Software Security Project Part 1: OWASP Top 10 Using OpenMRS 2.8.0 Standalone

Name	Unity ID
Aishwarya Sundararajan	asundar2
Abhash Jain	ajain28
Arjun Sharma	asharm33
Sachin Kumar	skumar26

A1: Injection

Test Case ID	Test_A1_1
Test Name	**SQL injection in Login Page**
Steps for test case	i) Go to login page: http://152.46.17.224:8082/openmrs-standalone/login.htm ii) Now provide following credentials on the page username: **admin**, password: **' or ' 1=1** iii) Now after trying the above provided credentials, authentication failed showing message “Invalid username/password. Please try again.”
Expected Result	Login should fail and response should be returned as Invalid username/password. Please try again.
Test Case Status	Passed.
Mitigation by OpenMRS team	Parameterized queries are used along with usage of ORM(Object Relationship Mapping) frameworks like Hibernate.

Test Case ID	Test_A1_2
Test Name	**SQL injection in Find Patient Record**
Steps for test case	i) Login using valid credentials. http://152.46.17.224:8082/openmrs-standalone/login.htm ii) Click on Find Patient Record. (http://152.46.17.224:8082/openmrs-standalone/coreapps/findpatient/findPatient.page?app=coreapps.findPatient) iii) In search box enter the query **' union select 1, 1, 1, user(), version()** #, which shows message “**No matching records found**”
Expected Result	It should not return any records through union with another query.
Test Case Status	Passed.
Mitigation by OpenMRS team	Parameterized queries are used along with usage of ORM(Object Relationship Mapping) frameworks like Hibernate.

A2: Broken Authentication

Test Case ID	Test_A2_1
Test Name	**Weak password acceptance in Register Patient**
Steps for test case	i) Go to http://152.46.17.224:8082/openmrs-standalone/index.htm, there i was logged in as admin ,then click on admin dropdown top bar of that page,then Myaccount button will appear,so click on it. ii) Next click on Change password button. iii) Now in the page that appears, provide the old password, and provide a weak new password such as **'Test1234'**, and then save it. iv) Now logout and login again with new password
Expected Result	The application should not have allowed to set a weak password while changing the password, as it makes system vulnerable to dictionary and brute-force attacks.
Test Case Status	Failed
Mitigation to be taken by OpenMRS team	The password rules for authentication should be strong and not easily open to dictionary and brute-force attacks. A combination of upper and lower case characters with a mix of alpha-numeric characters that don't usually form a sequence is a good password rule.

Test Case ID	Test_A2_2
Test Name	**Session ids in the url and its generation and rotation after successful login**
Steps for test case	i) Go to http://152.46.17.224:8082/openmrs-standalone/referenceapplication/login.page, fill in the username and password, and also click on inspect in chrome to open network tab's console. ii) Now press login button after filling in login details, and observe in url, session id was not present. iii) Also see the post payload in the network tab of console, there click on login.page row and click on Headers tab in right side window, there observe the session id generated and no-caching done as evident by following data: Cache-Control: max-age=0 Connection: keep-alive Cookie: referenceapplication.lastSessionLocation=4; _REFERENCE_APPLICATION_LAST_USER_=92668751; JSESSIONID=F315C3309D0593996E8279B7FC6AE7CC; referenceapplication.lastSessionLocation=4; _REFERENCE_APPLICATION_LAST_USER_=92668751

	iv) Now logout and repeat steps i to iii, to see the cookie details again Cache-Control: max-age=0 Connection: keep-alive Cookie: referenceapplication.lastSessionLocation=4; _REFERENCE_APPLICATION_LAST_USER_=92668751; JSESSIONID=6B42EDD70DE4CC5F85FDBB1B0B85AC39; referenceapplication.lastSessionLocation=4; _REFERENCE_APPLICATION_LAST_USER_=92668751 v) Now as you observe session id's are not rotated and change and randomized for every new session.
Expected Result	Session id's should not be transmitted in plain text as part of url and session ids should not be rotated and randomized for every session.
Test Case Status	Passed
Mitigation by OpenMRS team	Secure built-in session manager used to generate random session ids and also sessions ids are included in the post payload but not in the url

