Measures to Mitigate Cybersecurity Risks and Vulnerabilities in Service-Oriented Architecture

Carmen Elena CÎRNU, Carmen Ionela ROTUNĂ*,
Adrian Victor VEVERA, Radu BONCEA

National Institute for Research and Development in Informatics, 8-10 Averescu Avenue, Bucharest, 01145, Romania

carmen.cirnu@ici.ro, carmen.rotuna@rotld.ro (**Corresponding author*),

victor.vevera@ici.ro, radu.boncea@rotld.ro

Abstract: Raising awareness of cybersecurity issues and improving the digital literacy and skills in terms of recognition and management of threats is considered as a high-priority action. Service-Oriented Architecture(SOA) provides several benefits, including greater efficiency and open access to applications, services, and information, but their very openness creates unique and significant security challenges for organizations. SOA principles are widely used because they provide loose-coupling, service automation, extensible architecture and enhanced reuse. This paper addresses the security challenges of SOA considering their business and technical impact and performs a mapping to mitigation measures and tools. The process involves identifying the main security vulnerabilities and researching solutions for cyber attacks prevention. It also proposes business level measures to mitigate cybersecurity risks and vulnerabilities in SOAs. Several European projects and open source solutions whose aim is to ensure the security and the privacy of SOA are identified, analyzed and proposed as tools for enhanced security in the second part of the paper.

Keywords: SOA, Cybersecurity, Vulnerability classification, Mitigations, Open-source solutions.

1. Introduction

Service-Oriented Architecture (SOA) raises new challenges related to the information security level. Information security is concerned with what needs to be protected and why, what it needs to be protected from, and how to protect it. Establishing security involves securing data communications, data in rest, managing access rights, etc. In SOA, a coherent security view must be shared, a trust network should be established, and global security policies' enforcement should be accomplished [33].

The distributed nature of the SOA, brings new security challenges concerning unauthorized access because it involves several services and service providers. In traditional architectures, data travels from the sender to the receiver, it is processed by the receiver, and results are returned to sender. In an SOA environment, data originating from sender may travel through multiple intermediate points before arriving at the ultimate recipient. The message must be secured and the confidentiality and integrity of the data must be ensured. Therefore, SOA requires additional security components, as well as the adoption of new standards and specifications.

Researchers in [34], [1], [10] ,[35], [9] have shown many of the security challenges in deploying SOA. The current study aims to research and provide an inventory of the security and privacy vulnerabilities of SOA, investigate prevention methods and propose mitigation measures.

Section 2 provides a review of the capabilities and specific aspects of SOA while Section 3 identifies the main vulnerabilities that are classified by their nature and impact. Section 4 analyzes security standards, open source tools to mitigate security risks and performs a mapping of prevention measures, including good practices, procedures, techniques, and open source tools, to vulnerabilities previously identified. The organizational and business security measures provided by ENISA security guidelines [7] are also recommended in this section.

2. Service Oriented Architecture

Service-oriented architecture (SOA) is an architectural approach based on the use of services (such as RESTful, Web services) carrying out some small functions, for instance producing or validating data, or providing simple analytical

https://doi.org/10.24846/v27i3y201811

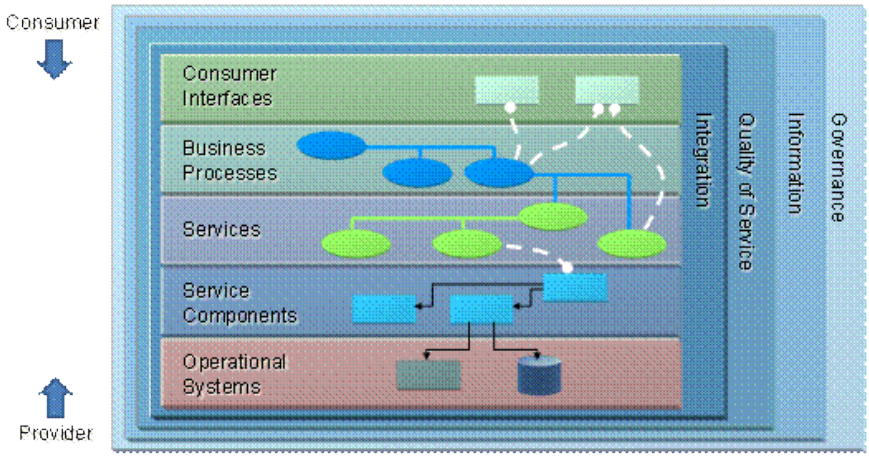

Figure 1. SOA Logical Solution View - taken from [11]

