Acta Informatica Pragensia, 2018, 7(1), 40–57
DOI: 10.18267/j.aip.113
Original Article

Classification and Evaluation of Cloud-Based Testing Tools: The Case Study of Web Applications' Security Testing

Martin Lněnička [1], **Jan Čapek** [1]

Abstract

The purpose of the article is to give a survey of research fields related to test and manage applications from the cloud, i.e. cloud-based testing, so that it can facilitate security requirements associated with the testing. This article has two main aims. The first one is the survey of published results attained by the synergy of these research fields – cloud-based testing, testing strategies and types of tests, and related architectures, which is followed by the classification of testing tools based on their testing strategies. The second part is focused on security testing of Fire and Rescue Service portals in the Czech Republic and identification of vulnerabilities in these portals. The results suggested that it is more appropriate to manage only one unified portal than a lot of portals on the regional level, also due to the economies of scale. Finally, the most suitable tool for cloud-based security testing was recommended based on these results and a typical cloud-based testing methodology was described.

Keywords: Cloud computing, Cloud-based testing, Web applications, Security, Case study.

1 Introduction

Cloud computing provides an infrastructure for resource sharing, software hosting and service delivering in pay-per-use approach (Gao et al., 2011; Harikrishna & Amuthan, 2016). It represents an efficient way to manage and deliver computing resources and services such as Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), and Infrastructure-as-a-Service (IaaS). Although there are many published papers discussing related cloud architectures, technologies, models, design and management, cloud-based testing is still quite a new subject. Most authors agree on the fact that cloud-based testing refers to testing and measurement activities in a cloud environment and infrastructure by using cloud technologies and solutions (Lněnička, 2013; Malhotra & Jain, 2013). According to Tung et al. (2014), "*its effective unlimited storage, quick availability of the infrastructure with scalability, flexibility and availability of distributed testing environment translate to reducing the execution time of testing of large applications and hence lead to cost-effective solutions*." Therefore, it should help to ease the testing procedure with much greater efficiency (Tung et al., 2014). A provider of cloud-based testing service leases a standardized infrastructure and a variety of options in pricing, performance, and feature set (Li et al., 2010). However, this new service model also

[1] Institute of System Engineering and Informatics, Faculty of Economics and Administration, University of Pardubice, Studentska 84, 532 10 Pardubice, Czech Republic
✉ martin.lnenicka@gmail.com, capek@upce.cz

brings along challenges as well as issues that should be addressed (Ajay & Umamaheswari, 2016; Priyanka et al., 2012; Riungu et al., 2010). Especially, the question of services for security testing of web applications needs to be explored more thoroughly.

Shklar and Rosen (2009) defined a web application as "*a client/server application that uses a web browser as its client program, and performs an interactive service by connecting with servers over the internet (or intranet)*". The application presents dynamically tailored content based on request parameters, tracked user behaviors, and security considerations (Shklar & Rosen, 2009). An application should be able to handle the sporadic traffic as well as heavy load from large amount of users. Success of any application highly depends on its features and whether it is correct, secure and perform well with varying user load. This characterizes cloud-based testing since cloud infrastructures of most providers are geographically distributed across the world (Gao et al., 2011; Lněnička, 2013).

Riungu et al. (2010) argued the research issues that cloud computing imposes on software testing, which were gathered during interviews with industry practitioners. This research issues were then categorized according to application, management, legal and financial issues. Later, Riungu-Kalliosaari et al. (2016) conducted another set of interviews with professionals to address the adoption, utilization and effects of cloud-based testing in different organizational contexts. They identified four categories, with which they were able to explain the adoption and use of cloud-based testing in practice. Comparing this issue, Priyanka et al. (2012) reported on a systematic review of cloud-based testing techniques published in major software engineering journals and conferences and classified them into four categories according to issues they addressed. Gao et al. (2011) then discussed the special objectives, features, requirements and especially needs to provide a comprehensive tutorial on cloud-based application testing. They summarized and compared different commercial products and solutions supporting cloud-based testing. Similarly, Inçki et al. (2012) provided an overview regarding main contributions, trends, challenges, gaps, opportunities and possible research directions of cloud-based testing. They classified research activities performed in the cloud-based testing area according to the problem/solution domain of the paper. However, there is lacking of research that clearly defines the basic strategies and type of tests of cloud-based testing.

Jun and Meng (2011) analyzed related questions to the deployment of cloud-based testing tools, including reasons for and against using of this approach. Selected tools to test cloud-based systems at various layers including hardware interface, platform interface, storage system, and application systems were introduced and compared by Bai et al. (2011). In spite of this, they omitted the area of web applications. Gao et al. (2011) offered their comparison, but only four tools were compared. They also did not provide any explanation of how these tools may be used. Thus, there can be found another issue, which needs to be solved, i.e., a systematic classification and comparison of related tools for cloud-based testing of web applications. Comparatively little attention also appears to have been given to security testing.

Therefore, this paper provides an overview of cloud-based testing and discusses the related challenges and new opportunities. The presented paper follows this structure. Firstly, it focuses on the research methodology and methods. Then, cloud-based testing is defined, followed by testing strategies, type of tests, architectures, and related tools comparison and classification, case study and its evaluation, new methodology, and finally, the summary.

2 Research Methodology and Methods

As mentioned above, there are various issues that need to be resolved for making cloud-based testing processes feasible in practice. At first, the main benefits and risks behind the emergence of cloud-based testing have to be identified. Then, the most widely used testing strategies, techniques and types of tests have to be compared against traditional testing approaches and new classification of them has to be provided. Based on these results, a new classification of tools has to be developed in order to aid the users with a better understanding about the suitability of these tools for different testing strategies. Finally, a case study has to be conducted to show the applicability of the selected tools and an appropriate methodology derived from this study has to be proposed. Thus, the major contributions of this paper are as follows:

- Review and discuss the key benefits and risks of cloud-based testing.
- Provide a systematic evaluation and classification of cloud-based testing strategies, techniques, and types of tests.
- Provide a comprehensive overview of cloud-based testing tools and classify them based on their testing strategies.
- Demonstrate a case study on the selected cloud-based testing tools focused on the security testing, describe the level details of their functions and compare the results.
- Define a methodology serving as a complex guide for stakeholders in the development of their own tests using cloud computing.