A3: Sensitive Data Exposure

Test Case ID	Test_A3_1
Test Name	**Data encryption in HTTP data transmission**
Steps for test case	i) Go to http://152.46.17.224:8082/openmrs-standalone/registrationapp/registerPatient.page?appId=referenceapplication.registrationapp.registerPatient to register a patient. ii) Fill in the details in all fields and click on Confirm button in confirm tab to submit. iii) In Chrome browser,right click on one of the fields and click inspect and in the open window click on Network tab iii) Then submit the page and notice in the network console many entries now have appeared iv) Now click on the link http://152.46.17.224:8082/openmrs-standalone/coreapps/clinicianfacing/patient.page?patientId=dc00030d-5ff0-4367-bf27-005b4256f6be appearing in the network console, once you click on it ,then click on the Preview tab in the subwindow that appears ,so there you will be able to see all the details submitted by the user being in clear text in the POST request payload. As in that payload all the details were in plaintext : TestGiven pMiddle 2Family Name Male 27 year(s) (02.Feb.1991) Edit Show Contact Info Hide Contact Info Test address, RaleighNCUS27607 Address 919112345 Telephone Number Edit Patient ID 1003EY
Expected Result	Submitted patient details after registration should not have appeared in the post payload as plain text, since a attacker can execute man in middle attack while capturing the payload data at transport layer.
Test Case Status	Failed
Mitigation to be taken by OpenMRS	The data submitted in the post payload must be encrypted to prevent a man in the middle attack.

Test Case ID	Test_A3_2
Test Name	**Checking presence of Browser Security directives**
Steps for test case	i) Go to http://152.46.17.224:8082/openmrs-standalone/login.htm and provide the details as admin and Admin123. ii) Now right click on page and click on Inspect and navigate to Network tab in the window that appear. iii) Now hit the log in button. iv) Now in the Network tab click on home.page in the pages that appear in Name column and then click on headers tab in the corresponding subwindow that appears on left. v) Now in the headers data that appears in the subwindow look for the ContentSecuritypolicy or CSP which is usually declared by <meta http-equiv="Content-Security-Policy" content="default-src 'self'; img-src https://*; child-src 'none';"> vi) As we observed we didn't found in the headers or page source any security policy directive being used in the openmrs which makes it prone to XSS and injection attacks.
Expected Result	There should have been a ContentSecurityPolicy or any other browser security policy directive should have been implemented in the OpenMRS.
Test Case Status	Failed

A4: XML External Entities (XXE)

Test Case ID	Test_A4_1
Test Name	**Apache Standard Taglibs (standard-1.1.2.jar)**
Vulnerabilities	Apache Standard Taglibs before 1.2.3 allows remote attackers to execute arbitrary code or conduct external XML entity (XXE) attacks via a crafted XSLT extension in a (1) <x:parse> or (2) <x:transform> JSTL XML tag.
CVE Search List	https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-0254
Mitigation Steps	i) Admin is advised to apply the appropriate updates, allow only trusted users to have network access, and monitor affected systems. ii) They may consider using IP-based access control lists (ACLs) to allow only trusted systems to access the affected systems.

Test Case ID	Test_A4_2
Test Name	**Apache POI (poi-3.12.jar)**
Vulnerabilities	Apache POI is vulnerable to a denial of service, cause by an XML External Entity Injection (XXE) error when processing XML data. By using a specially-crafted OOXML file, a remote attacker could exploit this vulnerability to consume all available CPU resources
CVE Search List	https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-5644
Mitigation Steps	Users with applications which accept content from external or untrusted sources are advised to upgrade to Apache POI 3.15 or newer.

