Security Testing Project

January 8, 2019

Francesco Minna *206181*

Abstract

This report describes my activity and work of security testing on the Schoolmate web application, written in PHP and MySQL, for the Cross Site Scripting vulnerabilities.

Project delivery

Follows a list of the contents with a brief description attached to the project delivery:

- *Report.pdf*: this document describes the activities that I did;
- *Test_cases.zip*: the JUnit test cases as a proof of the vulnerabilities presence and the relative fixing (this zip contains an Eclipse project);
- *Schoolmate_fixed.zip*: the web application PHP files with fixed code;
- *DB_snap.sql*: snapshot of the database's data used to run the test cases and verify the presence of the vulnerabilities.

Structure of the report

Follows a brief description of how the code and the test cases it was organized: first, I separated the false positives from the true positives.
The False positives are reported with a brief description of why are false positives, while the True positives are divided in different packages, where for each one i wrote:

- the *package* where i put the test cases;
- the *name* of the test cases;
- the *type* of Cross Site Scripting vulnerability (e.g. Reflected XSS);
- a brief *description* of the vulnerability;
- a possible *sanitization*.

All the true positives are organized in a table, based on the package: because some test cases are similar, I wrote a description for only the first test case, while others require a better explanation.
Each table contains the following columns:

Name	Variable	Description	Sanitization

Brief description of each column of the table:

- *Name*: each test cases are named with the following convention:
  - *Test*

  - *Action*: for example add, edit, view;
  - *Object*: the object of the action, for example Grades or Assignments.
  - *Number*: the vulnerability's number.

  Example: *TestAddStudent99*

- *Variable*: this column contains one or more variable that are not properly sanitize;
- *Description*: this column contains a brief description of the action in which the vulnerability occurs;
- *Sanitiation*: this column contains the file and the line where the variable is sanitize. Example: "index.php :10", which means: file "index.php" and line 10.

Sanitization: each PHP variable, vulnerable to XSS, it was sanitized with these two functions:

- *htmlspecialchars()*: converts all the HTML tags to a normal string;
- *strip_tags()*: remove all the HTML tags.

Example:

```
$bkc = strip_tags(htmlspecialchars($_POST[var]));
```

In my opinion, this is the best strategy to sanitize the user input, firstly converting the possible malicious code to a normal string, and then removing all the eventually HTML tags inside the string.

Also, at the end of the document, there is a Glossary with a brief explanation of the different types of the XSS attacks.

Test Cases

Follows the description of the false positives and the tables for each package of the positives test cases.

False positives

From the output of *pixy*, i found these false positives:

- *xss_index.php_2_min*: in this case, we have to look at the $schoolname variable, that it's printed in maketop.php. As we can see, in header.php, $schoolname is sanitized in the query at line 11, and then is globally sanitize in the next line, so this is a false positive;
- *xss_index.php_3_min*: same as previous;
- *xss_index.php_4_min*: same as previous;
- *xss_index.php_6_min*: same as previous;
- *xss_index.php_10_min*: same as previous;
- *xss_index.php_53_min*: same as previous, but in this case $schoolname is printed in the header.php file;
- *xss_index.php_321_min*: as we can see at line 26 of index.php, ReportCards.php is required only if page2 is equal to 1337, but this functionality doesn't work so we can not test this case.

True positives

Package addVulnerabilities

This package contains all the test cases regarding the operation of adding new information.
XSS type: all the test cases in the addVulnerabilities package are *Reflected XSS* vulnerabilities.

Name	Variable	Description	Sanitization
TestAddAnnouncement16	$page2 $page	This vulnerability occurs when the admin or the attacker tampering and inject the malicious code during the loading of the "Add Announcement" page.	AdminMain.php :7 index.php :36
TestAddAssignment11	$page2 $selectclass $page	Similar to previous, but in the "Add New Assignment" page.	TeacherMain.php :8, :9 index.php :36
TestAddAttendance13	$semester $student $page $page2	Similar to previous, but in the "Add New Attendance" page.	AddAttendance.php :3, :4 AdminMain.php :7 index.php :36
TestAddClass141	$page2 $page	Similar to previous, but in the "Add New Class" page.	AdminMain.php :7 index.php :36
TestAddClass269	$fullyear $page2 $page	Similar to previous.	AddClass.php :177 AdminMain.php :7 index.php :36
TestAddParent93	$page2 $page	Similar to previous, but in the "Add New Parent" page.	AdminMain.php :7 index.php :36
TestAddSemester71	$page2 $page	Similar to previous, but in the "Add New Semester" page.	AdminMain.php :7 index.php :36
TestAddStudent70	$page2 $page	Similar to previous, but in the "Add New Student" page.	AdminMain.php :7 index.php :36
TestAddTeacher63	$page2 $page	Similar to previous, but in the "Add New Teacher" page.	AdminMain.php :7 index.php :36
TestAddTerm19	$page2 $page	Similar to previous, but in the "Add New Term" page.	AdminMain.php :7 index.php :36
TestAddUser18	$page2 $page	Similar to previous, but in the "Add New User" page.	AdminMain.php :7 index.php :36