In the first part of the paper, the systematic literature review, defined by Petticrew and Roberts (2006), i.e. "*A review that aims to comprehensively identify all relevant studies to answer a particular question, and assesses the validity (or ''soundness'') of each study taking this into account when reaching conclusions*", was applied. Thus, the following steps were performed:

- define search terms and keywords search strategies based on the main purpose of this paper, i.e.: "cloud computing AND testing AND tools/platforms/services";
- select sources (digital libraries) on which to perform search, i.e.: papers that are indexed Web of Science and Scopus;
- application of search terms and keywords on sources, i.e.: "classification, case study, applications, security";
- assess the validity of studies identified in the search (in the context of the keywords); and
- selection of primary studies by application of inclusion and exclusion criteria on search results.

Further, this paper offers added value by means of a classification of existing approaches based on the problem they try to solve. For this purpose a comparative research method is used. For the other part of this paper, research was carried out via the service providers' websites, blogs, articles, etc., who provide the cloud-based testing tools for the various kinds of functional and non-functional testing. A case study approach is taken to address the research question, which is formulated as follows: "*Which free or open-source cloud-based testing tools can be used by stakeholders to further improve the security of their web portals?*"

This paper aims to serve as a useful guidebook of problems and solving techniques in the cloud-based testing, as well as a point of reference for future work in improving this framework or introducing novel tools and frameworks. It also provides a survey of

representative approaches, testing strategies and typical tools for cloud-based testing, classified based on the type of tests. Further, it recommends the most suitable tool for cloud-based security testing and presents a new methodology. Finally, it compares several major providers, and offers a comparative review of platforms, tools, and services used in cloud-based testing. In this context, this paper assumes the definition of cloud-based testing, provided by Riungu-Kalliosaari et al. (2016), as "*the use of computing resources, environments and infrastructures hosted in the cloud to perform testing*."

3 Cloud-Based Testing, Type of Tests and Architectures

3.1 Cloud-based testing definition, benefits and risks

Traditional testing of new software, usually performed in-house, requires costly server, storage and network devices only for a limited time. After this kind of testing, these resources are often unused for a long period and thus gaining extra cost on budget (Lněnička, 2013; Nachiyappan & Justus, 2015). The lifecycle of the traditional testing is often performed offline by test engineers before product delivery. On the contrary, cloud-based testing with its unique lifecycle stages and services provides new testing capabilities using continuous online testing and also massive scalability testing (Bai et al., 2011; Nachiyappan & Justus, 2015).

Nachiyappana and Justus (2015) introduced cloud-based testing as a form of evaluation methodology. Cloud-based testing basically aligns with the concept of cloud and SaaS (Nachiyappan & Justus, 2015). In the case of web applications, the main goal is to test them against a broad range of web technologies and protocols. It involves testing both the client and server side components of the web application. Akerele et al. (2013) stated that cloud-based testing is "*the practice of carrying out the testing phase of the software development process in the cloud, hence preventing the need for the vast capital expenditure on acquiring infrastructure, licenses and setup on customer side*." Another definition by Gao et al. (2011) introduces cloud-based testing as a process that "*refers to testing and measurement activities in a cloud-based environment and infrastructure by leveraging cloud technologies and solutions.*"

Generally, cloud-based testing can be divided into two main domains: testing in the cloud and from the cloud. According to Riungu et al. (2010), testing in the cloud supports testing by availing computing power and virtualization capabilities for software applications that are available on demand. On the contrary, testing from the cloud then generates a high load of Virtual Users (VUs) from various locations for performance testing of websites, web applications and Application Programming Interface (API)s, as would real users. These are called concurrent users. This type of testing is nowadays the most popular commercial service and it is also easier to use. Testing in the cloud is more complex task and requires much more attention to the details and stages of testing, including the definition of business requirements (Lněnička, 2013; Robinson & Ragusa, 2011). Therefore, this review is further focused only on the issue of testing from the cloud.

The findings indicated that cloud-based testing has several advantages: cost-effectiveness, rapid customization of hardware resources, effective use of resources (green ICT), pre-configured software images, easily control-scalable cloud system infrastructure, calculate and predict the application performance and scalability, on demand test services, pay-per-use, availability of services for small businesses and customers, support of better delivery of services, and reduced time to market for key business applications (Bai et al., 2011; Jun & Meng, 2011; Malhotra & Jain, 2013; Priyanka et al., 2012; Riungu-Kalliosaari et al., 2012;

Tung et al., 2014). According to the findings of Riungu-Kalliosaari et al. (2012), the primary benefit of cloud-based testing lies in the fact that it reduces costs for putting up, maintaining, and licensing internal testing environments. The flexibility to acquire a testing environment as needed and global access for both providers and customers are other benefits (Malhotra & Jain, 2013). On the other hand, performing this kind of testing may require special technical skills to generate test cases and scripts as well as the necessity to monitor security may also incur additional costs (Malhotra & Jain, 2013).

Security and privacy are one of the most important requirements for effective cloud computing and have often been cited as a risk by previous studies (Riungu-Kalliosaari et al., 2016). Web applications are constantly exposed to various threats and attacks, such as cross-site scripting, SQL injection, insecure configurations, or remote command execution vulnerabilities (Shklar & Rosen, 2009; Tung et al., 2014). Since the success of the testing requires fast, reliable, and robust internet connection, it is necessary to pay attention to this issue. A transmission path between the provider and the customer is also important (Akerele et al., 2013; Zissis & Lekkas, 2012). Further, the organizations should test the security, prevention capability, protection strategy, and recovery standards of a cloud service provider (Ajay & Umamaheswari, 2016). For this purpose, Zech (2011) proposed a model-driven methodology for security testing of cloud environments, ingesting misuse cases, defined by negative requirements derived from risk analysis. Zech et al. (2012) extended this methodology to test the security of a cloud computing environment among all layers. Their approach then exploited the public service interfaces, as they are a major source of newly introduced vulnerabilities, possibly leading to severe security incidents. However, researchers and practitioners are still in debate to agree on security and uncertainties with the management and legal aspects surrounding cloud-based testing (Riungu et al., 2010; Riungu-Kalliosaari et al., 2016).