A5: Broken Access Control

Test Case ID	Test_A5_1
Test Name	**Acting as admin when logged in as low-privilege user**
Steps for test case	i)Go to Admin Home Page using Admin credential. (URL: http://152.46.17.224:8082/openmrs-standalone/referenceapplication/home.page) ii) Click on System Administration on the home page and then click on“Advanced Administration”. iii) On the next page Select “Manage Users” Anchor link. iv) From the Role drop-down menu select “Organizational: Nurse” as roles. It will list Nurse user Select the hyperlink under System-ID **4-2**.that brings to next page. v) Change the username and password from the input item User password’s and confirm it in the next password box the same Password. I have selected the password “Ncsu1234”. And select the save user button to save the information. vi) Now login as nurse, then click on Appointment scheduling,where you don’t have access to manage service type, as this privilege was available to admin but not available for this role iii)But you open a new tab and openManage Service types page http://152.46.17.224:8082/openmrs-standalone/appointmentschedulingui/manageAppointmentTypes.page , then it will open where you can click on New service type button.
Expected Result	Nurse role didn’t had access to create new service type as that page was not available for that role in UI ,so manage services pages shouldn’t have appeared for nurse role
Test Case Status	Failed
Mitigation to be taken by OpenMRS	Various level of users of the application should be authorized to access different parts of the application after authentication.

Test Case ID	Test_A5_2
Test Name	**Elevation of privilege of standard user**
Steps for test case	i)Go to Admin Home Page using Admin credential. (URL: http://152.46.17.224:8082/openmrs-standalone/referenceapplication/home.page) ii) Click on System Administration on the home page and then click on“Advanced Administration”. iii) On the next page Select “Manage Users” Anchor link. iv) From the Role drop-down menu select “Organizational: Nurse” as roles. It will list Nurse user Select the hyperlink under System-ID 4-2.that brings to next page. v) Change the username and password from the input item User password’s and confirm it in the next password box the same Password. I have selected the password “Ncsu1234”. And select the save user button to save the information. vi) Now login as nurse,where on the dashboard you don’t have access to System administration option. vii) Now open manage users link which is available to admin user, http://152.46.17.224:8082/openmrs-standalone/admin/users/users.list?name=admin&role=&action=Search. viii) Now type admin in find user or name field, in search result admin records will appear, click on admin link appearing in first column of the record of the search result. viiii) Now in the page that appear, we can change the admin;s password and other details,so let’s change the password from Admin123 to Admin1234 and save it ix) Now logout and login with username as admin and password as Admin123, it will fail and now login with password Admin1234. x) So here , a standard user with not having access to manage the users did changed the admin account details.
Expected Result	User ‘nurse’ not having access to admin pages of manage users, shouldn’t have been able to access it and ability to also modify the data over there
Test Case Status	Failed
Mitigation to be taken by OpenMRS	Various level of users of the application should be authorized to access different parts of the application after authentication.

A6: Security Misconfiguration

Test Case ID	Test_A6_1
Test Name	**Unauthorized access to Admin Role Procedure To replicate**
Steps for test case	i) Go to Admin Home Page using Admin credential. (URL: http://152.46.17.224:8082/openmrs-standalone/referenceapplication/home.page) ii) Click on System Administration on the home page and then click on“Advanced Administration”. iii) On the next page Select “Manage Users” Anchor link. iv) From the Role drop-down menu select “Organizational: Nurse” as roles. It will list Nurse user Select the hyperlink under System-ID **4-2**.that brings to next page. V) Change the username and password from the input item User password’s and confirm it in the next password box the same Password. I have selected the password “Ncsu1234”. And select the save user button to save the information. Vii) Logout from admin role and login as username “nurse” and password which you have given in your password field. My case password was “Ncsu1234”. Viii) Now you have access to the nurse Dashboard which comes with limited access. Ix) Now login in another browser as Admin and copy the “Advanced Administration page” URL link and copy it to the browser where you have login as nurse. (URL : http://152.46.17.224:8082/openmrs-standalone/admin/index.htm) X) Now you see that you are able to access the Admin page, even though you was having limited role access.
Expected Result	When you login as nurse you should not be allowed the page of Admin only. We should get a suitable error message on the page when you try such link.
Test Case Status	Failed
Mitigation to be taken by OpenMRS	When a non-admin accesses an admin URL, an error message reading - “Access is denied” should be displayed. Appropriate checks should be placed to make sure that the level of user accessing the URL is authorized to do so.

