Journal of Engineering and Applied Sciences 14 (10): 3482-3491, 2019
ISSN: 1816-949X
© Medwell Journals, 2019

Applying a Mathematical Approach to Interpreting the Results of Testing Software and Hardware Data Security Tools During the Verification Process

Andrey M. Kanner and Tatiana M. Kanner
National Research Nuclear University (MEPhI), 31 Kashirskoe Highway, 115409 Moscow, Russia

Abstract: The study is devoted to the issues of applying existent mathematical apparatus from optimization and decision making theories for verification of software and hardware Data Security Tools (DST). The study contains an algorithm which can be used to verify such DST and which presupposes formal assessment of criticality of the errors found during testing in the security functions of such security tools and algorithms used to calculate the total error criticality before the DST is integrated into an information system. The study also describes software implementation of the proposed verification algorithm which represents a decision support system making it possible to automatically evaluate criticality of the errors found in the security tools. Presented software implementation takes into account not only criticality of the errors and their number but also criticality of the security functions in which such errors were detected during testing of the software and hardware DST. In case of taking into account the criticality of the security functions of DST the new level of individual criteria is added to the hierarchical structure of the decision making task for which criteria with errors will be considered as embedded local criteria.

Key words: Software and hardware DST verification, optimization and decision making theories, AHP, SAW, verification algorithm of software and hardware DST, program “verifier of software and hardware DST”, verification of access control subsystem

INTRODUCTION

Implementation of security functions in software and hardware Data Security Tools (DST) includes the following items (Kanner, 2014):

- Design (including establishing requirements for the DST security functions)
- Direct development of software and hardware components of the DST (including its internal software) implementing its security functions
- Testing of the above components
- Verification based on the test results
- Correction of errors found during testing
- Finalization, that is preparation of the DST for release which consists of fixing versions of internal and external DST Software, describing measures to compensate the errors left based on the results of the DST verification, describing features of the DST functioning, preparing the necessary documentation, etc.
- Release of the DST

The design phase is associated with the development of requirements to the security functions of the software and hardware DST. Requirements to a specific DST are formed based on the end user’s tasks, using the requirements of data protection regulatory authorities to the class of data protection tools, to which it belongs. After that, the software and hardware components of the DST are developed which implement the required security functions. Regardless of the type of the software and hardware security tool (Kanne, 2016a), prior to finalization and release, its security functions shall be checked for compliance with the requirements. The compliance is checked not only at the moment of first release of the DST but also at the moment of subsequent updates of its versions. To do this, the security functions of the software and hardware DST are tested and on the basis of such tests the DST is verified including classification of errors and features found during testing and a decision is made about success of the testing or a need to correct such errors (Kanner, 2014). Thus, all the above stages, except for the first one have to be repeated several times during the life cycle of the software and hardware DST. However, the most attention shall be paid to verification as a fundamental stage on which the release of the software and hardware DST depends.

Corresponding Author: Andrey M. Kanner, National Research Nuclear University (MEPhI), 31 Kashirskoe Highway, 115409 Moscow, Russia

In general terms, verification means a confirmation that the manufactured product complies with the requirements imposed thereon (Kulyamin, 2008). The decision on the results of verification is taken on the basis of a comprehensive testing of the software and hardware DST because this is testing that leads to identification of all errors and shortcomings in its functioning, concerning both the ease of use and malfunction and analysis of the results allows to conclude that the security functions of the DST being tested correspond to the stated requirements. Thus, the test results are a basis for the verification process and verification results can lead to the need of a new test that is, these two processes influence each other.

The sequence of actions when testing and verifying the DST is as follows (Kanner and Sultanahmedov, 2014):

Testing:

- Identifying incorrect behavior of the DST which demonstrate presence of errors (including in the security functions)
- Recording the manifestation of errors
- Localization of the errors manifestations-search for other manifestations and relationships
- Analysis of localized errors manifestations
- Recording errors and features

Verification:

- Classification of errors and features: establishing the type, the compensation possibility, the degree of criticality
- Making a decision on the results of verification and on the possibility of finalizing the DST or returning for revision with subsequent retesting

In the process of testing software and hardware DPT, Programs and Test Methods (PTM) are used which contain specific sequences of actions and descriptions of expected results, taking into account the peculiarities of functioning of a specific DST. The PTM are developed in such a way as to cover all the product functionality and to get the most complete picture of its performance. The object being tested can be in different initial conditions (different operating systems, different versions of hardware components, etc.) which are input data for testing. Testing using the PTM is performed separately for each such set of conditions. On the basis of each test cycle performed, a table of results is compiled containing a list of all identified errors. After that, the tester conducts localization of errors by examining their identified manifestations and analyzing possible relationships with previously encountered incorrect product behavior, if any. Based on the data received from the tester, the developer establishes the error and informs the tester what functions the detected error may affect. In its turn, the tester checks whether this error affects the specified functions of the product and then compiles the final list of the found errors and features (Kanner and Sultanahmedov, 2014).