The security testing is often applied in cloud computing as a Testing as a Service (TaaS) (Harikrishna & Amuthan, 2016). For example, Tung et al. (2014) proposed a framework of TaaS for security testing. They conducted the experiments and the results indicated that their prototype system can provide quality and stable service. In a cloud computing environment security testing should be applied on three layers: infrastructure, platform, and the service layer. Further, using PaaS or IaaS, the cloud provider itself cannot assure a customer's application security, as application specific code often introduces its own risks (Zech, 2011). This conflict in the security may be solved in the Service Level Agreement (SLA). A disaster recovery test may also provide an insight into the dependability of the testing service provider (Akerele et al., 2013).

Another aspect to take into consideration is to guarantee data security during cloud-based testing, especially in the public cloud, because data are stored in a remote location beyond an organization's legal and regulatory jurisdiction. This issue is discussed e.g. in Čapek (2012) or Zissis and Lekkas (2012) and should be solved using encryption techniques to ensure the authentication, integrity, and also confidentiality of involved data and communications on the provider's side as well as on the user's side.

3.2 Types of tests and their classification

The whole concept of cloud-based testing is categorized into three parts. The first one is the level based on the traditional V-Model of software testing which consists of unit, integration, system and acceptance testing. Next is the test type part addresses the type of test that is investigated/performed (functional, performance, security, and interoperability). The last one is the delivery model (SaaS, PaaS and IaaS). More can be found e.g. in Inçki et al. (2012).

From a practical point of view, Jun and Meng (2011) defined six steps of the main process cloud-based testing:

- logging in to provider's cloud testing website and register user information,
- applying for the test platform resources, applications need to describe the configuration requirements of virtual machine environment such as operating system version, browser version, memory size, hard disk size, hard disk speed, network bandwidth, firewall, etc.,
- reviewing the application and configuring the test platform by the service provider,
- paying a service fee by the user (online) under the rental agreement,
- logging in to the cloud testing platform until the end of the testing,
- according to the flow or time to pay the actual costs, the lease ends.

While functional testing is an established practice, non-functional testing of web applications is a challenge. Thus, the following classification is mostly focused on web application based tests that help ensure that all the expected outcomes are met. For this purpose, the following papers were reviewed: Gao et al. (2011), Harikrishna and Amuthan (2016), Malhotra and Jain (2013), Nachiyappan and Justus (2015), and Narayanan (2010).

Functional testing ensures that the business requirements are being met for both remote and local applications. This process of verification against specifications or system requirements is usually focused on how the system handles the response to various user queries and system requirements, data management, search and other features of the application. However, most of the papers reviewed did not take into account the architecture of applications as well as testing of standards. Therefore, there are these tests: (1) system testing – aiming to prove the system behavior within its own boundaries, (2) integration testing – integration of applications and their features and verification that they will work within the current infrastructure and environments, (3) user acceptance testing – on demand and on premise testing to meet all specified business requirements, and (4) service oriented architecture testing – layers and their composition; and testing of standards – together with the requirements based on the location.

Non-functional testing is focusing on the quality of application requirements. However, some authors exclude interoperability and compatibility testing and also disaster recovery testing from non-functional testing. While disaster recovery testing may be open to debate, interoperability and compatibility are ever more essential due to the existence of various communications and data standards in the cloud computing environment. Therefore, these are the most important non-functional tests.

Security testing is focused on SaaS/web applications and data security requests, vulnerabilities and risk analysis report. This provides assurance that business critical data are stored and transported safely. The main advantage is a scalable scan detection technique used by cloud providers.

- identity testing – authorization, authentication and access control,
- SQL and PHP code injection testing,
- cross-site scripting and cross-site request forgery testing,

Performance testing is performed to determine how applications perform in terms of responsiveness and stability under a particular workload of geographically targeted VUs.

- load/scalability testing – identifies the application's behavior under various conditions in conducting more realistic large-scale tests,

- stress testing – determines the stability of the application beyond normal operational capacity (peak loads under extreme conditions),
- availability testing – measures query response time and application availability to guarantee that there are no immediate downtimes,
- reliability testing – measures performance degradation over longer periods at varying load levels,
- latency testing – measures the difference between action and response time of the application,
- web browser testing – checks the performance of the application in combination with different web browsers under various conditions,

Interoperability and compatibility testing verifies if the application interacts and functions as expected with other software and hardware technologies and their combinations. Applications should be designed to be executed across various technologies and platforms with different components. Disaster recovery testing aims to ensure the recovery of the application, related data and services in the cloud computing environment. Verification have to be done to ensure the service is back online with minimum effects on the business (business continuity plan).

3.3 Cloud-based testing architectures and services

Cloud testing system is designed based service-oriented concepts, platform, and architecture. It should include the eight types of function modules: scalable test environment service, multi-tenant test modeling and adequacy service, on demand automated test and control service, digital test management service, test solution integration and composition service, test tracking and monitor service, large-scale test simulation service, testing contracting and billing service for continuous testing (Gao et al., 2013; Lněnička, 2013).

From a technical point of view, for all testing tasks a provider's infrastructure is used, when small and medium-size providers sometimes use computing resources of Amazon, Google, Microsoft or the other big cloud computing providers. Their know-how is then only in the service orchestration and pre- and post-processing scripts (Bai et al., 2011; Li et al., 2010). Some new architectures were proposed in recent years to provide testing functionalities as online services such as TaaS, which benefits from virtualized platform and services, massive resources, and parallelized execution of the cloud, Cloud Testing Infrastructure-as-a-Service (CTIaaS) that provides secured access to storage, hardware, networking components (including load balancers) and servers over the internet for testing and development purposes, Cloud Testing Platform-as-a-Service (CTPaaS), which provides a platform to development teams for functional testing purposes, Cloud Testing Software-as-a-Service (CTSaaS) or Test Support-as-a-Service (TSaaS). In these cases, the provider offers a so called virtual lab, which covers only the selected or all the cloud-based testing lifecycle stages. The customer does not need to have in-depth knowledge required for parameters setup or architecture configuration (Akerele et al., 2013; Bai et al., 2011; Gao et al., 2013; Harikrishna & Amuthan, 2016; Malhotra & Jain, 2013; Riungu et al., 2010).

