Thread Based XSS Detection System – Technical Design Document

Version 1.0

Document Reference & History

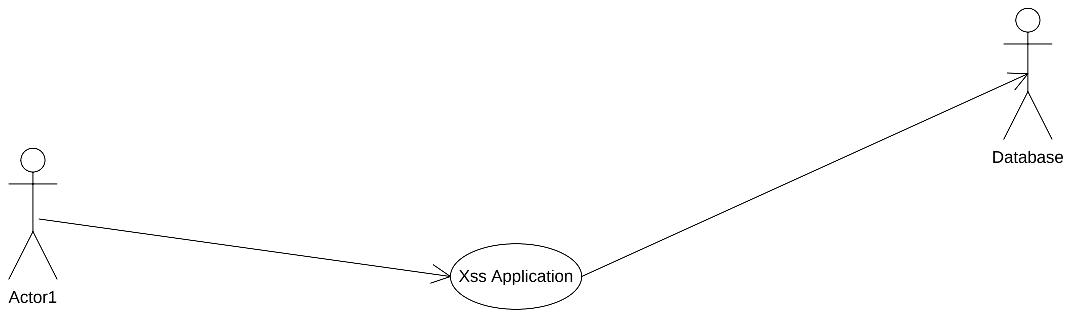
Revision	Author
1.0	Jayamsakthi Shanmugam

Contents

1 Introduction

XSS thread based application which tracks the hacker’s intentions by their request’s inputs. This will not allow malicious inputs to go further.

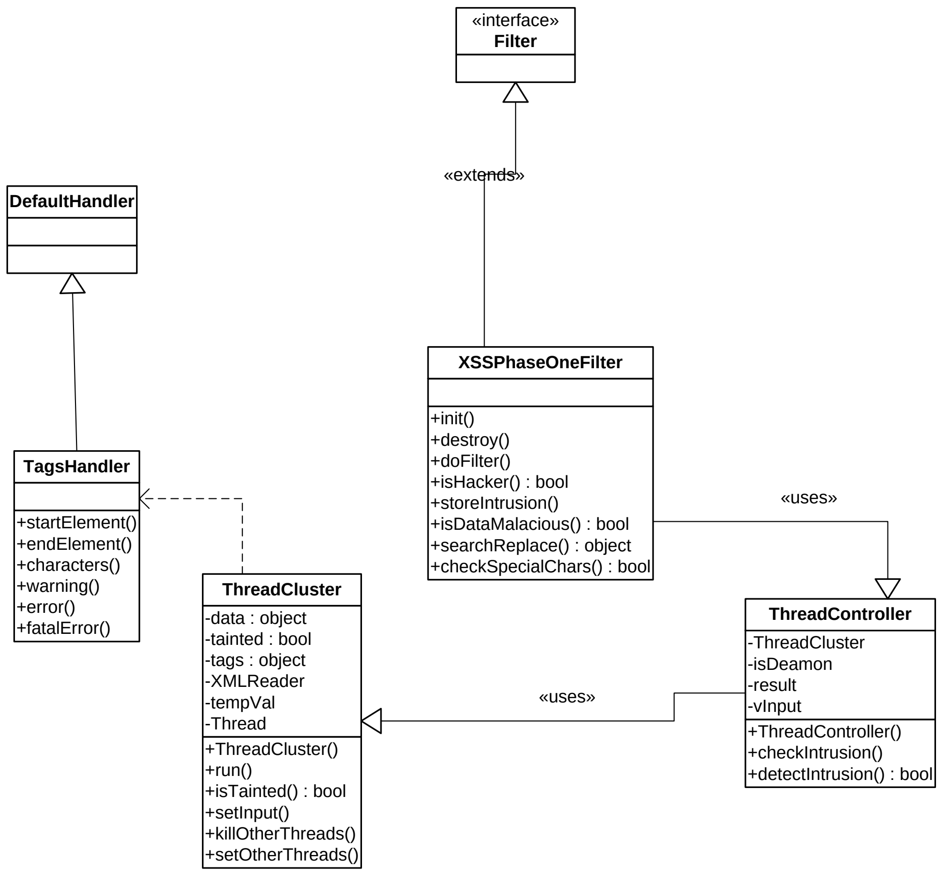
2 Usecase Realizations



XSS thread based application consists of use case tracking of malicious inputs which have done by hackers. It tracks the hackers' details from database and also it updates the details into the database.