Based on the list of the found errors and features, it is necessary to perform their classification including determination of the error type, the possibility of its compensation as well as the degree of its criticality (Kanner and Sultanahmedov, 2014). As a result of all the testing cycles on different sets of input data, it is necessary to create a final verification table of the software and hardware DST which is used to analyze performance of the product as a whole and to decide on successful completion of testing and verification.

Some errors found during the testing process can be corrected quickly enough before the verification is completed. Taking into account that each correction made to one of the modules of the software or hardware component of the DST may entail a change in the operation of other modules, it is necessary to retest the corrected product using all the PTM. Thus, the constant change of the tested and verified object can lead to confusion in the results and to the so-called “endless testing” which in its turn will lead to a delay in the release of the DST. To prevent entry into the infinite testing cycle, you shall either decide to make corrections to the next version and complete the verification of the current version with a verdict on its release or stop testing and verifying the current version and immediately start testing the new one which shall be the most updated current version of the software and hardware DST and its security functions.

For the DST including hardware and software ones, the verification process has some peculiarities in terms of analyzing the impact of errors detected during implementation of the security functions on the system security in which such tools are used (Kanner and Sultanahmedov, 2014). Such peculiarities shall be taken into account in the program and test methods, testing of the DST security functions cannot be considered complete without them.

The errors identified during testing of the DST security functions should be attributed to one of the following types: minor typos and errors (for example, a typo in the displayed message or an incorrect name of any function) that do not affect the correctness of the security functions, errors leading to inoperability of one or more of the DST security functions, errors leading to inoperability or damage to the system security in which the DST is used.

Such types of errors are unequal from the point of view of the DST operation, therefore, a certain gradation of all the found errors is needed in terms of their impact on the performance of the main task-protection of information resources and security.

The scale of error criticality is dependent and as a rule is intended for internal use by the company producing the specific data protection tool. At the same time, the correctness of assessing the error criticality is a fundamental factor in deciding on the possibility of finalizing and issuing the DST.

Let us consider an example of the criticality scale of the errors found when testing the security functions of the software and hardware DST. The errors can be divided into several types (Kanner and Sultanahmedov, 2014).

Interface errors: These include faults in the convenience of the user interface, correctness of the display of all its elements, typos in system messages and the like. Such errors do not affect the performance of the DST security functions, functionality and security of the system in which the DST is applied. Their correction is necessary to ensure comfortable work of the end user. The presence of such defects is not dangerous for the protected system and they are assigned a minimum level of criticality.

Errors that limit the DST functionality without damaging its security functions: Such type of errors imposes some restrictions on the DST functionality while not endangering the protected system. In this case, either improperly functioning DST capabilities are not directly responsible for the security of the protected system or the lack of these functions can be compensated by applying other means and measures without compromising the security level.

The errors related to compromising of the DST security features: This type includes errors that can affect the system or data security due to a damage to the DST security features which creates prerequisites for successful implementation of an attack using the resulting vulnerability. They are the most critical and presence of even one error of this type can lead to verification resulting in prohibition of product finalization and release, except in cases where such damage can be compensated by additional means and measures for example by adjusting the OS.

It shall be noted that the tester is not always able to determine the error type based on the detected manifestations thereof without participation of the developer. Therefore, involvement of the developer in the analysis of the detected error manifestations is a prerequisite without which it is impossible to accurately eliminate the errors and classify them correctly. For example, if an error found by the tester is that when checking the Digital Signature (DS) of a file that has been modified after it has been written, a message on the DS correctness appears, then this may be a second type error when the function of generating verification results messages works incorrectly. However, such a manifestation may also be caused by the situation when the function of checking the DS does not work correctly. In this case, it will already be a third type error which just may lead to a breach of the system security and a possibility of any attack from a potential intruder. In the described case, the tester alone will not be able to figure out exactly the essence of the arising error manifestation and the developer shall be involved to analyze and eliminate it.

After classifying the errors and features as well as analyzing the results obtained, it is necessary to summarize whether all the requirements for the DST security functions are met or there are critical errors that lead to their damage (even taking into account compensatory measures) and do not allow to make a decision about the beginning of the finalization stage. If there are no such errors, then, a decision is made to release the DST, otherwise, the DST shall be reviewed, re-tested and re-verified.

It should be noted that the decision on the DST release (or the prohibition of the release) on the basis of the testing results is generally taken quite informally for example, only on the basis of presence/absence of critical DST errors. Such an approach cannot guarantee a qualitative assessment of the DST performance and its security functions, therefore, it is necessary to use other methods that allow a more accurate assessment to be made to make the final decision.

MATERIALS AND METHODS

To decide on the possibility of releasing the DST, it is proposed to use the methods mentioned in optimization and decision making theories in particular Analytic Hierarchy Process, AHP in which this task can be considered as a task of making a decision in the context of absolute certainty (presence of a certain type of errors leads to a “deterioration” of the DST to some known extent). For example, to solve this task you can use a well-known simple choice algorithm (Simple Additive Weighting or SAW, known as weighted sum method) based on the methods of Saaty (1977, 1980), Cogger and Yu (1985), Takeda *et al.* (1987), Chernorutsky (2001) and Kim and Weck (2006), evaluating the DST (hereinafter referred to as the “alternatives”) according to the evaluation criteria, taking into account the "importance" of each of them while calculating a certain amount the value of the generalized evaluation criterion.