Further, there also various studies focused on a proposal of own solution or improvements of already existing architectures. For example, Hsieh et al. (2014) proposed a cloud testing service that combines cloud computing and multi-testing tools. This service provides users an option to deploy the testing clients in different countries to access the testing targets. Kuo et al. (2015) proposed a resource monitoring and management service and an automated virtual machine monitoring mechanism for an OpenStack-based cloud testing platform. Their solution ensures that all tests are effectively allocated to the virtual machines. It also reduces the waiting time for users running tests and the time required for manual management. Dai et

al. (2014) proposed a configurable cloud-based validation environment for interoperability tests between various distributed automation systems.

Together, the systematic literature review comprehensively indicates that it is critical to support easier access to cloud-based testing and related tools because it may help to increase competitiveness through a faster response to opportunities, i.e. to collaborate more effectively, to test across multiple platforms with less hardware, and to spend less on testing efforts. Also, there are not only commercial solutions, but businesses can develop and implement their own architectures and infrastructures for cloud-based testing with the use of open-source platforms such as OpenStack or CloudStack. Finally, in the context of delivery models, it can be recommended to use SaaS to test a development or deployment of new services, PaaS is focused on the testing of web services and applications or real-time systems, and IaaS is particularly used to test distributed and parallel applications.

4 Cloud-Based Testing Tools Classification and Comparison

Cloud testing tools can be basically distinguished in three categories (Lněnička, 2013). The first two categories are mostly focused on the testing from the cloud; the third category can be also used for testing in the cloud (some providers offer tools to test the infrastructure of a specific cloud provider such as Amazon, Rakspace, Microsoft or Google).

- browser based tools (accessible through a web browser) which are freely available without registration and offer a number of VUs used to test the performance from the cloud, it is possible to buy more VUs – e.g. Load Impact;
- browser based tools available after registration which usually allow the user to try out all the essential functions of this testing tool for one week to one month, then it is necessary to buy a version with the desired functionality – e.g. BlazeMeter, PractiTest or LoadStorm;
- tools that have to be installed on the client's side, registration is required, after which it is possible time to use the selected functions for some, then it is necessary to buy a version with the desired functionality – e.g. Apica LoadTest, Janova, Silk-Performer or TestMaker.

Tab. 1 shows a classification of the selected cloud-based testing tools which was proposed by the authors based on the type of tests suggested in the previous section. The first column covers the area of the functional testing. The following three columns belong to the non-functional testing. Disaster recovery testing was omitted because any of the following tools did not offer this option. It is mostly because of the complexity of this type of testing and it is usually a part of the SLA. Furthermore, the classification is not more elaborated and does not include the subcategories of the tests because of an effort to provide a comprehensive overview of the tools, rather than describing each tool in detail. This limitation may be overcome in future studies. It should be noted that this table provides a mere qualitative comparisons between the selected cloud-based testing tools and is not intended to bring any quantitative judgments about these tools. The significance of a tool should be quantified only in the context of a specific application or a problem at hand.

TOOL'S NAME	TYPE OF TESTS				FREE OR TRIAL OFFER
	functional	security	performance	inter-operability	
AgileLoad	NO	NO	YES	YES	only registration (with 10 VUs)
Apache JMeter	YES	NO	YES	NO	open-source
Apica LoadTest	NO	NO	YES	YES	only registration (with 500 VUs)
AppPerfect	YES	NO	YES	YES	none
Appvance	YES	YES	YES	YES	none
BlazeMeter	NO	NO	YES	NO	14 days trial
Blitz	NO	NO	YES	NO	none
Burp Suite	YES	YES	YES	NO	free edition
CloudTest Lite	YES	NO	YES	YES	only registration (with 100 VUs)
Dynatrace	YES	NO	YES	YES	30 days trial
Flood IO	NO	NO	YES	NO	free account
Gatling	NO	NO	YES	NO	open-source
HP LoadRunner	NO	NO	YES	YES	only registration (with 50 VUs)
Janova	YES	YES	NO	YES	none
LoadBooster	NO	NO	YES	NO	50 VUs
LoadComplete	YES	NO	YES	NO	free edition
LoadFocus	NO	NO	YES	NO	only registration
LoadStorm	NO	NO	YES	NO	10 VUs
Load Impact	NO	NO	YES	NO	25 VUs
Monitis	YES	NO	YES	YES	15 days trial
NeoLoad	NO	NO	YES	YES	only registration (with 50 VUs)
Netsparker Cloud	NO	YES	NO	NO	free edition
nResult	YES	YES	YES	YES	none
OWASP ZAP	NO	YES	YES	NO	open-source
Parasoft SOAtest	YES	YES	YES	NO	none
PractiTest	YES	NO	NO	YES	14 days trial
Proxy Sniffer	NO	NO	YES	NO	20 VU
Performance Tester	NO	NO	YES	NO	none
ReQtest	YES	YES	NO	NO	10 days trial
SandStorm	NO	NO	YES	YES	free trial
Sauce Labs	YES	NO	NO	YES	14 days trial
SoapUI	YES	NO	YES	YES	open-source
Silk-Performer	NO	NO	YES	YES	45 days trial
TestComplete	YES	NO	YES	YES	30 days trial
TestMaker Community	YES	NO	YES	YES	open-source
Test Studio	YES	NO	YES	NO	free trial

Tosca Testsuite	YES	NO	YES	YES	14 days trial
Wapiti	YES	YES	NO	NO	open-source
WAPT	NO	YES	YES	NO	30 days trial
Watcher	NO	YES	NO	NO	open-source
Websecurify Suite	YES	YES	NO	NO	free trial

Tab. 1. *Comprehensive comparison of the selected cloud-based testing tools. Source: Authors.*

5 Security Testing of the FRS Regional Portals

As stated above, the research question was formulated as follows: "*Which free or open-source cloud-based testing tools can be used by stakeholders to further improve the security of their web portals?*" The main goal of this case study is thus to deploy and compare selected cloud-based testing tools in a real life situation. For this purpose, the official portals of the Fire Rescue Service (FRS) of the Czech Republic (CR) on the regional level were tested. The reason is that the FRS of the CR protects life, health of citizens and property against fires and offers assistance at extraordinary events (e.g. natural emergencies, industrial break downs or terrorist attacks). It is an important part of the Integrated Rescue System (IRS) together with other basic IRS bodies: Police of CR and Medical Rescue Service. In the case of an extraordinary event, an official portal may be an important source of information for the citizens.