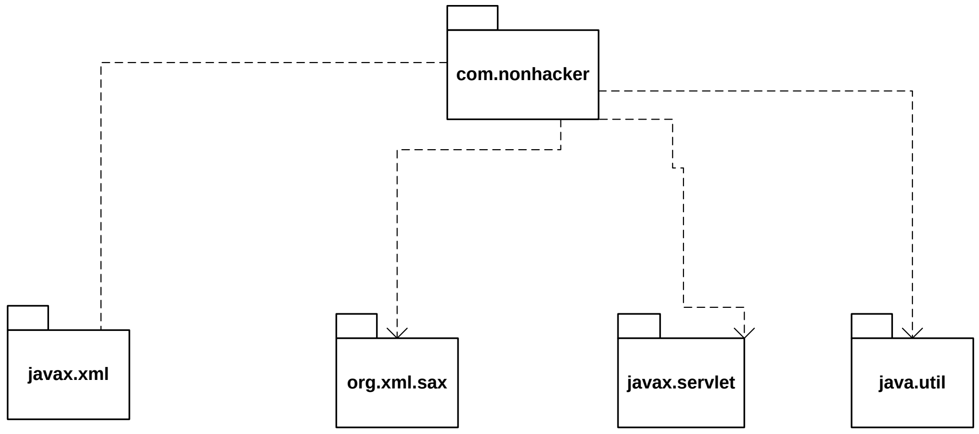
2.1 Static view

2.2 View of participating classes



Class	Description
ThreadCluster.java	The class which differentiates the 3 threads and compares the input and tags of White, Black and Malicious tags.
XSSPhaseOneFilter.java	This Filter Servlet which invokes for every request and it filters all the request inputs and passes the control to ThreadController.
ThreadController.java	This class intiates all the Threads for processing of the input requests.
TagsHandler.java	This is the inner class which parses all the xml tags.