services, with the aim to achieve a loose coupling amongst interacting components. The main goals for SOA consist in facilitating the continuous development of large scale interoperable networks of systems, facilitating online provisioning and use of services and reducing costs at an organizational level [17].

SOA is referred as a business process execution environment according to the OASIS SOA reference model and architecture [11]. The differentiation between visibility, interaction, and real world effect of services will be followed during the vulnerability analysis as, according to OASIS, these are the three key concepts of the SOA paradigm. Computer systems and in particular distributed systems face several security risks which also affect a SOA. A SOA vulnerability is a vulnerability present in such an environment and vulnerabilities can exist in any of the SOA layers. Figure 1 illustrates Logical Solution View of the SOA RA, the multiple separations of concern in the nine layers of the SOA RA [11].

The basic structure of SOA comprises three main components: Service Provider, Service Registry, and Service Requestor as shown in the figure bellow. The service is a key concept and core of an SOA. It is the technical representation and encapsulation of high-level business functionality. The Service Provider is the entity that creates and provides the services; it also creates metadata for the services and publishes them in a central repository, called Service Registry (Universal Description, Discovery, and Integration - UDDI). Service Requestor is an entity that requires certain services which are published by Service Providers. The operations that are performed in a SOA are: Publish, Find, and Bind as presented in the following figure:

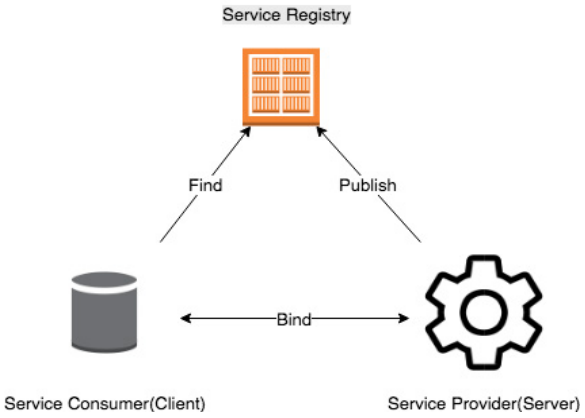

Figure 2. Basic SOA Architecture

Taken from "Alwadain, A., Korthaus, A., Fielt, E., & Rosemann, M. Integrating SOA into an Enterprise Architecture: a comparative analysis of alternative approaches. In Proceedings of the 5th IFIP International Conference on Research and Practical Issues of Enterprise Information Systems -CONFENIS. Brazil" [36]

SOA technologies, such as UDDI, and security and privacy standards such as SAML [8] and WS-Trust [2], introduce another role which addresses these issues, called a service broker [6], [34].

The key concepts of the SOA paradigm are: services visibility, interaction, and real world impact. Computer systems and in particular distributed systems, including SOA, are subject to several security risks and flaws that can occur in any of the SOA layers.

3. Identifying Most Dangerous SOA Security Vulnerabilities

Software weaknesses are errors that can lead to software vulnerabilities. A software vulnerability is an error of the software that can be directly used by a hacker to gain access to a system or network.

Software weaknesses are flaws, faults, bugs, vulnerabilities, and other errors in software implementation, code, design, or architecture [20].

Unless all weaknesses have been properly recognized and addressed, networks and systems will become vulnerable to attacks.

Example of software weaknesses include: buffer overflows, format strings, structure and validity problems, common special element manipulations, channel and path errors, handler errors, user interface errors, pathname traversal and equivalence errors, authentication errors, resource management errors, insufficient verification of data, code evaluation and injection.

Common Weakness Enumeration (CWE™) is a formal list or a dictionary of common software weaknesses that can occur in software's architecture, design, code or implementation that can lead to exploitable security vulnerabilities. CWE was created to serve as a common language for describing software security weaknesses and to provide a common baseline standard for weakness identification, mitigation, and prevention efforts.