Furthermore, it should be noted that the case study is focused on the testing from the cloud, i.e. it deploys and compares tools, which are a part of the cloud-based testing ecosystem comprising a range of tools using the cloud computing technologies to provide online services. Although some of the tools used in the study may not be hosted in the cloud, they are used as an extension of more robust tools to give a complex results in a more efficient way than traditional testing. Finally, the case study should also help to answer a question whether the unified portal is more secure than own portal on the regional level.

There is a unified portal for all the regional FRS headquarters in the CR, which is available at http://www.hzscr.cz/. Nevertheless, some regions also have their own portal, namely Liberec Region and South Moravia Region, as can be seen from the Tab. 2. The main reason to choose these portals is the fact that they can be used as backdoors to penetrate other information systems or databases. A reduction of these vulnerabilities that may affect critical infrastructure is one of the major objectives of the European Union (EU). Nowadays, it is crucial to update these requirements in the context of cybersecurity.

REGION'S NAME	WEB PORTAL – OFFICIAL (ONLINE)	WEB PORTAL – ARCHIVE
Central Bohemia Region	under http://www.hzscr.cz/	http://www.hzskladno.cz/
Hradec Králové Region	under http://www.hzscr.cz/	http://www.hzshk.cz/
Karlovy Vary Region	under http://www.hzscr.cz/	http://www.hzs-kvk.cz/
Liberec Region	http://www.hzslk.cz/	-
Moravia-Silesia Region	under http://www.hzscr.cz/	http://www.hzsmsk.cz/
Olomouc Region	under http://www.hzscr.cz/	http://archiv.hzsol.cz/
Pardubice Region	under http://www.hzscr.cz/	http://www.hzspa.cz/
Plzeň Region	under http://www.hzscr.cz/	http://www.hzspk.cz/
Region of the Capital City of Prague	under http://www.hzscr.cz/	http://www.hzspraha.cz/
South Bohemia Region	under http://www.hzscr.cz/	http://www.hzscb.cz/
South Moravia Region	under http://www.hzscr.cz/ http://www.firebrno.cz/	-
Ústí nad Labem Region	under http://www.hzscr.cz/	-
Vysočina Region	under http://www.hzscr.cz/	-
Zlín Region	under http://www.hzscr.cz/	http://archiv2.hzszlk.eu/

Tab. 2. *A list of the tested FRS regional portals. Source: Authors.*

For the purpose of this case study, five cloud-based testing tools were selected from the Tab. 1. The main criteria for the selection were: free or open-source license and support of security testing. The first tool selected was Burp Suite, which is an integrated platform for performing security testing of web applications. It offers various tools supporting the entire testing process, from initial mapping and analysis of an application's attack surface, through to finding and exploiting security vulnerabilities. The second one was Netsparker, which can find and report vulnerabilities like SQL injection and cross-site scripting and security issues on all web applications and websites as well as the Wapiti. OWASP ZAP is an easy to use integrated penetration testing tool for finding vulnerabilities in web applications. Watcher is an extension of Fiddler to find web application vulnerabilities. In this case study, only the functions of the selected tools, which are comparable, were used. Every test for the selected tool was repeated 10 times (two times a day from Monday to Friday) to ensure consistency and accuracy of results. The settings of the tools were used in their default mode, i.e. the main difference between them is represented by their databases of vulnerabilities, which they are able to uncover in the tested portals.

Since the most of the FRS regional portals are available under the unified portal, this one, together with the portal of Liberec Region and South Moravia Region, was tested. The results are shown in the Tab. 3, when errors are classified into four categories namely high, medium, low, and informational / information to determine aspects of standards compliance. It should be also noted that the errors found are aggregated into a single category based on the concrete vulnerability, because each Uniform Resource Identifier (URI) of the portal is tested and this vulnerability is found in most of them. These statistics were calculated for each set of tests: minimum and maximum value, arithmetic mean, and standard deviation.

TOOL'S NAME AND NUMBER OF ERRORS		UNIFIED PORTAL www.hzscr.cz			SOUTH MORAVIA REGION www.firebrno.cz			LIBEREC REGION www.hzslk.cz		
Selected statistics		**Min/ Max**	**Mean**	**Std. dev.**	**Min/ Max**	**Mean**	**Std. dev.**	**Min/ Max**	**Mean**	**Std. dev.**
Burp Suite	High	0/0	0	0	3/3	3	0	5/5	5	0
	Med.	5/7	6.1	0.7	5/6	5.4	0.5	8/11	9.9	1
	Low	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	Infor.	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Netsparker	High	0/0	0	0	1/2	1.2	0,4	3/5	3.9	0.9
	Med.	2/3	2.8	0.4	1/1	1	0	3/4	3.4	0.5
	Low	4/6	4.8	0.6	7/8	7.2	0.4	7/10	8.1	1.1
	Infor.	3/3	3	0	3/4	3.8	0.4	5/5	5	0
OWASP ZAP	High	0/0	0	0	1/1	1	0	3/3	3	0
	Med.	1/2	1.3	0.5	2/2	2	0	1/2	1.8	0.4
	Low	5/6	5.4	0.7	7/7	7	0	6/8	7.5	0.8
	Infor.	2/2	2	0	1/2	1.2	0.4	3/3	3	0
Wapiti	High	0/0	0	0	2/2	2	0	3/4	3.7	0.5
	Med.	6/7	6.3	0.5	5/7	6	0.9	6/6	6	0
	Low	5/5	5	0	6/8	7.4	1	8/10	9	0.9
	Infor.	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Watcher	High	0/0	0	0	1/1	1	0	2/3	2.4	0.5
	Med.	5/5	5	0	5/6	5.2	0.4	5/5	5	0
	Low	7/9	8	0.8	3/4	3.2	0.4	8/8	8	0
	Infor.	2/2	2	0	2/2	2	0	3/3	3	0
Mean value for all tools	High		0.0	0.0		1.6	0.1		3.6	0.4
	Med.		4.3	0.4		3.9	0.4		5.2	0.4
	Low		5.8	0.5		6.2	0.5		8.2	0.7
	Infor.		1.8	0.0		2.0	0.2		3.4	0.1

