D2.5 Analysis and evaluation of a dedicated security assurance methodology for multimodal transport

Work Package	WP2
Task	Security assurance methodology and tools
Authors	Sammy HADDAD
Dissemination Level	PU
Status	Final Version
Due Date	30/11/2021
Document Date	30/11/2021
Version Number	1.0

-QUALITY CONTROL

	Name	Organisation	Date
Editor	Sammy HADDAD	OPPIDA	26/11/2021
Peer review 1	Pietro De Vito	STAM	29/11/2021
Peer review 2	Konstantinos Malliatsos	UPRC	29/11/2021
Authorised by (Technical Coordinator)	Jason Sioutis	ICCS	30/11/2021
Authorised by (Quality Manager)	Vasileios Sourlas	ICCS	30/11/2021
Submitted by (Project Coordinator)	Angelos Amditis	ICCS	30/11/2021

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 883321. Content reflects only the authors' view and European Commission is not responsible for any use that may be made of the information it contains.

CONTRIBUTORS

Name	Organisation	Date
Sammy HADDAD	OPPIDA	02/09/21

DOCUMENT REVISION HISTORY

Version	Date	Modification	Partner
0.1	02/09/2021	Table of content definition	OPPIDA
0.2	15/11/2021	First Draft	OPPIDA
0.3	26/11/2021	Second Draft	OPPIDA
0.4	29/11/2021	Third Draft	OPPIDA
1.0	30/11/2021	Final	ICCS

LEGAL DISCLAIMER

CitySCAPE is an EU project funded by the Horizon 2020 research and innovation programme under grant agreement No. 883321. The information and views set out in this deliverable are those of the author(s) and do not necessarily reflect the official opinion of the European Union. The information in this document is provided “as is”, and no guarantee or warranty is given that the information is fit for any specific purpose. Neither the European Union institutions and bodies nor any person acting on their behalf may be held responsible for the use which may be made of the information contained therein. The CitySCAPE Consortium members shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials subject to any liability which is mandatory due to applicable law.

Table of Contents

List of Abbreviations and Acronyms

Abbreviation	Meaning
ITC	Information Technology
ANSSI	Agence National de Sécurité des Systèmes d'Information
CC	Common Criteria
CSPN	"Certification de premier niveau"
IT	Information Technology
ITS	Intelligent Transportation System
ITSEF	Information Technology Security Evaluation Facility
OS	Operating System
OSP	Organizational Security Policy
PP	Protection Profile
SFR	Security Functional Requirements
ST	Security Target
TOE	Target Of Evaluation
TSF	TOE Security Functions

Executive Summary

Security by design, i.e., the process of specifying and implementing a corresponding secure architecture, is of utmost importance in order to secure large and complex systems. In real ITC systems, security rarely happens by chance, thus, explicit efforts need to be made to specify efficient security properties. However, security by design is not sufficient to guarantee the fulfilment of the appropriate security requirements.

If software and architecture design can enforce good security principles, real security of systems depends on their full, complete implementation and configuration, once they are in operation.

Regarding CitySCAPE, in order to guarantee the fulfilment of security requirements identified in WP3 based on threats identified in WP2, in this deliverable, we define different sets of tailored assurance requirements, i.e., evaluation activities for the different system life-cycle stages.

To achieve this goal, we start by studying the state-of-the-art in assurance methodologies in order to be able to reuse best practices in a dedicated and tailored-made assurance framework defined to fulfil CitySCAPE needs, in order to obtain efficient assurance to match multimodal systems need.

In this deliverable, we define a dedicated approach that provides efficient and adapted assurance that we design based on the project risk analysis. We define evaluation tasks for the different stakeholders of the system at the different lifecycle stages, including code quality review, functional tests, offline penetration testing, operational penetration testing, operational configuration review, etc.

Based on the final assurance task sets, we will specify the different tools required to fulfil those different tasks (code analysis tools, test manager tools, vulnerability tests tools, system configuration review, etc.). OPP leads Task 2.4 given its experience in security assurance methodology and will be supported by UPRC and ICCS (both involved in assurance related projects and activities.

1 INTRODUCTION

1.1 Project Introduction

The traditional security controls and security assurance arguments are becoming increasingly inefficient in supporting the emerging needs and applications of the interconnecting transport systems, allowing threats and security incidents to disturb all dimensions of transportation.

CitySCAPE is a project funded by the EU's Horizon 2020 research and innovation program, which consists of 15 partners from 6 European countries united in their vision to cover the cybersecurity needs of the multimodal transportation.

More specifically, the CitySCAPE software toolkit will:

- Detect suspicious traffic-data values and identify persistent threats
- Evaluate an attack's impact in both technical and financial terms
- Combine external knowledge and internally-observed activities to enhance the predictability of zero-day attacks
- Instantiate a networked overlay to circulate informative notifications to CERT/CSIRT authorities and support their interplay.

The project duration extends from September 2020 to August 2023.

1.2 Deliverable Purpose

This deliverable proposes an assurance methodology that will enable efficient cybersecurity assurance in multimodal systems.

Multimodal systems are large and complex systems composed of several independent subsystems owned and managed independently by different operators. When it comes to security assurance, this is the most challenging contexts.

In fact, the well-known and established approach, the Common Criteria (CC) [1], take already several months up to years to certify just products for one specific version. Those evaluations require the full commitment of the developer to provide the required detailed input of the product development and specification details. In fact, the well-known and established approach, the Common Criteria (CC) [1], take already several months, up to years, to certify products for just one specific version. Those evaluations require the full commitment of the developers to provide the required detailed input of the product development and specification details.

Multimodal systems are composed of hundreds of thousands of different products, deployed by different entities, without any specific support of the developers. Knowing that they are regularly if not daily updated. So clearly, CC cannot be used at the system level in that case. Also, it can be argued

that assurance levels provided by the CC are too high and not necessary at the system level in our context.

In this deliverable, we discuss both (i) how to identify adapted assurance requirements: if the whole system cannot/doesn't require full high assurance evaluation and (ii) which components need assurance evaluation to provide appropriate confidence at the system level.

In fact, we discuss in section 2 how the state of the art demonstrates that at system level, product evaluation procedures ([1], [2], [3], [4]) do not adapt well, and security at the system level requires more generic approaches requiring preliminary security problem analysis and then security assurance adapted to the analysis. We discuss in section 3 the security assurance needs of the project to finally determine in section 4 the adapted methodology tailored to CitySCAPE needs.

Section 5 presents tools requirement and quick overview of existing ones for later assurance validation based on the CitySCAPE approach.

2 SECURITY ASSURANCE STATE OF THE ART

Several IT evaluation schemes exist. Their main objective is to validate the security functions of a product or system. A subset of them ends with an official certificate delivered by a certification authority. In all cases, there is an evaluation process to be done and then the level of recognition/acceptance depends on the level of maturity of the method and the one who runs it. In the case of a certification, it is the certificate that determines the level of the achievable recognition.

CitySCAPE does not aim at defining a certification process (i.e., no need for an official certification body) but only a faster evaluation process that does not have to go through the burden of an administrative task associated with certification, that helps system owners to determine if their systems provide appropriate security to their users.

However, in this section we present the state of the art for both general evaluation and certification processes. If not targeted, existing certification schemes are still the most complete and mature references when it comes to cybersecurity assurance.

There is a difference in evaluating or certifying a product and approving an entire system. In this document, and more generally in the CitySCAPE project, we aim at studying mainly the evaluation of the most important multimodal systems components i.e., vehicles command and control, user information apps and services, traffic management centres, and security components (IDS/IPS, SIEM, etc.).

