REPORT - 4

1. **Protection Poker (20 points)**

   Write 5 new functional requirements for OpenMRS to add functionality that is not in the system yet. Play protection poker, using the OpenMRS database tables on these requirements. The data model is described on OpenMRS wiki - https://wiki.openmrs.org/display/docs/Data+Model. Rank the security risk for the 5 requirements. Justify your relative values for ease and values when you compare the five new requirements. (For example, why would one requirement be 5 times easier to attack when compared to another. Based upon the security risk, document the steps you would take to mitigate, eliminate, transfer or accept (META) the risk for each of the requirements.

Possible point values: 1, 3, 5, 8, 13, 20, 40, 100

The following are the domains for the datatables which are used in the below table:

1. **Encounter: Contains the meta-data regarding health care providers interventions with a patient.**
2. **Observation: This is where the actual health care information is stored. There are many observations per Encounter.**
3. **Patient: Basic information about patients in this system.**
4. **User: Basic information about the people that use this system.**
5. **Person: Basic information about person in the system.**

All assets mentioned in the below table are the tables from one of the domains above. Any asset with patient information & obs are considered to be of high value.

Number	Feature	Assets	Value points	Ease points	Risk Exposure	Rank
1	A Patient should be notified if their medical records were accessed or modified.	Patient, patient_identifier, patient_identifier, obs, encounter, person.	40	13	520	4

		(This includes details about medical record, patient information and staff who modifies/accesses this information)				
2	Adding multi-factor authentication. This will ensure a 2-step verification and proper authentication of the legitimate user.	Users, user_role, role, role_privileges, privileges, role_role, person, Patient, patient_identifier, obs, encounter (This includes details about the people who use the system - these users have access to all the medical information)	100	3	300	5
3	Ensuring strong password constraints (using special characters like, $,%, etc.) for each user login.	users , user_roles,role_role, role, role_privileges, obs, Patient, patient_identifier, obs, encounter (this includes all the user related tables , especially the one that contain hashed and salted password)	100	13	1300	2
4	Ensure that a user can reset the password, without much intervention from the system administrator. Currently, the user can not reset the password and the administrator	users , user_roles,role_role, role, role_privileges, Patient, patient_identifier, obs, encounter	100	8	800	3

	has to reset the password	(this includes all the user related tables , especially the one that contain hashed and salted password and role based attributes)				
5	Users (as in patient) should be able to access their medical records directly. They should also be able to request to delete the data that contains personal identifiable information	Patient, patient_identifier_type, visit, visit_type, visit_attribute, obs (this includes all the tables related to patients and other health related information such as medications.	40	40	1600	1

Either all patient records are accessible or a particular patient record is accessible by these features, thus thy have the value points of 100/40 respectively

Comment:

1. Feature 5 has the highest risk exposure factor - for patients to view their records they would need their own account using which they can login to the system to check their details. If an adversary can get these credentials he can get all the details about this patient. Since it involves patient information we have given value points as 40, the details of which can be obtained from the patient database table.

   To get these details, the adversary has to just get the credentials of the user/patient, if a strong password isn't set he can easily get these details. Thus we have given ease points as 40.

2. The next one would be the 3rd feature. If an adversary is able to get the credentials, he can access all the medical records for all the patients and can also try to get most of the passwords stored in Database(user table). Thus, we have given value points as 60.

Since an adversary has to perform Brute Force/Dictionary attack to get the credentials, this has lower ease points. Because these involve - trying a lot of combinations till he gets the needed information.

3. An attacker can use the reset password feature to login to the system, if he has the username details. If he logs in he can view all patients details. Thus it has value points as 100.
   But he has to know the account associated with the reset password option to get the code/temporary password. Thus we have given ease points of 13.

   Thus, when we compare this with feature 1, it is 5 times easier to attack feature 1.

4. The risk exposure factor for feature 1 is comparatively lower - the reason being even if the adversary tries to attack he can only get a particular patients' record and only those details which are associated with the record that was accessed/modified. Thus, we have given value points as 40.
   It is not easy for an adversary to get these details because he would have to get information about how the patient is being notified and try to access the device/account. Therefore, we have given the ease points as 30

5. Feature 2 is the one with least risk exposure factor. The reason being 2FA/multi factor authentication can protect users from multiple security bugs. This reduces the chances of unauthorized access. But if an adversary can get these details/credentials, he can get all patient records stored, thus, it has value points as 100. He can get all the details from the patient table.
   But it is the hardest to attack, the reason being an attacker would have to get the credentials for both the steps of verification. Hence, the ease points is set to 3.

META:

1. **Patient Notification:**
   Risk Mitigation:
   The mitigation would be to ask the customer for consent every time someone wants to view a patient record. If the customer has not consented for data access , that means that an unauthorised person is trying to access the data. Also, all the data access events should be logged properly (with the identity of the user you accessed it and timestamp) , so that it can be analysed later if some concern comes up.

2. **Multi Factor Authentication:**
   Risk Acceptance:
   Although the value points associated with it is very high, the risk can be accepted here, since it has the least risk exposure factor. Furthermore, cracking a 2 factor authentication is a complex task.

3. **Strong Password:**
   Risk mitigation:
   This can be mitigated by enabling other forms of biometric login such as fingerprint or iris scan. This is inherent in a user and he/she cannot forget it. Even if passwords are used, steps should be taken to ensure that the passwords meet the necessary constraints like the length of password, using numerals, capital and small characters etc.

4. **Reset password:**
   Risk Mitigation:
   This can be mitigated by verifying that the user trying to login by claiming to have lost his password is legitimate or not. Enforcing the use of secondary email for any passwords forgotten or losing access to an account as a backup is a good solution. Also, setting up strong security questions whose answers would be known only to the legitimate user shall be a good idea when raising alerts like forgotten passwords.

5. **Records to be accessed by patients:**
   Risk Mitigation:
   The risk here is that the patient account can be compromised. The possible way to mitigate that is by enabling 2 factor authentication (or multi level authentication).

2. **Vulnerability Fixes (35 points)**

Using OpenMRS and Eclipse on the VCL, develop fixes to vulnerabilities you have found this semester. The vulnerabilities could be those found during any of Project Parts 1, 2, or 3. The vulnerability must be a security-related bug.

Specify which test case, tool, etc. from a previous report you are fixing. Reference your old report + provide a screen shot from that report. If your reference a black box test that does not have repeatable steps, enhance the test case to be repeatable now so you do not lose points now.
If you really want to generate a new vulnerability, also submit a complete, repeatable black box test to generate it.
Submit these fixes by (1) copying (e.g. text or screenshot) a diff of the fix into your report; (2) explaining the vulnerability and the fix; and (3) screenshot of old vulnerability OR new test case to a vulnerability.
Each person on the team should submit a vulnerability fix, though the team can collaborate on helping the team member with the fix. Put your name on the vulnerability fix you submit.
A person who does not submit a vulnerability fix can not be awarded the points for this part of the project while the rest of the team can.

For those team members who do submit vulnerability fixes, the grade will be an overall team grade -- so you should still be motivated to collaborate with each other.

Bug fix 1

Project Report 1 - Coverity - Testcase - 2 - SQL Injection

Screenshots from Report 1 – Coverity testcase:

2. **Module Name** – htmlformentry
CID 10996: SQL Injection
Category: Medium Impact Security
ASVS 5.3.4 :
Verify that data selection or database queries (e.g. SQL, HQL, ORM, NoSQL) use parameterized queries, ORMs, entity frameworks, or are otherwise protected from database injection attacks
CWE: 89
Vulnerability type: True positive

Description:

The parameter 'searchAttribute' is tainted as it is one of the request params of the incoming HTTP request. In the function, getPersonIdHavingAttributes(), a SQL query statement is created using this param. As the SQL command string flows out of the function, it can be used to access the database elsewhere.

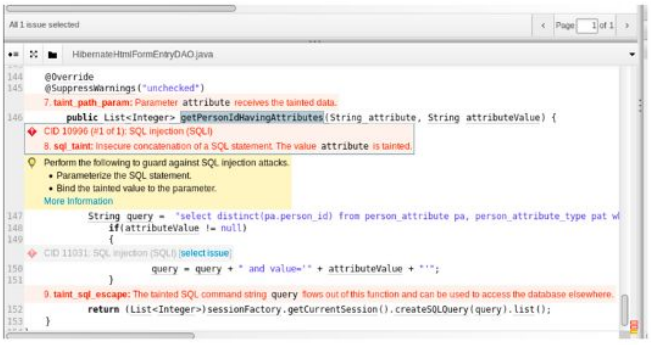
Vulnerability Fix:

To remedy both SQL injection and preserve the meaning of the query, one could:
1.Parameterize the SQL statement.

2.Escape the percent sign (%, U+0025) and underscore (_, U+005F) characters within the string used in the LIKE query '%...%'.

