Applied Dynamic Chain of Responsibility in Web Application Security

Mahdi Negahi Shirazi, Maryamsadat Hejazi, and Hossein Dolatabadi

***Abstract*—Software design patterns provide a proven and comprehensive solution for a series of problems that designers confront with them. Web security becomes a critical issue in the online environments. Many techniques strive to achieve web applications to acceptable level of confidence. One of the useful behavioral design patterns is the chain-of-responsibilities pattern that includes several command objects in a series of processing objects. There is a set of logic in each processing object that can handle a command and transfer it to the next processing object. This series of processing objects is stable, and also, the position of each ring in the chain is unchangeable. In this paper, a dynamic chain of responsibilities design pattern that provides a new mechanism for enhance flexibility in security of web application would be presented.**

***Index Terms*—Design pattern, chain of responsibility, web application, web security.**

I. Introduction

Nowadays the web is the most popular technology for wide a range of users. The security issue is a wide area in the web application development. The secure host and secure application are the most important concerns in the web security. However, when network and host-level entry points are relatively secure, the public interfaces of Web applications become the focus of attacks [1]. Hence, a trustable application and host could provide an appropriate infrastructure for online applications.

There are commonly problems that occur in design of applications. Each problem has a general and reusable solution in the same situation that it is called Design Pattern. A design pattern is not a finished design that can be transformed directly into code [2] but they are as a mechanism that represents how to solve a problem in the software design. Nowadays, software designers use design patterns to improve their quality of attribute such as performance, usability, security, and also; they cause to write code clearly. Chain-of-responsibility is one of the design pattern that use in a sequential process. Each ring in the chain is responsible to handle a command. It can receive a request without specifying the handler explicitly. The chief advantage of this pattern is that the number of the chain's rings is changeable. The rings' position is stable in every process. There is no flexibility in the position of rings. All requests process in the same way and there is no priority for which ring should be processed first.

In this paper, we represent a new kind of chain-of–responsibilities that make possible to have dynamical chain's rings. This dynamical chain's rings refer to the rings position that can be changeable according to their priority and can be used in the process of priority. We also describe the following (I) How to apply priority in the chain (II) present Dynamic Chain-Of-Responsibilities (III) How to apply web application security in Dynamic Chain-Of-Responsibilities.

II. Ring's Priority

Priority always refers to schedule tasks. According to which policy of priority applied in the context, appropriate task is chosen and executed.

In the traditional COR, all the rings in the chain have the same priority. A command is sent to the chain and is processes by one of the rings. In addition, each ring has a stable position in the queue. One of the disadvantages of the current pattern is that there is no mechanism for assigning priority over each ring. Hence, all the rings are processed in the same way, which leads to the lack of flexibility in COR. The process of handling a request on COR is the same as following:

- Sending a command to the chain
- Not specifying the appropriate ring
- Traveling of commands through the chain until the appropriate ring handles it

For solving COR weaknesses, we present a new mechanism that each ring of the chain has the own priority. The ring that processes the request increases its priority. For new command the chain is rebuilt, and consequently the position of the ring could be changed. For example, the Ring2 whose position is in the end of the chain handles the command. After handling the command, its priority is more than others. Therefore, for a new command the chain is rebuilt. The position of the Ring2 is changed, which has been shows in Fig. 1.

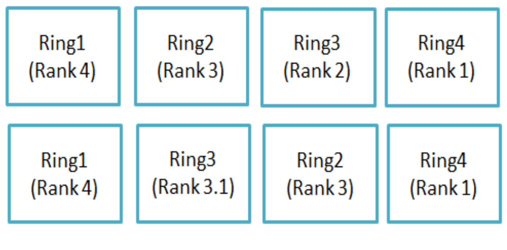


Fig. 1. Example of priority rings and rebuild of the chain.

There are three types of priority that make the chain more

Manuscript received August 1, 2012; revised September 30, 2012.

Mahdi Negahi Shirazi and Hossein Dolatabadi are with the Software Engineering and Software Architecture, Multimedia University of Malaysia, Malaysia, Cyberjaya.

Maryamsadat Hejazi is with the Computer Science in Software Engineering and Software Architecture from Multimedia University, Cyberjaya, Malaysia.

dynamic and flexible. Two of these types make the rings stable, and the rest one is for the changeable rings.

The first type of priority (TPI) makes a ring in the first position of the chain at all times. This priority must appear in the chain only once.

The second type of priority (TPII) sets a ring position in the end of the chain, and also it must be participated in the chain once.

The last type of priority (TPIII) has changeable value in each process. The range of this type must be between above types.