We do not aim to assess the complete system security at once, but only evaluate a critical part of it and validate the general security architecture to provide enough assurance at system level. We try to avoid among other things cascading effects.

Thus, in this section we present the existing evaluation processes and schemes for IT products available in the state of the art. To better understand the global challenge addressed by CitySCAPE, we start by describing general challenges for security assurance.

2.1 Security evaluation methodologies generalities and common challenges

Over the last three decades many researchers and practitioners have addressed the general problem of IT products validation, to try to find more specific and formalized approaches. So far, no fully satisfying (i.e., universal recognition with no drawbacks) solution has been found (and it will probably never be). Several comprehensive overviews of the various efforts made on the evaluation and measurement of IT security domain exist [5], [6], [7], [8], [9].

Before comparing different evaluation schemes and methodologies, we start by identifying the main and most important aspects that make the difference between existing security evaluations.

All existing IT security evaluation methods address the following three topics:

- What must be evaluated?
  - Which product and which version of the product?
  - Which function of the product?
  - In which environment and for which type of threat?
- Which evaluation activities?
  - Evaluate the development
  - Evaluate the product architecture
  - Test the external/internal interfaces
  - Analyse the code, the guides, etc.
- Who is competent and in charge of what activity:
  - Who is the evaluation authority in charge of defining and managing the evaluation activities to guarantee the overall evaluations expectations?
  - Who will pay and be the sponsor of the evaluation?
  - Who has the expertise and required test environment?
  - What does the developer have and what information must he provide for the evaluation of its product?
  - What is the end user's point of view?

The above three dimensions correspond to what is generally called:

- The Security Target (ST).
- The assurance components.
- The evaluation schemes.

All IT security evaluation schemes have their own identification of what is most important to tackle for these three dimensions and how to tackle them.

It is important to understand that there is no universal solution for the problem of IT security evaluation and all known solutions are disputed and criticized. In fact, they all have distinct advantages and drawbacks.

Security evaluation is a difficult problem and would probably remain so because IT systems are complex, and they evolve rapidly. Whether or not it will be feasible one day to obtain a fully automated formal proof process for any systems security, the current state of the IT technologies attempts to enhance existing approaches to lower assurance cost and provide higher assurance levels for even more complex systems.

Multimodal systems are directly affected by this observation. These systems are modern, and constantly evolving, so they do not benefit from the years of real security experience, and they are significantly complex (system of systems, large applications, etc.). So, it will not be a simple task to define and adopt a universally recognized evaluation scheme for ITS products and systems.

The general problem of IT products validation has been addressed by many researchers and practitioners, in search of finding more specific and formalized approaches. So far, no fully satisfying (i.e., universal recognition with no cons) solution has been found.
The main identified challenges and gaps are [8]:

(a) Elucidation, Modelling, and Validation of Security Requirements.
(b) Security Assurance in Component-based Development.
(c) Operational Security Assurance.
(d) Security Assurance in Service Selection (assurance for systems providing configurable service-oriented architectures).
(e) Security Assurance Aggregation (combination of assurance of different system components).
(f) Security assurance Tool.
(g) CC Protection Profile for Trusted Computing Features.
(h) Automation of Security Assurance.
(i) Identification and Prediction of Security vulnerability

In the end, we can identify four main evaluation approaches categories that we discuss briefly here.

2.2 Conformity Checks

Conformity Checks (also called compliance assessment) is a form of evaluation that validates a product or system compliance to a specific reference. This approach needs to have a reference conformity list. This list has to be kept up to date and has to be relevant for the product type and its actual needs in terms of functionality and security. There are two main limitations to the conformity check approach. First, the definition and maintenance of relevant conformity lists can be difficult or even infeasible in an industrial context (i.e., too many updates needed, no agreement on the conformity requirements, scope of conformance too restrictive, etc.). Also, anything not conformant to (a part of) the conformity list cannot be validated. On the other side, conformity checks provide usually the fastest and cheapest evaluation scheme compared to other methods, providing comparable levels of confidence. Also, the evaluation results are simpler to understand and easily comparable since every test is known in advance and they are the same for every product evaluated.

A main certification (and thus evaluation) scheme that defines a normalized test suite suitable for Conformity Checks is the FIPS 140-2 standard [3] by the Federal Information Processing Standard (FIPS). This certification only concerns cryptographic products. The FIPS are public standards developed by the United States federal government, aiming at ensuring some computer security and interoperability for the US governmental Information Systems.

Contrary to other frameworks, such as ITSEC, CC or the French CSPN [10] [2] [1] [11] [12] FIPS evaluations do not need the specification of a security target. The list of functions and tests to be done is directly defined by the FIPS 140-

2 standard, which indirectly defines the security target together with the assurance component through the list of conformity checks.

In this approach, since the test requirements are defined in the standard, they age with it and the standard has to be rewritten every time new security paradigms are required (i.e., new threats, new needs, etc.). For this reason, the FIPS 140-2 standard foresees to be reviewed every five years, whereas such a standard in the multi-modal transportation and ITS world should be typically reviewed every 6 months considering the rapid evolution of the system. Also, even if cryptographic functions are quite well recognized and very limited in complexity and numbers, this is not the case when we consider the full implementation of an ITS architecture. Such architecture includes Operating Systems (Oss), communication and security stacks, sensors, applications and so on. Cryptographic functions are a very limited subset of those systems and scaling the methodology would be at least as expensive as developing the system themselves.

In many industrial sectors and when feasible, this scheme is the preferred one – see for example the Compliance Assessment process specified by the C-ITS Platform[1], the US Certification program for Connected Vehicles and ETSI ITS validation platform for standardized protocols. But such an approach can only partially cover Multimodal security validation and so far, nothing close to the beginning of a recognized and validated set of security requirements and their associated tests exists for such complex systems (even though this approach is regularly promoted).

2.3 Vulnerability tests

This approach simply defines an evaluation perimeter, not necessarily forming a real complete ST. Usually, it only defines the product, the tests environment, and associated limitations. Then an expert runs any tests of his/her choice during a predefined time on the defined scope. At the end, the result is the set of potential vulnerabilities identified by the tester. If no vulnerability is found, then the evaluation result states that the product resisted to an attacker a certain number of days equal to the evaluation duration.

Thus, this method allows validating the product's security level, providing low to medium assurance level. Also, on average, the results are obtained faster than other methodologies; note that common tests take 20 to 30 days.

The drawbacks of this methodology are that there is a great need of confidence in the tester competences, and they do not try to provide high assurance and do not cover other potentially efficient assurance activities (such as developers tests review, code review, etc.). Also, results are not fully consistent or directly comparable since two testers are free to use completely different tests for the very same product. The fact that they rarely

[1] https://ec.europa.eu/transport/sites/transport/files/2017-09-c-its-platform-final-report.pdf

rely on proper ST definition limit the efficiency of the tests, since they cover mostly evaluator concerns and not necessarily final user needs.

A formalized approach falling under this category is the French CSPN [2] where a detailed ST is required and the number of vulnerability test days is pre-defined - 25 days for every product. This process is the only one that provides a certificate signed by the prime minister and recognized nationally.

2.4 Assurance framework

The Assurance framework approach is the most complete and exhaustive approach. It provides the highest assurance levels (i.e., level of confidence in the product security), but it is generally more expensive and time consuming. It also requires the involvement of rare and expensive accredited evaluators expertise.

There are two main types of security validation, evaluation or accreditation processes: those made for products and those for systems. The more formal and structured one are for products: CC, ITSEC, TCSEC; when system security assessment includes more generic definitions of procedures, it is also called Information Security Management System (ISMS) such as [13], [14].

