businesspeople” and developers.
- Projects are built around motivated individuals.
- Face-to-face conversation as the best form of communication.
- Working software is the primary measure of progress.
- Continuous attention to technical excellence and good design.
- Simplicity is essential.
- Regularly, the team reflects on how to become more effective and adjusts accordingly.

Software development activities are organized in sprints of two weeks, where each sprint starts with a sprint planning meeting and ends with a sprint review meeting and a release of the developed software. The sprint planning meeting is held between all software developers. Whereas, the sprint review meeting is organized as a demonstration of the developed software, where all developers and all stakeholders participate.

Development, done primarily in Typescript/Javascript and C# programming languages makes extensive use of static and dynamic analysis tools.

During the daily activity, static analysis is performed locally using Visual Studio (C#) and Visual Studio Code (Typescript /Javascript) development tools.

If no major critical issues emerge from the analysis, an automated pipeline compiles the project and releases it to the testing environment. The compilation relies on the static validation tools: Net Compiler (C#) and ESLint (Typescript /Javascript). Unit tests performed through Jest (Typescript /Javascript) and NUnit (C#) that facilitate functional verification are performed and constantly updated. Depending on the required functionality and its complexity, a Test-Driven Design is also adopted.

Once development is completed, the code is uploaded to the versioning system and:

- further static analysis is performed using the SonarQube tool.
- tests are re-run overnight (nightly build) via automatic pipeline.

If no major critical issues emerge from the analysis, the pipeline proceeds with the execution of End-to-End (E2E) tests via Playwright software to check for functional regressions and if no errors are present, it proceeds with the release to the test environment. The E2E test loads a standard test environment, different browsers and applications that run on the devices and automatically reproduces the main interactions of a real user on the ANIMA software, providing images, videos, and reporting of any results that differ from the expected ones.

During execution in the test environment and in production, the ANIMA software is monitored by the development team using the three main observability tools:

- Logging through Serilog.

- Metrics through Prometheus and Grafana.
- Tracing through OpenTelemetry.

ANIMA workflow is based on Microsoft SDL methodology https://www.microsoft.com/en-us/securityengineering/sdl/

Among all practices, the actual implementation of the process in ANIMA is structured on a subset of the whole methodology:

- Practice #2 - Define Security Requirements: the whole system has been designed from the beginning with security in mind, following Microsoft best practices for microservices development which includes authentication and authorization with well-known frameworks like IdentityServer and OpenIdDict. Both frameworks have been designed to ensure standard-compliant OAuth 2.0/OpenID Connect servers. Authorization and logging have been implemented using middleware to reduce the maintenance and potential errors on new developments.
- Practice #5 - Establish Design Requirements: a common agreement has been found on authentication protocols (OAuth2.0), logging (Serilog) and cryptography (asymmetric).
- Practice #6 - Define and Use Cryptography Standards: Microsoft SDL Cryptographic Recommendations document is used as a reference.
- Practice #8 - Use Approved Tools: the same development tools are used across the whole development team as well as static and dynamic analyzers. All the adopted frameworks use Long Term Service versions to ensure proper bug and security fixes. All dependencies are updated regularly to minimize security risks.
- Practice #11 - Perform Penetration Testing: as described in section 2.2

4 Release Management

The release management of INFINITYS follows these principles:

- The work is planned in time-boxed iterations of two weeks.
- A backlog of features is maintained in prioritized order, and the priority is reviewed every iteration.
- At the start of each iteration the highest value items are selected from the backlog and a detailed planning for those items is done.
- Integration/release is done after each iteration.

The goal of having features completed at the end of each iteration allows us to determine what will be available to ship. The prioritization of the backlog allows us to have the most important features developed first, so that the release date can be met. The always shippable goal means that we can, if needed, accelerate our release schedule.

This agile approach enables better release planning by combining planning disciplines, which helps to focus on the highest value work, and engineering discipline, which helps to identify and fix problems early, giving us more predictability. This practice makes shipping a release a decision that our product owners can make without worrying if the team will meet a date far in the future.

