TEST CASE SELECTION FOR PENETRATION TESTING IN MOBILE CLOUD COMPUTING APPLICATIONS: A PROPOSED TECHNIQUE

[1]AHMAD SALAH AL-AHMAD, [2] HASAN KAHTAN
[1]Universiti Teknologi MARA, Department of Computer Science, Malaysia
[2]Universiti Malaysia Pahang, Department of Software Engineering, Malaysia
E-mail: [1]ahmad.salah.85@gmail.com, [2]hasankahtan@ump.edu.my

ABSTRACT

The extensive use of mobile applications in terms of user's number and size of diverse data has introduced additional security threats which make uncovering these vulnerabilities complex for testers. Testers use certain types of software security testing to detect software vulnerabilities, particularly penetration testing. Test case selection is an essential phase of penetration testing, especially when testing complex and large applications. Multiple techniques have been proposed for selecting test cases to be used in penetration testing. In general, the majority of such techniques select a set of test cases that cover the designated paths and fit well with the user requirements. This study reviews existing techniques and models that are used for test case selection. Methods, strengths and weaknesses are the main factors that are presented in this study. This study shows that offloading, that is, the technology used in mobile cloud computing applications, has been disregarded by existing techniques and models for test case selection. Therefore, this study proposes an enhanced test case selection technique for penetration testing. This proposed technique considers offloading parameters when selecting test cases to improve coverage paths and reflect user preferences in terms of cloud and mobile priority percentages. Moreover, test cases for both mobile and cloud in the mobile cloud computing applications are considered to be selected in list of test cases to be executed. Besides, user preferences feature is provided in the selection process to reflect the importance of each parties, cloud and mobile sides of the application under test. The proposed technique will improve the security of mobile cloud computing applications by exposing the possible vulnerabilities from both mobile and cloud sides application.

Keywords*: Penetration Testing, Test Case Selection, Offloading, Path Coverage*

1. INTRODUCTION

Mobile computing is a technology that has emerged in the present and will certainly affect the future by enabling real-time communication anywhere at any time [1, 2]. Nearly all human life disciplines are influenced by mobile computing technology, as manifested in the number of users and the huge business investment in the mobile technology sector either by developing additional applications or enhancing the infrastructure [3]. The increasing number of users and additional domains, where mobile computing is implemented, has resulted in increased security vulnerabilities which make security testing essential [4]. A key security testing type is penetration testing which is used to uncover unknown vulnerabilities.

Penetration testing is a well-known technique that is used to evaluate the security posture of any software application in which a tester aims to compromise a software application security [5]. This technique is considered widely and commonly used method among all software application security best practices [6]. Penetration testing has been used to ensure the security of web applications. In this technique, a tester can discover vulnerabilities and weaknesses by simulating possible attacks on the target web application. An efficient penetration testing requires the tester to rely on techniques that gather input vector information on the target web application, analyse the responses of the application to check if the attack can possibly exploit any existing vulnerability and determine if an attack is successful. However, several existing techniques are lacking and frequently incomplete, thereby possibly leaving a part of the application untested and vulnerabilities undiscovered [7].

Penetration testing involves designing and generating a set of test cases, analysing the software application using these test cases and examining the results to determine if the software application security has been compromised. Test case generation processes generate test cases that cover all application paths, uses all attributes and adopts variable domains, ranges and types. The execution and testing of all generated test cases are generally infeasible given time and cost limitations [8]. Therefore, finding mechanisms that select test cases or prioritise the generated test cases is mandatory to reduce the resources required whilst maintaining high path coverage and fault detection ratios [9].

As long executing all generated test cases is not visible due to limited budget and time available for testers [10]. Selecting the set of test cases to be executed becomes more critical task. This process reduces the size of test case set while insuring that this reduction will not affect the quality of test results. Test case selection has a significant impact on the test coverage on the intended parts to be tested for the applications under test [10]. For instance the traditional test case selection select among the generated test cases based on user preferences [11] or based on pre-established techniques and formulas [12, 13].

Test case selection is still at initial stages and more research is required to make it applicable in today's world [14]. Therefore, for new technologies such as Mobile Cloud Computing (MCC) application which is one of the complex applications that uses new technologies needed an exceptional factors to be used once selecting the set of test cases. Thus together, the high complexity of new technologies and the simplicity of test case selection techniques, make is required to conduct this study to come up with a technique that help testers to select test cases to be executed.

MCC is a convergent technology which consists of three main heterogeneous technologies, namely, mobile computing, cloud computing and networking [15]. MCC is simply a set of tools and techniques that utilise cloud resources to empower mobile applications. According to literature, MCC has recently attracted significant attention. The direct revenue of the mobile cloud market was approximately $68 billion in 2017. The number of MCC subscribers worldwide has increased from 42.8 million in 2008 to more than 998 million in 2014. Mobile cloud is expected to eventually become the dominant method for mobile application operation [16].