In fact, the main problem in security assurance frameworks comes from the fact that assessing security properties of an IT product fully depends on the product itself: its purposes, the technologies used to implement it, its functional and security architecture, its operational environment (e.g., users, interconnections), etc; and finally, the current state of the art of attacks.

All these parameters cannot be constantly standardized for every possible IT product in an up-to-date manner.

Clearly, we cannot evaluate in the same way products such as: a firewall, a data base, a web site or an operating system. It is also very hard to compare the results of any evaluation of this product, since even if they belong to the same categories, they will still be different and not subject to the same sets of attacks and threats; that is because of the technologies used and/or their operational environment.

Thus, every credible evaluation framework takes this observation as an axiom and does not try to provide a methodology to assess overall security rating, since there is no such universal security scale. All known methodologies adopt the same general structure:

1. Identify the product to be evaluated.
2. Define the security problem.
   a) Identify the assets to be protected.
   b) Identify the threats for the assets to be mitigated.
3. Define the security functions to be validated for that product in order to mitigate the identified threats.

4. Define a set of evaluation tasks to apply for the validation of the product's security functions (possibly set of tasks dedicated to the specific product type or category).
5. Define specific tests for the product to be evaluated.

A main difference between the methodologies relies in the fact that:

- either each of these points is directly defined by the methodology and thus directly constrained by it (limiting the possible application of the methodology);
- or the methodology asks for these points to be defined, leaving it more flexible.

Another main difference in the referenced approaches is the fact that the scope of evaluation (functionality evaluated) and the assurance level (evaluation tasks to validate the functions under evaluation) may be independent from each other or not.

Concerning security assurance for products, there is one primary reference. In fact, the state-of-the art only identifies one evaluation process that is recognized internationally. This reference is the Common Criteria for Information Technology Security Evaluation, known as Common Criteria (CC). [1] The CC is inspired by two important assurance schemes existed in United States and Europe: [15] and [10].

The first version of the Common Criteria for Information Technology Security Evaluation, known as Common Criteria (CC) dates back to 1994 and the last version to be standardized [1] was released in 2009. Since then, regular revisions have been done, but the global approach has not changed. The current version that is accessible on the common criteria portal and used for evaluations is the 3.1 Release 5.

It keeps the main concepts of ITSEC: (i) the notion of the need of a proper ST target, (ii) the decomposition of the evaluation in generic evaluation tasks independent of any product or security requirements, (iii) the definition of several evaluation assurance levels, each providing a set of more stringent evaluation tasks and evidence requirements.

Eventually, the CC provides a complete description and a reference set of security requirements to write formalized STs and the most extensive list of evaluation activities, including any activities empirically recognized as having a potential impact on the final product security.

The CC global approach consists of the evaluation of every life cycle element that helps to demonstrate that security requirements identified in the ST can be traced to the real product delivered to the end user. It proposes to evaluate the product life cycle management, the product architecture and full specification, the guides provided with the product to demonstrate that it can be easily used with the proper security configuration, the functional test run on the product and finally the vulnerability test to complete the whole assessment that the product fulfils the requirements stated in the ST

and that those requirements cannot be bypassed. Vulnerability tests and conformity checks are included in the CC and are only subparts of a complete CC evaluation. No other methodology covers so many aspects or is as well structured. That is why it is the best approach and accordingly, the most expensive one. Also, it is the only one to benefit from an official international recognition agreement.

The main drawback of this approach is that the assessments are static, time-consuming, and do not scale well to the extensive, networked, IT-driven system. It also does not offer continuous security assurance. Many researchers have made efforts to resolve these challenges. However, it is still an open issue [8].

2.5 Security metrics and other evaluation approaches

The three aforementioned approaches are the most commonly used ones. However, as mentioned before many researchers and practitioners have addressed the general problem of IT products validation, to try to find more specific and formalized approaches.

Several comprehensive overviews of the various efforts made on the evaluation and measurement of IT security over the last 20 years ago can be found in [5], [6], [8]. It covers software, standards ([16], [17]), taxonomies ([18], [19]), metric definitions ([20], [6], [21]), methodologies ([22], [23]), security databases ([19]), etc.

However, many of them face the criticism of security evaluation challenges ([24], [25]), relying on sole security expert's knowledge or not being adapted to real-world dynamic systems. Even though works are still on going and efforts are made to try to enhance evaluation methodologies, there is no new proposed solution, and the same three main (aforementioned) approaches are used.

3 CITYSCAPE ASSURANCE NEEDS

In order to define the proper assurance framework for CitySCAPE we first need to (roughly) estimate the type and level of risks that CitySCAPE system will face. Those risks will help us to identify the specific assurance needs of the solution.

In fact, several parameters will help us to define the project assurance requirements for the system and its component:

1. Risk levels: the higher the risk faced by the system components, the higher the required assurance.
2. System component exposition: the more a system component is exposed the more likely this component will be exposed to attacks and will need higher assurance assessments.
3. System components life cycle and updates frequency: the more often the system components are updated the more often assurance process shall be run.

To assess these parameters, we will analyse the project assets and threat identification provided by deliverable D2.3.

3.1 Threat identification inputs and risk analysis

The complete risk analysis results of the project use cases will be provided by WP7 and WP5 later in the project. We do not aim in this section to provide either a full analysis or fully validated by CitySCAPE partners. We only provide a rough estimation of risks for the different system components of the use cases.

For the present document and our assurance analysis we will use only the threat analysis provided by D2.3. It will help us identify the distinct threats that can apply to the different parts of the system.

These threats correspond to more or less critical risks that will justify the required level of assurance of the different components of the system. In fact, some important elements of the system such as user information display or journey planner tools present some important risks that clearly need to be mitigated for service and business purposes, but we can also identify even more critical risks affecting the autonomous vehicle which have a more direct safety impact.

In the first case, if cascading risks must be considered (potentially triggering any higher risk) and if user transportation services are important (but not life threatening) the size of the system and the complexity of assessing security assurance providing does not justify requiring high assurance level.

While in the second case, the safety risks are more severe, even if less likely and the part of the system to be protected much smaller. Thus, assurance assessment can be higher in that case.

To further complete this study, we partially estimate here risks for the different use cases system parts (composite assets). Risks will be evaluated

in detail and continuously updated in the CitySCAPE solution and are out of scope of this deliverable. Therefore, the objective here is to make a first identification of the most critical part of the system and the severity of the risk they face to choose our assurance approach. Obviously, this is not a complete analysis, since system owners and managers did not take part in the assessment of the impact, just a preliminary study.

Tallinn Composite Assets			
Composite Asset ID	**Asset Name**	**Most critical risks**	**Asset exposition**
COM-TAL-AS-01	Autonomous Vehicle (AV) Shuttle	AV Accident and road user death (**critical**)	Medium
COM-TAL-AS-02	Autonomous Vehicle (AV) Shuttle Remote Operator	AV Accident and road user death (**critical**)	Medium
COM-TAL-AS-03	Communications Platform-as-a-Service (CPaaS)	Service disruption (**high**)	**High**
COM-TAL-AS-04	Payment Service System	Stilling or payment data modification (**critical**)	**High**
COM-TAL-AS-05	Roadside Unit (RSU)	Service disruption (high)	Medium
COM-TAL-AS-06	Tram	Service disruption (high)	Low
COM-TAL-AS-07	Bus	Service disruption (high)	Low
COM-TAL-AS-08	Trolleybus	Service disruption (high)	Low
COM-TAL-AS-09	Autonomous Vehicle (AV) logging server	Service disruption (high)	Low
COM-TAL-AS-10	Telemetry Server	Service disruption (high)	Medium

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 883321. Content reflects only the authors' view and European Commission is not responsible for any use that may be made of the information it contains.

