

Federal Institute of science and Technology  
Angamaly  
Mookkannoor

# A LITERATURE SURVEY ON PERFORMING VULNERABILITY TESTING IN WEB SERVICES

Bimal Varghese  
bimalvv2005@gmail.com  
Simi Stephen  
simistephen@fisat.ac.in

NCIE

March 18, 2015



## Introduction

SOA & Web Service

## Survey Works

Automated Robustness Testing

Perturbation Based Testing

Message Exchange by SOAP Bundling Framework

Using Data Perturbation

Adaptive Random Testing

XML Perturbation

Combinatorial Mutation

API based security solutions

Worst-Input Mutation Approach



### SOA & Web service

- ▶ An architectural style that aims to achieve loose coupling among interacting software agents.
- ▶ Service Provider & Service Consumer are implemented via software agents
- ▶ **Web Service:** Popular way of implementing a SOA.
- ▶ Way of Integrating web based applications using standards such as XML, SOAP, WSDL & UDDI
- ▶ To ensure quality and reliability of Web Service, proper testing must be conducted

# Survey Works

# [1] Automated Robustness Testing of Web Service



Evan Martin et al presents a framework for automatically generating and executing web service request.

## Method

Consist of 3 steps

### 1. Code Generator

1.1 Generate necessary code to implement a service consumer

### 2. Test Generation

2.1 Generated test class is supplied to a test generation tool such as JCrasher inorder to generate JUnit test.

### 3. Test Execution

3.1 Invoke test case and call web service.  
Collect response from web service.



## Advantages

- ▶ Easy framework for automatically invoking web service given a service provider's WSDL
- ▶ Leverage existing automated unit test generation tools to generate unit test cases.
- ▶ No knowledge of underlying service implementation is required.

## Disadvantages

- ▶ Generalized black box testing method.
- ▶ Cannot categorize generated errors.
- ▶ Cannot identify errors within data  
Eg: Age=-10

## [2] Exploring Perturbation Based Testing for Web Service



An extended approach based on XML message perturbation has been proposed in the paper **Exploring Perturbation Based Testing for Web Service**.

- ▶ Utilized SOAP Perturbation operators and a Web service Testing tool ( **SMAT-WS** )

### SOAP Perturbation Operator

SOAP perturbation primitive operators are

1. *Null* ( $n$ )
2. *Incomplete* ( $n$ )
3. *Inversion* ( $n$ )
4. *ValueInversion* ( $n$ )
5. *Mod\_Len* ( $n$ )
6. *Space* ( $n$ )



## Advantages

- ▶ Consider SOAP message perturbation for Web service Testing.
- ▶ Consider various data perturbation operators for testing
- ▶ Created a Testing tool (SMAT-WS) for testing Web Services

## Disadvantages

- ▶ Designed mutation operators is not sufficient for comprehensive testing.
- ▶ Issues related to amount of test cases generated by the operators is not considered.



# [3] Efficient Web Services Message Exchange by SOAP Bundling Framework

Toshiro Takase, Keishi Tajima



- ▶ A SOAP message bundling framework is Proposed.
- ▶ Framework enables bundling of multiple messages into one message.
- ▶ Application developers do not have to consider the service granularity for performance reasons.



## Advantages

- ▶ Existing service providers do not have to re-implement their service application.
- ▶ Bundling framework generates new operations for all combinations of the original operations in the WSDL
  - ▶ If original WSDL has  $N$  operations,  $2^{N-1}$  operations are generated.

## Disadvantages

- ▶ Existing service requesters have to change their application a little to take advantage of the bundled operations.
- ▶ Multiple Interdependent functions are not considered.

# [4] Generating Test Cases for Web Services Using Data Perturbation



- ▶ Existing XML messages are modified based on rules defined on the message grammars, and then used as tests.
- ▶ Data perturbation uses 2 methods to test Web services:
  - Data value perturbation:**  
modifies values according to the data type.
  - Interaction perturbation:**  
classifies the communication messages into two categories:  
RPC communication and data communication



## Adaptive Random Testing

### Principle

- ▶ Given a set of previously executed test cases that have not revealed any failures, new test cases located away from these old ones are more likely to reveal failures.