The U.S. National Vulnerability Database (NVD) is a federal government repository of standards-based vulnerability management data. These data enable automation of vulnerability management, security measurement, and compliance (e.g., FISMA). NVD integrates CWE into the scoring of Common Vulnerabilities and Exposures (CVE®) entries, upon which NVD is built, by providing a cross section of the overall CWE structure. NVD analysts score CVEs using CWEs from different levels of the hierarchical structure.

The CWE classification with percentage of NVD entries shows the following values [30]: SQL Injection 17.85, Cross-Site Scripting (XSS) 14.58, Buffer Errors 12.88, Permissions, Privileges, and Access Control 9.04, Input Validation 7.98, Code Injection 7.8, Path Traversal 6.79, Resource Management Errors 4.97, Information Leak Disclosure 3.91, Numeric Errors 3.06, Authentication Issues 2.72, Link Following 2.26, Cross-Site Request Forgery (CSRF) 1.47, Credentials Management 1.32, Configuration 1.12, Cryptographic Issues 0.97, Format String Vulnerability 0.59, Race Conditions 0.53, OS Command Injections 0.16.

3.1. Vulnerabilities Classification

Web application vulnerabilities are: Injection attack**,** Cross-site scripting (XSS) attack**,** Cross Site Request Forgery (CSRF)**,** Protocols and session management vulnerabilities**,** Security misconfiguration, Cryptographic storage vulnerabilities**,** Username enumeration.

SOA specific vulnerabilities are:

1. Web Services Layer [29],[10] : WSDL scanning, Metadata spoofing, Attack obfuscation, Over-sized cryptography, Insufficient Logging, Insecure Configuration, Inadequate Testing, Information Leakage.

2. Business Processes Layer: BPEL scanning, Metadata spoofing, BPEL state deviation, Instantiation flooding (direct and indirect), WS-Addressing spoofing, Workflow engine hijacking.

3. SOA oriented vulnerabilities:

(a) SOAP vulnerabilities [31]: Harmful SOAP attachments, SOAP Action spoofing.

(b) XML vulnerabilities [32]: XML External Entity (XEE) attack, XPath injection, XML Denial of Service (DoS) attacks, Schema poisoning.

4. SOA - Security Vulnerabilities and their Mitigation

SOA is a collection of loosely coupled and independent services (or resources), each with a well-defined interface. Services are offered on demand and can range from a simple service, to a high level service repository, composed of multiple services. Services can be delivered to an end-user, to an application or to another service with no need for human intervention.

Table 1. SOA Web Service Security Requirements relationship with Standards

Dimension	Requirement	Specifications
Messaging	Confidentiality and Integrity	WS-Security SSL/TLS
	Authentication	WS-Security Tokens SSL/TLS X.509 CERTIFICATES
Resource	Authorization	XACML XrML
	Privacy	EPAL XACML
Negotiation	Registries	UDDI ebXML
	Semantic Discovery	SWSA OWL-S
	Business Contracts	ebXML
Trust	Establishment	WS-Trust XKMS X.509
	Trust Proxying	SAML WS-Trust
	Federation	WS-Federation
Security Properties	Policy	WS-Policy
	Security Policy	WS-Security
	Availability	WS-Reliable Messaging WS-Reliability

According to NIST, a vulnerability is defined as "a flaw or weakness in system security procedures, design, implementation, or internal controls that could be exercised (accidentally triggered or intentionally exploited)" [19] and results in a security breach or in a violation of the system security policy. In this context, in the following we examine crucial SOA security issues - including confidentiality, integrity, and availability - while providing mitigation measures helpful in increasing SOA security.

Security means integrity, reliability, and confidentiality of the system. In particular, security in a SOA ecosystem focuses on those aspects of assurance that involve the accidental or malicious intent of other people to damage, compromise trust, or hinder the availability of SOA-based systems to perform desired capability. SOA Security Oasis definition: "The set of mechanisms for ensuring and enhancing trust and confidence in the SOA ecosystem." [11].