The task can be solved with the help of such a mathematical apparatus due to the fact that (Kanner, 2018).

The described task is a task of a multi-criteria decision making in the context of certainty with a small number of criteria and alternatives.

You may use both qualitative (presence/absence of an error) and quantitative characteristics (number of errors of various types) as evaluation criteria that is the objective of the task can be changed without changing the apparatus used, if necessary.

When recording errors during the testing, you may always identify the necessary additional information (the number of errors, the degree of their importance and so on).

It is important to get an unambiguous answer for testing and verification if this is possible to release the DST (for example, knowing some acceptable margin of error criticality in the DST) and if such methods are most suitable in this case while others can yield many optimal answers, the so-called Pareto set (Lin, 1976; Chernorutsky, 2001; Kim and Weck, 2006).

We apply the simple choice algorithm based on the methods of Saaty (1977, 1980) and Cogger and Yu (1985) to solve the task of making a decision about the possibility of releasing the DST. This algorithm assumes that we shall first construct a weight vector (weighting coefficients) for the evaluation criteria (in our case, these are the criteria f_1-f_3 presence of errors of three levels of criticality):

$$\alpha = (\alpha_1, \alpha_2, \alpha_3)$$

having the property of normalization:

$$\sum \alpha_i = 1, i = 1, 2, 3$$

To do this, according to a predetermined criticality scale, the relationship of pairwise superiority of evaluation criteria between each other shall be first determined:

$$\alpha_{12} = \alpha_1/\alpha_2$$
$$\alpha_{13} = \alpha_1/\alpha_3$$
$$\alpha_{23} = \alpha_2/\alpha_3$$

Thus, a transition from qualitative characteristics of the criteria to quantitative ones is made and then the values of weighting coefficients α_1 and α_3 are determined by solving a linear equation obtained from the normalization condition.

After that, the values of the individual optimality criteria corresponding to the alternatives are calculated in the same way:

$$\alpha^{(1)} = (f_1(alt_1), ..., f_1(alt_k))$$
$$\alpha^{(2)} = (f_2(alt_1), ..., f_2(alt_k))$$
$$\alpha^{(3)} = (f_3(alt_1), ..., f_3(alt_k))$$

where, alt_1, ..., alt_k, $k \in N$ are the evaluated alternatives (DST). That is, on the basis of qualitative and quantitative characteristics of the alternatives, a transition is made to the quantitative relationship:

$$\alpha^1_{12} = \alpha^1_1/\alpha^1_2$$
$$\cdots$$
$$\alpha^1_{k-1k} = \alpha^1_{k-1}/\alpha^1_k$$
$$\alpha^2_{12} = \alpha^2_1/\alpha^2_2$$
$$\cdots$$
$$\alpha^1_{k-1k} = \alpha^2_{k-1}/\alpha^2_k$$
$$\alpha^3_{12} = \alpha^3_1/\alpha^3_2$$
$$\cdots$$
$$\alpha^3_{k-1k} = \alpha^3_{k-1}/\alpha^3_k$$

Then, taking into account the normalization condition of vectors $\alpha^{(1)}$-$\alpha^{(3)}$, the system of linear equations is solved and the values are determined the weighting coefficients of significance of one or another type of errors in a particular alternative:

$$f_1(alt_1)$$
$$f_2(alt_1)$$
$$f_3(alt_1)$$
$$\cdots$$
$$f_1(alt_k)$$
$$f_2(alt_k)$$
$$f_3(alt_k)$$

To assess the criticality of errors in alternatives, it is necessary to calculate and compare the values of the generalized criteria using the following linear convolution formula (Adamcsek, 2008):

$$J(alt_i) = \sum \alpha_j f_j(alt_i), j = 1, 2, 3, i = 1, ..., k, k \in N$$

A higher value of the generalized criterion corresponds to presence of larger critical errors in one or another alternative, so, the task is reduced to minimizing the value of the generalized criterion.

Using the described mathematical apparatus, it is possible to compare different versions of the same DST to each other or use a specific version as a standard and release the DST only if it possesses indicators which are close to or exceed those of the standard one (however, this evaluation may always be inaccurate due to presence of the DST errors not detected during testing). In addition, it is possible to introduce a new level of the evaluation criteria hierarchy-criticality of the security functions themselves in which errors of one or another criticality level have been revealed, the former mathematical apparatus and similar calculations can be used to solve such a task.

Based on the above, it is possible to suggest an algorithm for verifying software and hardware DST implementing security functions, based on the classification of detected errors, analysis of their criticality and impact on the system security or data which involves the following steps:

Formation of an error criticality scale for the hardware and software DST being verified (for example in accordance with the above proposed division of errors into types: insignificant and not affecting the correctness of performing security functions, those leading to inoperability of one or several security functions, those leading to inoperability or damage to the system security in which the DST is used).