Genoa Composite Assets			
Composite Asset ID	**Asset Name**	**Most critical risk**	**Asset exposition**
COM-GEN-AS-01	AVM (Automated Vehicle Monitoring) System	AV Accident and road user death (**critical**)	Medium
COM-GEN-AS-02	Passenger Mobile Device and Application	Service disruption (high)	**High**
COM-GEN-AS-03	Smart Display	Service disruption (high)	Low
COM-GEN-AS-04	Subscription System	Service disruption (high)	**High**
COM-GEN-AS-05	Ticketing System	Stilling or payment data modification (**critical**)	**High**
COM-GEN-AS-06	Validator Mobile Device and Application	Service disruption (high)	Medium
COM-GEN-AS-07	Info-mobility Server	Service disruption (high)	Medium

For asset exposure evaluation we use the following very simple scale:

- High: WAN referenced public address.
- Medium: WAN unreferenced public address.
- Low: LAN access.

Since transport systems are critical infrastructures, we can easily estimate that all components of the system face at least high risks.

From our point of view, even if they are not necessarily the most exposed once, the most critical elements are the one that can imply safety issues or include payment transactions.

Service disruptions are important risks but not critical in the sense that the system can always recover while physical damages or money stealing can have permanent effects. The most critical elements correspond to a very small part of the system, while the main components face lower risks.

So, for us, this brief study clearly shows that we have two different assurance needs.

Thanks to our rough study, we identify the need for two different assurance levels.

- High assurance needs for a small part of the system having potential implications on safety issues or payment transactions, or being highly exposed
- Medium assurance needs for the biggest part of the system, providing all parts of end-user travelling experience with multi-modal services and potential targets for cascading effects

3.2 Assurance level requirements

Definition of adapted assurance levels for a system is depending on different factors. The first one discussed in the previous section is the level of risk faced by the system or product under study.

The second one is the cost of assurance assessment. Assurance is expensive and existing high assurance level evaluations are not affordable for large and complex systems.

Common assurance evaluation (CC and CSPN, for example) cannot scale to systems for costs reasons. They already take months to run and they are valid for one specific version of a single product (firewall, VPN server, signature service, etc.). They would take potentially decades for full-scale complex systems composed of tens of critical components.

Therefore, this parameter must be considered in CitySCAPE and find the right balance between the following elements:

- Level of risks faced by the system.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 883321. Content reflects only the authors' view and European Commission is not responsible for any use that may be made of the information it contains.

- Complexity of the system that require assurance and associated assurance cost.
- Pre-existing quality and assurance processes in the domain ecosystem that could make the assurance assessment more efficient.

The most critical elements

i. Need individual and dedicated assurance evaluation providing high assurance level
ii. This includes safety critical components and the modules of the CitySCAPE cyber-security solutions

Less critical elements

iii. Need global protection of the system provided through "operational assurance" of secure elements (demonstrating that the system configuration allows the component to be protected by the CitySCAPE security solution

3.3 Cascading threats management

As identified earlier, the entire system cannot be evaluated at once to assess high assurance. The targeted systems will be too large, too complex, and constantly evolving (component version, system, or network configuration, etc.). For this reason, assurance in that context will not be assessed at once. The global assurance of the system, which will help us guarantee the proper security properties of the entire system, will have to be gained by composing assurance elements of local components.

As identified in the section 3.1 only the most critical element will be evaluated to guarantee the whole system properties.

To **prevent cascading threats**, our approach is to **identify global system requirements** that avoid such threats. In fact, most of the risks come from the fact that some components are considered trustworthy and so no specific protection is used against those components, and thus in case of corruption the attack, the threat can spread "easily". But if those risks are identified at system level, **specific security counter measures can be found**. Once identified, the security controls can be implemented and counter those threats.

Therefore, security assurance composition is handled through security requirement decomposition. Thus, cascading threats will be handled in our methodology by the fact that our assurance approach will guarantee specific security requirements fulfilment that will mitigate those risks.

When security requirements are properly defined to counter cascading threats, then assurance validates that the system is, in fact, protected. However, specific mechanisms must be defined to make sure that proper security requirements are made to mitigate those specific risks. This is also true for privacy, safety and general cyber-security issues.

This is why the assurance methodology that we define in this deliverable defines a security target based on the global risk analysis conducted thanks to the methodology defined in D2.3.

Cascading threats, as well as all security assurance composition needs for privacy, safety and general cyber-security issues are handled by definition of local security targets based on risk analysis made at the system level which address cascading threats, safety, privacy and general IT risks.

3.4 Assurance continuity

Assurance continuity is one of the challenges of assurance security [8]. IT systems evolve rapidly, while security assurance assessment (mostly) apply to one single version of a tested product regarding one instance of the state of the art at the time of the evaluation. But IT product (for one version) lifetime rarely exceed months, as well as cyber-security state-of-the-art and threats evolve almost on a daily base. Therefore, assurance evaluation results can lose value overtime.

It is often hard to understand why assurance security extends from product version X to version X + 1. This is simply related to IT security principals. Security issues lies into implementation details and if you change any of those details, state-of-the-art has already demonstrated that it could have high security impacts. Thus, in order to assess that a code modification has no security impact, most often the majority of the assurance evaluation tasks are required to be repeated.

In our assurance methodology, the assurance continuity challenge will be tackled differently, depending on the asset's critical aspect. Since we define two different assurance levels, the way to handle assurance continuity will be different for each of them.

One of the main properties of the CitySCAPE solution is to provide means to monitor security. We propose to use these tools to monitor security assurance evolution:

i. its adaptation to new/real system threats
ii. adaptation to software and architecture updates
iii. current fulfilment of security requirements

For highest assurance level, the proposed assurance continuity, that is to be provided, concerns security target evolution. Thanks to (i) and (ii) we will constantly monitor if components evaluated at high assurance level still meet an adapted requirement. If not, their security target will be updated, and new evaluation will be requested.

For lower assurance components, the assurance assessment will be provided by supervision results (iii). This will allow us to manage assurance continuity for updated components, for which we will be able to continuously provide assurance via the same means.

- Security targets updates based on continuous risk assessment and threat assessment based on real system observation
- Continuity provided by operational assurance (system supervision provided by IDS, IPS and SIEM) for lower assurance needs

4 CITYSCAPE ASSURANCE METHODOLOGY

4.1 Risks analysis and security requirements definition

The first innovative step we provide in the CitySCAPE assurance approach is to handle assurance composition by identifying risks at system level in order to project them onto local components. From each risk, we derive local requirement that will counter local projection of global risks, and thus guarantee assurance at system level. The idea is to provide assurance composition by system risk and security objective decomposition.

4.1.1 Assets' assurance requirement identification

The **first step** in our methodology is to identify the **sensitivity** of the different **components of the system** (cf. section 3) to first define if they need high or medium assurance assessment.