Package editVulnerabilities

This package contains all the test cases regarding the modification of the stored information.
XSS type: all the test cases in the editVulnerabilities package are *Reflected XSS* vulnerabilities.

Name	Variable	Description	Sanitization
TestEditAnnouncement41	$id[0] $page2 $page	This vulnerability occurs when the admin or the attacker tampering and inject the malicious code during the loading of the "Edit Announcement" page.	EditAnnouncement :3 AdminMain.php :7 index.php :36

Name	Variable	Description	Sanitization
TestEditAssignment37	$id[0] $selectclass $page2 $page	Similar to previous, but in the "Edit Assignment" page.	EditAssignment :3, :5 AdminMain.php :7 index.php :36
TestEditClass239	$id[0] $page2 $page	Similar to previous, but in the "Edit Class" page.	EditClass :3 AdminMain.php :7 index.php :36
TestEditGrade76	$assignment $id[0] $page2 $selectclass $page	Similar to previous, but in the "Edit Grade" page.	EditGrade :3, :5, :6 TeacherMain.php :8 index.php :36
TestEditParent161	$id[0] $page2 $page	Similar to previous, but in the "Edit Parent" page.	EditParent :3 AdminMain.php :7 index.php :36
TestEditSemester85	$id[0] $page2 $page	Similar to previous, but in the "Edit Semester" page.	EditSemester :3 AdminMain.php :7 index.php :36
TestEditStudent115	$id[0] $page2 $page	Similar to previous, but in the "Edit Student" page.	EditStudent :3 AdminMain.php :7 index.php :36
TestEditTeacher111	$id[0] $page2 $page	Similar to previous, but in the "Edit Teacher" page.	EditTeacher :3 AdminMain.php :7 index.php :36
TestEditTerm44	$id[0] $page2 $page	Similar to previous, but in the "Edit Term" page.	EditTerm :3 AdminMain.php :7 index.php :36
TestEditUser149	$id[0] $page2 $page	Similar to previous, but in the "Edit User" page.	EditUser :3 AdminMain.php :7 index.php :36

Package mainVulnerabilities

This package contains all the test cases regarding the visualization of the home page for all the different users.
XSS type: all the test cases in the mainVulnerabilities package are *Reflected XSS* vulnerabilities.

Name	Variable	Description	Sanitization
TestAdminMain186	$page2 $page	This vulnerability occurs when the admin or the attacker tampering and inject the malicious code during the loading of the "Admin Main" page.	AdminMain.php :7 index.php :36
TestParentMain194	$selectclass $page2 $page	Similar to previous, but you login as Parent.	ParentMain.php :8, :9, :10 index.php :36

Name	Variable	Description	Sanitization
TestStudentMain165	$selectclass $page2 $page	Similar to previous, but you login as Student.	StudentMain.php :8, :9 index.php :36
TestTeacherMain180	$selectclass $page2 $page	Similar to previous, but you login as Teacher.	TeacherMain.php :8, :9 index.php :36

Package manageVulnerabilities

This package contains all the test cases regarding the visualization of the Manage page.
XSS type: in the manageVulnerabilities package, the TestManageAssignments207 and TestManageSemesters234 test cases are *Stored XSS* vulnerabilities, because the malicious code are persistently stored in the database.
All the other test cases are *Reflected XSS* vulnerabilities.