This paper will review the current available techniques and models used to conduct test case selection. The outcome of review will be analysed based on MCC applications complexity implementation models that affect the selection process. The analysis result will be used to propose and enhanced technique that fill the research gap and tackle the issues of previous test case techniques and models when testing MCC applications.

In this study, a test case selection technique is proposed to solve the issues of previous techniques. The objective of this study is to investigate the limitations of previous test case selection techniques in the MCC domain and propose an enhanced technique that overcomes these limitations. Section 2 describes the motivation for studying the test case selection for penetration testing in the domain of MCC. Section 3 examines the test case selection techniques in accordance with the specific analysis technique used and evaluates the strengths and limitations of each approach. Section 4 introduces the test case selection techniques. Section 5 and section 6 demonstrate the results of the enhanced test case selection technique and discuss the implications of these results. Section 7 presents the limitations of this technique. Finally, paper is concluded in Section 8.

2. MOTIVATION

The addition of new variables to test case generation increases the number of generated test cases exponentially [17-22]. In addition, the test case selection technique is a complex issue, especially in penetration testing given the numerous paths, input types, input methods, environment factors and vulnerability types. Scholars, industry experts and practitioners attempt to construct new mechanisms that enhance the ratio between the number of selected test cases versus the path coverage and fault detection ratios [23].

Test case selection technique is essential to penetration testing. However, such technique is challenging and requires test case selection technique that can generate well-defined and specific test cases, utilise minimal resources and reduce time whilst maintaining the test case coverage acceptable [24, 25]. Such test case selection technique is important, especially for MCC applications.

In the MCC application penetration testing, offloading should be used as a variable when generating test cases because offloading affects the application execution paths. Offloading means remote execution and is the method used when outsourcing mobile tasks to the cloud [26-28]. This process adds uniqueness and complexity to the MCC application penetration testing in terms of generating, selecting and executing test cases.

Offloading in the MCC application affects the execution path for the mobile applications [29] as this process uses multiple factors to determine where to execute certain tasks. In dynamic offloading, the decision to augment the task to the cloud is made whilst the application is running. In static offloading, the decision is predetermined by the developers and is unaffected by the running applications. However, the MCC applications can use dynamic and static offloading collectively; this combination is called hybrid-offloading [30, 31]. Offloading exponentially increases the complexity of MCC application which in turn increases the required resources when conducting penetration testing. An offloading parameter is a significant factor to be studied to improve efficiency and effectiveness of penetration testing for the MCC application.

3. PENETRATION TEST CASE SELECTION TECHNIQUES: RELATED WORK

Many techniques for test case selection have been proposed in the literature [32-37]. This section provides a critical review of several works and proposed models for test case selection for penetration testing. The result of the critical review shows the strength and limitations of each study based on the main MCC application characteristics that makes it unique and complex, which is the offloading. These results are summarized in Table 1.

The survey was conducted by following the steps presented in the methodology illustrated in Figure 1. This survey for publications included an systematic search using several digital libraries as they are the most relevant sources in software and security engineering [38]. These digital libraries were ACM digital library (http://portal.acm.org), IEEE Xplore (http://ieeexplore.ieee.org), ScienceDirect (http://www.sciencedirect.com) and SpringerLink (http://linkspringer.com).

Certain search strings used to search over the selected repositories that reflect the domain of test case selection on penetration testing. These search strings were as follows: (i) ‘penetration testing’ and ‘test case selection’, (ii) ‘test case selection’ and ‘penetration testing’, (iii) ‘penetration testing’ and ‘case selection’ and (iv) ‘penetration testing’ or ‘test case selection’.

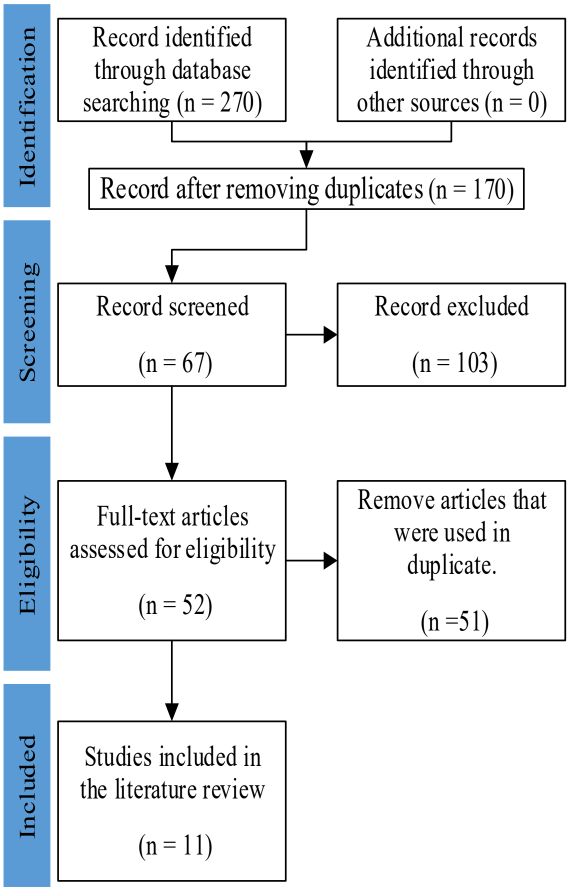
Identification
Record identified through database searching (n = 270)
Additional records identified through other sources (n = 0)
Record after removing duplicates (n = 170)
Screening
Record screened (n = 67)
Record excluded (n = 103)
Eligibility
Full-text articles assessed for eligibility (n = 52)
Remove articles that were used in duplicate. (n =51)
Included
Studies included in the literature review (n = 11)