Identification and fixation of computable manual or automatic checks of security functions (Kanner, 2016a) whose completion result is negative.

Identification and fixation of the remaining manual or automatic checks of security functions of those that are not computable as a result of a negative completion of the checks under point 2.

Classification of the errors resulting from negative completion of manual and automatic checks under points 2 and 3 in accordance with the selected error criticality scale under point 1.

Deciding on the possibility of successful test completion or a need to correct the identified errors using the provisions of the optimization and decision making theory (for example, a simple choice algorithm based on the methods of Saaty, Cogger and Yu) or on the basis of other evaluations (Saaty, 1977, 1980; Cogger and Yu, 1985; Takeda *et al.*, 1987; Chernorutsky, 2001; Kim and Weck, 2006).

The testing including points 1-5 shall be carried out for a fixed version of the software and hardware DST (software and hardware components) and in case of making any changes in the DST or correction of the detected errors before completion of the verification process, you shall start again from testing and then go to point 2.

The block diagram of the proposed verification algorithm is shown in Fig. 1.

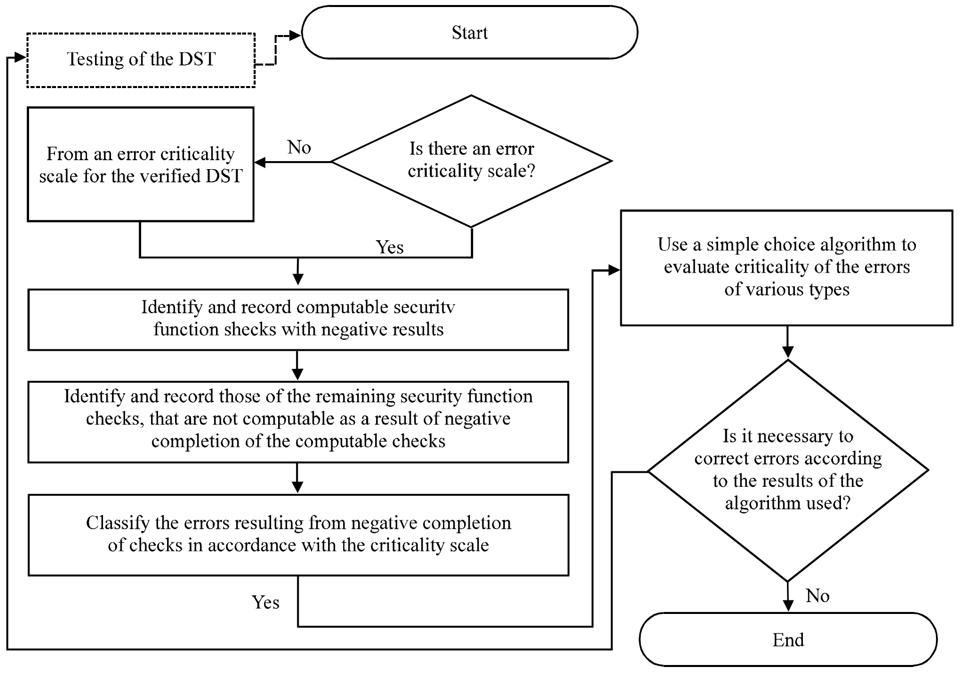

Start
Testing of the DST
Is there an error criticality scale?
No
From an error criticality scale for the verified DST
Yes
Identify and record computable securitv function shecks with negative results
Identify and record those of the remaining security function checks, that are not computable as a result of negative completion of the computable checks
Classify the errors resulting from negative completion of checks in accordance with the criticality scale
Yes
Use a simple choice algorithm to evaluate criticality of the errors of various types
Is it necessary to correct errors according to the results of the algorithm used?
No
End

Fig. 1: Block diagram of the verification algorithm for the software and hardware DST implementing security functions which is based on classification of the detected errors, analysis of their criticality level and impact on the system and data security

The proposed algorithm has a wide purpose and is applicable not only to all types of software and hardware DST (Kanner, 2016a, b) but also to software security tools. However, this algorithm is irrelevant for software or any other software and hardware tools as it considers error criticality of security functions which are absent in any other tools that are not intended to protect information. This algorithm allows to evaluate the criticality of errors that occur when performing manual and automatic checks of criteria-computable security functions (Kanner, 2016a, b) that is in cases where such checks are computable but the result of their completion is negative, analyze the results obtained and the impact of errors on the system security and based thereon decide on successful completion of testing or on return of the DST for revision. Before applying the proposed verification algorithm, it is necessary to test the software and hardware DST for example, on different hardware platforms using virtualization tools and the testing algorithm proposed in (Kanner, 2015, 2017) and/or auxiliary testing tools and additional hardware equipment.