### Types of ART methods

- ▶ Fixed Size Candidate Set ART (FSCS-ART):  
Candidate with largest distance from current Test case is considered next.
- ▶ Restricted Random Testing (RRT):  
Create Exclusion zone for current test case.  
Take random selection if it lie outside the zone.



- ▶ Web services uses XML to describe and transmit data.
- ▶ XML schema is utilized to generate data formats and test cases.

### XML Data Model

- ▶ An XML schema can be modeled as a tree.  
XML tree  $T = (N, D, X, E, n_r)$ , where:
  - ▶  $N$  is a finite set of elements and attribute nodes.
  - ▶  $D$  is a finite set of built-in and derived data types.
  - ▶  $X$  is a finite set of constraints (integrity and representation).
  - ▶  $E$  is a finite set of edges.
  - ▶  $n_r$  is the root node.

# [7] Combinatorial Mutation Approach to Web Service Vulnerability Testing



Combinatorial mutation testing focuses on using combinations of at least two faulty input data parameter to find faults within the software

- ▶ A set of operators that can be combined are presented
- ▶ SOAP message is obtained by parsing the WSDL file, and data perturbation techniques are adopted to generate simple initial test data.
- ▶ A combinatorial testing algorithm is developed.

# [8] API based Security solutions for Communication among Web Services



2 existing web service attacks are considered.

1. Message Alteration Attack (MAA)
2. Data Injection Attack (DIA)

Proposed solution consists of

- ▶ **Middleware services:**  
containing set of security service
- ▶ **Domain web service:**  
set of pluggable API's



## Advantages

- ▶ Encrypts all the outbound messages
- ▶ Installable Plug-ins as API

## Disadvantages

- ▶ High overhead of encryption and decryption
- ▶ Attacks like Reply of Message Attack and Denial of service attack is not considered.





Proposed an approach based on SOAP message mutation and the worst-input technique.

## Methods

- ▶ Worst Input Mutation:  
Utilizing characteristics of SOAP message.
- ▶ Automatic Test Case Generation:  
Test Case Generation based on Farthest Neighbor (TCFN) algorithm
- ▶ A prototype Web Service Vulnerability Testing Tool is implemented



Basis of mutation object is SOAP, which is a message protocol based on an XML document.

## eRTG

a Regular Tree Grammar with 6 tuples  $\langle \mathbf{E}, \mathbf{N}, \mathbf{DT}, \mathbf{P}, \mathbf{A}, n_s \rangle$

- ▶ E finite set of elements.
- ▶ N finite set of non-terminals.
- ▶ DT finite set of data types defined as {int, string, bool, numerical, char, object}
- ▶ P finite set of production rules.
- ▶ A a 2-tuple  $\langle n, \text{type} \rangle$ 
  - ▶ **n** : number of parameters.
  - ▶ **type** : parameter type.
- ▶  $n_s$  is the starting non-terminal



## Mutation Operator

Given a set of all element instances  $N$ , a mutation operator is  $r=f(n_1, n_2 \dots n_i)$ , where  $f$  is a function,  $i \geq 1$ , each  $n_1, n_2 \dots n_i \in N$ , and has an arbitrary data type, and  $r$  outputs the mutated  $n_1 \dots n_i$  with the same data type as the input  $n_1 \dots n_i$ .