Figure 1: Survey Methodology

The set of publications found with the strategy of the survey was filtered using inclusion and exclusion criteria for selecting relevant primary studies. The survey included peer-reviewed publications written in English and cover test case selection for penetration testing. Papers related to testing approaches other than penetration testing or do not propose a new test case selection method have been excluded. The initial search based on the search strategy resulted in 51 papers, where 11 proposed new test case selection techniques. These techniques are summarised in Table 1.

Table 1 Test Case Selection Models and Techniques

Ref.	Method	Strength	Limitation
[39]	Divide the testing time into time slots. Iteratively select and prioritise test cases for each time slot using integer linear programming.	Shows effectiveness in exposing faults and enhancing the branch coverage.	Disregards offloading and is valid only for request quota constraints.
[40]	Select factors based on prioritisation goal. Use factor values for all requirements from the person involved. Compute each requirement weight and its mean of a factor.	Improves the rate of fault detection.	Disregards offloading. Incomplete requirements affect the result.
[23]	Select test cases based on similarities between test paths using triggers and guards using the generating algorithm.	Reduces the number of test cases selected whilst preserving the fault detection rate.	Disregards offloading and requires highly complex programmes to analyse the source code in each version.
[41]	Prioritise test cases based on multiple variables, and add another variable that is calculated based on the number of the variables used in the first versions and changed to the modified version.	Selects a minimal number of test cases, and highly detects faults.	Disregards offloading and requires highly complex programmes to analyse the source code in each version.
[8]	Standardise the requirements. Determine the neighbourhood F(x) for every requirement. Select neighbourhood function order of the functions to benefit to F(x). Generate order, and calculate the appropriate test case number.	Has high reduction	Disregards offloading and changing requirements.
[9]	Use the test case selection technique on the basis of detecting the changes in the applied case descriptions between versions.	Detects the changed test cases efficiently.	Disregards offloading and cannot be used for an initial test.
[36]	Prioritise the test cases based on four groups of practical weight factors: customer allotted priority, developer observed code execution complexity, changes in requirements and fault effect, completeness and traceability.	Improves fault detection rate and preserves time and resources.	Disregards offloading.
[42]	Minimise the number of test cases and maximise path coverage by determining the similarity between test cases.	Reduces the number of test cases selected with high coverage path.	Disregards offloading and is highly resource-intensive.
[43]	Calculate test case weight based on a differential control flow graph.	Represents a new factor for improving coverage.	Disregards offloading and is time-consuming.
[44]	Prioritise selection of flow graph-based composite services through modelling behaviour of service interactions and control flow.	Composite services and modelling behaviour of service improve the path coverage.	Disregards offloading and is time-consuming.

[45]	Test selection technique based on functional modelling by selecting a subset of the set of test cases that provides a coverage level above a predetermined coverage.	Selected subset fits with user requirements and pretests the coverage-based test case subset.	Disregards offloading. The functional model is time-consuming.

The majority of the reviewed techniques for selecting test cases are based on the requirements of the user, tester, modelling and developer as presented in Table 2. In addition, none of these test case selection techniques and models use offloading or device state as a parameter in the selection technique. The number of test cases to be executed increases when offloading is ignored or the testers sacrifice the coverage path percentage to reduce time and efforts required.

Table 2: Test Case Selection Models and Technique Analysis

Reference	Requirement-Based	Model-Based	Developer-Based	Offloading
[39]	✓	×	×	×
[40]	✓	×	×	×
[23]	×	✓	×	×
[41]	✓	×	✓	×
[8]	✓	×	✓	×
[9]	×	×	✓	×
[36]	×	×	✓	×
[42]	×	×	✓	×
[43]	×	×	✓	×
[44]	×	✓	×	×
[45]	×	×	×	×

4. PROPOSED TECHNIQUE

The proposed technique in this study has adopted and enhanced the existing work in [40, 46]. The proposed technique has embedded offloading parameters and vulnerability domain in the selection technique to tackle the uniqueness and complexity of MCC applications that previous techniques have disregarded.