The advantage of this algorithm is that in contrast to the probabilistic approach to risk management and reliability evaluation of information systems (Drobotun, 2009) adopted in well-known papers and regulatory documents, this algorithm offers a deterministic approach that is most suitable for verifying software and software-hardware DST. Verification of the DST shall be carried out immediately prior to their integration into the Information System (IS) and not the probability of the IS failure or its frequency shall be determined but expediency of correcting specific errors that have already been identified and may appear during the system operation and affect its security immediately prior to application of security tools. This approach is a preventive measure to ensure reliability and eliminates deterioration of the IS security even prior to integration of the DST as well as to organize regression testing of the security tools themselves.

The proposed algorithm for verifying the software and hardware DST allows a more formal evaluation of the error criticality that occur when implementing security functions that are computable by criteria found during checks (Kanner, 2016a) to analyze the results obtained and the extent to which the errors affect the system security. Moreover, on the basis of the analysis carried out using the provisions of the optimization and decision-making theory, a decision is made on successful completion of testing or the need to correct the errors found.

RESULTS AND DISCUSSION

On the basis of the proposed algorithm for verifying the software and hardware DST, software has been developed that provides a possibility to automatically solve the task of assessing criticality of the errors detected during error testing in security functions the program called “Verifier of software and hardware DST”.

The results of testing the security functions of the software and hardware DST in a predefined format shall serve as an input for the verification program. Upon receipt of the input data, the program “Verifier of software and hardware DST” allows to evaluate criticality of the errors detected during testing and the degree of their influence on the information system security. In this case, the input data can be set either manually (determining criticality of all the detected errors) or the results of the testing of the DST security functions described in (Kanner, 2017) shall be transmit as the input data. In the latter case, criticality of the errors is determined automatically and set in the results of the testing programs, depending on the security functions checks which ended with a negative result.

Evaluation objects can include various versions of the same DST tested in various operating systems and with different versions of the hardware component. As a result of this, it becomes possible to compare the tested software and hardware DST both with its previous versions in order to conduct regression testing and with some abstract “reference” version in which no errors were detected in order to assess the degree of influence of the errors found on the quality of the security tool.

In this case, two operation modes of the verification program are assumed: basic one the DST is evaluated only on the basis of the detected errors of different levels of criticality, advanced one the DST is also evaluated on the basis of criticality of the errors but taking into account the criticality level of the security functions in which they were identified.

The basic mode of operation can be used for any software and hardware DST while the advanced mode can be used for those for which testing programs are developed by Kanner (2017) means of cryptographic data protection and access control subsystems of various manufacturers. Both operation modes allow us to rank the evaluation objects according to the degree of influence the detected errors have on the quality of the software and hardware DST and moreover in addition to assessing the criticality of the detected errors, to take into account their number. The application of one or another method of evaluation (only based on criticality of the errors or on both criticality and quantity) depends on the specific objectives of verification.

Suppose that it is necessary to verify the next version of the software and hardware DST (for example, access control subsystem V1.4) based on the errors found during testing. For comparison, the verification program should

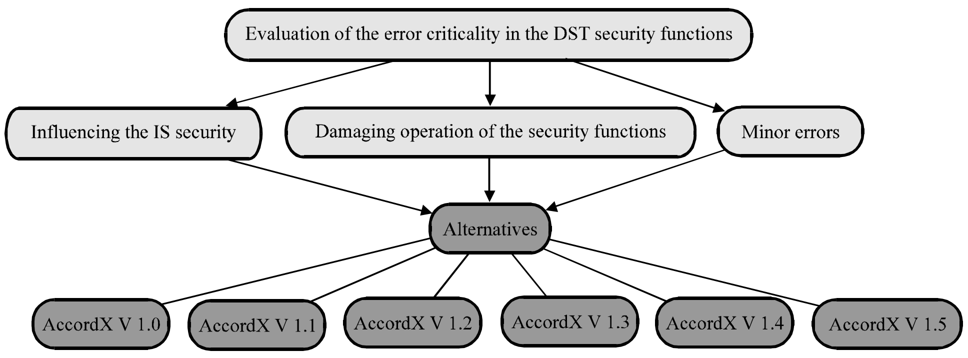

Evaluation of the error criticality in the DST security functions
Influencing the IS security
Damaging operation of the security functions
Minor errors
Alternatives
AccordX V 1.0
AccordX V 1.1
AccordX V 1.2
AccordX V 1.3
AccordX V 1.4
AccordX V 1.5

Fig. 2: Visualization of the hierarchical structure of the task connected with evaluation of the DST error criticality: purpose individual criteria and alternatives

use the results of testing previous versions (for example, V 1.0, 1.1, 1.2 and 1.3) and a certain "reference" version of the DST. At the same time, suppose that while testing the specified DST versions, the following number of critical errors affecting the IS protection, errors in security functions and non-critical errors were recorded-you need to specify them as a vector whose element values correspond to the number of the listed types of errors, respectively, V1.0-(1, 3, 1); V 1.1-(2, 2, 1); V 1.2-(0, 5, 8); V 1.3-(0, 2, 5); V 1.4-(0, 1, 10).