3. Bind the value to a parameter within the LIKE clause.

Snapshot:



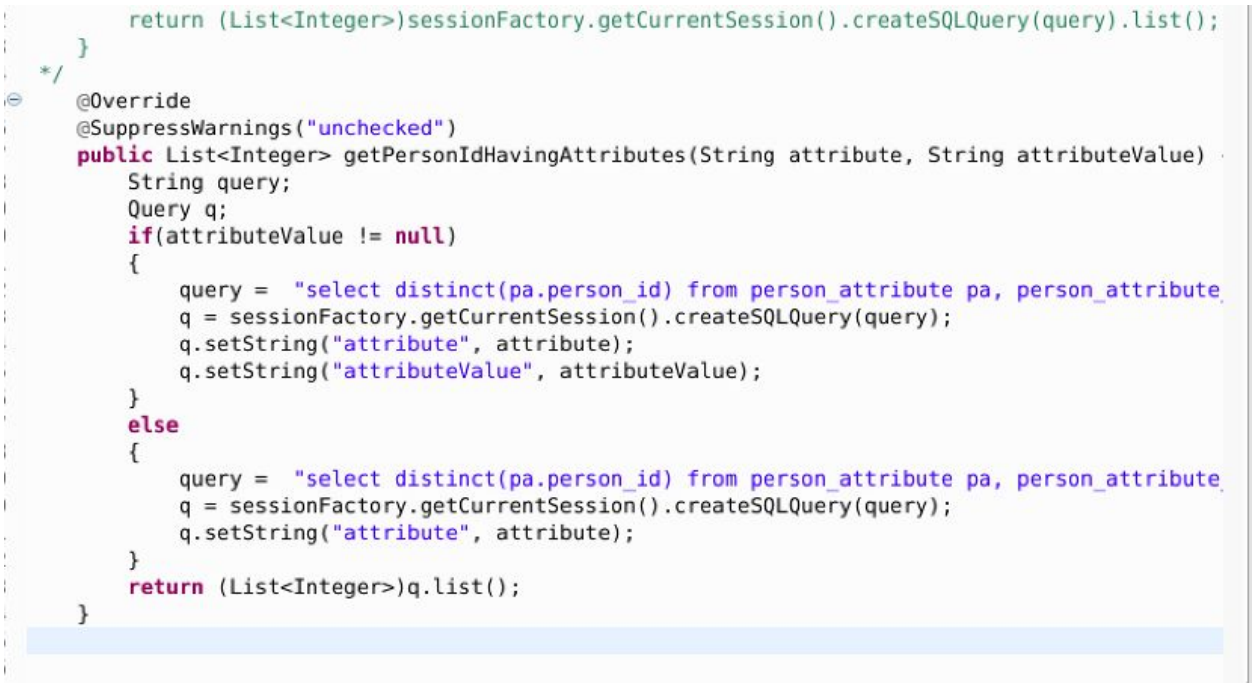
Added the following fix to prevent SQL injection attack – used prepared statements:

Given Code	Fix

@Override
@SuppressWarnings
("unchecked")
public List<Integer>
getPersonIdHavingAt
tributes(String
attribute, String
attributeValue) {
String query =
"select
distinct(pa.person_id
) from
person_attribute pa,
person_attribute_typ
e pat where
pa.person_attribute_t
ype_id =
pat.person_attribute_
type_id and
pat.name='" +
attribute + "'";
if(attributeValue !=
null)
{
query = query + "
and value='" +
attributeValue + "'";
}
return
(List<Integer>)sessio
nFactory.getCurrentS
ession().createSQLQ
uery(query).list();
}
	@Override
	
	@SuppressWarnings("unchecked")
	
	public List<Integer> getPersonIdHavingAttributes(String attribute, String attributeValue) {
	
	String query;
	
	Query q;
	
	if(attributeValue != null)
	
	{
	
	query = "select distinct(pa.person_id) from person_attribute pa, person_attribute_type pat where pa.person_attribute_type_id = pat.person_attribute_type_id and pat.name=:attribute and value=:attributeValue";
	
	q = sessionFactory.getCurrentSession().createSQLQuery(query);
	
	q.setString("attribute", attribute);
	
	q.setString("attributeValue", attributeValue);
	
	}
	
	else
	
	{
	
	query = "select distinct(pa.person_id) from person_attribute pa, person_attribute_type pat where pa.person_attribute_type_id = pat.person_attribute_type_id and pat.name=:attribute";
	
	q = sessionFactory.getCurrentSession().createSQLQuery(query);
	
	q.setString("attribute", attribute);
	
	}

	`return (List<Integer>)q.list();`
	
	`}`

Fix added in Eclipse:



```
        return (List<Integer>)sessionFactory.getCurrentSession().createSQLQuery(query).list();
    }
*/
    @Override
    @SuppressWarnings("unchecked")
    public List<Integer> getPersonIdHavingAttributes(String attribute, String attributeValue)
        String query;
        Query q;
        if(attributeValue != null)
        {
            query =  "select distinct(pa.person_id) from person_attribute pa, person_attribute
            q = sessionFactory.getCurrentSession().createSQLQuery(query);
            q.setString("attribute", attribute);
            q.setString("attributeValue", attributeValue);
        }
        else
        {
            query =  "select distinct(pa.person_id) from person_attribute pa, person_attribute
            q = sessionFactory.getCurrentSession().createSQLQuery(query);
            q.setString("attribute", attribute);
        }
        return (List<Integer>)q.list();
    }
```

Explanation:

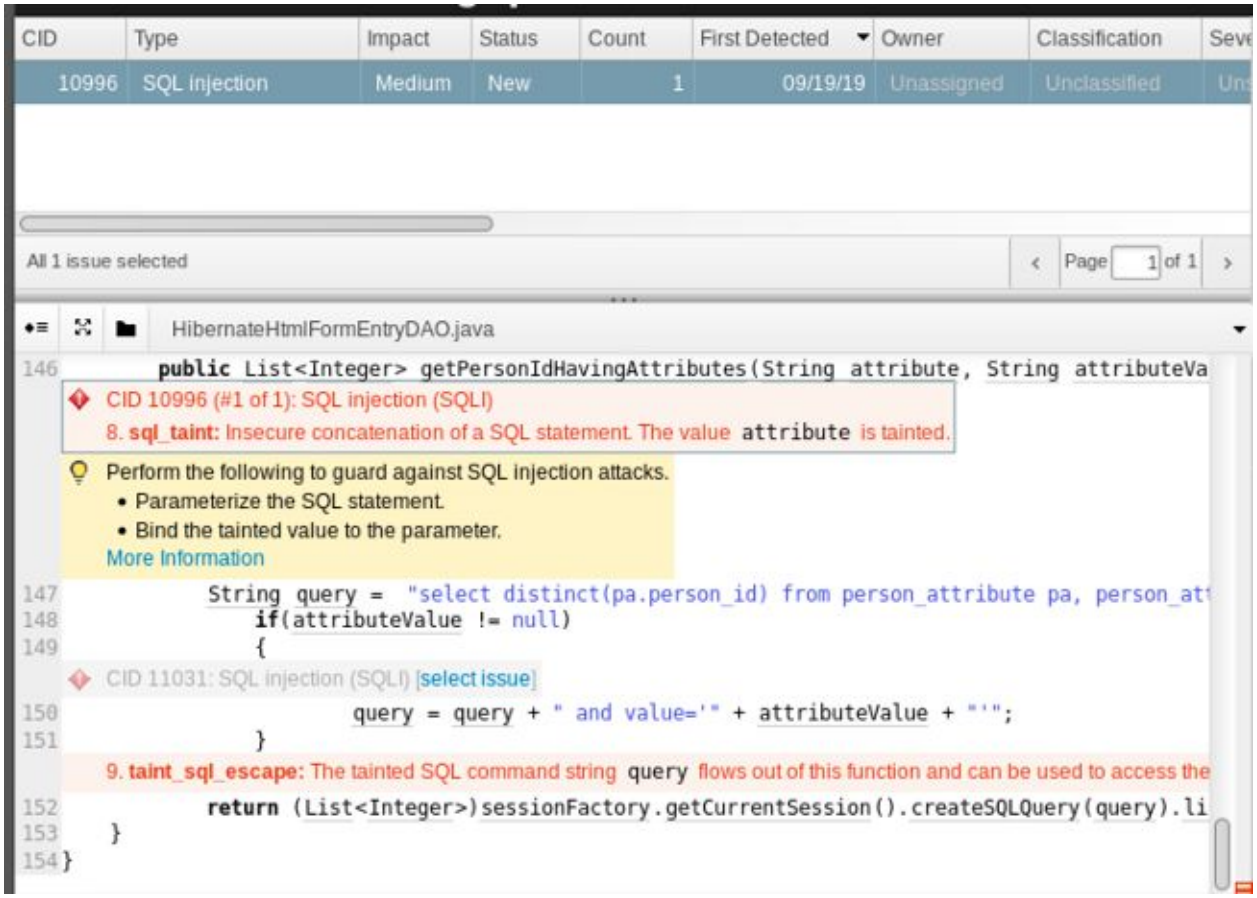
The code that was there earlier in HIbernateHTMLFormEntryDao.java was normal SQL statements to query the database.

Reason - This makes it vulnerable to SQL injection attack because the input that is passed to the query is not validated.

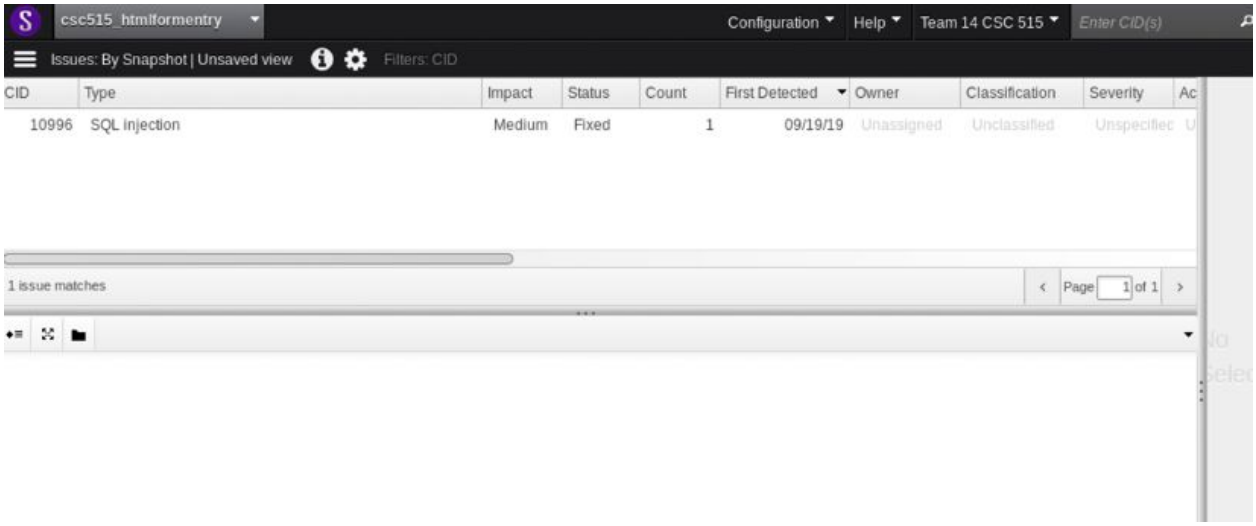
Fix - I have updated these to use prepared statements. This forces the user input to be handled as the content of a parameter (and not as a part of the SQL command).