To take this decision, several steps and parameters should be evaluated:

1. *Assurance assessment capability:* in the best-case scenario, all components of the system should be CC-certified. In the CitySCAPE context, we know these resources can be limited. Thus, the first step of assurance requirement is to identify assurance capabilities and how many resources can be provided (providing an upper bound on how many system components can be evaluated at high assurance level cf. section 4.2.1).
2. *Level of risks applicable to each system asset*: only the systems' assets facing the most critical risks are considered critical. If the number of critical components can be covered by assurance assessment capabilities, then go to the security target definition process for those elements. If not go to 3.
3. *Size of the critical sub-system*: if too many components are facing critical risks (considering assurance assessment resources), then a subset of them must be identified as candidate for high assurance. This can be due to low risk analysis maturity or simply due to too high assurance requirements (very sensitive systems). In the first case go back to step 1. Otherwise, the following elements can be further added to the assurance requirement classification:
    a. *Exposure of the component*: in fact, the most exposed component (typically public interfaces on the internet, or any service accessible form internet even if not publicized). This is due to the fact that most exposed components are more likely to be subject of attacks.
    b. *Feasibility of high assurance evaluation of a specific component:* finally, a last parameter can be used. If some critical components cannot be evaluated due to resources limitation, cost estimation of each critical component assurance evaluation has to be made. In fact, not all component evaluation is equivalent: size/complexity,

developer commitment into deployment and evaluation, product assurance maturity (evaluation of previous versions), etc; are example of parameters that can help to identify potentially more efficient evaluation. Only the most efficient once shall be chosen.

If, at the end of this process, too many elements are still identified, it is for the system manager to decide which elements are to be considered critical. The final output of this step of our methodology is the list of products to be evaluated following section 4.2.1. Otherwise, they fall under the section 4.2.2 assurance evaluation process.

4.1.2 Security target definitions for high assurance components

Critical components need to follow an advance assurance evaluation process (cf. section 4.2.1). But for that, we need to define a Security Target identifying:

- TOE overview.
- Security problem definition including:
  - **Assets** to be protected by the TOE.
  - The **threats** that are to be countered by the TOE.
  - **Assumptions**.
- Security requirements (security functions to be evaluated).
- Security requirement **rational**.

One very critical point in our methodology is to identify security requirements that enforce global protection of a large multidomain, multi-technology system, i.e., multimodal transport systems. To do this, we rely on the proper definition of security problems. The security problem is the base of the security implementation choices.

In security assurance processes, the only and best way to guarantee that security requirement identified for a product meet system needs, is the identification of the assets to be protected, the threats identified for those assets and the assumptions on the operational environment required for the product to work securely.

Ensuring exhaustivity of threat identification is not possible (at least the state-of-the-art does not provide such solution). That's why using a good risk analysis method is very important. That is why in our methodology, those are obtained thanks to the risk analysis methodology provided by CitySCAPE (D2.3). This methodology allows us to provide a complete risk analysis at system level that facilitates the identification of specific threats encountered by multimodal systems (safety, privacy, cascading threats, etc.). Therefore, enforcing the use of this methodology will help to identify efficiently most threats applicable to the system and from them, those threats that apply to a specific element under study for this system.

Threats can be countered usually in many ways. Enforcing an ST, to identify the set of threat considered will note be equivalent to a formal proof of security requirement fulfilment but will help evaluators and the final user to

know against what the product should be protected, and evaluate based on their own expertise, if the security requirements are sufficient to counter those threats.

Thus, we propose the following process to define dedicated security targets for a specific system:

1. Identification by the system owner for each critical component identified during the "assets' assurance requirement identification phase" of other systems **assets** they have or provide access to (using system risk analysis result).
2. Identification by the system owner of each of those critical component threats applicable to these **assets** affecting the component under study.
3. Definition of mandatory assumptions on the operational environment by the systems manager and product developer for the component to work properly.
4. Update those elements after every risk analysis update thanks to operational supervision using RITA and FIMCA engines. to assess the risk and financial impact, respectively.

Introduction of the last step allows our methodology to provide assurance continuity. In fact, continuous operational monitoring of risks and threats allows the system owner to validate their assurance objective continuously and identify when exactly their system components need assurance re-assessment due to other system updates, threats evolutions, underestimated or overestimated threats, etc.

4.2 Evaluation tasks and assurance levels definition

4.2.1 Assurance evaluation for critical components

We cannot aim and do not want to aim at very high assurance levels, such as full EAL 3 or higher CC certification. We are looking for cheaper and more adapted assurance level.

An assurance level is the amount of evidence that is provided for an evaluation of a product to assess the conformity to its security requirements (specification review, quality development processes review, code review, functional testing, vulnerability testing, etc.).

Assurance activities are well covered by the state-of-the-art (cf section 2.4) and we do not need to define new ones. What we do for CitySCAPE assurance framework is to simply identify the appropriate and most efficient activities for the CitySCAPE context. Also, based on the CC assurance requirement, we relax some of the format and thus the following lighter activities are followed:

- Security target evaluation.
- Specification validation.
- Functional tests.

- Vulnerability tests.

4.2.1.1 Assurance activities

The assurance activities we propose are inspired by the CC assurance activities [12] and CSPN activities [2]. CitySCAPE Assurance activities for critical component have the same objectives as CC, but they relax many of the CC constraints in terms of format and evaluation requirements, making the evaluation lighter. This will affect the final assurance level obtained, but we clearly do not aim at having CC equivalent assurance. Otherwise, we would directly recommend CC certification.

We think that relaxing CC structure and expertise requirements as well as limiting involvement of accredited labs will be a good trade-off between assurance cost decreases and finally obtained assurance level.

A simple argumentation about obtained assurance is that we require strictly more evaluation tasks than the CSPN but less structured ones that average EAL 3 or 4 CC evaluations. Thus, providing a good intermediary evaluation process from our point of view.

The following assurance activities are to be run iteratively, as for CC evaluations. The developer provides the required inputs to the identified evaluator of the task. Then the evaluator assesses if those inputs:

- succeeds (if no problem is identified),
- fails (if incomplete or contain identified issues),
- or are inconclusive to pass the evaluation task (mostly due to lack of precision in the inputs).

This process is repeated, and the inputs are updated until they obtain the success result.

Evaluators can find complementary information on how to run assurance tasks evaluation in [26].

4.2.1.1.1 Security target evaluation

Assurance requirements

The core concept of any assurance approach is to evaluate a specific set of security functions of a specific product under clearly identified environmental configuration, to assess that a product meets its security requirement in the identified environment. For that, those parameters have to be explicitly defined. The parameter definition takes the form of what is called a Security Target. Thus, any assurance evaluation process shall start by evaluating that a proper ST has been defined for the evaluation. For the CitySCAPE assurance framework, this document is even more crucial, since it is not only the evaluation specification, but also the supporting mechanism for proper assurance composition (cf. section 3.3) and continuity support (cf. section 3.4).

The CitySCAPE security targets must thus contain the following mandatory elements:

- The commercial **name** and the exact **reference** of the evaluated version.
- Product **developers** name.

- Target of Evaluation (TOE) overview:
  - Description of the **usage and major security features** of the TOE intended to give a very general idea of what the TOE is capable of in terms of security, and what it can be used for in a security context.
  - Many TOEs (notably software TOEs) rely on **additional, non-TOE, hardware, software and/or firmware**. TOE overview will identify such non-TOE hardware, software and/or firmware composing the technical environment in which the product can be executed (required hardware, OS, services such as: time server, AD, mail; Hardware Security Module (HSM), etc.).
- Security problem definition.
  - **Assets** to be protected by the TOE.
  - The **threats** that are to be countered by the TOE, its operational environment, or a combination of the two. A threat consists of an adverse action performed by a threat agent on an asset. These actions influence one or more properties of an asset from which that asset derives its value.
  - **Assumptions** that state requirements on the operational environment in order to be able to provide security functionality. In fact, most (if not all) products need a minimum confidence in their environment to run securely. This confidence is translated in terms of assumptions. If the TOE is placed in an operational environment that does not meet these assumptions, the TOE may not be able to provide all of its security functionality anymore. Assumptions can be made on physical components (e.g., physical protection of physical interfaces of the TOE), personnel (typically trustworthy administrators) and connectivity of the operational environment (e.g., need of an active directory, timestamps server).
- Security requirements:
  - **Security functions** to be implemented by the TOE to counter the threats identified in the security problem.
- Security requirement **rationale** to justify how threats are mitigated by the security requirements.