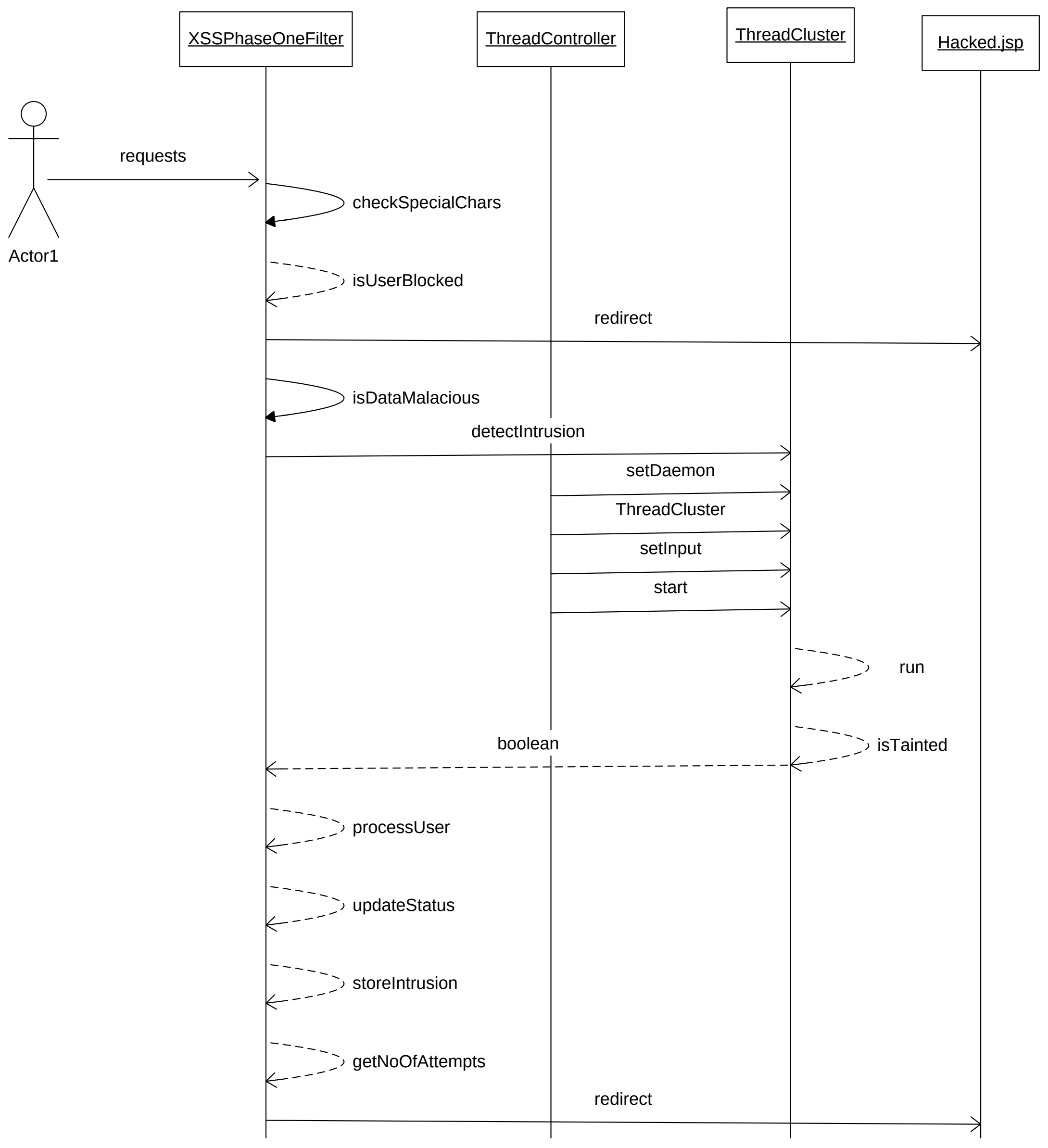
2.2.1 Package dependencies



Package	Description
com.nonhacker	The main package of XSS Application
Org.xml.sax	Used for xml parsing.
Javax.servlet	Servlet package for the Filtering Requests
Java.util	Utility package for Java

2.3 Dynamic view

2.3.1 Sequence diagrams



2.3.2 Method Description

Method of XSSPhaseOneFilter	**Description**
checkSpecialChars(str)	It checks whether there are any special characters are there in request inputs.
isDataMalacious(Vector vInput)	This method invokes ThreadController and checks whether request input is malicious are not and accordingly it returns Boolean value. It return true when the data is malicious and false when data is not malicious.
isUserBlocked(req,userid)	Checks whether user is blocked or not and if user is blocked control will be forwarded to hacker.jsp
storeInstrusion(String userid,request,noofAttempts,status)	Inserts the user details into database, if this user trying to insert malicious data.
processUserStatus()	This method contains logic of processing user status based on no of attempts, Lower limit for no of attempts, Max No. of Attempts a user can be made and accordingly it updates the database tables.
updateStatus(userid, status)	It is a db method which updates the status of user.

Method of ThreadCluster	**Description**
ThreadCluster(String name,String type)	This constructor loads the xml files based on the type of xml file and creates SAX parser for parsing the different tags of xml files.
Run()	Life cycle method of Thread class in which it determines whether input data is malicious or not based on tags of xml files and accordingly it sets the Boolean result.
isTainted	It returns the result set by run method.
setInput	This method sets the input data.
setOtherThreads	This method sets Threads t1, t2
killOtherThreads	This methods interrupts the running threads if malicious data is found.