The program "verifier for software and hardware DST" uses the proposed mathematical apparatus proposed in the optimization and decision-making theory. This solves the task of evaluating criticality of the errors detected in security functions of the software and hardware DST and ranking alternatives (DST versions) according to the evaluation criteria. The hierarchical structure of this task is shown on Fig. 2 and consists of a purpose individual criteria (presence of errors of three criticality levels) and the alternatives being evaluated.

In the basic operation mode, the program "verifier for software and hardware DST" takes into account only criticality of the errors detected during testing of the DST security functions and their number. At the same time, to ensure proper operation of this program, it is necessary to set the above quantitative data on the detected errors (the testing results) and also to determine the criticality scale describing the "importance" (superiority) relationship of one criterion in relation to the other. In accordance with the Saaty's scale (Saaty, 1980; Chernorutsky, 2001), it would be advisable to set the following values as a result of a pairwise comparison of superiority of the errors of the accepted criticality levels: critical errors compared to errors in the security functions -3 (medium superiority), critical errors compared to minor ones -9 (absolute superiority), errors in the security functions in comparison with non-critical ones -7 (strong superiority). Taking into account the given superiority values, the verification program should automatically calculate the numerical values of the weighting factors for the evaluation criteria. It is important to note that the criticality scale should be determined only once and can be used without further changes to solve new similar tasks of evaluating criticality of the errors in software and hardware DST.

Based on the input data, the verification program allows you to automatically evaluate criticality of the errors in the alternatives identified during testing. When using the proposed mathematical apparatus, quantitative indicators of the detected errors are also taken into account.

In the advanced mode of operation, the program "verifier of software and hardware DST" takes into account not only criticality of the errors and their number but also criticality of the security functions in which such errors were detected during testing of the security functions of the software and hardware DST. In this mode, a new level of individual criteria is added to the hierarchical structure of the task being solved (Fig. 3), characterizing the tested DST security functions for which criteria with errors will be considered as embedded local criteria (for each criterion of a higher level of hierarchy). In such conditions, for example, for the access control subsystem, it is advisable to consider the following security functions that will represent the criteria of a higher level: user identification and authentication, access control, integrity control, creation of an isolated software environment for users to ensure that the current security policy is not violated (Kanner, 2016b), other minor security functions (clearing of the random access memory and residual information, printing control, etc.).

Let us consider the use of the developed program "verifier of software and hardware DST" to verify one of the selected DST the Access Control Subsystem DST Protecting from Unauthorized Access (PUA) "Accord-X"

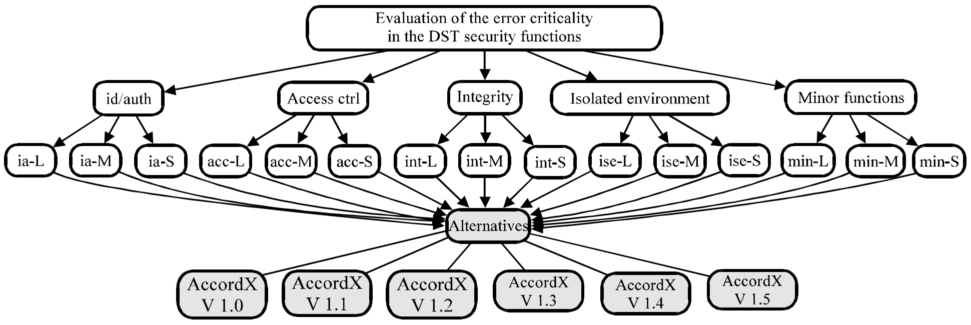
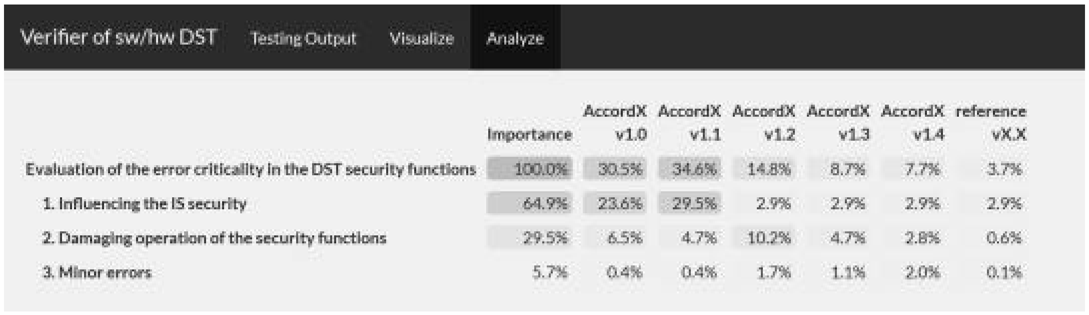


Fig. 3: Visualization of the hierarchical structure of the task connected with evaluation of the DST error criticality with respect to criticality of security functions (L, M, S denote the level of criticality of errors from major to minor)