No standardised format is required. This approach is inspired by ST requirements defined by the French CSPN. Using a more open format will ease the ST definition and evaluation, and thus the global evaluation.

The counter part is that STs might be (slightly) less precise or uniform and their evaluation prone to more evaluator's subjective interpretations. This can also lead to later difficulties in other evaluation task, where those interpretations can influence the evaluation result. The evaluators will be mainly responsible for more interpretation and normalisation efforts over

the different evaluations they conduct to standardize as much as possible assurance results.

Evaluator identification and activity

In common CC evaluations, all assessment activities are done by an evaluator of the chosen accredited evaluation lab. In the CitySCAPE security assurance framework, the evaluation of the ST will be made by the end user (e.g., in our use case, a Tallinn or Genoa security manager).

The evaluation task includes verifying that the ST contains all the mandatory parts, and that the content of those mandatory parts is clear, complete, understandable and contains no inconsistency. A specific care will be taken to review the security requirement rational and verify its completeness.

ST evaluation cannot go much further than completeness and quality review. There is no known scale or identified methodology to assess assets definition, threat identification or security requirement definition correctness. However, since in our methodology the ST review is made by the end user, the end user can assess the matching of the security target with their needs and potentially influence the developer in order to change or adapt the ST and the product to the user needs, if commercial agreement can be found.

4.2.1.1.2 Specification validation

Assurance requirement

The objective of this evaluation task is to verify the existence and the validity of the functional specification of the TOE and its interfaces with respect to the security requirements defined in the ST, i.e., identification of what the CC defines as the TOE Security Functionality Interfaces (TSFI). The TSFI comprises all means by which external entities (or subjects in the TOE but outside of the TSF) supply data to the TSF, receive data from the TSF and invoke services from the TSF. This family provides assurance directly by allowing the evaluator to understand how the TSF meets the claimed SFRs. This evaluation task verifies the proper identification and specification of the different external interfaces of the TOE, as well as the TSFI identification:

- HMI,
- API,
- network interfaces,
- physical interfaces.

For each interface, the security functions accessible through the interface shall be provided.
For each security function, the developer then describes:

- the purpose and SFR enforced (extract from the ST),
- interfaces and exchanged data,
- description of operations,
- logs and Error messages,
- how to configure the function (parameters).

The description must also correspond to the description of the TOE actions described in the ST.

This identification only concerns the implementation of the different TOE functions and, more specifically, the security functions of the product.
This is useful in order to guarantee that the developer can demonstrate that the product really satisfies the ST and that the coverage of the SFR defined in the ST is correct; later on, it helps to make sure that the functions specified in this architecture are in fact implemented in accordance with this specification.

Evaluator identification and activity

In common CC evaluations, all evaluation activities are done by an evaluator of the chosen accredited evaluation lab. In the CitySCAPE security assurance framework, the evaluation of the specification documentation will be made by the end user (e.g., in our use case, a Tallinn or Genoa employee).

The functional description doesn't contain too sensitive information. They mainly help to complete the full tracing proof to get from the SFR defined in the ST to the product function and its interfaces implementing the SFR to be evaluated. The functional description here is at the interface level and must provide the whole interfaces description, including protocol used and interface usage. For each interface, the security functions accessible through the interface shall be provided.
The main parameter to be verified is the exhaustivity of the description, the clear identification of the TSFI.

4.2.1.1.3 Functional tests

Assurance requirements

Here, we propose to verify that the TOE and its TSF behave as described in the ST and the functional specification provided for the evaluation. It does not contain any penetration testing, only functional testing.
On one hand the developer shall demonstrate that he/she has sufficiently tested his/her product, and on the other end, the evaluator shall verify the proofs provided and repeat some of the developer tests and possibly add independent testing when deemed appropriate (i.e., too few developer tests, not tested parameters or function, etc.).
The developer must provide its test plan, including test scenarios or test scripts. For each scenario, the developer shall describe the prerequisite, operations and expected results.
The test results, as executed by the developer for the TOE, must be provided to verify their validity and the validity of their description.
Then the second evaluation activity consists of tasks to prove that all TSFI are covered by tests, tasks to verify that all TSFI have tested and that they have been tested deeply enough, i.e., testing the correctness of TOE's internal functions and interactions (as far as the TOE specification allows to judge it). For these evaluation tasks, there is no expectation of exhaustive nor complete test coverage of every possible TOE behaviour, only sufficient coverage of all TSFI.

Evaluator identification and activity

Once again, it is proposed to have these evaluation tasks done by the end user. This is interesting for two reasons, (i) first for cost reduction, since the end user integration team tend to be less expensive and because during integration, functional tests must be run anyway, (ii) because this will force the end user to test and review all TOE security functionalities, and thus help to guarantee an efficient use of those functions.
The evaluator shall confirm that the information provided meets all requirements identified below for content and presentation of evidence.
First, the evaluator must determine that the pre-requisites are complete and appropriate, i.e., that they will not bias the observed results towards the expected test results.
Then, the evaluator must determine that the test steps and expected results are consistent with the descriptions of the TSFI in the functional specification.
Finally, they determine that each TSFI has been sufficiently tested against the behavioural claims in the functional specification.

4.2.1.1.4 Vulnerability analysis

The goal of this task is to identify potential vulnerabilities using all information gained during the evaluation and to test the exploitation of these potential vulnerabilities for an attacker with different resources.
The main input for this task is the TOE. The output is a vulnerability analysis report, listing the potential vulnerabilities tested with the corresponding potential attack, when and if the vulnerability is exploitable.
The vulnerability analysis is usually done compared to a certain attacker level. The CC provides a way to evaluate to which attacker level a vulnerability exploitation corresponds [26]. If, during the test, an exploitable vulnerability has been found, but the corresponding required attacker level is higher than the one considered for the evaluation, the vulnerability is considered as residual. In the CitySCAPE context, we do not recommend using an attacker level, but rather the evaluator minimum resources, mostly in terms of spent days to run the tests as for the CSPN [2].

Evaluator identification and activity

The report can contain very sensitive results and very technical skills are required for this activity. Thus, we require this task to be done by independent security experts. Most preferably CC accredited ITSEF or well-recognized entities with equivalent competences.
From our point of view, the best reference to use on how to perform a vulnerability analysis in the context of an assurance methodology is [26]. We summarize here the specific approach.

The evaluator will examine the sources of information publicly to identify potential vulnerabilities in the TOE, based on known vulnerabilities for related products, products of the same developers, in libraries used by the TOE, related to the technologies used, etc.

There are many sources of publicly available information, which should be considered, e.g., mailing lists, security forums, Common Vulnerability Enumeration (CVE) data bases, conference presentation or articles on the world wide web that report known vulnerabilities in specified technologies, etc. In the context of CitySCAPE, we also highly recommend using CERT and CSIRT vulnerability knowledge input to identify potential public vulnerabilities. This might require for the evaluator to go through the end-user CERT and CSIRT contact, but clearly, they are important knowledge sources for vulnerability tests.

The accessibility of vulnerability information and attack tools shall be used to identify potential vulnerabilities in the TOE and exploit them. Regular search tools make such information easily available to the evaluator, and well-known generic attacks can be achieved in a cost-effective manner.

The evaluator shall devise penetration tests based on the independent search for potential vulnerabilities. The evaluator prepares for penetration testing as necessary to determine the susceptibility of the TOE, in its operational environment, or towards the potential vulnerabilities identified during the search of the sources of information publicly available.