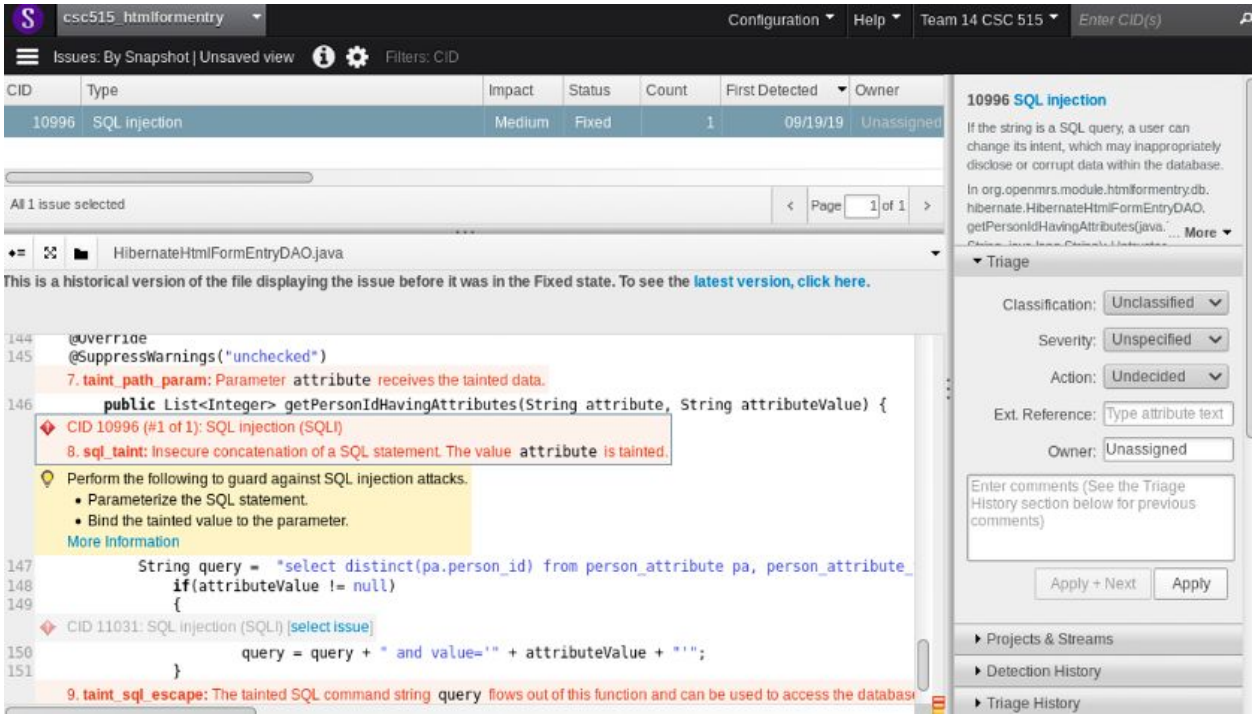
Coverity report before adding the fix for the module HTMLformentry – CID - 10996:

CID	Type	Impact	First Detected	Owner	Classification	Severity	Action	Component	Category	Issue K
11018	Resource leak on an e:	Low	09/19/19	Unassigned	Unclassified	Unspecifiec	Undecide	Other	Exceptional resource le	Variou
12486	Resource leak on an e:	Low	01/26/20	Unassigned	Unclassified	Unspecifiec	Undecide	Other	Exceptional resource le	Variou
11008	Filesystem path, filenar	High	09/19/19	Unassigned	Unclassified	Unspecifiec	Undecide	Other	High impact security	Secur
11060	Filesystem path, filenar	High	09/19/19	Unassigned	Unclassified	Unspecifiec	Undecide	Other	High impact security	Secur
11025	Unsafe reflection	Low	09/19/19	Unassigned	Unclassified	Unspecifiec	Undecide	Other	Low impact security	Secur
11028	Unsafe reflection	Low	09/19/19	Unassigned	Unclassified	Unspecifiec	Undecide	Other	Low impact security	Secur
11039	Unsafe reflection	Low	09/19/19	Unassigned	Unclassified	Unspecifiec	Undecide	Other	Low impact security	Secur
10995	Open redirect	Medium	09/19/19	Unassigned	Unclassified	Unspecifiec	Undecide	Other	Medium impact security	Secur
10996	SQL injection	Medium	09/19/19	Unassigned	Unclassified	Unspecifiec	Undecide	Other	Medium impact security	Secur
11016	Open redirect	Medium	09/19/19	Unassigned	Unclassified	Unspecifiec	Undecide	Other	Medium impact security	Secur
11031	SQL injection	Medium	09/19/19	Unassigned	Unclassified	Unspecifiec	Undecide	Other	Medium impact security	Secur
10939	Resource leak	High	09/19/19	Unassigned	Unclassified	Unspecifiec	Undecide	Other	Resource leaks	Variou
10972	Resource leak	High	09/19/19	Unassigned	Unclassified	Unspecifiec	Undecide	Other	Resource leaks	Variou
10984	Resource leak	High	09/19/19	Unassigned	Unclassified	Unspecifiec	Undecide	Other	Resource leaks	Variou
10992	Resource leak	High	09/19/19	Unassigned	Unclassified	Unspecifiec	Undecide	Other	Resource leaks	Variou
11006	Resource leak	High	09/19/19	Unassigned	Unclassified	Unspecifiec	Undecide	Other	Resource leaks	Variou

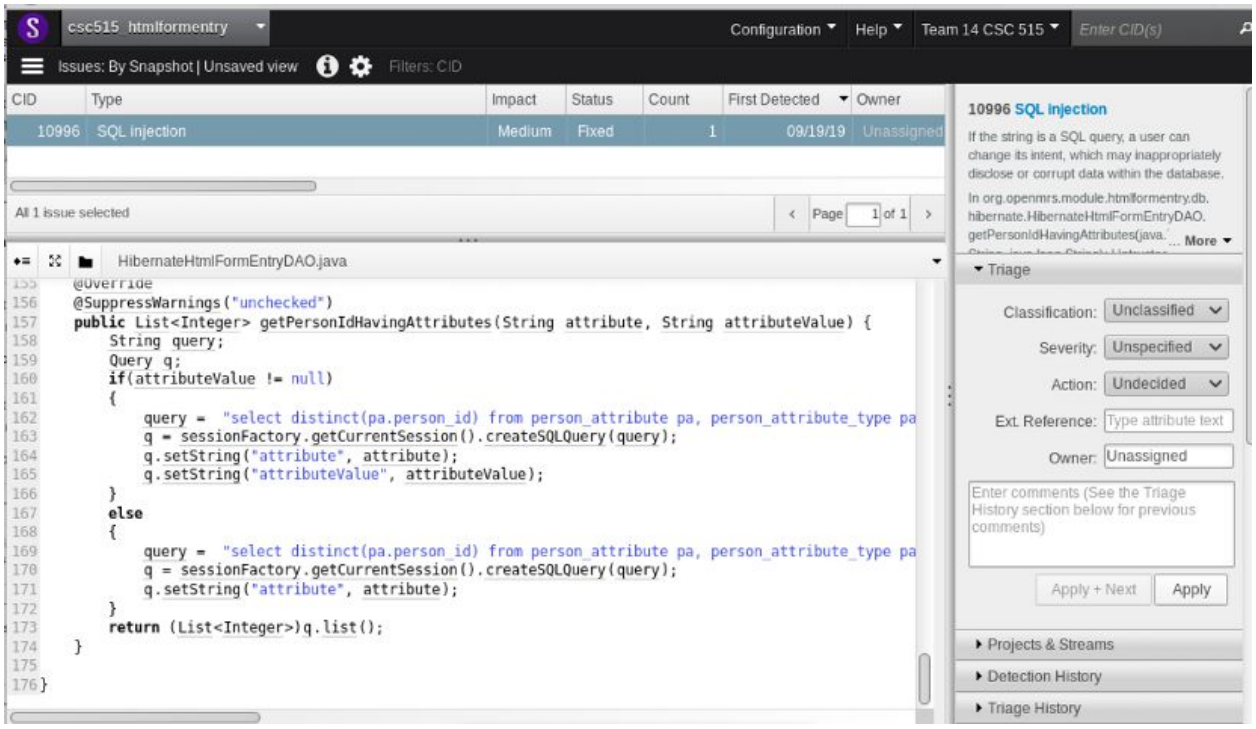


After adding the fix, ran coverity tests again for the module - HTMLFormentry:
It displays that the error is fixed under the status column, also shows the changes in the code, as displayed in the below screenshot.





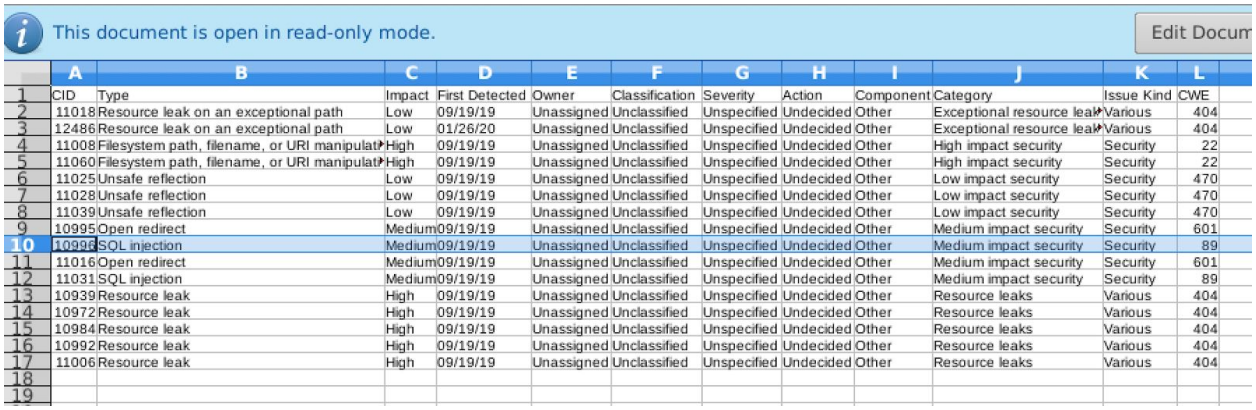
On clicking the latest version:



[The report does not state these changes - it does not display status column, adding screenshots of report for your reference -
Old:

CID	Type	Impact	First Detected	Owner	Classification	Severity	Action	Component	Category	Issue Kind	CWE
11018	Resource leak on an exceptional path	Low	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Exceptional resource leaks	Various	404
12486	Resource leak on an exceptional path	Low	01/26/20	Unassigned	Unclassified	Unspecified	Undecided	Other	Exceptional resource leaks	Various	404
11008	Filesystem path, filename, or URI manipulation	High	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	High impact security	Security	22
11060	Filesystem path, filename, or URI manipulation	High	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	High impact security	Security	22
11025	Unsafe reflection	Low	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Low impact security	Security	470
11028	Unsafe reflection	Low	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Low impact security	Security	470
11039	Unsafe reflection	Low	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Low impact security	Security	470
10995	Open redirect	Medium	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Medium impact security	Security	601
10996	SQL injection	Medium	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Medium impact security	Security	89
11016	Open redirect	Medium	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Medium impact security	Security	601
11031	SQL injection	Medium	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Medium impact security	Security	89
10939	Resource leak	High	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Resource leaks	Various	404
10972	Resource leak	High	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Resource leaks	Various	404
10984	Resource leak	High	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Resource leaks	Various	404
10992	Resource leak	High	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Resource leaks	Various	404
11006	Resource leak	High	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Resource leaks	Various	404

New:



This document is open in read-only mode. Edit Docum

	A	B	C	D	E	F	G	H	I	J	K	L
1	CID	Type	Impact	First Detected	Owner	Classification	Severity	Action	Component	Category	Issue Kind	CWE
2	11018	Resource leak on an exceptional path	Low	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Exceptional resource leak	Various	404
3	12486	Resource leak on an exceptional path	Low	01/26/20	Unassigned	Unclassified	Unspecified	Undecided	Other	Exceptional resource leak	Various	404
4	11008	Filesystem path, filename, or URI manipulati	High	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	High impact security	Security	22
5	11060	Filesystem path, filename, or URI manipulati	High	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	High impact security	Security	22
6	11025	Unsafe reflection	Low	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Low impact security	Security	470
7	11028	Unsafe reflection	Low	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Low impact security	Security	470
8	11039	Unsafe reflection	Low	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Low impact security	Security	470
9	10995	Open redirect	Medium	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Medium impact security	Security	601
10	10998	SQL injection	Medium	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Medium impact security	Security	89
11	11016	Open redirect	Medium	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Medium impact security	Security	601
12	11031	SQL injection	Medium	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Medium impact security	Security	89
13	10939	Resource leak	High	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Resource leaks	Various	404
14	10972	Resource leak	High	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Resource leaks	Various	404
15	10984	Resource leak	High	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Resource leaks	Various	404
16	10992	Resource leak	High	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Resource leaks	Various	404
17	11006	Resource leak	High	09/19/19	Unassigned	Unclassified	Unspecified	Undecided	Other	Resource leaks	Various	404
18												
19												

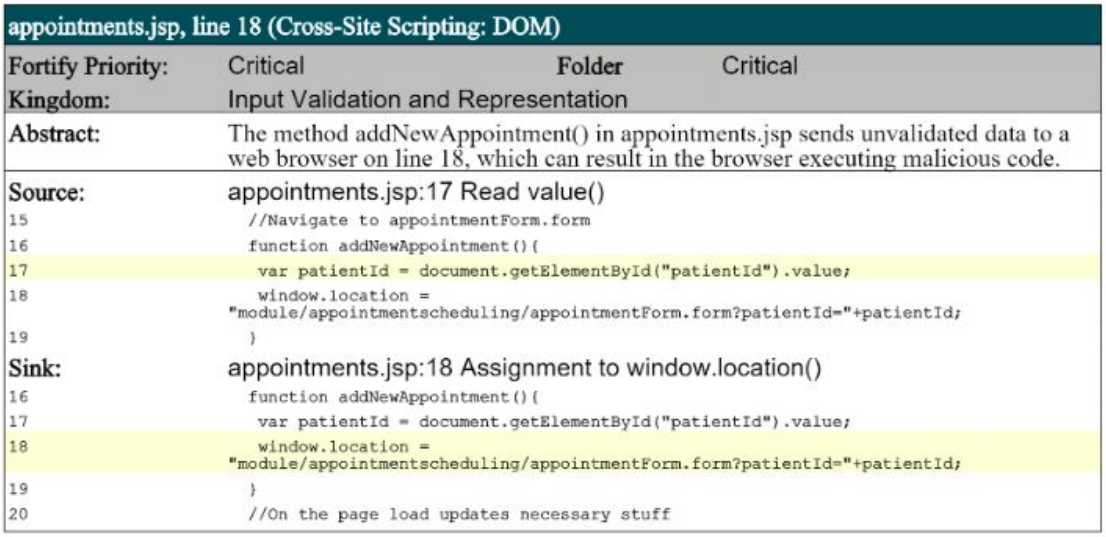
]

Bug fix 2

Project Report 1 - Fortify detected a true positive vulnerability.

Input validation and representation issue in appointments.jsp page. The user input is stored in the variable, patientId on line 17 as per the screenshot attached which is passed into a web address without a sanity check of the input. Hence, this can be considered as a potential True positiveresult. This vulnerability could result in phishing attacks, browserexecuting malicious code, allow unvalidated input to control the URL.

Module	RESULT	ASVS	CWE
appointmentscheduling	True Positive	5.1.3	20



appointments.jsp, line 18 (Cross-Site Scripting: DOM)			
Fortify Priority:	Critical	Folder	Critical
Kingdom:	Input Validation and Representation		
Abstract:	The method addNewAppointment() in appointments.jsp sends unvalidated data to a web browser on line 18, which can result in the browser executing malicious code.		

Source: appointments.jsp:17 Read value()

```
15    //Navigate to appointmentForm.form
16    function addNewAppointment(){
17     var patientId = document.getElementById("patientId").value;
18     window.location =
"module/appointmentscheduling/appointmentForm.form?patientId="+patientId;
19    }
```

Sink: appointments.jsp:18 Assignment to window.location()

```
16    function addNewAppointment(){
17     var patientId = document.getElementById("patientId").value;
18     window.location =
"module/appointmentscheduling/appointmentForm.form?patientId="+patientId;
19    }
20    //On the page load updates necessary stuff
```

Fix is added in appointments.jsp file in the appointmentscheduling module:

```
function addNewAppointment(){
                var patientId = document.getElementById("patientId").value;
                if (isNaN(patientId))
                {
                        return false;
                }
                else
                {
                        return true;
                }
                window.location =
"module/appointmentscheduling/appointmentForm.form?patientId="+patientId;
        }
```

Note: Running the fortify analysis on the module again did not trigger a vulnerability indicating that it has been fixed. The old and new Fortify reports have been attached for reference.

```
13
14ipt type="text/javascript">
15    //Navigate to appointmentForm.form
16    function addNewAppointment(){
17        var patientId = document.getElementById("patientId").value;
18        if (isNaN(patientId))
19        {
20              return false;
21        }
22        else
23        {
24              return true;
25        }
26        window.location = "module/appointmentscheduling/appointmentForm.form?patientId="+patient
27    }
28    //On the page load updates necessary stuff
29     $j(document).ready(function() {
30            //data table initialization
31            var oTable = $j('#AppointmentsTable').dataTable({
32             "aLengthMenu": [[5, 10, 25, 50, -1], [5, 10, 25, 50, "All"]],
33                "aoColumns" : [ {
```

Bug fix 3
Fortify test case 9
Note : in the previous report we misjudged the vulnerability to be in 'reportingcompatibility' module. I have corrected it here.

Module	RESULT	ASVS	CWE
registrationcore	True Positive	11.1.6	367

Here is a screen capture of the issue before I applied the fix.

IdentifierBuilder.java, line 80 (Race Condition: Singleton Member Field)

Fortify Priority:	High	Folder	High
Kingdom:	Time and State		
Abstract:	The class IdentifierBuilder is a singleton, so the member field iss is shared between users. The result is that one user could see another user's data.		
Sink:	IdentifierBuilder.java:80 AssignmentStatement()		

```
78      private IdentifierSourceService getIss() {
79          if (iss == null) {
80              iss = Context.getService(IdentifierSourceService.class);
81          }
82          return iss;
```

Reason for the the issue:

- It seems that there is an issue with the class declaration IdentifierBuilder. It has a private variable declared as iss.
- This is a singleton class. It means that only one object of the class has been created.

- Consider a scenario, if there are multiple processes trying to access a particular variable.
- There might be a race condition and that is exactly what Fortify is reporting. On the other hand fortify does not know if there are proper locking mechanisms for this variable in place. There are a lot of resource locking functions in place, if we go through the source code.

Fix for this issue:

- I have updated the declaration to ‘ private static volatile IdentifierSourceService iss; ‘
- The keyword volatile was used to indicate this variable will be accessed by multiple threads.
- The keyword static was used , because , if in future more objects of this class are initialised they all will have one common variable

Cross-checking for this vulnerability:

- After changing the source code in File : IdentifierBuilder.java , I rebuilt the module.
- After rebuilding the module ‘registrationcore’ , i rebuilt the entire openmrs project
- Then I ran static analysis on the module ‘registrationcore’.
- After checking the report there was no issues in the file IdentifierBuilder.java, for this particular piece of code
- I have attached the report for the module ‘registrationcore’ for reference.