This study adopts the [40, 46] techniques by injecting the offloading parameters and vulnerability domain in the selection technique. This study proposes that the initial test is different from the repeated test. The repeated test focuses on the changes in the application, whereas the new test focuses on all tasks. Therefore, the factors used in test case selection are classified into two groups, namely, new and repeated test factors. These factors are listed in Table 3 [40, 46].

Table 3: Test Case Factors

New Test Factors	Regression Test Factors
Complexity	Fault effect of requirements
Changes in requirements	Completeness
Priority of requirement	Traceability

The proposed test case selection for MCC penetration testing technique is demonstrated in Figure 2 and Figure 3. Figure 2 represents the first part of the technique that used to calculate the weight of each test case based on set of test case factors shown on Table 2. Figure 3 represents the second part of the technique where the mobile and cloud test cases sets are built to be merged as single order set of selected test cases.

This technique starts in Figure 2 by calculating the mean of the factors that have been selected previously for each of the requirements and then stores these results in the requirement record at the requirement repository. This calculation is followed by looping through all the requirements for each test case that has been generated to calculate the test case weight and add it to its parameters in the test case record at the test case repository.

The proposed technique will build two lists of test cases as shown in Figure 3. These lists are defined to differentiate between the test cases that would run on the mobile device or the cloud. Each test case will be added to the corresponding list in accordance with the offloading parameter that has been calculated during the test case generation or specified by the user. The two lists will be sorted by weight for the highest weight to be highly prioritised for selection.

The test case selection technique will select cloud and mobile test cases with the highest weight on the basis of the calculated preferred number of cloud and mobile test cases by using the number of preferred test cases to be selected multiplied by the percentage of cloud and mobile test cases priority. This enhance the technique test case result set coverage as it ensure the selected test case set covers vulnerabilities for mobile and cloud.

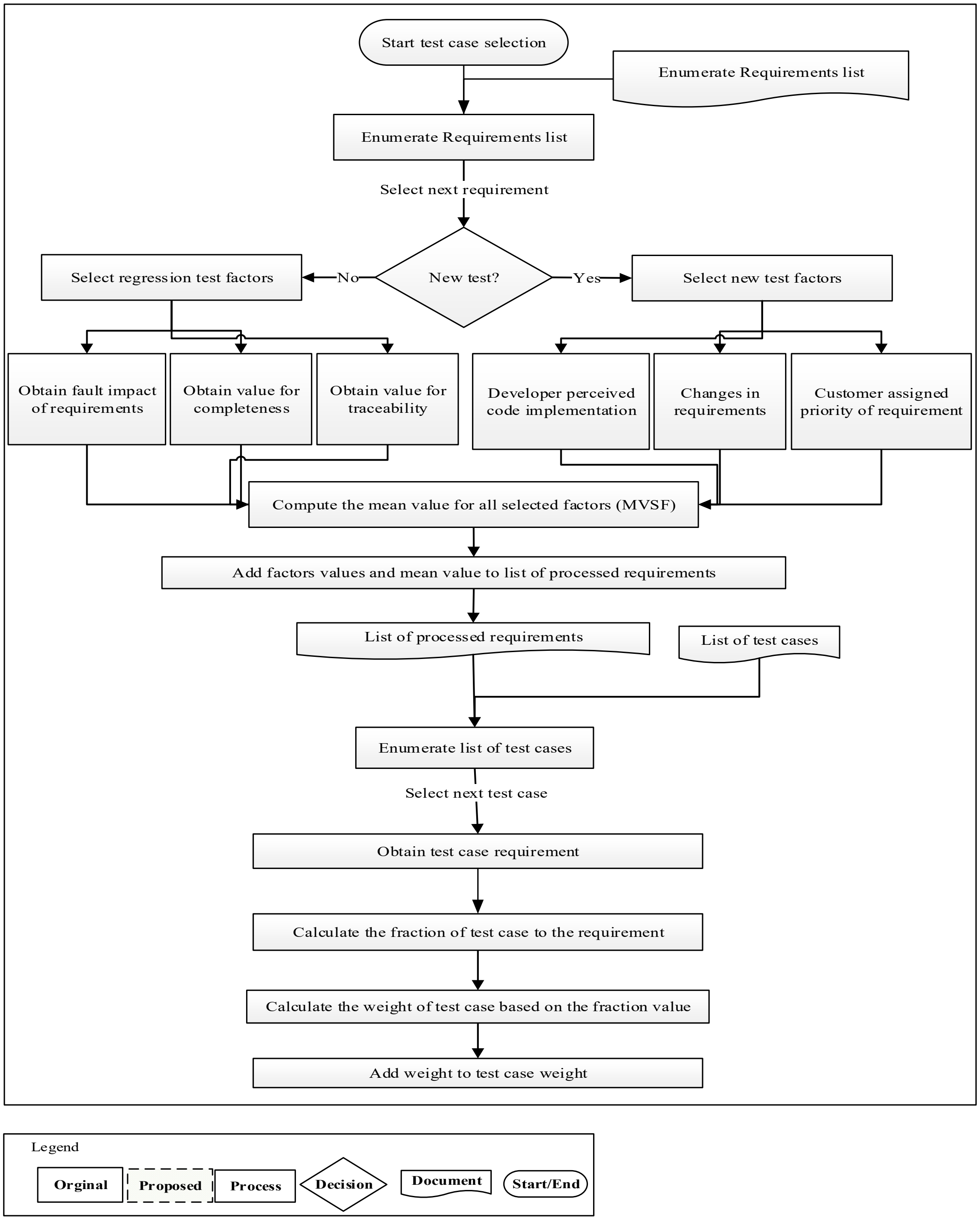
Start test case selection
Enumerate Requirements list
Enumerate Requirements list
Select next requirement
New test?
No
Yes
Select regression test factors
Select new test factors
Obtain fault impact of requirements
Obtain value for completeness
Obtain value for traceability
Developer perceived code implementation
Changes in requirements
Customer assigned priority of requirement
Compute the mean value for all selected factors (MVSF)
Add factors values and mean value to list of processed requirements
List of processed requirements
List of test cases
Enumerate list of test cases
Select next test case
Obtain test case requirement
Calculate the fraction of test case to the requirement
Calculate the weight of test case based on the fraction value
Add weight to test case weight
Legend
Orginal
Proposed
Process
Decision
Document
Start/End