III. DCOR in Detail

As mentioned earlier in Section II for solving COR problems, we suggested the applied priority on each ring's chain. Thus, each ring is a class. The class has the logic of handling of a request. Ring's class inherits from the Approver class. Responsibility of Rank field in the Approver class stores priority value and priority algorithm can use it for calculating ring's priority. Approver class is shown in Fig. 2.

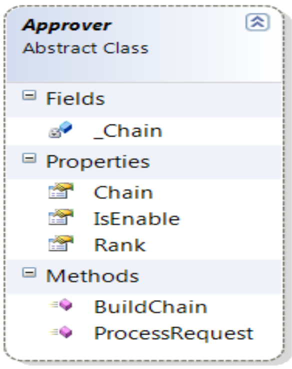
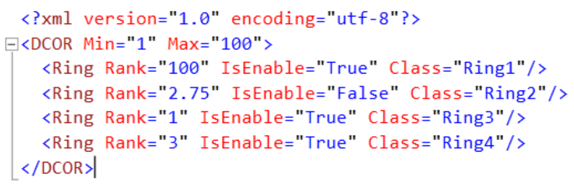


Fig. 2. Approver class as a super class.

Moreover, a chain is built by *Approver* class in runtime. Approver class has a property that it is called *Chain*. The responsibility of this property is to maintain a chain in itself. The succession of the rings in a chain is accessible by *Chain* property.

Furthermore, the client in COR is responsible for generating a chain, and DCOR uses an XML file for this purpose, and determines the order of each ring. The target of IsEnable attribute in an XML file and Approver class provides an interface for disabling a ring at runtime. A simple sample is shown in the Fig. 3.



```
<?xml version="1.0" encoding="utf-8"?>
<DCOR Min="1" Max="100">
  <Ring Rank="100" IsEnable="True" Class="Ring1"/>
  <Ring Rank="2.75" IsEnable="False" Class="Ring2"/>
  <Ring Rank="1" IsEnable="True" Class="Ring3"/>
  <Ring Rank="3" IsEnable="True" Class="Ring4"/>
</DCOR>
```

Fig. 3. A simple sample of XML.

The Fig.3. shows a chain that has three rings, which they are active and Ring2 is inactive. Hence, Ring2 does not corporate in a chain. The BuildChain method in an Approver class is responsible for updating an XML file. The Min and the Max attributes represent the first and last ring in a chain. The Min attribute refers to TPI, and also; the Max attribute refers to TPII. In consequence, Ring1 is the last ring in the chain, while Ring3 is the first one.

The major part of DCOR is how to load class in runtime. At COR, a client class knows all ring classes, but DCOR client only knows Approver. Hence, the chain is generated from the XML file. The chain is generated by reflection mechanism in runtime. Class attribute in the XML file represents class name in code.

The concept of reflection is first proposed by Smith B C in 1982, and mainly refers to the program's ability to access, test and modify its own state and behavior [3]. Reflection is simply a mechanism that allows components or classes to interrogate each other in runtime, also; it discovers information that goes beyond the information gleaned from the publicly available interfaces the objects expose [4]. As a result, the chain generates dynamically by using Reflection. Each ring might be in different DLL, Hence; Namespace or Package name should be mentioned in Class attribute on XML file. Fig. 4. indicates a corresponding chain that it is built according to Fig. 3.

IV. Units

Use either SI (MKS) or CGS as primary units. (SI units are strongly encouraged.) English units may be used as secondary units (in parentheses). This applies to papers in data storage. For example, write "15 Gb/cm^2 (100 Gb/in^2)." An exception is when English units are used as identifiers in trade, such as "3½ in disk drive." Avoid combining SI and CGS units, such as current in amperes and magnetic field in oersteds. This often leads to confusion because equations do not balance dimensionally. If you must use mixed units, DCOR's object is same as COR, which is avoid coupling the sender of a request to its receiver by giving more than one object a chance to handle the request. Chain the receiving objects and pass the request along the chain until an object handles it [5]. In addition, the objects' positions are changeable on the base of their priority.

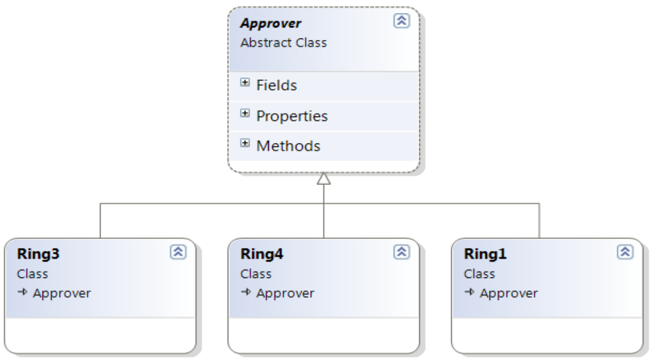


Fig. 4. Generate chain by using XML file

V. Applied Web-Security by DCOR

Seems the internet became popular in the world, making website has become one of the most popular activities on the internet. Nowadays, websites facilitate activity communications and exchanges in human life; thus, two features of websites are the most important in the design of them. These features are beauty and security.