Screen capture of the code fix :

```
public class IdentifierBuilder {

    private LocationService locationService;

    private static volatile IdentifierSourceService iss;

    private PatientService patientService;

    public void setLocationService(LocationService locationService) {
        this.locationService = locationService;
    }

    public void setPatientService(PatientService patientService) {
        this.patientService = patientService;
    }
```

Bug fix 4:

3. **Exploratory Penetration Testing (35 points)**

Video link 1:
PART1(50m 59s) **-**
https://drive.google.com/open?id=194mKakHhlkHiad3-lDpKoqrlYcJbFC8g
PART2(51m - 01h 55m 3s) **-**
https://drive.google.com/open?id=1_mJTG8VgbgwN8w9KSrKkqf5j9K38bQDn
PART3(01h 55m 38s - 03h 03m)**-**
https://drive.google.com/open?id=1Xc-HU6Xa2RMWumYfwR91H3xtLhUUGugC

Vulnerability/Testcase number	Elapsed Time	Ref Info for Video Traceability	CWE	Commentary
1	3m 30s	Video Link 1 - PART1	200	Password in plain text
2	25m 10s	Video Link 1 - PART1	209	Stack trace
3	29m 45s	Video Link 1 - PART1	285	Access control – Clerk can access admin
4	45m 07s	Video Link 1 - PART1	613	Once you login from API page, application never logs you out
5	26m 07s	Video Link 1 - PART1	613	Session logout incorrect
6	08m 59s	Video Link 1 - PART2	79	Stored XSS attack
7	39m 41s	Video Link 1 - PART2	79	Stored XSS attack
8	41m 30s	Video Link 1 - PART2	94	UI displays code
9	43m 30s	Video Link 1 - PART2	79	Stored XSS attack

10	45m 30s	Video Link 1 - PART2	94	UI displays code
11	01h 02m	Video Link 1 - PART2	544	Complete Stack trace displayed
12	01h 03m 03s	Video Link 1 - PART2	79	Reflected XSS attack
13	13m 37s	Video Link 1 - PART 3	544	Complete Stack trace displayed
14	30m 50s	Video Link 1 - PART3	79	Stored XSS attack
15	52m 22s	Video Link 1 - PART 3	285	If clerk gets the Patient ID, he can delete patient information
16	54m 19s	Video Link 1 - PART3	79	Stored XSS attack

Black Box test cases:

Testcase	**Unique ID**	**ASVS**	**CWE**
1	**1.6.4-1**	**1.6.4**	**200**

ASVS Name: Cryptographic Architectural Requirements

ASVS Description: Verify that symmetric keys, passwords, or API secrets generated by or shared with clients are used only in protecting low risk secrets, such as encrypting local storage, or temporary ephemeral uses such as parameter obfuscation. Sharing secrets with clients is clear-text equivalent and architecturally should be treated as such.

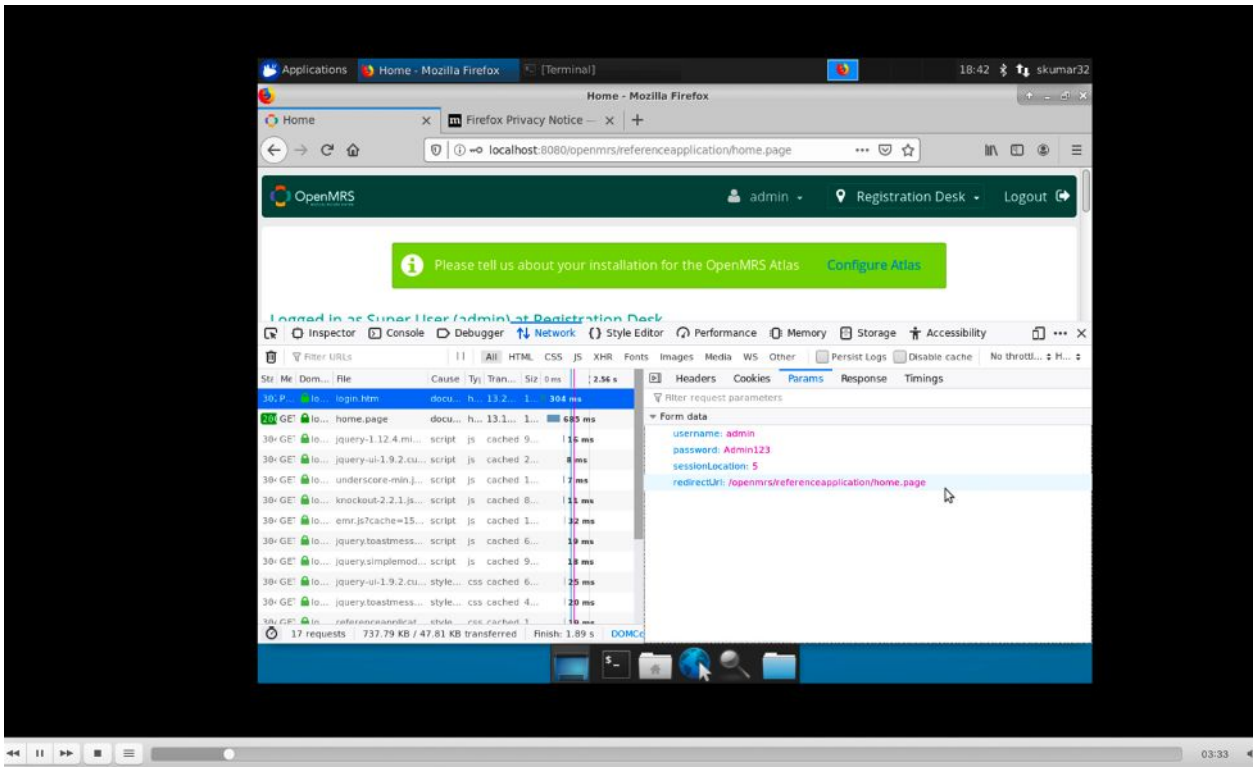
CWE Name: CWE320 - Key Management Errors

Repeatable Steps:

1. Navigate OpenMRS login page - localhost:8080/openmrs/index.htm
2. Click on Firefox settings > Web developer > Inspector > network tab
3. Enter credentials in the login page. Username – Admin; Password: Admin123
4. Choose registration desk as location
5. Click on login
6. Now, in the network tab, click on /login.htm request and check params option

Expected Results:
Password should not be displayed in plain text. Sensitive information should always be encrypted.

Actual Results:
Password is displayed in plain text



Testcase	Unique ID	ASVS	CWE
2	14.3.1-1	14.3.1	209

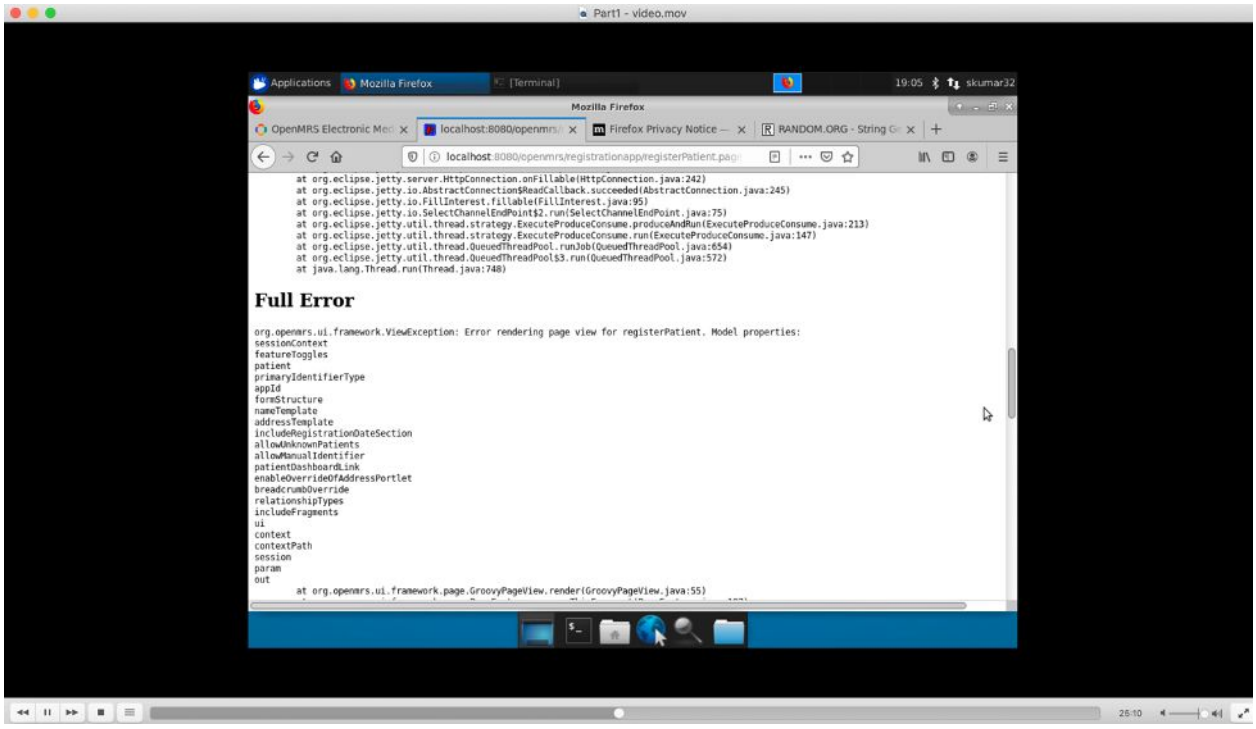
ASVS 14.3.1:
Verify that web or application server and framework error messages are configured to deliver user actionable, customized responses to eliminate any unintended security disclosures
CWE 209 : Information Exposure Through an Error Message
The software generates an error message that includes sensitive information about its environment, users, or associated data.

Repeatable Steps:

1. Navigate to - localhost:8080/openmrs/index.htm
2. Login with these credentials - Username: admin and Password - Admin123
3. Choose location as registration desk. Click on login.
4. Click on System Administration
5. Click on Advanced Administration
6. Click on Manage Users
7. Click on Add User
8. Under create a new person
9. Click on Next
10. Enter Given Name - test
    Family Name – test12
    Gender: Female
    UserName - test12
    User Password – Test1234
    Confirm Password - Test1234
    Roles: Organizational Clerk
11. Click on home and then click on logout
12. Login with credentials - Username: test12and password: Test1234
13. Choose registration desk as location and click on login.
14. Click on register Patient

Expected Results:
If there is any error, should display the error message.

Actual Results:
Displays the complete stack trace



Testcase	Unique ID	ASVS	CWE
3	4.1.3-3	4.1.3	285

ASVS:

4.1.3 Verify that the principle of least privilege exists - users should only be able to access functions, data files, URLs, controllers, services, and other resources, for which they possess specific authorization. This implies protection against spoofing and elevation of privilege.