End-to-end security capabilities include federated authentication, which intercepts service requests and adds the appropriate username and credentials; validation of each service request and authorization to make sure that the sender has the appropriate privilege to access the service; and, lastly, encryption/decryption of XML content at the element level for both message requests and responses. To address these intricate security requirements trust models, WS-Security [3] and other security related standards have been developed [34].

Relationship of SOA Web Service Security Requirements to Standards shows which security requirements are satisfied by the various specifications and standards as shown in Table 1.

Successfully implemented SOA security has to be defined, planned, and implemented to both threats and counter-measures in an agile manner. Other important aspect of SOA security are well defined service level agreements (SLAs)

and security metrics between service providers and service consumers. SOA security may also involve greater auditing and reporting to adhere to regulatory compliance established by governance structures.

4.1. Projects and Standards Addressing SOA Security

The following is a summary of security open-source tools for Web services:

secRT

The secRT is an Open Source security platform, developed by CORISECIO [13] in cooperation with the German Federal Office for Information Security (BSI), which provides a development framework that contains a full set of security functionality. Security methods may be combined with easy workflow methods to provide secure and customized business processes. Configured security processes are automatically distributed to the runtime environment and executed there.

The secRT Framework offers features such as: Entity Management which enables the integration in existing meta directories, Key Management - a complete infrastructure for PKI keys and certificates and Cryptographic functions which allow the choice of the algorithm.

WebSand

The overall goal of the WebSand FP7 project is to empower Web application developers, hosters, and users in designing, implementing, and running secure applications [27]. The project enables the specification and enforcement of three classes of security policies: fine-grained access control policies, information flow control policies and secure composition policies. As a necessary prerequisite for the enforcement of such policies, one of the aims of WebSand is to enforce a reliable separation of data and executable code, e.g. through a strict type system. This separation also counteract many types of injection attacks targeting a server or relying on injected scripts being reflected to a client-side end-user.

SWEPT

Started in March 2014 the SWEPT project [18] provides required software components that can be integrated within the current existing web development frameworks, which enable the creation of highly secure websites. SWEPT is the first technology which aims to create self-protected web applications and web services. The main goal is to develop a new multifaceted approach to mitigate malicious attacks on websites by maximizing the security posture of websites with a minimum of intervention needed by website owners and administrators. In addition, the project also aims to define a defacto standard and good practice for securing websites.

w3af

w3af is a Web Application Attack and Audit Framework with the goal to create a framework to help secure web applications by finding and exploiting all web application vulnerabilities [14]. The framework is easy to use and extend, and licensed under GPLv2.0.

OWASP SKF

The Open Web Application Security Project (OWASP) Security Knowledge Framework relies on OWASP Application Security Verification Standard and other resources and is intended to be a tool used as a guide for building and verifying secure software. OWASP SKF provides a free, open source web application security system, which also serves as a training tool to teach developers about application security. The SKF supports software developers throughout the product lifecycle, ensuring security in both pre-development and post-release updates. It analyzes the processing techniques that developers use to edit their data, then matches those patterns to known security vulnerabilities. After providing descriptions of linked vulnerabilities and offering feedback on how to implement solutions, the SKF validates if security fixes were implemented correctly.

The WS-security standard

WS-Security is a web services security standard published by OASIS consortium developed with the aim to enforce integrity and confidentiality on messages [22]. WSS relies on existing security standards and specifications (such as X.509, SAML assertions, Kerberos, XML digital

signatures, and XML encryption [26], etc.) to define a framework for embedding the security information within a SOAP message. WSS defines an XML element called Security which is inserted in the SOAP header and contains integrity, identity, and confidentiality information and gives the receiver the information necessary to decrypt and validate the message.

WS-Trust

WS-Trust describes the model for establishing both direct and brokered trust relationships including intermediaries [12]. The Web Services Trust Language (WS-Trust) uses the secure messaging mechanisms of WS-Security to define additional primitives and extensions for the issuance, exchange and validation of security tokens. WS-Trust also enables the issuance and dissemination of credentials within different trust domains. To secure a communication between two parties, the two parties must exchange security credentials (either directly or indirectly). However, each party needs to determine if it can 'trust' the asserted credentials of the other party. This specification defines extensions to WS-Security for issuing and exchanging security tokens and ways to establish and access the presence of trust relationships. Using these extensions, applications can engage in secure communication designed to work with the general Web services architecture.