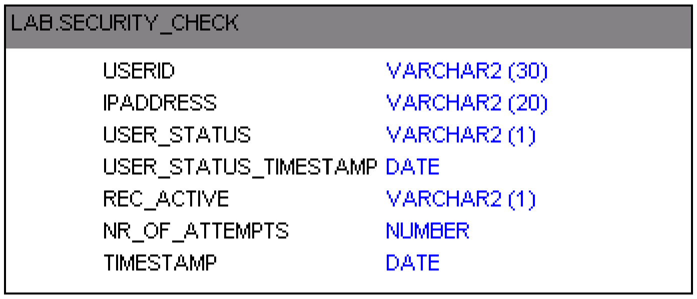
Method of ThreadController	**Description**
ThreadController	Constructor which sets the input data as vector
detectIntrusion	This method invokes all the three threads and sets the corresponding data to those threads. This method initiates the start processes of all the three threads and returns the Boolean values based on whether data is malicious or not

2.3.3 Data Specifications

One Database Table have been used for this Application ,

SECURITY_CHECK Table - This Table stores UserId, IPaddress, User Status and Timestamp when User Status is changed, Active State of User, No of Attempts made by user

User Status will be in N-Notice,W-Warning,B-Block.
Rec Active Status will either be in Y or N status at any instance of time.



LAB.SECURITY_CHECK	
USERID	VARCHAR2 (30)
IPADDRESS	VARCHAR2 (20)
USER_STATUS	VARCHAR2 (1)
USER_STATUS_TIMESTAMP	DATE
REC_ACTIVE	VARCHAR2 (1)
NR_OF_ATTEMPTS	NUMBER
TIMESTAMP	DATE

2.3.4 Configurable Parameters

There are Two Property files which can be configurable.

Two Property files which are used by application are

1) dbAccess.properties : This is file is used for retrieving database details and following parameters are used.

propDriver - Database Driver.
propURL – URL of Database.
propUser - UserName to access Database
propPassword - Password to access Database

UserIDParam - Login Id of Application.

2) IED.properties : This is file is used to restrict the no of attempts a user will be made and following parameters are used.

   BLOCKING_TIME_INTERVEL - No. of Mins for a User will be blocked.

   ATTEMPTS_THRESHOLD – Max No. of Attempts a User can be made , If User exceeds this limit, User status will changed to Blocked and He can't Access the Application anymore.

   ATTEMPTS_LOWER_LIMIT – Min No. of Attempts a User can be made, If User exceeds this no. of Attempts User status will be changed to Warning

2.3.5 Pseudo Code

This application can be deployed on any Application where it supports J2EE 1.3 Specs.This application tracks the malicious attempts made by user and accordingly it will not allow the user to access the application based on his no. of attempts.

The following is Pseudo Code for this application

1) User login into application
2) XSS thread based applicationtracks the User logged in and if user is blocked then it redirects to hacker.jsp
3) If User login for first time and a malicious attempt is found then a record will be inserted into Security_Check Table with "Notice" as Status.
4) If he continosly makes malicious attempts then User Status will be changed to "Warning".
5) If No of Attempts made by him exceeds ATTEMPTS_THRESHOLD i.e Max No of ATTEMPTS , then Status will be changed to "Blocked".
6) If User has got Status Blocked , Then User cannot access the application.

Pseudo Code for Checking Whether Request Input is malicious or not

1) Application tracks the input tags and it compares input with xml tags of White Listed, BlackList and Malicious tags which are predefined in xml files.
2) Input tags should be part of white listed tags or it would be treated as Malicious Attempts.
3) Input tags should not be part of Black Listed and Malicious tags and if match is found then input will be treated as Malicious Attempts.
4) If malicious attempts are found and exceeds defined Max No. of Attempts, then User is blocked.