	Importance	AccordX v1.0	AccordX v1.1	AccordX v1.2	AccordX v1.3	AccordX v1.4	reference vX.X
Evaluation of the error criticality in the DST security functions	100.0%	30.5%	34.6%	14.8%	8.7%	7.7%	3.7%
1. Influencing the IS security	64.9%	23.6%	29.5%	2.9%	2.9%	2.9%	2.9%
2. Damaging operation of the security functions	29.5%	6.5%	4.7%	10.2%	4.7%	2.8%	0.6%
3. Minor errors	5.7%	0.4%	0.4%	1.7%	1.1%	2.0%	0.1%

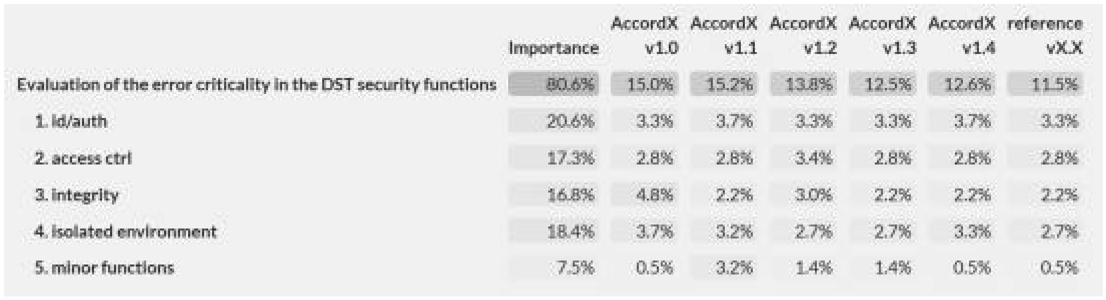
Fig. 4: Results of verifying Accord-X based on the errors found during testing (basic mode)

(Kanner and Ukhlinov, 2012; Kanner, 2014) Versions -V 1.0, 1.1, 1. 2, 1.3 and 1.4. The result of verifying the specified versions is shown in Fig. 4 (the percentage is indicated instead of the values of the weighting coefficients for convenience).

It can be seen from the obtained results that Accord-X version V 1.4 has the value of the generalized criterion that is the closest to the "reference" version as well as the smallest criticality of the detected errors as compared to all other versions. This indicates a positive trend of regression testing-compared to previous versions, the number of DST errors has increased while their overall criticality has decreased. Thus, verification of Accord-X Version V 1.4 can be completed and the identified errors can be taken into account when developing the next version of the DST. Moreover, the verification results in Fig. 4 show that Accord-X Version V 1.1 has the greatest criticality of the detected errors due to a greater number of errors affecting the IS or data security. Therefore, verification of this version was interrupted, although, uncompleted and the DST was sent for revision. Version V 1.2 was developed as a result of the revision in which all the most critical errors were corrected but several new errors of a lower criticality level were found. However, due to the fact that the overall criticality of all the detected errors was reduced including in comparison with Version V 1.0, the verification of Accord-X Version V 1.2 was successfully completed.

It is necessary to clarify that the values of the generalized and individual criteria (for specific types of errors) for the "reference" version of the DST in Fig. 4 have a non-zero value due to the use of the mathematical apparatus and the scale of superiority adopted in it to solve the task, even taking into account the zero values of all evaluation criteria. The given criteria values for the "reference" version should be interpreted as the minimum acceptable values that cannot be potentially exceeded in any real version of the software and hardware DST.

At the same time, errors in user identification and authentication as well as in creating an isolated software environment for users, will have the highest priority, the priority of the errors in access control and integrity control is lower and the priority of errors in other functions (cleaning of the random access memory and residual information, printing control, etc.) is minimal. The previously obtained "importance" (superiority) relations between the types of errors themselves (identified in the corresponding security function) as well as the criticality scale can be used without change. In accordance with this



	Importance	AccordX v1.0	AccordX v1.1	AccordX v1.2	AccordX v1.3	AccordX v1.4	reference vX.X
Evaluation of the error criticality in the DST security functions	80.6%	15.0%	15.2%	13.8%	12.5%	12.6%	11.5%
1. Id/auth	20.6%	3.3%	3.7%	3.3%	3.3%	3.7%	3.3%
2. access ctrl	17.3%	2.8%	2.8%	3.4%	2.8%	2.8%	2.8%
3. integrity	16.8%	4.8%	2.2%	3.0%	2.2%	2.2%	2.2%
4. isolated environment	18.4%	3.7%	3.2%	2.7%	2.7%	3.3%	2.7%
5. minor functions	7.5%	0.5%	3.2%	1.4%	1.4%	0.5%	0.5%

Fig. 5: Results of verifying Accord-X based on the errors of various criticality levels of the security functions in which they were identified

in the advanced operation mode of the program "verifier of software and hardware DST", the results shown in Fig. 5 were obtained for the same input data.

In this operation mode the results of Accord-X verification are different: the overall criticality of the errors in V 1.0 and 1.1 is not so different due to the fact that all critical errors in V 1.1 were detected in the least important security function. It is possible that with such results, verification of this version of Accord-X could be completed because as compared with Version V 1.0, the critical errors in a more significant security function were eliminated.