Figure 2 Mcc Application Penetration Test Case Selection (Part 1)

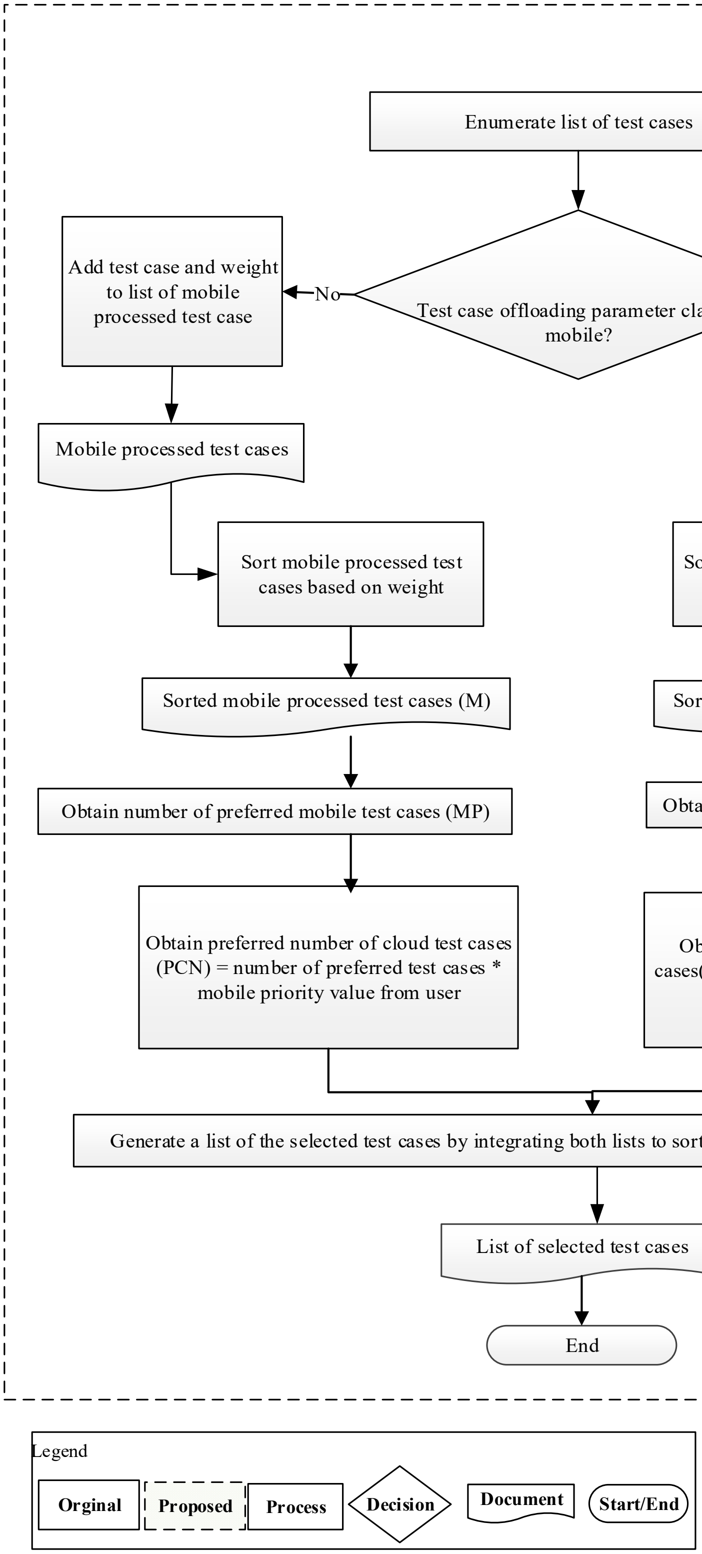
Enumerate list of test cases
Test case offloading parameter class is mobile?
No
Yes
Add test case and weight to list of mobile processed test case
Add test case and weight to list of cloud processed test case
Mobile processed test cases
Cloud processed test cases
Sort mobile processed test cases based on weight
Sort cloud processed test cases based on weight
Sorted mobile processed test cases (M)
Sorted cloud processed test cases (C)
Obtain number of preferred mobile test cases (MP)
Obtain number of preferred cloud test cases (MC)
Obtain preferred number of cloud test cases (PCN) = number of preferred test cases * mobile priority value from user
Obtain preferred number of mobile test cases(PMN) = number of preferred test cases * cloud priority value from user
Generate a list of the selected test cases by integrating both lists to sort the result list by offloading parameter value
List of selected test cases
End
Legend
Orginal
Proposed
Process
Decision
Document
Start/End