UDDI

Produced by OASIS, Security for Universal Description, Discovery and Integration (UDDI) allows Web services to be easily located and subsequently invoked. Security for UDDI enables publishers, inquirers and subscribers to authenticate themselves and authorize the information published in the directory.

4.2. SOA Vulnerabilities Mapping to Mitigation

An attacker may attempt to compromise the security of a SOA in several ways but the main sources of attacks are third parties, that seek to subvert interactions between legitimate participants, and social engineering techniques. In such a complex system with multiple ownership boundaries and trust boundaries, it is important to understand the threats in order to effectively secure all components: messages, services and data storage. Any of the contributing technologies can be the subject of a threat but the risks can be minimized by using mitigation measures as cryptography standards and security technologies.

SOA vulnerabilities classification takes into account vectors such as exploitability, preva-lence, detectability and impact and is in line with the top flaws identified by OWASP [24],[23]. The top threats of SOA with easy exploitability, common prevalence, easy detectability and severe impact are: threats on Web Service implementation, insufficient authentication and authorization and XML specific attacks.

4.2.1. Threats on Web Service Implementation

These threats, also called Injection flows, target at finding and exploiting a vulnerability in the code of the deployed web service. The attacker sends malicious code to the interpreter in order to alter the existing data, to produce an error, cause repudiation issues such as voiding transactions, gain database administrator privileges, destroy the data or make it unavailable.

Exemples of vulnerabilities are SQL Injection, XPath Injection, Cross-site Scripting.

An efficient mitigation measure is automated dynamic scanning of the application for web vulnerabilities such as CSRF, SQL Injection, XSS. A simple technique for preventing Injection is parameterized query usage where placeholders are used for parameters and supplied when executing the query.

Other measures include:

- checking the code is an efficient method to see if the application uses interpreters safely;
- statements received from users should be validated by means of appropriate threat-detection rules before executing;
- prevent usage of weak password or default password;
- use encrypted connections – HTTPS.

Open-source solutions W3af Web Application Attack and Audit Framework, InSpec and WebSand can be used to increse the level of security.

4.2.2. Insufficient Authentication and Authorization

Authentication verifies the identity of the user while Authorization establishes whether the interaction is legitimate, ensuring that the involved participants have permission to participate in the interaction. An authenticated, but unauthorized attacker can try to access unauthorized data and functions when access control information is not properly defined.

Examples of vulnerabilities are Dictionary attacks, IP Spoofing, Message Eavesdropping, Brute force attacks, Credential theft, Elevation of privilege.

To prevent these type of attacks PKI, multi-factor authentication, or the newer XML security-based technologies such as XML-Signature and SAML should be used. Also strong password policies, including storing credentials in a secure manner and using authentication mechanisms, that do not require clear text credentials to be passed over the network are also efficient mitigation measures.

Other recommended mitigation measures are:

- Authentication Confirmation Using Tokens & Certificates;
- Access Control usage controls what an „authorized" user is allowed to do;
- Cryptographic random number generators to generate session IDs.

4.2.3. XML Specific Attacks

XML is a versatile data-encoding standard. However, parsing XML can be processor intensive and complex, which can lead to security issues. One common issue is a denial of service (DOS) against a web service. If an attacker crafts an XML message with very large payloads, recursive content, excessive nesting, malicious external entities, or with malicious DTDs (Data Type Documents), a DOS can occur.

The use of XML for messaging makes the infrastructure particularly prone to attacks targeting those components processing the messages [29]. These threats may at least make the infrastructure unavailable and at worst compromise the whole system by giving access to unauthorized users.

Examples of XML specific attacks are XML-bombs (XML documents with endless recursions), schema poisoning (an alteration in the XML Schema of a message, leading to inconsistencies), XEE atack (XML External Entity attack is a type of attack against an application that parses XML input), XML Denial-of-Service (an attacker tries to prevent legitimate users from accessing a service by flooding the service with thousands of requests) or Recursive Payloads (where the attacker tries to send too deeply nested data to the web server so that the XML parser will be heavily stressed).