Tab. 3. *Security testing results' comparison of the FRS regional portals. Source: Authors.*

While Wapiti and Watcher finished the testing in about ten minutes, Burp Suite, Netsparker, and OWASP ZAP needed almost an hour. The unified portal had no error with high severity. The portal of Liberec Region had between 2-5 categories of errors with high severity and the South Moravia Region portal had between 1-3 categories of errors. The other categories and their results are also rather against these portals. Cross-site scripting error with high severity, which enables attackers to inject client-side script into web pages viewed by other users, was found at the portal of Liberec Region, as well as South Moravia Region portal (two errors of this type). This error may be reduced by safely validating untrusted HyperText Markup Language (HTML) input and disabling JavaScript. Web browser testing may be also used to solve this problem. On the portal of South Moravia Region were also found errors with high severity in too many links that redirect users to an untrusted site.

In the category with medium severity, most of the errors came from HTML, insecure frame (external) – x-frame-options header is not included in the HTTP response to protect against

'ClickJacking' attacks, cross-site request forgery detected, application error disclosure – a page contains an error/warning message that may disclose sensitive information like the location of the file that produced the unhandled exception, etc. The majority of errors found were sorted into the category with low severity. These are: cookie not marked as HttpOnly, which should be solved by set up the HttpOnly modifier on the cookie; cross-domain JavaScript source file inclusion, where should be ensured that JavaScript source files are loaded from only trusted sources, and the sources can't be controlled by end users of the application; a private IP such as 10.x.x.x, 172.x.x.x, 192.168.x.x has been found in the HTTP response body; web browser cross-site scripting protection is not enabled, etc. The unified portal had the average of errors obtained by the tools used: medium (4), low (6). For the portal of Liberec Region it was: medium (4), low (6), South Moravia Region portal: medium (5), low (8). In the category Informational / Information, which was not primarily targeted in this case study, the highest number of errors was found in the portal of Liberec Region. It has significant problems with the current standards of the website development. Also, developers should check their portals for path-relative style sheet import vulnerabilities, which can be resolved by not using path-relative URLs in style sheet imports, and HTML uses unrecognized charset.

Then, a single factor ANOVA (Analysis of Variance) in Microsoft Excel was performed to find statistical significance between the means. The null hypothesis that the means for the selected portals are all equal was tested. It was tested for each of four categories (high, medium, low, and informational / information) of vulnerabilities. Based on the results, the null hypothesis was rejected. The means of the three portals are not all equal. Thus, from the security point of view, it may be suggested that it is more appropriate to manage only one unified portal than a lot of portals on the regional level, also due to the economies of scale. The same analysis was also performed to compare the selected tools for each portal. All the null hypothesis were rejected, except of the null hypothesis for informational / information vulnerabilities, which means that in this category there is no difference among the tools. However, the other three categories of vulnerabilities had to be explored more thoroughly. For this purpose, a paired-sample t-test was applied. Here, most of the null hypothesis were not rejected. The observed differences between the means are not enough to claim that there is a significant difference among the tools. On the other hand, since the sample sizes were too small, there is still a need to continue this research. Nevertheless, this study provided important findings regarding the way how to use and interpret the results offered by these tools as well as compare functionalities of them.

Therefore, based on the results presented above, authors' own observations and the use of the tools in the real life situation, OWASP ZAP may be recommended as the most suitable open-source cloud-based testing tools, which can be used by stakeholders to further improve the security of their web portals. The main reasons are as follows: although Burp Suite offers a robust combination of useful features that can be used both manually and automatically to check the application, most of them are not available in the free edition, Netsparker does not support free (community) edition anymore, Wapiti and Watcher do not provide a comprehensive testing capabilities such as other tools compared, Wapiti is also a command-line application and Watcher requires to do the testing manually in most cases. These are usually used as extensions of more robust cloud-based testing tools.

6 Methodology for the Cloud-Based Testing

The purpose of methodology is to ensure that the cloud-based testing process is foreseeable and that the right results are achieved in a given amount of time. The contribution of a new

methodology is based on the definition of cloud-based testing provided by Nachiyappana and Justus (2015). The methodology was derived from the previous case study. The aim is to create a methodology serving as a complex guide for stakeholders in the development of their own tests using cloud computing. Together, the methodology and relevant tools and services help to ensure the quality of cloud-based testing. A typical cloud-based testing methodology is comprised of the following phases:

- Briefing – Realize requirement analysis for the web application in the context of business goals, priorities and strategies. Understand the key factors to consider in preparation for testing and the metrics and outcomes to focus on and ignore at each step. Some features of the tools, which are often charged, may distract the attention from the testing.
- Test planning – Define the test strategy and goals of the testing. Specify the time plan and test schedule, technical restrictions, deadlines and outcomes, type of tests, test sizes, etc. Choose the most suitable tool and the number of VUs. Pay attention to SLAs, terms of service, legislation, data security and privacy, etc. Stakeholders need training on key features of the tool chosen (trial version is usually available).
- Test design – Develop user scenarios and test cases for the selected tools. User scenario scripts define how the VUs will behave during a test, i.e. what they will do and what resources hosted in specific geographic areas to load. Validate test scenario script by using the integrated debugger for a more detailed look at its execution. In this phase, the recommendation is to use graphical editors, if available, for the scripts.
- Test execution – Based on the test scenario requirements, additional resources may be required. Monitoring of testing goals and comparing the expected and actual outcomes is the key step. Otherwise, testing parameters have to be changed.
    - The first step deals with the test configuration. The goal is to create a test case, by uploading the test scenario script, which can be executed continuously with any number of VUs throughout the test. It also includes which user scenarios to include in each test, how the load will be allocated between them, or what network types the VUs should emulate. These requirements vary between tools.
    - The second step is focused on creating a set of benchmarks to compare and find bottlenecks. The goal is to build an understanding of the current situation and iterate tests to identify issues. This step is repeated until the goals are reached.
    - The third step is designed for proactive management and monitoring. It includes monitoring of all the test cases and test results. In this step, the recommendation is to classify the test cases according to their relevance to goals of the testing. Then, export them for use in another tool and delete all the others.
- Reporting and remediation – Analyze the test results in order to generate a report and to identify possible issues. Provide these results in various formats and details for different stakeholders and purposes. Solve these vulnerabilities. Ensure that these findings will be used to update security policy, procedures, and security mechanisms on a continual basis.
- Debriefing – When all goals have been achieved, new plans should be made regarding on-going testing. The testing should be also repeated to control whether the vulnerabilities were fixed. Testing tools provide storage at low cost and may be used for archiving test cases, test results and test data. Deliver the right outcomes for the right stakeholders.

7 Discussion, Limitations and Future Research

Cloud-based testing offers more strategies and types of tests than security testing, hence, the web application may be also tested using performance tests such as load, stress and web browser testing. Cloud-based testing will require testing of additional aspects and parameters in the context of cloud computing infrastructure, e.g. load balancing, network latency, multitenancy, interoperability, etc. The literature review and the case study conducted in this paper suggested that cloud-based testing may improve testing. However, many problems related to the testing remain unsolved. Riungu-Kalliosaari et al. (2016) reported that although cloud-based testing extends both manual and automated testing offerings (in-house vs. cloud-based resources), it does not offer a generic test automation environment that covers these needs.

Nowadays, security and privacy have been primary concerns for many businesses, especially due to privacy regulations in the EU, which prohibit some types of personal data from being distributed outside the EU. Therefore, while using some of the cloud-based testing services, it is important to ensure that data centers are located in the EU and testing from other geographical regions cannot be used. There are also concerns regarding the lack of interoperability between services from different providers since they offer the services through their own interfaces (Dai et al., 2014; Inçki et al., 2012; Riungu-Kalliosaari et al., 2016). This is also closely connected to the last stage of cloud-based testing: summary of the testing results. Various tools may offer different results, especially in the context of severity classification into categories. To meet the business needs and goals of the testing, it is needed to interpret and compare the results of more than one testing cycle measurement or to use more cloud-based testing tools. This all may be barriers to the wider use of cloud-based testing.

Therefore, the first limitation of this research is the existence of different platforms, tools, and services that may provide different results. In this regard, the case study is limited to cloud-based testing tools applied. Another limitation is that the topic of cloud-based testing is still quite recent and it may be questioned if the results are sufficiently reliable for the intended purpose. To overcome this issue, it may be suggested to compare these results with non-cloud testing tools or even to conduct a comparison analysis of security standards and how these tools are supporting them.

Sometimes, it is a difficult task to meet the needs of the organizations with the most relevant and sustainable solutions. Thus, the research focusing on the business needs has to be conducted with sufficient regards to fulfil the business and industry demands. Another research activities are needed to understand the models, strategies, typical tools and frameworks for managing, planning and assessing cloud-based testing as it is applied in practice. More real-life projects and processes should be also explored to evaluate the real benefits and limitations of cloud-based testing. Finally, since a lot of web applications are nowadays hosted in the cloud, they should be also tested in the cloud using other tools focusing on this area.

8 Conclusion

Testing by using cloud computing has its own specifics that demand for original testing strategies, tests, and tools. Since no two web applications are alike but the computing resources, usage pattern and other common characteristics can be utilized in cloud-based testing, which will lead to decrease of costs and increased efficiency of the application.

Cloud-based testing is a new approach, which can help to measure accurate performance and security requirements of these applications and overcome the limitations of traditional testing. Compared with traditional testing methods, cloud-based testing from the cloud focuses more on online testing in which it takes the advantages of cloud computing.

This paper contributed to the growing interest in adopting and using of cloud-based testing. It provided an overview of how cloud-based resources can be used to perform testing, especially in the case of a need to quickly test security. The classification of the most important tools then enables to select the appropriate tool and apply it to cloud-based testing. The contribution of this paper is not only this classification but also a systematic review of related research papers and studies, which clarifies the main testing strategies and type of tests as well as related benefits and risks. Besides that, the case study presented may offer an important insight into the ways in which the risks and security issues associated with web applications are evaluated.

References

Ajay, D. M., & Umamaheswari, E. (2016). An Initiation for Testing the Security of a Cloud Service Provider. In V. Vijayakumar & V. Neelanarayanan (Eds.), *Proceedings of the 3rd International Symposium on Big Data and Cloud Computing Challenges* (pp. 33-41). Cham: Springer. doi: 10.1007/978-3-319-30348-2_3

Akerele O., Ramachandran M., & Dixon M. (2013). Testing in the Cloud: Strategies, Risks and Benefits. In Z. Mahmood & S. Saeed (Eds.), *Software Engineering Frameworks for the Cloud Computing Paradigm. Computer Communications and Networks* (pp. 165-185). London: Springer. doi: 10.1007/978-1-4471-5031-2_8

Bai, X., Li, M., Chen, B., Tsai, W.-T., & Gao, J. (2011). Cloud Testing Tools. In *Proceedings of the IEEE 6th International Symposium on Service Oriented System Engineering* (pp. 1-12). New York: IEEE. doi: 10.1109/SOSE.2011.6139087

Čapek, J. (2012). Cloud Computing and Information Security. *Scientific Papers of the University of Pardubice, Series D, Faculty of Economics and Administration*, 18(24), 23-30.