## Mutation Operators Considered

ID	Operator	Brief description	Cases/Examples
01	SVB	Set the value of $n$ to be blank	Change value $n$ to ""
02	SVN	Set the value of $n$ to be "null"	Change value $n$ to "null"
03	IPO	Insert parameter operator into the value assigned to node $n$	Insert absolute value symbol into the value assigned to node $n$
04	DNS	Delete node $n$ and its child nodes from the SOAP message	Delete root nodes and child nodes from the SOAP message
05	FVS	Format the value of string	"%n%n .....(256)", "%s% s(1024)" etc.
06	IIV	Integer irregular value	0,+/(1,2 <sup>8</sup> ·1,2 <sup>8</sup> ·2 <sup>8</sup> +1,2 <sup>16</sup> ·2 <sup>16</sup> +1,2 <sup>16</sup> ·1,2 <sup>32</sup> ·2 <sup>32</sup> +1,2 <sup>32</sup> ·1,2 <sup>64</sup> ·2 <sup>64</sup> +1)
07	FIV	Float irregular value	0, 1, -1, +/(the max float point +/-1), +/(the min float point +/-1),5E+324,1.7E+308,pi,e
08	CIV	Char irregular value	'A', 'Z', Null, 'a', 'z', ' ', ' ', ' ', ' ', '{', '(', 'T', '\n', '\0', '\s', '\d'
09	EOV	Exchange the order of values assigned to nodes	Exchange the order of the values assigned to $n_1, n_2$
10	EON	Exchange the order of nodes	Exchange the order of $n_1, n_2$
11	RSV	Random string value	Escape character string "\e \n \r \d \x \s", "\xff \xfe \x00 \x01 \x42 \xb5 \nnnn \h9cc..."
12	LSV	Long string value	Generate String(int n) such as: "AAA.....(256)", "AAA.....(1024)", "AAA.....(15000)"
13	UVF	URL and the value of file directory string	"http://dddddddeeeceerrtttt", "//system32/Notepad.exe", "H:\ABC\killvirus.exe", "D:\AA.exeexe"
14	SSI	SQL string injection	"a or 1=1", "delete", "drop table users", "sql attempt5--"
15	PFB	Parameter flip bit	Use ReverseBit() to flip the value assigned to node $n$

Figure: Mutation Operators

# Conclusion



- ▶ Web service Testing is considering for various mutation approach.
- ▶ An attack on XMI itself was not considered.
- ▶ XML Injection attack of web service is hidden vulnerability in web service technology and has to be considered.

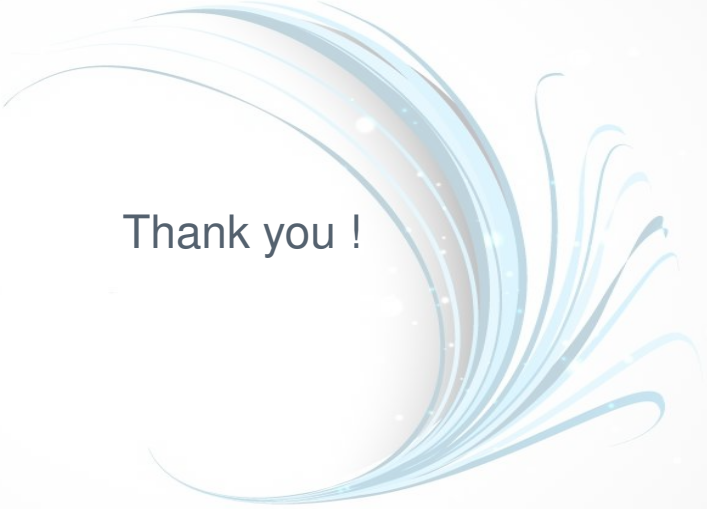


- 1 Automated Robustness Testing of Web Service.  
Evan Martin, Suranjana Basu, Tao Xie
- 2 Exploring Perturbation Based Testing for Web Service  
Lourival F de Alemeida and Silvia R Vergilio
- 3 Efficient Web Services Message Exchange by SOAP Bundling Framework  
Toshiro Takase, Keishi Tajima
- 4 Generating Test Cases for Web Services Using Data Perturbation  
Jeff Offutt & Wuzhi Xu
- 5 Adaptive Random Testing: the ART of Test Case Diversity  
Tsong Yueh Chen, Fei-Ching Kuo, Robert G. Merkel, T.H. Tse
- 6 Testing Web Services by XML Perturbation  
Wuzhi Xu, Jeff Offutt and Juan Luo



- 7 Combinatorial Mutation Approach to Web Service Vulnerability Testing  
Qing Li , Jinfu Chen , Yongzhao Zhan, Chengying Mao, Huanhuan Wang
- 8 API based Security solutions for Communication among Web Services  
A. Kanchana Rajaram, B. Chitra Babu, and C. Kishore Kumar R
- 9 Worst-Input Mutation Approach to Web Services Vulnerability Testing Based on SOAP Messages  
Jinfu Chen, Huanhuan Wang, Dave Towey, Chengying Mao, Rubing Huang, and Yongzhao Zhan





Thank you !