CWE-285: Improper Authorization

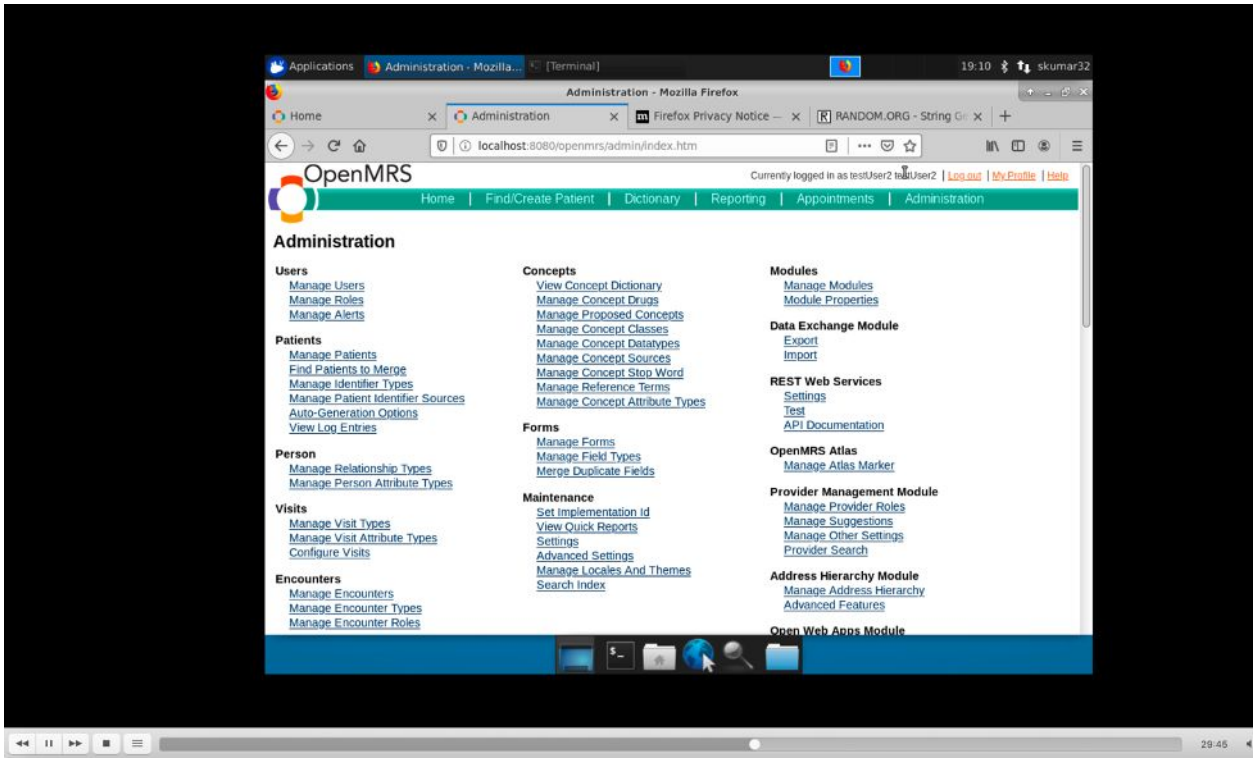
Repeatable Steps:

1. Navigate to - localhost:8080/openmrs/index.htm
2. Login with these credentials - Username: admin and Password - Admin123
3. Choose location as registration desk. Click on login.
4. Click on System Administration
5. Click on Advanced Administration
6. Click on Manage Users

7. Click on Add User
8. Under create a new person
9. Click on Next
10. Enter Given Name – testUser2
    Family Name – testUser2
    Gender: Female
    UserName - testUser2
    User Password – Test1234
    Confirm Password - Test1234
    Roles: Organizational Clerk
11. Click on home and then click on logout
12. Login with credentials - Username: testUser2 and password: Test1234
13. Choose registration desk as location and click on login.
14. Access this URL - http://localhost:8080/openmrs/admin/index.htm

Expected Results:
User with Clerk privileges should not be able to access admin page

Actual Results:
User with Clerk privileges is able to access admin page



Testcase	Unique ID	ASVS	CWE
4	**3.3.1-4**	**3.3.1**	**613**

ASVS:

3.3.1 Verify that logout and expiration invalidate the session token, such that the back button or a downstream relying party does not resume an authenticated session, including across relying parties. (C6)

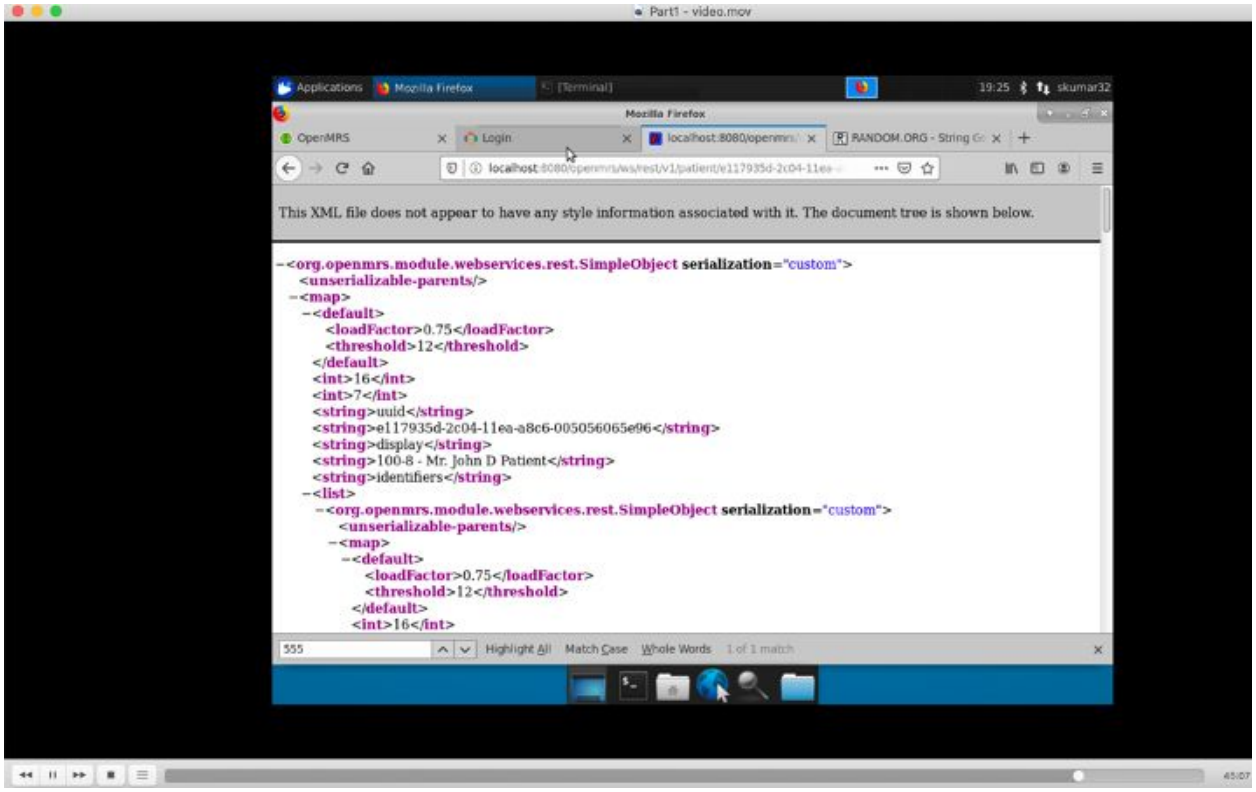
CWE-613: Insufficient Session Expiration

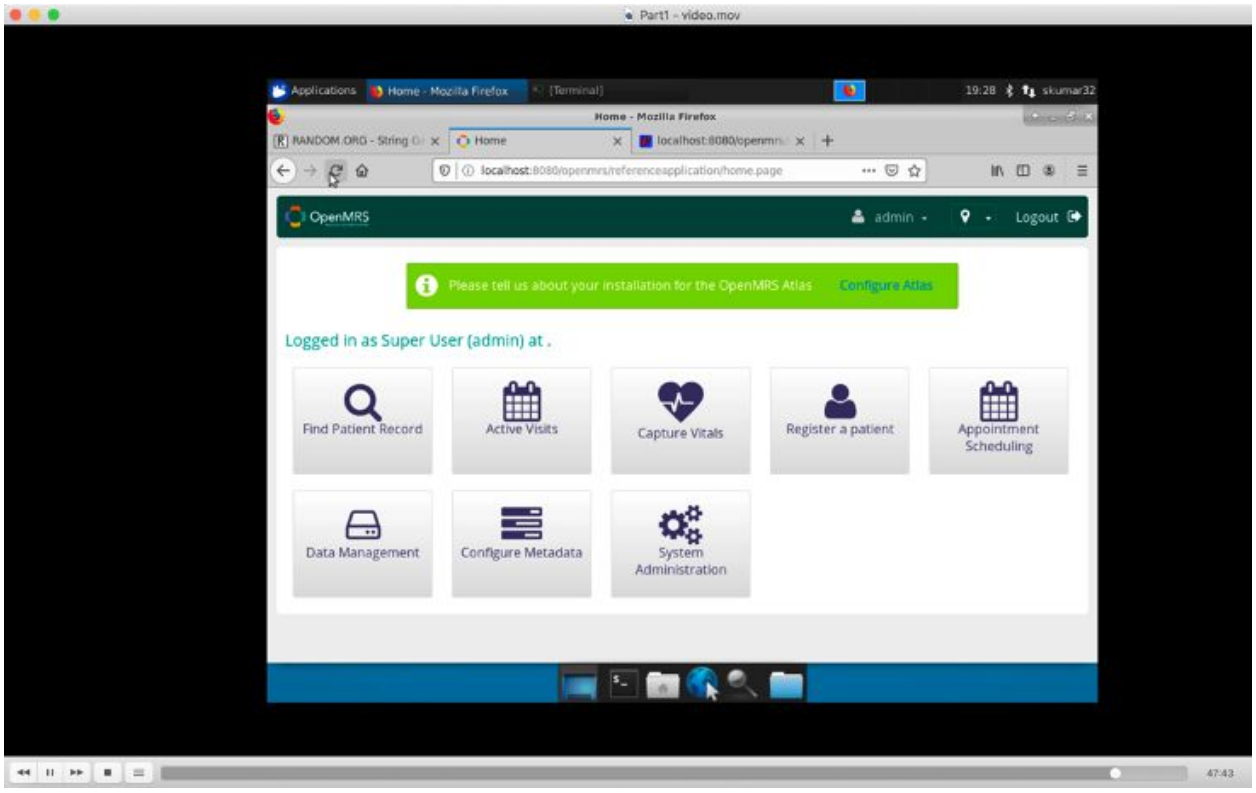
Repeatable Steps:

1. Navigate to - localhost:8080/openmrs/index.htm
2. Login with these credentials - Username: admin and Password - Admin123
3. Choose location as registration desk. Click on login.
4. Click on system administration > advanced administration > API documentation
5. Get the API to get patient details, which would be of the format: http://localhost:8080/openmrs/ws/rest/v1/patient/e117935d-2c04-11ea-a8c6-005056065e96
   [e117935d-2c04-11ea-a8c6-005056065e96 – patient ID for John D patient]
6. On a different tab access the above mentioned URL.
7. Navigate to the first tab, click on logout.
8. Now, refresh the API page. It will ask for credentials,enter these - Username: admin and Password - Admin123
9. Navigate back to the first, click on logout.
10. Refresh the page.