Version V 1.3 has a lower overall error criticality as compared to V 1.4 due to the fact that the latter revealed a lot of minor errors in two most significant security functions. It might be better to send this version of the DST for revision.

CONCLUSION

The study proposes a verification algorithm based on the results of testing software and hardware DST and considering criticality of the errors not in terms of abnormal operation of the test object but in terms of the possibility of damaging the IS or data security in case of incorrect operation of the security functions. A new level of evaluation criteria hierarchy is introduced - criticality of the security functions themselves. In this case, the criticality of each error takes into account the interdependence of the computability of the security functions from each other and is made up of the totality of criticality levels of the error itself and the security function in which it was detected. This algorithm implements formal evaluation of criticality of the errors detected during testing in security functions of the software and hardware DST and algorithms for calculating the overall error criticality value before integration of the security tools in the information system and used to compare both, different versions as a part of regression testing and those with an abstract "reference" version in order to determine the degree of influence of the detected errors on the system or data security and to make a decision on the possibility to release a specific version of the DST.

The study also describes the program "verifier of software and hardware DST" developed on the basis of the proposed algorithm for verifying software and hardware DST which makes it possible to automatically evaluate criticality of the errors detected in the security functions of software and hardware DST after using the corresponding testing programs. At the same time, the proposed program "verifier of software and hardware DST" can analyze both the results of testing different versions of the DST between themselves as a part of regression testing and of comparing them with an abstract "reference" version to evaluate the degree of influence of the detected errors on the security quality of this or another version (or when testing them in certain conditions).

In addition, to take into account various verification objectives, the developed program provides for two operation modes, the results of which may differ depending on "importance" of the security functions in which errors of different criticality levels were detected. Due to dependence of the advanced operation mode on the hardware and software DST in terms of the implemented security functions and their significance in the verification framework, it is possible to apply it to security tools for which testing programs have already been developed. The basic mode of operation can be applied without changes with respect to any software and hardware or only software DST.

REFERENCES

Adamcsek, E., 2008. The analytic hierarchy process and its generalizations. Master Thesis, Eotvos Lorand University, Budapest, Hungary.

Chernorutsky, I.G., 2001. Methods of Optimization and Decision Making. LAN Publishing, Saint Petersburg, Russia.

Cogger, K.O. and P.L. Yu, 1985. Eigenweight vectors and least-distance approximation for revealed preference in pairwise weight ratios. J. Optim. Theor. Appl., 46: 483-491.

Drobotun, E.B., 2009. [Criticality of errors in software and analysis of their consequences (in Russian)]. Fundam. Res., 1: 73-74.

Kanner, A.M. and L.M. Ukhlinov, 2012. [Access control in GNU/Linux (in Russian)]. Inf. Secur. Questions, 1: 35-38.

Kanner, A.M., 2014. [Linux: Process life cycle and access control (in Russian)]. Inf. Secur. Questions, 1: 37-40.

Kanner, A.M., 2016b. Correctness of data security tools for protection against unauthorized access and their interaction in GNU/Linux. Global J. Pure Appl. Math., 12: 2479-2501.

Kanner, T.M. and K.h.S. Sultanahmedov, 2014. [Features of verification of data security tools (In Russian)]. Inf. Secur. Questions, 4: 55-57.

Kanner, T.M., 2015. [Features of using virtualization for testing software and hardware data security tools (In Russian)]. Inf. Secur., 18: 416-419.

Kanner, T.M., 2016a. Applicability of software testing methods to software and hardware data security tools. Global J. Pure Appl. Math., 12: 167-190.

Kanner, T.M., 2017. [The effectiveness of using supporting tools for testing software and hardware DST (In Russian)]. Inf. Secur. Questions, 1: 9-13.

Kanner, T.M., 2018. [Adaptation of existing verification methods for software and hardware DST (In Russian)]. Inf. Secur. Questions, 1: 13-19.

Kim, I.Y. and O.L. De Weck, 2006. Adaptive weighted sum method for multiobjective optimization: A new method for Pareto front generation. Struct. Multidiscip. Optim., 31: 105-116.

Kulyamin, V.V., 2008. Software Verification Methods. Institute for System Programming of the RAS, Moscow, Russia, (In Russian).

Lin, J., 1976. Multiple-objective problems: Pareto-optimal solutions by method of proper equality constraints. IEEE. Trans. Autom. Control, 21: 641-650.

Saaty, T.L., 1977. A scaling method for priorities in hierarchical structures. J. Math. Psychol., 15: 234-281.

Saaty, T.L., 1980. The Analytical Hierarchy Process. 2nd Edn., McGraw-Hill, New York, USA., ISBN:9780070543713, Pages: 287.

Takeda, E., K.O. Cogger and P.L. Yu, 1987. Estimating criterion weights using eigenvectors: A comparative study. Eur. J. Oper. Res., 29: 360-369.