Test Case ID	Test_A6_2
Test Name	**Unable to handle invalid urls**
Steps for test case	i) Login to portal as admin. ii) Navigate to System Administration > Advanced Administration > View Log Entries. iii) Click on Search then you'll get the list of log entries. iv) change value of generatedBy parameter to any invalid value for eg. http://152.46.17.224:8082/openmrs-standalone/module/idgen/viewLogEntries.list?source=&identifier=&fromDate=&toDate=&comment=&generatedBy=asdbhjsfs&action=Search v) Instead of getting an error such as "No Log Entries Found", we get complete error stack trace "HTTP Status 500 Request processing failed; nested exception is TypeMismatchException".
Expected Result	The application should be able to handle invalid urls and not throw error stack trace.
Test Case Status	Failed

A7: Cross-site scripting

Test Case ID	Test_A7_1
Test Name	**Reflected XSS in EditProfile**
Steps for test case	i) Go to Find patient page http://152.46.17.224:8082/openmrs-standalone/coreapps/findpatient/findPatient.page?app=coreapps.findPatient ii) Now click on the row for patient Paul Walker iii) Now click on Edit link on the page that appears. iv) Now click on Name link appearing on left sidebar, followed by which fill in details “</script><script>alert(1234)</script><script>” in the Middle name section and then on Save form on top of the page and finally click on Confirm button on the page that appears. v) Now again Paul Walker profile details page appear,so click on Edit button on top of the page. vi) Now alert box will popup showing message as ‘12345’, click on OK and then you will reach again back to edit page with all the details.
Expected Result	Script code should not have been accepted as a valid value for the Middle name field in edit profile field,as it allowed the script execution to print some value.
Test Case Status	Failed

Test Case ID	Test_A7_2
Test Name	**Check stored XSS in creating Service types**
Steps for test case	i) Go to Manage Service types page http://152.46.17.224:8082/openmrs-standalone/appointmentschedulingui/manageAppointmentTypes.page and then click on New service type button. ii) Now in service name, type in " onmouseover="('page hacked')" and fill other parameters and click on save to save that record. iii) Now go to http://152.46.17.224:8082/openmrs-standalone/appointmentschedulingui/home.page and then again go to service type pagehttp://152.46.17.224:8082/openmrs-standalone/appointmentschedulingui/manageAppointmentTypes.page iv) Now do a mouseover on the page and you won't notice the alert as per the script we injected in the service name.
Expected Result	There shouldn't be an alert message popup for the script injected in the name field of new service type
Test Case Status	Passed
Mitigation by OpenMRS team	Name input field had been sanitized before loading it.

A8: Insecure Deserialization

Test Case ID	Test_A8_1
Test Name	**Jackson (jackson-databind-2.8.1.jar)**
Vulnerabilities	A deserialization flaw was discovered in the jackson-databind in versions before 2.8.10 and 2.9.1, which could allow an unauthenticated user to perform code execution by sending the maliciously crafted input to the readValue method of the ObjectMapper. This issue extends the previous flaw CVE-2017-7525 by blacklisting more classes that could be used maliciously.
CVE Search List	https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-15095
Mitigation Steps	Mitigation to this problem is to not trigger polymorphic deserialization globally by using: objectMapper.enableDefaultTyping() and rather use @JsonTypeInfo on the class property to explicitly define the type information. For more information on this issue please refer to https://www.github.com/mbechler/marshalsec/blob/master/marshalsec.pdf?raw=true