Figure 3 Mcc Application Penetration Test Case Selection (Part 1)

Both of these lists will be sorted in order to select these with highest weight in their category, mobile or cloud, and then a formula that use the tester preference number of the cloud priority or mobile priority will be used to be multiplied with the number of preferred test cases to be selected in order to find the number of test cases to be selected of each category.

A combination of cloud and mobile priority values in the selection technique using the preferred number of test cases to be selected will provide two lists of test cases. The two lists are integrated and then sorted by weight to provide the tester with the list of selected test cases in accordance with the requirements.

The result set of test cases will have same as the number of preferred test cases the testers selected. This list also includes set of test cases that will be executed on the cloud or on the mobile with same percentage as also selected by the tester. These list of test cases also sorted as per the calculated weight based on other preferences that test selected after injecting the weight or cloud or mobile in the formula.

5. RESULTS

The proposed technique enables the tester to implement the user requirements in terms of number of test cases to be selected and covering client and cloud applications that are implemented in the MCC application under test by differentiating between each test case in terms of execution location, either to be executed on the mobile or on the cloud. In addition, the proposed technique provides the result as an ordered set of test cases after implementing the offloading user requirements that are sorted by mobile and cloud priority values.

In detail, this technique starts by calculating the mean value for each requirement by using all factors proposed by [40, 46] as summarised in Table 1. Then, the proposed technique determines the fraction that each test case represents the user requirements to calculate the test case weight. The test case weight is calculated based on the mean value of the requirement it represents, as described in [40, 46].

A new procedure has been added to define the class of each test case using offloading details after calculating the weight of each test case. This technique continues to sort based on test case weight after determining if the test case is cloud or mobile. Then it selects the top test case of each class (mobile and cloud) on the basis of mobile and cloud priority values. Finally, the result set of test cases are sorted based on the test case weight.

6. DISCUSSION

The goal of this work is to build a test case selection technique that embed offloading parameters in the selection process. The offloading parameters embedded by selecting test cases of both mobile and cloud computing. Moreover, user preferences feature is provided in the selection process to reflect the importance of each parties, cloud and mobile sides of the application under test.

The proposed test case selection for MCC penetration testing uses offloading as a parameter in selecting test cases by building two sets. First set includes test cases that will be executed on the cloud and second test case set include test cases executed on the mobile. This segmentation process use the data provided by the tester along with the test cases or using the MCC application description from the developers.

The returned set of test cases has managed to provide the penetration testers with a set of test cases that consider offloading parameters and represent the user preferences. This will improve the results coverage path percentage due to considering the offloading parameters when selecting test cases as it ensures that mobile and cloud implementations are tested. Furthermore, the user requirement in terms of resources which are represented by the number of preferred test cases is also considered when generating the result of test cases, thereby improving the resource allocation based on the user preferences.

This paper contributes to the domain of MCC by reviewing penetration test case selection techniques which will serve as base line references for future research. Moreover, the outcome of the review highlighted that there is a room for MCC improvement in terms of offloading features. Furthermore, this paper address a new direction for researchers in the area of penetration testing to enhance current technique and models in order to cope with the new technologies, such as offloading, by using this paper as a reference.

7. LIMITATIONS

The test case selection technique proposed in this paper is one of the main phases of penetration testing that used to conduct penetration testing on MCC applications effectively and efficiently. Consequently, to evaluate the proposed technique based on the common evaluation criteria for penetration testing it should be integrated into a penetration testing model or framework.

This integration requires a test case generation and test case execution techniques that also built specifically to be used on MCC application due to the complexity and uniqueness of these applications. Therefore, evaluating this proposed technique requires a new test case generation technique that provides the data required and to propose a new test case execution technique that read and execute the result of our proposed technique.