Expected Results:
User should be logged out – including the page displaying API results

Actual Results:
Once logged in from API page, even of the user clocks on login from application page, it doesn't log out.


Part1 - video.mov
Applications
Mozilla Firefox
[Terminal]
19:25
skumar32
Mozilla Firefox
OpenMRS
Login
RANDOM.ORG - String Ge
This XML file does not appear to have any style information associated with it. The document tree is shown below.
<org.openmrs.module.webservices.rest.SimpleObject serialization="custom">
<unserializable-parents/>
<map>
<default>
<loadFactor>0.75</loadFactor>
<threshold>12</threshold>
</default>
<int>16</int>
<int>7</int>
<string>uuid</string>
<string>e117935d-2c04-11ea-a8c6-005056065e96</string>
<string>display</string>
<string>100-8 - Mr. John D Patient</string>
<string>identifiers</string>
<list>
<org.openmrs.module.webservices.rest.SimpleObject serialization="custom">
<unserializable-parents/>
<map>
<default>
<loadFactor>0.75</loadFactor>
<threshold>12</threshold>
</default>
<int>16</int>
555
Highlight All
Match Case
Whole Words
1 of 1 match
45:07


Part1 - video.mov
Applications
Home - Mozilla Firefox
[Terminal]
19:28
skumar32
Home - Mozilla Firefox
RANDOM.ORG - String Ge
Home
localhost:8080/openmrs/referenceapplication/home.page
OpenMRS
admin
Logout
Please tell us about your installation for the OpenMRS Atlas
Configure Atlas
Logged in as Super User (admin) at .
Find Patient Record
Active Visits
Capture Vitals
Register a patient
Appointment Scheduling
Data Management
Configure Metadata
System Administration
47:43

Testcase	Unique ID	ASVS	CWE
5	**3.3.3-5**	**3.3.3**	**613**

ASVS

3.3.1 Verify that logout and expiration invalidate the session token, such that the back button or a downstream relying party does not resume an authenticated session, including across relying parties. (C6)

CWE-613: Insufficient Session Expiration

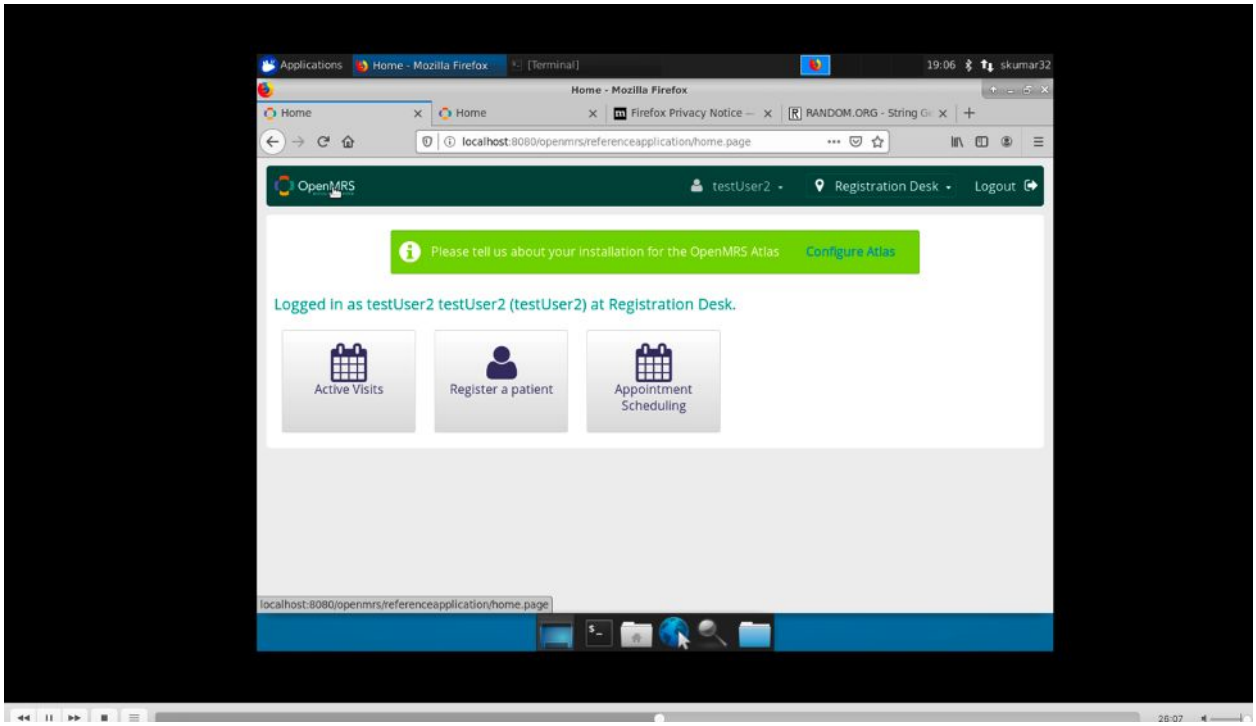
Repeatable Steps:

1. Navigate to - localhost:8080/openmrs/index.htm
2. Login with these credentials - Username: admin and Password - Admin123
3. Choose location as registration desk. Click on login.
4. Click on System Administration
5. Click on Advanced Administration
6. Click on Manage Users
7. Click on Add User
8. Under create a new person
9. Click on Next
10. Enter Given Name – testUser2
    Family Name – testUser2
    Gender: Female
    UserName - testUser2
    User Password – Test1234
    Confirm Password - Test1234
    Roles: Organizational Clerk
11. Click on home and then click on logout
12. Login with credentials - Username: testUser2 and password: Test1234
13. Choose registration desk as location and click on login.
14. From a different tab, open this URL - http://localhost:8080/openmrs/referenceapplication/home.page
15. From the first tab, click on testUser2 > my account > change password
    Old password: Test1234
    New password: Test12345
    Confirm password: Test12345

16.Click on save
17.Refresh the page on second tab

Expected Results:
It should ask the user to enter details again, since password was changes

Actual Results:
It doesn't ask the user to re-enter any details, the page is refreshed.



Testcase	Unique ID	ASVS	CWE
6	5.3.3-6	5.3.3	79

ASVS 5.3.3: Verify that context-aware, preferably automated - or at worst, manual – output escaping protects against reflected, stored, and DOM based XSS.
CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')

Repeatable Steps:

1. Navigate to - localhost:8080/openmrs/index.htm
2. Login with these credentials - Username: admin and Password - Admin123
3. Choose location as registration desk. Click on login.
4. Once logged in, click on ‘Register a patient’.
   Enter all the details:
   Given name: </script><script>alert(‘test’);</script>
   Family name: test
   Gender: Male
   BirthDate: Day - 01 Month - Jan Year -1995 Address: 123, abc st
   City:Raleigh
   State:NC
   Country:USA
   Postal Code:12345
   Phone Number: 123456789 Select Relationship type: Sibling Person name: Mukesh

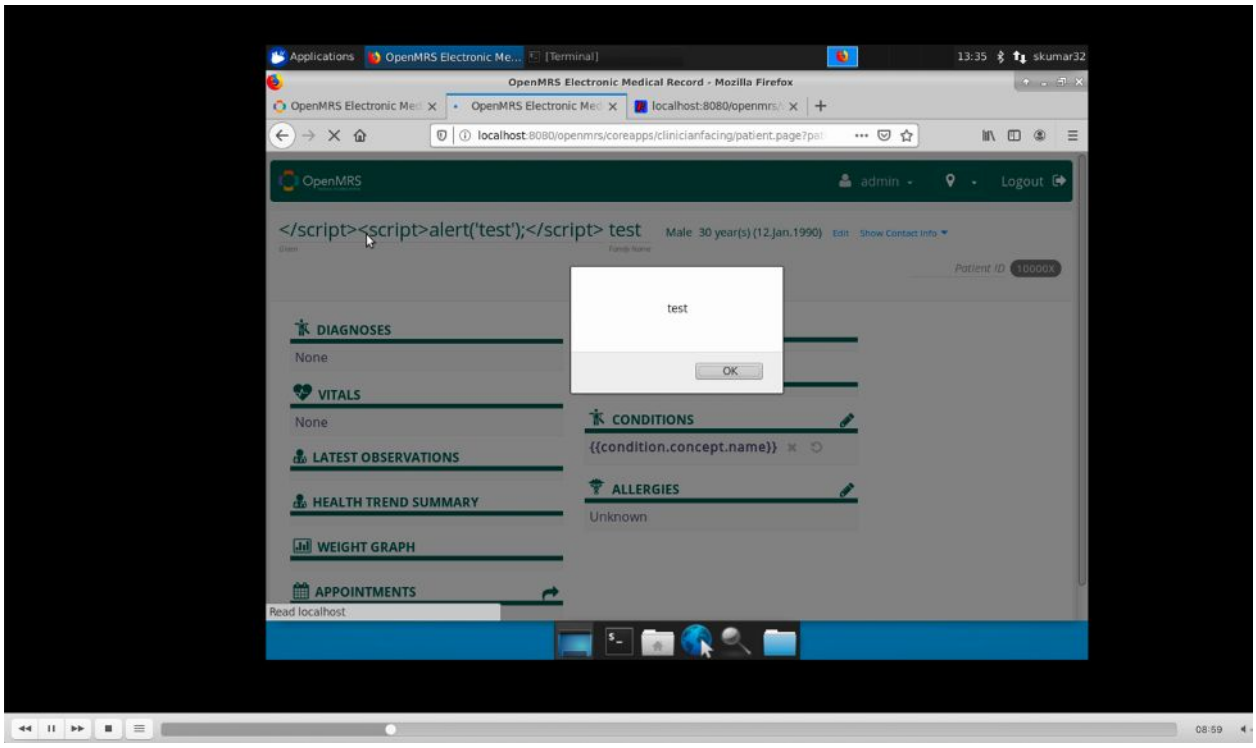
5. Click on confirm.