As requested by [26], the evaluator reports shall provide the following detail to enable the tests to be repeatable:

a) identification of the potential vulnerability the TOE is being tested for;
b) instructions to connect and setup all required test equipment as required to conduct the penetration test;
c) instructions to establish all penetration test prerequisite initial conditions;
d) instructions to stimulate the TSF;
e) instructions for observing the behaviour of the TSF;
f) descriptions of all expected results and the necessary analysis to be performed on the observed behaviour for comparison against expected results;
g) instructions to conclude the test and establish the necessary post-test state for the TOE.

4.2.2 Assurance for noncritical components

Assurance for noncritical components will be addressed indirectly.

In fact, we have already identified at this stage of our evaluation process that noncritical components are: either mostly protected by other security critical components (firewalls, IDS/IPS, SIEM, etc.); are not sufficiently exposed to be endangered; or they are simply not critical enough for the system's good operation.

In this context, we have identified that we do not really need to assess their security (or we cannot attempt it and only best efforts can be done).

However, we have identified risks and threats applying to them and we should verify at the lowest cost possible that:

- if an unidentified critical risk was missed for those components, then updating of the risk analysis and the "assets' assurance requirement identification" consequently is performed.

- those are not materialized.

To do that, we fully rely on CitySCAPE security solution developed by the project:

- IDS/IPS
- And SIEM

Those elements will be used to monitor for threat realization on those elements. In fact, if detection mechanisms are used to verify that security requirement are not violated, then, it is an indirect proof that they are fulfilled.

The idea is for the security managers together with tools manager to define dedicated detection rules whenever possible (depending on threat precision, known associated vulnerabilities, threat consequences formalization, etc.) to operationally verify security requirements violation.

5 CITYSCAPE ASSURANCE TOOLS

In this section, we discuss the requirements and try to identify tools for two different topics concerning assurance evaluation for critical components (cf section 4.2.1):

- Development of assurance evidence input.
- Functional and vulnerability tests.

In the first section, we identify tool requirements that could ease assurance assessment and thus lower the costs of assurance evaluation. In fact, providing assurance evidence is not always trivial for developers. Dedicated tools could help developer in providing them examples or document templates that would speed up their documentation developments and increase their documentation quality. Implementation of these tools is out of scope, but implementation of such tools would greatly enhance our methodology. Thus, we provide these requirements as potential development for further enhancement of CitySCAPE solution.

Tools for functional and vulnerability tests, if existed, usually speed up tests execution significantly and allow higher test coverage, whereas identification of those tools is not so easy for none-domain experts. That's why we provide these reference lists.

5.1 Assurance inputs developments

5.1.1 Risk analysis update

One main contribution of CitySCAPE assurance methodology is the use of risk analysis operational update to define STs input (assets and threats).

This update is realized thanks to feedbacks provided by two project partners:

- RITA, which will be developed by ED and UPRC.
- FIMCA, which will be developed by ENG and STAM.

Those tools are fully specified in D3.2 and D3.3.

5.1.2 Security targets definition

The state-of-the-art has already studied the importance of implementing supporting tools for assurance evaluations ([27], [28]) and more specifically STs ([29], [30]).

Writing security targets is difficult, even in a context such as the CitySCAPE methodology, where the ST is only required to provide roughly specified elements, i.e.:

- The commercial **name** and the exact **reference** of the evaluated version.
- Product **developers** name.
- TOE overview.
    - Description of the **usage and major security features**.
    - **Additional, non-TOE, hardware, software and/or firmware**. composing the technical.
- Security problem definition.
    - **Assets**.
    - **Threats**.

- **Assumptions**.
- Security requirements.
- Security requirement **rational**.

If we intentionally choose not to require formal and standardized structure or description of these elements, the level of freedom is left to the ST writer and this fact might complicate their ST writing tasks.

A simple tool predefining the document structure (e.g., by providing specific interface fields to be field in) and providing examples or de facto standards of the different elements (e.g., assumption of trustworthy administrators, user identification and authentication mechanisms, etc.) would greatly speed up ST definition.

Additionally, support for preparing the requirement rationale, simply by helping the ST writer to select threats and requirements from the ones specified in the ST and identify unjustified requirements (identification of threat or security requirement not yet present in any coverage rational) would be a simple way to provide significant help.

5.1.3 Functional specification

The functional specification document to be provided for our evaluations, have to contain very specific elements that are not normally present in regular developers' documentation.
Thus, in order to help developers to take into account evaluation requirements, a redaction tool (simple web interface) taking as input the ST defined for the product evaluation, could automatically verify that all SFRs present in the ST are linked to at least one TSFI.
Further verification could be done automatically, such as user roles definition and identification, cryptographic algorithms consistency, etc.

5.2 Functional and vulnerability tests tools

Tools required to test security functions of multi-modal transportation and ITS elements and systems are fully dependant of the specific details of their implementation e.g., API, HMI, technologies including type of code (C, C++, Python, SQLlite etc.), database engines (MySQL, SQL lite, etc.), drivers, communication technologies and protocols (IP stacks, Bluetooth BLE, WIFI, etc.).
Thus, to test either functionally and/or make vulnerability tests, one can rely on already existing and most widely used testing tools.
Let's note that the state-of-the-art does not provide a benchmark or detailed characterisation study of security vulnerability testing tools. Experts' or users' opinions may be found, but they are all subject to unknown subjective analysis that depends on the person's interest, expertise, habits, subjective "look and feel" experience or again very specific testing cases feedback, etc. Also, most of the arguments we could provide could be biased in the same. Thus, providing opinions based on too specific usage that may not apply to other tests and thus not relevant in the present document.

Also, it greatly depends on budget and licence cost the tester can afford, which may vary over time or usage.
Thus, we only provide here a list of known relevant tools. By no means this list is meant to be exhaustive since, among other things, commercial tools are not all publicly advertised. For instance, we did not manage to identify testing tools for other technologies than road vehicle (i.e., train technologies, plan technologies, etc.), however it is possible that non-commercial tools may exist.
This list will need to be updated regularly.
However, here, we propose a useful reference list with links in order for the reader to find the tools or their documentation. This is done in order to avoid paraphrasing or poor summaries – knowing that tool capabilities identified by developers might not be efficient in many contexts and thus have to be reviewed for each specific evaluation:

- Scanning tools
  - Nmap, https://nmap.org/
  - Masscan, https://github.com/robertdavidgraham/masscan
  - Nexpose, https://www.rapid7.com/products/nexpose/
  - Nessus, https://www.tenable.com/products/nessus/nessus-professional
  - Dirbuster, https://sourceforge.net/projects/dirbuster/
  - Dirb, http://dirb.sourceforge.net/
  - Findsploit, https://github.com/1N3/Findsploit
  - Sslyze, https://github.com/iSECPartners/sslyze
  - Detectify, https://detectify.com/
- Vulnerability tests and exploitation tools
  - Web
    - Arachni, http://www.arachni-scanner.com/
    - Burp, https://portswigger.net/burp/
    - Parameth, https://github.com/mak-/parameth
  - Fuzzing and brute force
    - Wfuzz, https://github.com/xmendez/wfuzz/
    - Patator, https://github.com/lanjelot/patator
    - Hydra, https://github.com/vanhauser-thc/thc-hydra
  - Peach, https://www.peach.tech/
- Network analysis, interception and packet manipulation
  - IP networks
    - Wireshark , https://www.wireshark.org/
    - Tcpdump, http://www.tcpdump.org/
    - Netzob, https://github.com/netzob/netzob
    - Ettercap, http://www.ettercap-project.org/ettercap/
    - Dsniff, https://www.monkey.org/~dugsong/dsniff/
    - Netsed, http://silicone.homelinux.org/projects/netsed/
    - Scapy, https://scapy.net/
    - Packet Sender, https://packetsender.com/