A validation service acting as a security proxy or "filter" to any application service can efficiently stop the threat. Another mitigation is to use filters, XML Gateways, or XML parser options when processing XML to prevent parsers from processing malicious messages.

Examples of tools for preventing XML attacks are OWASP SKF and SWEPT. WS-Security(WSS) and WS-Trust and UDDI standards implementation also increases the security level.

4.3. Assessment Frameworks/ Tools: ENISA Technical Guideline on Security Measures

Enisa defines security incidents as "A breach of security or a loss of integrity that could have an impact on the operation of electronic tele-communications networks and services " [7].

Enisa's technical guideline on security measures is useful for assessing cybersecurity, as the guideline basically addresses physical, network and software systems implemented for end-users, as well as the human resource management of these systems.

Also the security guideline addresses the security issues mainly in a post implementation scenario and provide mitigations to most frequent security vulnerabilities.

The security objectives within the guideline have been derived from a set of international and national standards that are commonly used by providers in the EU's electronic communication sector. For these security objectives there is a list of security measures which could be implemented by providers to reach the security objective depending on the level of security intended to be achieved.

The most important security measures that are recommended in the Enisa guideline are:

- Governance and risk management
- Human resources security
- Security of systems and facilities
- Operations management
- Incident management procedures
- Business continuity management
- Monitoring, auditing and testing.

Risks vary for different service providers and they depend on specific details like provider type, services type and so on.

5. Conclusion

SOA and Web Service technologies offer many benefits and are revolutionizing the way software interacts, but the real benefits are usually accompanied by serious security flaws. Being aware of the risks to Web Services before deploying Web Services in a SOA can help improve the overall security.

Better protection against cyber-attacks requires national and transnational collaboration mechanisms, allowing access to external resources (e.g. cybersecurity research and development, tailored information, certified training) and experiences (e.g.cooperation).

The study addressed the main SOA vulnerabilities considering exploitability, prevalence, detectability and impact and performed a mapping to mitigation measures and tools. The current study provides an analysis of the capabilities and specific aspects of SOA and identifies the main vulnerabilities that are classified by their nature and impact. It also analyzes security standards and open source tools to reduce security risks. The result consists in mapping research on prevention measures, including best practices, procedures, techniques and tools designed to mitigate the open source vulnerabilities previously identified. Thus performing threat assessments, creating mitigation strategies, and determining acceptable levels of risk are the keystone for an effective process to mitigating threats in an efficient way. The result can be an acceptable level of risk to the safety and integrity within any SOA ecosystem.

REFERENCES

1. Altaani, Noor A. & Jaradat, A. (2012). *Security Analysis and Testing in Service Oriented Architecture*.

2. Anderson, S. et al. (February 2005). *Web Services Trust language (WS-Trust)*. Public draft release, Actional Corporation, BEA Systems, Computer Associates International, International Business Machines Corporation, Layer 7 Technologies, Microsoft Corporation, Oblix Inc., OpenNetwork technologies, Ping Identity, Reacticity, RSA Security, and Verisign.

3. Atkinson, B. et al. (April 2002). *Web Services Security (WS-Security)*, Technical report, Microsoft, IBM and Verisign.

4. Boncea, R. & Bacivarov, I. C. (January-March 2016). Security in Internet of Things: Mitigating the Top Vulnerabilities, *Quality Assurance Journal*, *XXII*(85), 11-17. ISSN 1224–5410.

5. Bowen, P., Hash, J. & Wilson, M. (October 2006). Information Security, *NIST Special Publication 800-100*.

6. Colan, M. (April 2004). *Service-Oriented Architecture expands the vision of Web services, Part 2*, IBM DeveloperWorks.

7. *ENISA Technical Guideline on Security Measures*, Version 2.0, October 2014.

8. Farell, S. et al. (July 2003). *Assertions and protocol for the OASIS security*

assertion markup language (SAML), V1.1. Committee specification, OASIS.

9. Hafner, M. & Breu, R. (2008). *Security engineering for service-oriented architectures*. Springer Science & Business Media.

10. Harney et al. (2006). Web Services Vulnerability Assessment 2004, *Group Secure Association Key Management Protocol* (online), http://www.ietf.org/rfc/rfc4535.txt.