Name	Variable	Description	Sanitization
TestManage - Announcement257	$page2 $onpage $page	This vulnerability occurs when the admin or the attacker tampering and inject the malicious code during the loading of the "Manage Announcements" page.	AdminMain.php :7 ManageAnnouncements :3 index.php :36
TestManage - Assignment207	$coursename	This vulnerability occurs when the admin or the attacker injects the malicious code in the class name, so the teacher, that visualizes the assignments of that course, became the victim of the attack.	ManageAssignments.php :7
TestManage - Assignment309	$page2 $onpage $selectclass $page	This vulnerability occurs when the admin or the attacker tampering and inject the malicious code during the loading of the "Manage Assignments" page.	TeacherMain.php :7 ManageAssignments :9, :10 index.php :36
TestManage - Attendance272	$page2 $page	Similar to previous, but in the "Manage Assignment" page.	AdminMain.php :7 index.php :36
TestManage - Classes320	$page2 $onpage $page	Similar to previous, but in the "Manage Classes" page.	AdminMain.php :7 ManageClasses :3 index.php :36
TestManage - Grades316	$page2 $selectclass $page	Similar to previous, but in the "Manage Grades" page.	AdminMain.php :7 ManageGrades :3 index.php :36
TestManage - Parents288	$page2 $onpage $page	Similar to previous, but in the "Manage Parents" page.	AdminMain.php :7 ManageParents :3 index.php :36

Name	Variable	Description	Sanitization
TestManage - SchoolInfo92	$page2 $page	Similar to previous, but in the "Manage School Info" page. Notice that, in this case, pixy reports also the $address and the $phone variables as possibly dangerous, but both are never printed in the web application, so it's not possible to verify the vulnerability: for sure they are vulnerable because in header.php :11 they are not sanitized.	AdminMain.php :7 index.php :36
TestManage - Semesters234	$term	This vulnerability occurs when the admin or the attacker injects the malicious code in the term name, so the user that visualizes the semester, became the victim of the attack.	ManageSemesters.php :134
TestManage - Semesters268	$page2 $onpage $page	This vulnerability occurs when the admin or the attacker tampering and inject the malicious code during the loading of the "Manage Semesters" page.	AdminMain.php :7 ManageSemesters :163 index.php :36
TestManage - Students293	$page2 $onpage $page	Similar to previous, but in the "Manage Students" page.	AdminMain.php :7 ManageStudents :211 index.php :36
TestManage - Teacher273	$page2 $onpage $page	Similar to previous, but in the "Manage Teachers" page.	AdminMain.php :7 ManageTeachers :181 index.php :36
TestManage - Terms260	$page2 $onpage $page	Similar to previous, but in the "Manage Terms" page.	AdminMain.php :7 ManageTerms :163 index.php :36
TestManage - Users283	$page2 $onpage $page	Similar to previous, but in the "Manage Users" page.	AdminMain.php :7 ManageUsers :187 index.php :36

Package otherVulnerabilities

This package contains all the generic test cases.
XSS type: in the otherVulnerabilities package, the TestSiteMessage54 and TestLogin105 test cases are *Stored XSS* vulnerabilities, because the malicious code are persistently stored in the database.
All the other test cases are *Reflected XSS* vulnerabilities.

Name	Variable	Description	Sanitization
TestClassSettings89	$page2 $page $selectclass	This vulnerability occurs when the teacher or the attacker tampering and inject the malicious code during the loading of the "Class Settings" page.	TeacherMain.php :8 ClassSettings.php :12 index.php :36

Name	Variable	Description	Sanitization
TestLogin105	$message	This vulnerability occurs when the teacher or the attacker tampering and inject the malicious code during the loading of the first page of the web application. Notice that the $page variable is not vulnerable to injection before login into the web app, because is used as id-number in the index.php file.	Login.php :12
TestRegistration299	$page2 $page	Similar to previous, but in the "Registration" page.	AdminMain.php :7 index.php :36
TestSiteMessage54	$sitetext	This vulnerability occurs when the teacher or the attacker tampering and inject the malicious code during the loading of the "Class Settings" page.	Login.php :16

Package reportVulnerabilities

This package contains all the test cases regarding the visualization of the different report for each student.
XSS type: all the test cases in the reportVulnerabilities package are *Reflected XSS* vulnerabilities.