Dai, W. W., Riliskis, L., Vyatkin, V., Osipov, E., & Delsing, J. (2014). A configurable cloud-based testing infrastructure for interoperable distributed automation systems. In *Proceedings of the 40th Annual Conference of the IEEE Industrial Electronics Society* (pp. 2492-2498). New York: IEEE.

Gao, J., Bai, X., & Tsai, W. T. (2011). Cloud Testing – Issues, Challenges, Needs and Practice. *Software Engineering: An International Journal*, 1(1), 9-23.

Gao, J., Bai, X., Tsai, W. T., & Uehara, T. (2013). Testing as a Service (TaaS) on Clouds. In *Proceedings of the IEEE 7th International Symposium on Service Oriented System Engineering* (pp. 212-223). New York: IEEE. doi: 10.1109/SOSE.2013.66

Harikrishna, P., & Amuthan, A. (2016). A Survey of Testing as a Service in Cloud computing. In *Proceedings of the International Conference on Computer Communication and Informatics* (pp. 1-5). New York: IEEE. doi: 10.1109/ICCCI.2016.7479949

Hsieh, S. J., Yuan, S. M., Luo, G. H., & Chen, H. W. (2014). A flexible public cloud based testing service for heterogeneous testing targets. In *Proceedings of the 16th Asia-Pacific Network Operations and Management Symposium* (pp. 1-3). New York: IEEE. doi: 10.1109/APNOMS.2014.6996521

Inçki, K., Ari, I., & Sözer, H. (2012). A Survey of Software Testing in the Cloud. In *Proceedings of the IEEE Sixth International Conference on Software Security and Reliability Companion* (pp. 18-23). New York: IEEE. doi: 10.1109/SERE-C.2012.32

Jun, W., & Meng, F. (2011). Software Testing Based on Cloud Computing. In *Proceedings of the International Conference on Internet Computing Information Services* (pp. 176-178). New York: IEEE. doi: 10.1109/ICICIS.2011.51

Kuo, J. Y., Liu, C. H., & Yu, W. T. (2015). The Study of Cloud-Based Testing Platform for Android. In *Proceedings of the IEEE International Conference on Mobile Services* (pp. 197-201). New York: IEEE. doi: 10.1109/MobServ.2015.36

Li, A., Yang, X., Kandula, S., & Zhang, M. (2010). CloudCmp: Comparing public cloud providers. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement* (pp. 1-14). New York: ACM.

Lněnička, M. (2013). Cloud-Based Testing of Business Applications and Web Services. *Scientific Papers of the University of Pardubice, Series D, Faculty of Economics and Administration*, 20(26), 66-78.

Malhotra, R., & Jain, P. (2013). Testing Techniques and its Challenges in a Cloud Computing Environment. *SIJ Transactions on Computer Science Engineering & its Applications*, 1(3), 88-93.

Nachiyappan, S., & Justus, S. (2015). Cloud Testing Tools and its Challenges: A Comparative Study. *Procedia Computer Science*, 50, 482-489. doi: 10.1016/j.procs.2015.04.018

Narayanan, C. V. (2010). Testing, the 'Cloud' Way. *Siliconindia*. Retrieved November 15, 2017, from https://www.siliconindia.com/magazine_articles/Testing_the_%E2%80%98Cloud%E2%80%99_Way-NLJO444799615.html

Petticrew, M. & Roberts, H. (2006). *Systematic Reviews in the Social Sciences: A Practical Guide*. Malden: Blackwell Publishing.

Priyanka, C., Chana, I., & Rana, A. (2012). Empirical Evaluation of Cloud-based Testing Techniques: A Systematic Review. *ACM SIGSOFT Software Engineering Notes*, 37(3), 1-9. doi: 10.1145/180921.2180938

Riungu, L. M., Taipale, O., & Smolander, K. (2010). Research Issues for Software Testing in the Cloud. In *Proceedings of the IEEE Second International Conference on Cloud Computing Technology and Science* (pp. 557-564). New York: IEEE. doi: 10.1109/CloudCom.2010.58

Riungu-Kalliosaari, L., Taipale, O., & Smolander, K. (2012). Testing in the Cloud: Exploring the Practice. *IEEE Software*, 29(2), 46-51. doi: 10.1109/MS.2011.132

Riungu-Kalliosaari, L., Taipale, O., Smolander, K., & Richardson, I. (2016). Adoption and use of cloud-based testing in practice. *Software Quality Journal*, 24(2), 337-364. doi: 10.1007/s11219-014-9256-0

Robinson, P., & Ragusa, C. (2011). Taxonomy and requirements rationalization for infrastructure in cloud-based software testing. In *Proceedings of the IEEE Third International Conference on Cloud Computing Technology and Science* (pp. 454-461). New York: IEEE. doi: 10.1109/CloudCom.2011.67

Shklar, L., & Rosen, R. (2009). *Web Application Architecture: Principles, Protocols and Practices*. Chichester: Wiley.

Tung, Y. H., Lin, C. C., & Shan, H. L. (2014). Test as a Service: A framework for Web security TaaS service in cloud environment. In *Proceedings of the IEEE 8th International Symposium on Service Oriented System Engineering* (pp. 212-217). New York: IEEE. doi: 10.1109/SOSE.2014.36

Zech, P. (2011). Risk-Based Security Testing in Cloud Computing Environments. In *Proceedings of the IEEE Fourth International Conference on Software Testing, Verification and Validation* (pp. 411-414). New York: IEEE. doi: 10.1109/ICST.2011.23

Zech, P., Felderer, M., & Breu, R. (2012). Towards a Model-Based Security Testing Approach of Cloud Computing Environments. In *Proceedings of the IEEE Sixth International Conference on Software Security and Reliability Companion* (pp. 47-56). New York: IEEE. doi: 10.1109/SERE-C.2012.11

Zissis, D., & Lekkas, D. (2012). Addressing cloud computing security issues. *Future Generation Computer Systems*, 28(3), 583-592. doi: 10.1016/j.future.2010.12.006

The article has been reviewed. | **Received:** 5 October 2017 | **Accepted:** 5 December 2017

Academic Editor: Stanislava Mildeova