Security is a fundamental issue for protecting assets [6] which can be as distributed items such as web-pages. It should be emphasized that security is a way, not a destination. It is one major concern in all the world that always have some users with a diverse range of educational backgrounds as well as different purposes.

Firstly, it seems to be helpful to describe some of the key words.

- Asset: a source of a system such as a database or an external file.
- Threat: a potential occurrence that can be detrimental to an asset.
- Vulnerability: a weakness feature of a system that is an entrance point for attack.
- Attack: an action that is taken by someone or something in order to harm to assets.

Security is one of the most important features in each website because website's user can trust to information exchange in website. The chief sources of website are databases, users' private information and etc, which are as website assets. For this reason, protection of the assets is a major consideration in the creation of website. In addition, all kinds of web attacks are as threats such as SQL Injection, Cross-Site Scripting (XSS) and so on. Vulnerability can make websites a suitable location for hacking. The aim of this paper is not to discuss how can detect attacks or how can correct them. This paper discusses how DCOR provides a new mechanism to apply security in the online applications.

A web request comes to applications. DCOR's role is to check the request and to provide an acceptable confidence in security. Each ring's responsibility is detection and correction an attack. If a ring detects an attack, it will correct that attack, otherwise the request is transferred to next ring. If all rings travel and process is not terminated, the output will be sent to a main component of application.

In order to detect and correct an attack, two abstract methods should be added to Approver class. These methods' duties are to detect and correct an attack in a request. These methods are necessary because a ring's class load in runtime by reflector mechanism and the base class has to have the interface for execute detection and correction process. Approver class that it is adapted to security is represented in Fig. 5.

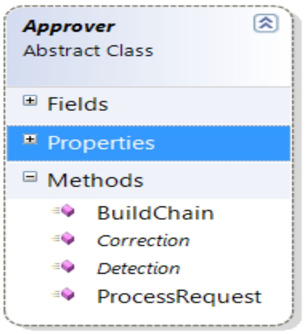


Fig. 5. Approver class.

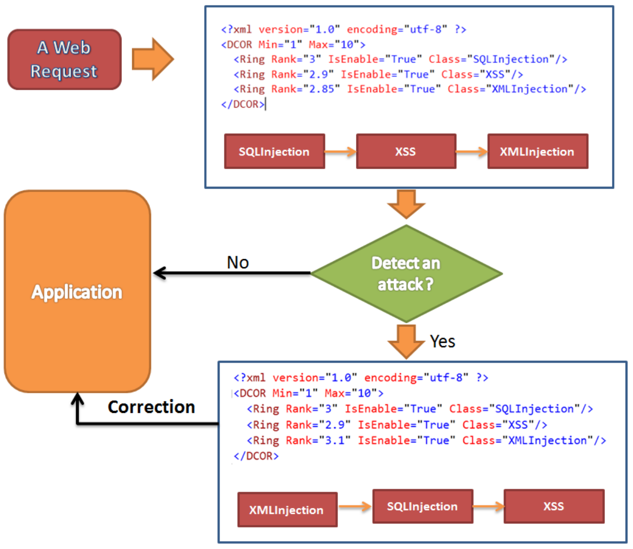
Each concrete object that they inherent from approver is responsible for detecting and correcting one attack. The correction result may be to terminate the request.

Fig. 6 represents a chain which includes 3 rings. Each ring has its own priority. When a request comes to the application, it is transferred to the chain. Each ring in the chain checks the request and verifies it according to a detection algorithm. If detection module detects a potential attack, the correction module will executes and perform a correction algorithm. Detection module has another responsibility that it is to change the priority of a ring.

When a new request comes to the application, Approver class calls the BuildChain module. This module builds a chain according to rings' priority.

Fig. 6 represents a hierarchy of a chain that it consists 3 rings. A request comes to the application that it uses DCOR for its security. In the request, XML injection attack is detected and the request is cleared by a correction module. Detection module modifies corresponding ring's priority. For the next request the chain is rebuilt and the position of XML injection rings will be changed. For this DCOR all priorities are in type three that means all priorities are dynamic. In some cases, a correction algorithm terminates a request because there is not any way to correct a request.