        - Haka, http://www.haka-security.org/download/haka.html
    - CAN bus
        - CANoe, https://www.vector.com/int/en/products/products-a-z/software/canoe/
        - CANalyser https://www.isit.fr/fr/produit/ixxat-cananalyser.php
        - Can4linux https://gitlab.com/hjoertel/can4linux
        - Can-utils https://elinux.org/Can-utils
        - Slcan https://github.com/topics/slcan
        - UDSim https://github.com/zombieCraig/UDSim/
        - CANalyzat0r http://github.com/schutzwerk/CANalyzat0r
        - Vehicle Spy Entreprise https://intrepidcs.com/products/software/vehicle-spy/
- Exploit
    - Metasploit, https://www.metasploit.com/
    - Canvas, https://www.immunityinc.com/products/canvas/
    - Core impact, https://www.coresecurity.com/core-impact
- Radio
    - GNU Radio, https://gnuradio.org/
- Conformity
    - ETSI
        - Java Implementation of ITS Intelligent Transport Systems (ITS) Security header and certificate formats http://pvendil.github.io/c2c-common/
        - Titan https://forge.etsi.org/rep/ITS/ITS/tree/STF525

For everything else, i.e., all tests for which those tools cannot provide any support, which is generally more than 50% of vulnerability tests, dedicated tools or code, has to be developed. In these cases, the most common programming languages should be used: C, C++, java, python, bash, etc.

6 CONCLUSION

The CitySCAPE assurance methodology provided in this deliverable is an innovative solution that enables:

- Cheaper assurance:
    - Using the subset of CC full evaluation (which allows us to maintain good assurance level) but with lesser formalization.
    - Not relying on formal certification scheme which suppresses administrative burdens and using parallelized and distributed evaluation tasks that involve (expensive) accredited labs - further lowering costs and evaluation duration.
    - Specification of evaluation enhancing tools.
- Covers new and specific assurance needs:
    - Assurance at system level by introducing the concept of security assurance composition by security problem decomposition, which allows us to consider cascading threats thanks to CitySCAPE risk analysis method.
    - Includes operational and assurance continuity thanks to continuous monitoring of system security properties and threat landscape evolution.

Moreover, we have identified existing tools that can be used as a reference list to ease functional and security tests of high assurance activities. We have also provided some potential tools specification that could also help developers to produce more easily higher quality evaluation inputs.

Executing our assurance methodology as well as developing assurance enhancement tools, is out of scope of the project. In fact, security assurance is an activity that suite systems with high level of maturity (TRL 9 mostly), which is not the case of the project developments and use cases. So, it would not be fully efficient to try to run it, but it could greatly help real CitySCAPE deployments after the project, since it would provide strong evidence to systems owners that their system is secured as they intend it to be, which is valuable to help solution adoption. Also, the development of tools following our specification could further enhance the project results and further ease its adoption.

7 REFERENCES

[1] ISO/IEC, *15408-1 Information technology – Security techniques – Evaluation criteria for IT security - Part 1: Introduction and general model,* 2009.

[2] ANSSI, *First Level Security Certification For Information,* April 7th, 2014, 2014.

[3] NIST, *FIPS PUB 140-2 - Security Requirements for Cryptographique Modules,* 2002.

[4] ENISA, Cybersecurity Certification: Candidate EUCC Scheme, https://www.enisa.europa.eu/publications/cybersecurity-certification-eucc-candidate-scheme.

[5] B. B. K. G. a. T. W. N. Bartol, "Measuring Cyber Security and Information Assurance: a State-of-the-Art Report," *Information Assurance Technology Analysis Center (IATAC),* 2009.

[6] I. Freiling, Dependability Metrics, Springer, vol. 4909, 2008.

[7] L. r. Stéphane Paul, "Over 20 years of research into cybersecurity and safety engineering: a short bibliography," WIT Transactions on the Built Environment Vol 151., May 2015.

[8] B. K. L. O. N. P. K. Y. G. K. W. Ankur Shukla, "System Security Assurance: A Systematic Literature Review," arXiv:2110.01904, 2021.

[9] J.-P. S. R. S. Mazen Mohamad, "Security assurance cases—state of the art of an emerging approach," Empirical Software Engineering, 2021.

[10] SOG-IS, *ITSEC: Information Technology Security Evaluation Criteria.*

[11] ISO/IEC, "15408-2 Information technology – Security techniques – Evaluation criteria for IT security - Part 2: Security functional requirements.," 2008.

[12] ISO/IEC, "15408-3 Information technology – Security techniques – Evaluation criteria for IT security - Part 3: Security assurance requirements," 2008.

[13] ISO/IEC., " 27001:2005 – Information technology – Security techniques – Information security management systems – Requirements.," 2005.

[14] U. D. o. Defense, *DoD Instruction (DoDI) 8510.01, Risk Management Framework (RMF) for DoD Information Technology (IT).*

[15] U. D. o. defense, *Trusted Computer System Evaluation Criteria (TCSEC), DoD 5200.28-STD,* l985.

[16] ISO/IEC, *27004 : Information technology – Security techniques – Information security management – Measurement..*

[17] ISO/IEC, *27005: Information technology – Security techniques – Information security risk management..*

[18] R. H. a. A. S. R. Vaughn, "Information assurance measures and metrics-state of practice and proposed taxonomy," *Proceedings of Hawaii International Conference on System Sciences, vol. 1., Citeseer,* 2003.

[19] P. P. M. A. L. B. N. J. a. H. A. S. Nabil, "Current trends and advances in information assurance metricsFredericton," *Proceedings of Second*

Annual Conference on Privacy, Security, and Trust (PST 2004), NB, Canada, p. 1315, 2004.

[20] A. Jaquith, Security metrics, replacing fear, uncertainty, and doubt, MA: Addison-Wesley Reading, 2007.

[21] N. B. J. S. J. H. a. L. G. Marianne M Swanson, "Security metrics guide for information technology systems.," Technical report, 2003.

[22] J. P. a. J. W. M. Howard, "Measuring relative attack surfaces," *Computer Security in the 21st Century Eds. Springer US,* p. pp. 109–137., 2005.

[23] S. Payne, A guide to security metrics, SANS institute, 2001.

[24] J. McHugh, "Quality of protection: measuring the unmeasurable?," in *Proceedings of the 2nd ACM workshop on Quality of protection*, New York, NY, USA, , 2006.

[25]]. S. Bellovin, "On the brittleness of software and the infeasibility of security metrics," 2006.

[26] ISO/IEC, "18045 Information technology — Security techniques — Methodology for IT security evaluation," 2008.

[27] S. F. G. G. a. E. W. Andreas Ekclhart, "Ontological mapping of common criteria's security assurance requirements.," In IFIP International Information Security Conference. Springer, 85–95., 2007.

[28] D. Rogowski, "Software implementation of Common Criteria related design patterns," *Federated Conference on Computer Science and Information Systems (FedCSIS),* 2013.

[29] K. Y. N. A. Y. G. J. C. Daisuke Horie, "GEST: A generator of ISO/IEC 15408 security target," *Computer and Information Science,* 2009.

[30] P. P. S. H. A. A. Angelos Stamou, "Enabling Efficient Common Criteria Security Evaluation for Connected Vehicles," *Conference: IEEE International Conference on Cyber Security and Resilience (CSR) At: Virtual Conference,* 2021.

[31] ANSSI, Security Certificationof Products, https://www.ssi.gouv.fr/uploads/2018/01/security-certification-of-products_security_visa_anssi.pdf.