8. CONCLUSION

This paper reviews the test case selection techniques used in the test case selection phase for penetration testing. These techniques are aimed at selecting the minimum number of test cases with the highest percentage of path coverage to conduct the penetration testing with minimal effort and achieve the optimal results in finding unknown vulnerabilities. The paper shows that offloading parameter, which is one of the development model for advanced technology, has been disregarded when selecting test case set using previous techniques and models. Therefore, applications that use offloading may require extra effort when carried out testing for vulnerabilities through penetration testing. This study proposes an enhanced test case selection technique that considers offloading. Several potential directions for future research should be identified to implement and evaluate the proposed technique over certain application domains such as: social and banking applications.

ACKNOWLEDGMENTS

This research is supported by the Department of Research and Innovation of University Malaysia Pahang under Fundamental Research Grant Scheme (FRGS) RDU170102.

REFERENCES

[1] Baccarelli, E., et al., Energy-efficient dynamic traffic offloading and reconfiguration of networked data centers for big data stream mobile computing: review, challenges, and a case study. IEEE Network, 2016. 30(2): p. 54-61.

[2] Schneider, S., M. Hirzel, and B. Gedik. Tutorial: stream processing optimizations. in Proceedings of the 7th ACM international conference on Distributed event-based systems. 2013. ACM.

[3] Chua, W.Y., et al. Improving mobile applications usage experience of novice users through user-acclimatized interaction: a case study. in Twentieth Americas Conference on Information Systems, AMCIS. 2014. Savannah, Georgia, USA.

[4] Khurana, P., A. Sharma, and P.K. Singh, A systematic analysis on mobile application software vulnerabilities: Issues and challenges. Indian Journal of Science and Technology, 2016. 9(32).

[5] Scandariato, R., J. Walden, and W. Joosen. Static analysis versus penetration testing: A controlled experiment. in Software Reliability Engineering (ISSRE), 2013 IEEE 24th International Symposium on. 2013. IEEE.

[6] Arkin, B., S. Stender, and G. McGraw, Software penetration testing. IEEE Security & Privacy, 2005. 3(1): p. 84-87.

[7] Halfond, W.G., S.R. Choudhary, and A. Orso, Improving penetration testing through static and dynamic analysis. Software Testing, Verification and Reliability, 2011. 21(3): p. 195-214.

[8] Lawanna, A., A Model for Test Case Selection in the Software-Development Life Cycle. International Journal of Computer, Information Science and Engineering, 2013. 7(4, 2013): p. 5.

[9] Raengkla, M. and T. Suwannasart. A Test Case Selection from Using Use Case Description Changes. in Proceedings of the International MultiConference of Engineers and Computer Scientists. 2013. Hong Kong, China.

[10] Khan, S.U.R., et al., A Systematic Review on Test Suite Reduction: Approaches, Experiment's Quality Evaluation, and Guidelines. IEEE Access, 2018.

[11] Enoiu, E. and M. Frasheri, Test Agents: Adaptive, Autonomous and Intelligent Test Cases. arXiv preprint arXiv:1802.03921, 2018.

[12] Coutinho, A.E.V.B., E.G. Cartaxo, and P.D. de Lima Machado, Analysis of distance functions for similarity-based test suite reduction in the context of model-based testing. Software Quality Journal, 2016. 24(2): p. 407-445.

[13] Bokil, P., P. Krishnan, and R. Venkatesh, Achieving Effective Test Suites for Reactive Systems using Specification Mining and Test Suite Reduction Techniques. ACM SIGSOFT Software Engineering Notes, 2015. 40(1): p. 1-8.

[14] Sagar, G.P. and P. Prasad, Survey on Test Case Prioritization Techniques for Regression Testing. Indian Journal of Science and Technology, 2017. 10(10).

[15] Sanaei, Z., et al., Heterogeneity in mobile cloud computing: taxonomy and open challenges. IEEE Communications Surveys & Tutorials, 2014. 16(1): p. 369-392.

[16] Rahimi, M.R., et al., Mobile cloud computing: A survey, state of art and future directions. Mobile Networks and Applications, 2014. 19(2): p. 133-143.

[17] Chang, B.R., et al. Access security on cloud computing implemented in hadoop system. in Fifth International Conference on Genetic and Evolutionary Computing (ICGEC). 2011. Kitakyushu, Japan: IEEE.

[18] de los Reyes, G., et al. Securing the Mobile Enterprise with Network-Based Security and Cloud Computing. in IEEE Sarnoff Symposium. 2012. Newark, New Jersey, USA: IEEE.

[19] Kim, T., et al. Monitoring and detecting abnormal behavior in mobile cloud infrastructure. in IEEE Network Operations and Management Symposium (NOMS). 2012. Maui, Hawaii, USA: IEEE.

[20] Ahern, M.M. and M.S. Hendryx, Social capital and trust in providers. Social science & medicine, 2003. 57(7): p. 1195-1203.

[21] Sumter, L.Q. Cloud computing: Security risk. in Proceedings of the 48th Annual Southeast Regional Conference. 2010. Oxford, MS, USA: ACM.