Name	Variable	Description	Sanitization
TestDeficiencyReport191	$page2 $page	This vulnerability occurs when the admin or the attacker tampering and inject the malicious code during the loading of the "Manage Semesters" page.	AdminMain.php :7 index.php :36
TestGradeReport241	$page2 $page	Similar to previous, but in the "Grade Report" page.	AdminMain.php :7 index.php :36
TestPointsReport212	$page2 $page	Similar to previous, but in the "Points Report" page.	AdminMain.php :7 index.php :36

Package viewVulnerabilities

This package contains all the test cases regarding the visualization of some pages of the web application. The tests are further divided by the login users, used to prove the vulnerability: each user is identified by the letter after Test. In particular:

- *A*: admin login
- *P*: parent login
- *S*: student login
- *T*: teacher login

XSS type: in the viewVulnerabilities package, the TestAViewAssignments30 and TestAViewAssignments31 test cases are *Stored XSS* vulnerabilities, because the malicious code are persistently stored in the database.
All the other test cases are *Reflected XSS* vulnerabilities.

Name	Variable	Description	Sanitization
TestAView - Assignments30	$coursename	This vulnerability occurs when the admin or the attacker injects the malicious code in the course name, so the student that visualizes this course, became the victim of the attack.	ViewAssignments.php :9
TestAView - Assignment31	$coursename	Similar to previous.	ViewAssignments.php :9
TestAVisualize - Classes230	$page2 $page	This vulnerability occurs when the admin or the attacker tampering and inject the malicious code during the loading of the "School Class Schedule" page.	AdminMain.php :7 index.php :36
TestAVisualize - Registration238	$page2 $page	Similar to previous, but in the "Registration" page.	AdminMain.php :7 index.php :36
TestPView - Announcements146	$page2 $onpage $page	Similar to previous, but in the "View Announcements" page.	AdminMain.php :7 ViewAnnouncements.php :67 index.php :36
TestPView - Assignments183	$page2 $onpage $selectclass $page	Similar to previous, but in the "View Assignments" page.	AdminMain.php :7 ViewAssignments :87, :88 index.php :36
TestPView - Assignments184	$page2 $onpage $selectclass $page	Similar to previous.	AdminMain.php :7 ViewAssignments :87, :88 index.php :36
TestPView - ClassSettings87	$page2 $selectclass $page	Similar to previous, but in the "View Class Settings" page.	AdminMain.php :7 ViewClassSettings.php :36 index.php :36
TestPView - Courses142	$page2 $page $student	Similar to previous, but in the "View Course" page.	StudentMain.php :8 ViewClassSettings.php :36 index.php :36
TestPView - Grades200	$page2 $selectclass $page	Similar to previous, but in the "View Grades" page.	ParentMain.php :8 ViewGrades.php :3 index.php :36
TestPView - Students90	$page2 $page	Similar to previous, but in the "View Student" page.	StudentMain.php :8 index.php :36
TestSView - Announcements147	$page2 $onpage $selectclass $page	Similar to previous, but in the "View Announcements" page.	StudentMain.php :8 ViewAnnouncements :67, :88 index.php :36
TestSView - ClassSettings88	$page2 $selectclass $page	Similar to previous, but in the "View Class Settings" page.	StudentMain.php :8, :9 ViewClassSettings.php :36 index.php :36

Name	Variable	Description	Sanitization
TestSView - Courses138	$page2 $page	Similar to previous, but in the "View Courses" page.	StudentMain.php :8 index.php :36
TestSView - Grades201	$page2 $selectclass $page	Similar to previous, but in the "View Grades" page.	StudentMain.php :8 ViewGrades.php :3 index.php :36
TestTView - Announcements148	$page2 $onpage $selectclass $page	Similar to previous, but in the "View Announcements" page.	TeacherMain.php :8 ViewAnnouncements :67, :88 index.php :36
TestTView - Courses126	$page2 $page	Similar to previous, but in the "View Courses" page.	TeacherMain.php :8 index.php :36
TestTView - Students181	$page2 $selectclass $page	Similar to previous, but in the "View Students" page.	TeacherMain.php :8 ViewStudents.php :83 index.php :36

Package utility

This package contains the *Utilities* java class, that contains some methods to simplify the writing of the test cases. The code is commented for a better understanding.

Glossary

Cross-site Scripting: Cross-site Scripting, also known as XSS, is a type of web attack, typically found in web application, that enables an attacker to inject malicious code, generally in the form of a browser side script, into web pages viewed by other users. This is possible because the developers dont validate correctly the user input (e.g. a form input field).

Non Persistent/Reflected XSS: this type of XSS vulnerability takes his name to the fact that the attacker injects malicious code that is not persistent in the web page (and also in the database of course), which means that the attack happens when the victim clicks on the malicious link.

Persistent XSS: in this case the attacker injects permanent malicious code inside the web application, and the victim has no defense measure: once he opens the infected page, the code is automatically executed.