Test Case ID	Test_A8_2
Test Name	**Spring JMS (spring-jms-3.0.5.RELEASE.jar)**
Vulnerabilities	Spring Framework 3.0.0 through 3.0.5, Spring Security 3.0.0 through 3.0.5 and 2.0.0 through 2.0.6, and possibly other versions deserialize objects from untrusted sources, which allows remote attackers to bypass intended security restrictions and execute untrusted code by (1) serializing a java.lang.Proxy instance and using InvocationHandler, or (2) accessing internal AOP interfaces, as demonstrated using deserialization of a DefaultListableBeanFactory instance to execute arbitrary commands via the java.lang.Runtime class.
CVE Search List	https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-2894
Mitigation Steps	Users should migrate away from serialization-based remoting in cases where the client cannot be trusted, as it is a potential source of vulnerabilities in both Spring and non-Spring applications. All users may mitigate this issue by upgrading to Spring Framework 3.0.6 and Spring Security 3.0.6. Spring Framework users should make use of the additional features introduced to prevent deserialization of malicious proxies.

A9: Using Components with Known Vulnerabilities

Test Case ID	Test_A9_1
Test Name	**Apache Solr Core (solr-core-4.10.4.jar)**
Vulnerabilities	Cross-site scripting (XSS) vulnerability in webapp/web/js/scripts/plugins.js in the stats page in the Admin UI in Apache Solr before 5.3.1 allows remote attackers to inject arbitrary web script or HTML via the entry parameter to a plugins/cache URI.
CVE Search List	https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-8797

Test Case ID	Test_A9_2
Test Name	**Hibernate Bean Validation (hibernate-validator-4.2.0.Final.jar)**
Vulnerabilities	In Hibernate Validator 5.2.x before 5.2.5 final, 5.3.x, and 5.4.x, it was found that when the security manager's reflective permissions, which allows it to access the private members of the class, are granted to Hibernate Validator, a potential privilege escalation can occur. By allowing the calling code to access those private members without the permission an attacker may be able to validate an invalid instance and access the private member value via ConstraintViolation#getInvalidValue().
CVE Search List	https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-7536

A10: Insufficient logging and monitoring

Test Case ID	Test_A10_1
Test Name	**Sensitive Information Exposure**
Steps for test case	i) From Admin Home page navigate to “System Administration” tab. URL for admin page is (http://152.46.17.224:8082/openmrs-standalone/index.htm). ii) In System Administration page if you select “Manage Module“ option, It shows 404 Error. Along with this information it shows the Apache tomcat version (7.0.50). iii) Using this information I can find some vulnerability in apache tomcat server and carry out the attack on this application. This can lead to attack of type A3: 2017-Sensitive Information Exposure.
Expected Result	Whenever there some page which can not be dynamically generate to user then we should handle these error with some Obscure message about the error without letting end user know about my software Stack information.
Test Case Status	Failed

Test Case ID	Test_A10_2
Test Name	**Logging the login information and all the Server based transactions and events**
Steps for test case	i) Go to http://152.46.17.224:8082/openmrs-standalone/admin/maintenance/currentUsers.list to see the list of logged in users in application. ii) Also go to http://152.46.17.224:8082/openmrs-standalone/admin/maintenance/serverLog.form to view the server log to see all the application based transactions and events logged there which did provide a comprehensive view to monitor the suspicious events or all the modules accessed by respective users
Expected Result	Application should log the logged in users and all the applications based transactions or events for monitoring the suspicious activity.
Test Case Status	Passed
Mitigation by OpenMRS team	Audit trail maintained to log high value transactions and also a view to see the logged in users to monitor the suspicious activity

References:

i) https://www.owasp.org/index.php/Category:OWASP_Top_Ten_2017_Project for top ten vulnerabilities
ii) https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP for understanding browser security policy directives
iii) https://issues.openmrs.org/secure/Dashboard.jspa to see what all issues are open ,so that instead find a new one rather than finding a existing one.
iv) https://cxsecurity.com/issue/WLB-2011100117 for mitigations steps for Spring JMS issue.
v) https://poi.apache.org/ for mitigations steps for Apache POI issue.