To integrate produced software as often as possible, and to test and verify it frequently, after each iteration a new release of Infinitys is published. This usually happens on Monday afternoon and is published on our BETA system. Specialized software testers verify that the software works correctly and meets the requested requirements. After

the software is verified and potential problems are fixed, usually every second Monday in late afternoon, the rollout on PRODUCTION system is done.

Since one of our goals is to keep quality always high, in every state of our software development process, on serious problems it is possible that a hotfix is released outside of this release plan. This will be decided at each occurrence and depending on the impact of the bug. Normal, non-blocking bugs are included in the normal release process of the software.

To keep track of issues notified by our users, the RADIX service desk management tool is used (https://www.infominds.eu/radixplus?lang=it).

The ANIMA solution provides an internal communications management feature for its user community. On each update, INFINITYS publishes news that, through notification, makes the community aware of the most interesting information related to the update itself. At the user's request, further, this communication can also be received by e-mail.

INFINITYS defines different maintenance time windows:

WHAT	WHEN	COMMUNICATION
FORTNIGHTLY UPDATE - new version of the software.	Monday (2 or 3 mondays per month, depending on the actual calendar) 9 p.m. to 10 p.m. (UTC+1, Rome)	Only in cases that differ from this rule.
PLANNED UPGRADE - modification of server infrastructure. - network configuration upgrade. - expansion of available storage.	Monday to Friday, from 1 p.m. to 2. p.m. (UTC+1, Rome).	Warning message through the ANIMA notification system, at least 48 hours before the update.
HOTFIX - Vulnerability reported on one of the components. - Software bug resulting in corruption or loss of data.	When needed.	Warning message through the ANIMA notification system, with as much notice as possible, reasonably with a few hours' notice.

During this time windows:

- the displays/players will play, without any interruption, the scheduled content (e.g. pictures, videos).
- in case the layouts provide widgets that require updated data from the Internet, they may not have access to the data themselves and activate, where applicable, the fallback behavior for loss of connectivity.
- the backend portal (https://cockpit.infinitys.it) may not be accessible.

Keep in mind that updates made do not always require downtime of services and, further, that outage times are typically limited to a few minutes, always within the above-mentioned window.

5 System Redundancy for Business Continuity

ANIMA SaaS is mainly based on a group of virtual machines with a „two level“ affordability:

- *Application redundancy:* all application components are based on multiple battery-based application servers that can be reached via virtual balancers or clustered solutions capable of making the system scalable while also ensuring the continuation of the activities in case of block of an application server or database server without the need for any intervention. These battery-based servers, on the other hand, are linked to a monitoring system capable of verifying service availability to guarantee restart in the shortest possible time.
- *Virtual Environment Redundancy*: in addition to the guarantees of application balancing/failover, the features offered by the virtualization platform guarantee business continuity solutions allowing the automatic restart of virtual machines, hot migration between the various physical servers, load balancing on the servers and offer a series of tools capable of controlling and monitoring the system itself.

6 Users, Administrators and End Users Support

INFINITYS Service Desk is INFINITYS the single point of contact for all users: end users (ex. store manager), content manager(s), admin(s). Multiple access channels are available:

- Phone: +39 0471 063333
- Mail: support@infinitys.it

Working hour for INFINITYS service desk is from 8:00 AM to 5:00 PM (CET), Monday to Friday (excluding italian national, Trentino Alto Adige and Bolzano province local holidays).

7 GDPR - Right to be Forgotten

In the European Union, the General Data Protection Regulation (GDPR), in force since May 2018, regulates the right to be forgotten, in Articles 17, 21 and 22.

The ANIMA solution provides specific functionality related to the right to be forgotten. In case a user requests it, a user with ADMIN rights can activate the appropriate procedure.

The specific functionalities ensure the consistency of data archives, media and programming, while providing the possibility to permanently delete data related to the affected account.

In addition to the execution by an ADMIN, we will say in a manual way, it is possible to define within ANIMA some policies that automatically provide for this action, in example after 365 days from the date of last login performed.