DCOR has capability to add rings at runtime. For this target after implementing a new ring, it should add a class name and its priority to XML file. For new request Approver class rebuilds a chain and also it adds new rings to the chain.


```xml
<?xml version="1.0" encoding="utf-8" ?>
<DCOR Min="1" Max="10">
  <Ring Rank="3" IsEnable="True" Class="SQLInjection"/>
  <Ring Rank="2.9" IsEnable="True" Class="XSS"/>
  <Ring Rank="2.85" IsEnable="True" Class="XMLInjection"/>
</DCOR>
```

```xml
<?xml version="1.0" encoding="utf-8" ?>
<DCOR Min="1" Max="10">
  <Ring Rank="3" IsEnable="True" Class="SQLInjection"/>
  <Ring Rank="2.9" IsEnable="True" Class="XSS"/>
  <Ring Rank="3.1" IsEnable="True" Class="XMLInjection"/>
</DCOR>
```

Fig. 6. A simple applied security mechanism in DCOR.

VI. Conclusion

This paper introduces a new type of Chain-Of-Responsibility. This type of COR is more flexible than original one. Rings' chain has its own priority and rings position can be changed in the next process. The mechanism uses reflection for loading ring's class at runtime. Furthermore the usage of the new designed mechanism in gaining security in web applications has been explained. DCOR act like firewall in web application. It stands before application and check all request for a probable attacks. DCOR encapsulates the recognition and solve a attach in itself; therefore, its Rings can change based on the number of a specific attack.

REFERENCES

[1] Y. Huang, C. Tsai, T. L Huang, D. T. Lee, and S. Kuo "A testing framework for Web application security assessment," *Computer Networks* vol. 48, pp. 739-761, 2005.

[2] M. Robert, *Design Principles and Design Patterns*.

[3] B. C. Smith, *Massachusetts Instituteof Technology*, Phd thesis, 1982.

[4] R. Kovacs, *Creating Dynamic Factories in NET Using Reflection*.

[5] W. F. Tichy, "A catalogue of general-purpose software design patterns," *Technology of Object-Oriented Languages and Systems*, pp. 330-339, 1997.

[6] J. D. Meier, A. Mackman, M. Dunner, S. Vasireddy, R. Escamilla, and A. Murukan, *Improving Web Application Security: Threats and Countermeasures*.

[7] Q. Wu, Y. Hu, and Y. Wang, "Unit testing and action-level security solution of struts web applications based on MVC," *Biomedical Engineering and Computer Science (ICBECS), International Conference*, pp. 1-4, 2010.

[8] Y. Ge, "Using design pattern to develop the quality MSC simulator software in performance test for GSM and GPRS system," *Proceedings. Third International Conference, Quality Software*, pp. 374-377, 2003.

[9] *Design Patterns–Elements of Reusable Object Oriented Software–Erich Gamma*, 1995.

[10] M. L. Nelson, "A design pattern for autonomous vehicle software control architectures," *Proceedings. The Twenty-Third Annual International Computer Software and Applications Conference, COMPSAC '99.* pp. 172-177, 1999.

[11] M. Wheatman and K. Liu, "Automating software design pattern transformation," *Industrial Informatics. INDIN 2009. 7th IEEE International Conference*, pp. 167-172, 2009.

Mahdi Negahi Shirazi was born in Iran, Tehran, 21 November 1981, Diploma in mathematics in Kosar high school educational institution, Iran Tehran, 2001, BC in computer software engineering, Iran, Maybod, Azad University of Maybod, 2006, Master in Software Engineering and Software Architecture, Multimedia University of Malaysia, Malaysia, Cyberjaya, 2012. He was an employee of TITG, Iranian software developer for 2 year, from 2003 to 2005 as a Software developer, he started to work with Hamrahan System in Iran as a Software Designer and Developer from 2006 to 2007 and continued his working experiments with Solic System in Iran as a Software Developer and Designer for more than two years from. Finally, He was an employee of GTB Malaysian financial company as a software designer.

Maryamsadat Hejazi was born in 1983, in Iran. She received B.Sc. degree in computer software engineering from University of Applied Science and Technology, Tehran, IRAN, in 2008. She received M.Sc. degree in computer science in software engineering and software architecture from Multimedia University, Cyberjaya, Malaysia, in 2012. She has a journal papers in the field of Support Vector Machines as a classification method. One of the papers is published in journal of Advanced Computer Science and Technology Research in 2012.

Hossein Dolatabadi was born in Iran, Sari, 8 May 1983, Diploma in mathematics in Beheshti high school educational institution, Iran Tehran, 2000, BC in computer hardware engineering, Iran, Qazvin, Azad University of Qazvin, 2007, Master in Software Engineering and Software Architecture, Multimedia University of Malaysia, Malaysia, Cyberjaya, 2012. He was an employee of LG, Iranian hardware company for 1 year, from 2004 to 2005 as a hardware engineer, he started to work with Bahar programming co in Iran as a financial software developer from 2005 to 2006 and continued his working experiments with Roshangar E-Learning material producer company in Iran as a software developer and software department manager for more than two years.