11. http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/cs01/soa-ra-v1.0-cs01.pdf (December 2012). *OASIS Reference Architecture Foundation for Service Oriented Architecture Version 1.0*.

12. http://docs.oasis-open.org/ws-sx/ws-trust/200512, Oasis WS Trust.

13. http://opensource.corisecio.com/?id=secrt_framework, Corisecio Framework.

14. http://w3af.org/, w3af Web Application Attack and Audit Framework.

15. http://www.cloudcomputing-news.net/ news/ 2014/nov/21/top-cloud-computing-threats-and-vulnerabilities-enterprise-environment/, Top Cloud Computing Threats.

16. http://www.computerworld.com/article/2565944/security0/sans-unveils-top-20-security-vulnerabilities.html, Top 20 Security vulnerabilities.

17. http://www.opengroup.org/soa/source-book/soa_refarch/p5.htm - SOA Reference Architecture, *The Open Group*.

18. http://www.swept.eu/, Swept Project.

19. https://csrc.nist.gov/cyberframework (2018). *NIST Cybersecurity framework*.

20. https://cwe.mitre.org/, Common Weakness Enumeration (CWE) Community.

21. https://www.kb.cert.org/vuls/, Vulnerability Notes Database.

22. https://www.oasis-open.org/committees/tchome. php?wg_abbrev=wss, OASIS Web Services Security (WSS) TC.

23. https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project, Owasp Top Ten Vulnerabilities.

24. https://www.owasp.org/index.php/OWASP_Security_Knowledge_Framework, Owasp Security Knowledge Framework.

25. https://www.sifassociation.org/NewsRoom/White%20Papers/Data%20Privacy%20Security%20and%20Interoperability.pdf, SIF Association, Data Privacy, Security and Interoperability.

26. https://www.w3.org/TR/xmlenc-core1/, XML Encryption Syntax and Processing Version 1.1.

27. https://www.websand.eu, Websand Project.

28. https://www2.opengroup.org, The Open Group, SOA white paper.

29. Lindstrom, P. (2004). *Attacking and Defending Web Services*, White-paper, Spire Security.

30. Lowis, L. & Accorsi, R. (2009). On a Classification Approach for SOA Vulnerabilities. In *33rd Annual IEEE International Computer Software and Applications Conference*, 2009, Seattle, WA (pp. 439-444). doi: 10.1109/COMPSAC.2009.173.

31. Mohamed, B. I. & Mohamed, S. A. R. (Mar-Apr. 2014). SOA Security Threats on SOAP Web Services–A Critical Analysis, *IOSR Journal of Computer Engineering (IOSR-JCE), 16*(2), Ver. XI, 135-141. e-ISSN: 2278-0661, p- ISSN: 2278-8727.

32. Moradian, E. & Hakansson, A. (2006). Possible attacks on XML Web Services, *IJCSNS International Journal of Computer Science and Network Security, Vol. 6*, 154-170.

33. Nassar, B. P., Badr, Y., Biennier, F. & Barbar, K. (2012). Securing Collaborative Business Processes: A Methodology for Security Management in Service-Based Infrastructure. In: Frick, J. & Laugen, B.T. (eds.) *Advances in Production Management Systems. Value Networks: Innovation, Technologies, and Management.* APMS 2011. *IFIP Advances in Information and Communication Technology*, *Vol. 384*. Springer, Berlin, Heidelberg.

34. Papazoglou, M. P. & Van Den Heuvel, W. J. (2007). Service oriented architectures: approaches, technologies and research issues, *The VLDB Journal*, *16*(3), 389-415.

35. Phan, C. (2007). Service Oriented Architecture (SOA) - Security Challenges and Mitigation Strategies. In *MILCOM 2007 - IEEE Military Communications Conference* (pp. 1-7).

36. Alwadain, A., Korthaus, A., Fielt, E. & Rosemann, M. Integrating SOA into an Enterprise Architecture: a comparative analysis of alternative approaches. In *Proceedings of the 5th IFIP International Conference on Research and Practical Issues of Enterprise Information Systems -CONFENIS*. Brazil.