[22] Zhou, Z. and D. Huang. Efficient and secure data storage operations for mobile cloud computing. in Proceedings of the 8th International Conference on Network and Service Management. 2012. Las Vigas, USA: International Federation for Information Processing.

[23] Hemmati, H., et al. An enhanced test case selection approach for model-based testing: an industrial case study. in Proceedings of the eighteenth ACM SIGSOFT international symposium on Foundations of software engineering. 2010. Santa Fe, NM, USA: ACM.

[24] Bahl, S. and M. Chaturvedi, Literature Review of Mobile Applications Testing on Cloud from Information Security Perspective. International Journal of Computer Applications, 2013. 79(14): p. 15-23.

[25] Pundhir, Y.S., Cloud Computing Applications And Their Testing Methodology. Bookman International Journal of Software Engineering, 2013. 2(1).

[26] Umair, S., et al., Mobile Cloud Computing Future Trends and Opportunities. Managing and Processing Big Data in Cloud Computing, 2016: p. 105.

[27] Yadav, D.S. and K. Doke, Mobile Cloud Computing Issues and Solution Framework. International Research Journal of Engineering and Technology, (IRJET), 2016. 03(11).

[28] Abolfazli, S., et al., Mobile Cloud Computing. Encyclopedia of Cloud Computing, 2016: p. 29.

[29] Al-Ahmad, A.S., S.A. Aljunid, and A.S.A. Sani. Mobile Cloud Computing Testing Review. in International Conference on Advanced Computer Science Applications and Technologies (ACSAT). 2013. Kuala Lumpur, Malaysia: IEEE.

[30] Sarrab, M. and H. Bourdoucen, Mobile Cloud Computing: Security Issues and Considerations. Journal of Advances in Information Technology Vol, 2015. 6(4).

[31] Khan, A.R., et al., A survey of mobile cloud computing application models. Communications Surveys & Tutorials, IEEE, 2014. 16(1): p. 393-413.

[32] Singh, Y., et al., Systematic Literature Review on Regression Test Prioritization Techniques. Informatica (Slovenia), 2012. 36(4): p. 379-408.

[33] Yoo, S. and M. Harman, Regression testing minimization, selection and prioritization: a survey. Software Testing, Verification and Reliability, 2012. 22(2): p. 67-120.

[34] Yoo, S., Extending the Boundaries in Regression Testing: Complexity, Latency, and Expertise, in Computer Science. 2009, King's College London: David Barber : Brml - Home Page.

[35] Hettiarachchi, C., H. Do, and B. Choi. Effective Regression Testing Using Requirements and Risks. in IEEE

International Conference on Software Security and Reliability (SERE). 2014. Redwood City, CA, USA: IEEE.

[36] Muthusamy, T., A New Effective Test Case Prioritization for Regression Testing based on Prioritization Algorithm. International Journal of Applied, 2014. 6(7): p. 21-26.

[37] Miller, T., Prioritisation of test suites containing precedence constraints. 2012, University of Melbourne: Department of Computing and Information Systems,. p. 15.

[38] Brereton, P., et al., Lessons from applying the systematic literature review process within the software engineering domain. Journal of systems and software, 2007. 80(4): p. 571-583.

[39] Hou, S.-S., et al. Quota-constrained test-case prioritization for regression testing of service-centric systems. in IEEE International Conference on Software Maintenance. 2008. Beijing, China: IEEE.

[40] Krishnamoorthi, R. and S. Sahaaya Arul Mary, Factor oriented requirement coverage based system test case prioritization of new and regression test cases. Information and Software Technology, 2009. 51(4): p. 799-808.

[41] Singh, Y., A. Kaur, and B. Suri, A Hybrid Approach for Regression Testing in Interprocedural Program. Journal of Information Processing Systems, JIPS, 2010. 6(1): p. 21-32.

[42] de Souza, L.S., R.B. Prudêncio, and F.d.A. Barros, Multi-Objective Test Case Selection: A study of the influence of the Catfish effect on PSO based strategies. Anais do XV de Testes e Tolerância a Falhas, 2014. 8.

[43] Akimoto, S., R. Yaegashi, and T. Takagi. Test case selection technique for regression testing using differential control flow graphs. in Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2015 16th IEEE/ACIS International Conference on. 2015. IEEE.

[44] Buck, P., Q. Shi, and Á. MacDermott. A Selective Regression Testing Approach for Composite Web Services. in Developments of E-Systems Engineering (DeSE), 2015 International Conference on. 2015. IEEE.

[45] Raz, O., et al., Method and system for creating functional model of test cases. 2017, Google Patents.

[46] Ramasamy, K. and S. Mary. Incorporating varying requirement priorities and costs in test case prioritization for new and regression testing. in International Conference on Computing, Communication and Networking (ICCCN). 2008. St. Thomas, USA: IEEE.