Expected Results:

Input should be sanitized and should execute the script tag and display the alert

Actual Results:

Executes the script tag and displays the alert



Testcase	Unique ID	ASVS	CWE
7	**5.3.3-7**	**5.3.3-7**	**79**

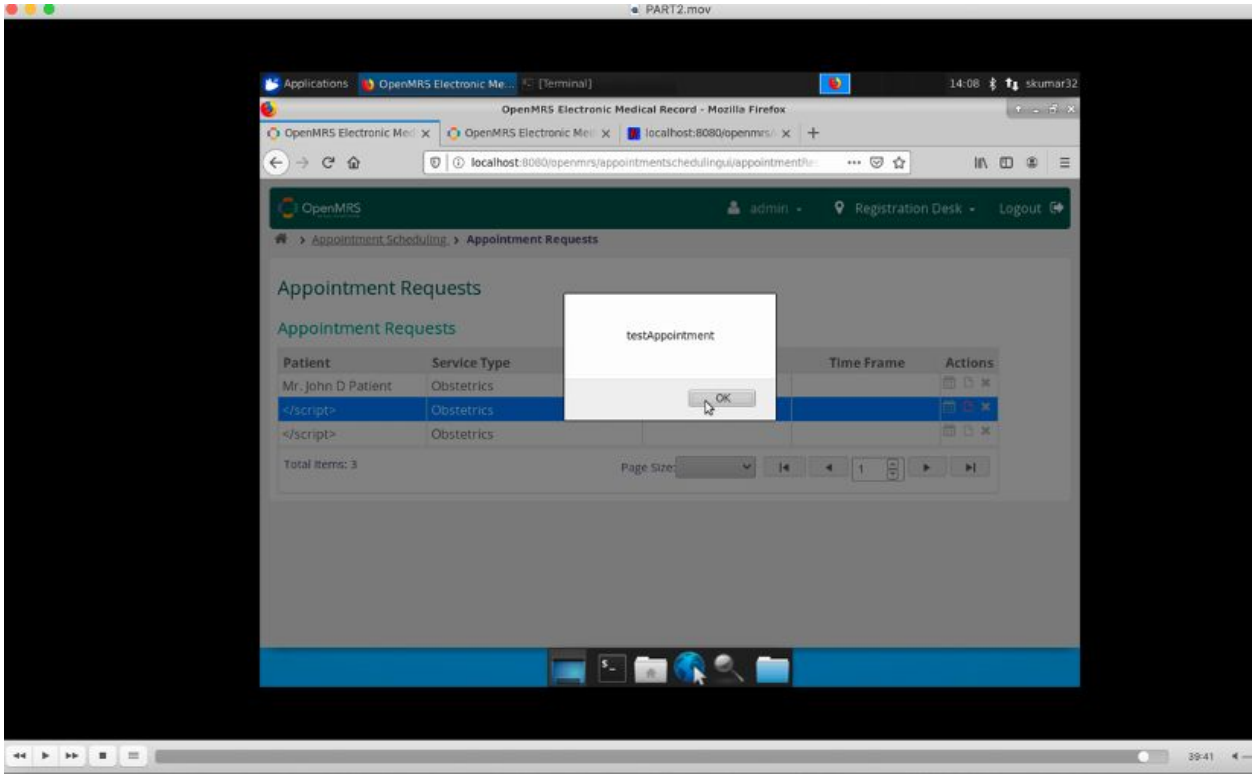
ASVS 5.3.3:
Verify that context-aware, preferably automated - or at worst, manual – output escaping protects against reflected, stored, and DOM based XSS.
CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')

Repeatable Steps:

1. Navigate to - localhost:8080/openmrs/index.htm
2. Login with these credentials - Username: admin and Password - Admin123
3. Choose location as registration desk. Click on login.
4. Click on 'Find Patient record'
5. Click on John D Patient
6. From 'General Actions' tab, click on request appointment.
7. Enter Appointment Type -Obstetrics & Notes - </script><script>alert('testAppointment');</script>
8. Click on save
9. Navigate to localhost:8080/openmrs/referenceapplication/home.page
10. Click on Appointment scheduling > Appointment requests.
11. Click on Notes button for John D Patient

Expected Results:
Input should be sanitized and should execute the script tag and display the alert

Actual Results:
Executes the script tag and displays the alert



Testcase	Unique ID	ASVS	CWE
8	**5.2.5-8**	**5.3.3**	**94**

5.2.5 Verify that the application protects against template injection attacks by ensuring that any user input being included is sanitized or sandboxed.

CWE-94: Improper Control of Generation of Code

Repeatable Steps:

1. Navigate to - localhost:8080/openmrs/index.htm
2. Login with these credentials - Username: admin and Password - Admin123
3. Choose location as registration desk. Click on login.
4. Once logged in, click on ‘Register a patient’.
   Enter all the details:
   Given name: </script><script>alert(‘test’);</script>
   Family name: </script><script>alert(‘test’);</script>

Gender: Male
BirthDate: Day - 01 Month - Jan Year -1995 Address: 123, abc st
City:Raleigh
State:NC
Country:USA
Postal Code:12345
Phone Number: 123456789 Select Relationship type: Sibling Person name: Mukesh

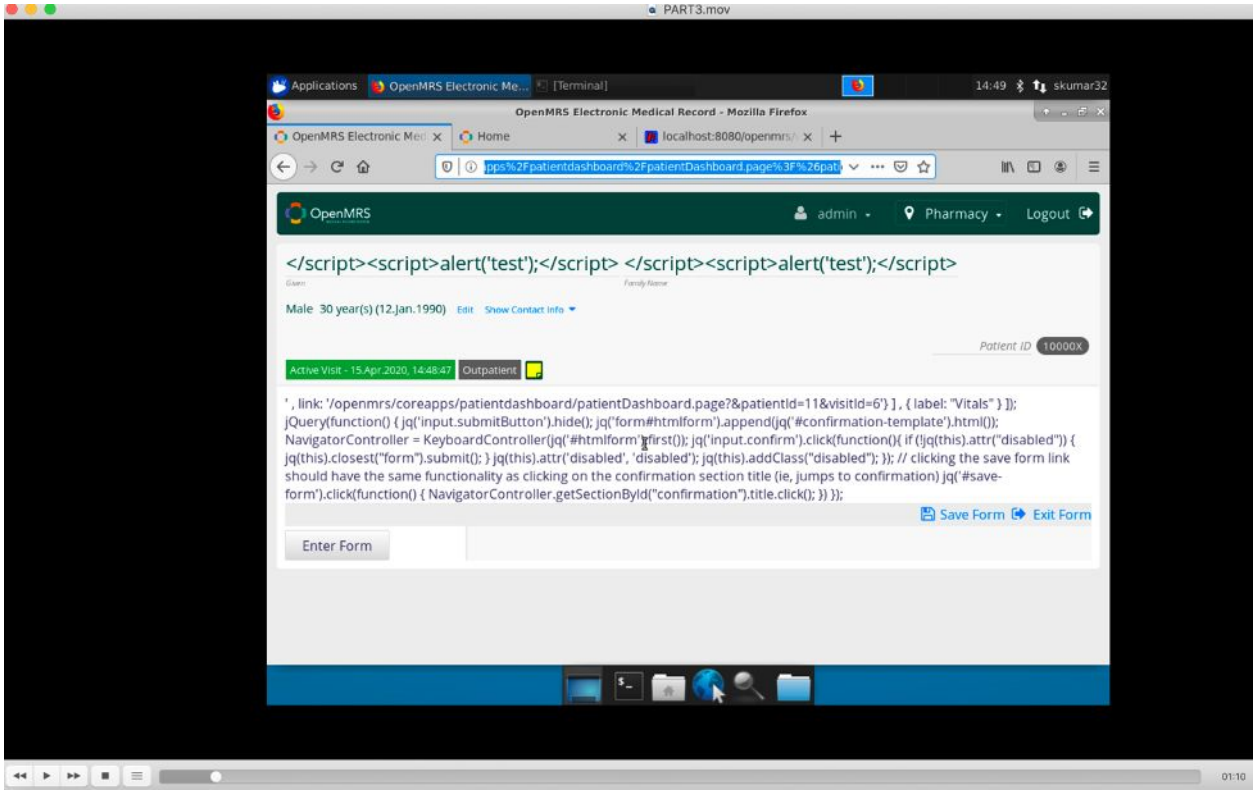
5. Click on confirm.
6. From ‘General Actions’ tab, click on Start visit > Choose outpatient > confirm
7. Click on capture vitals

Expected Results:

It shouldn’t consider the script tag as input and should be able to able handle erroneous input

Actual Results:

Displays code on the UI, because input was not handled correctly, it eas vulnerable to XSS attack.



Testcase	Unique ID	ASVS	CWE
9	5.3.3-9	5.3.3	79

ASVS 5.3.3
Verify that context-aware, preferably automated - or at worst, manual – output escaping protects against reflected, stored, and DOM based XSS.
CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')

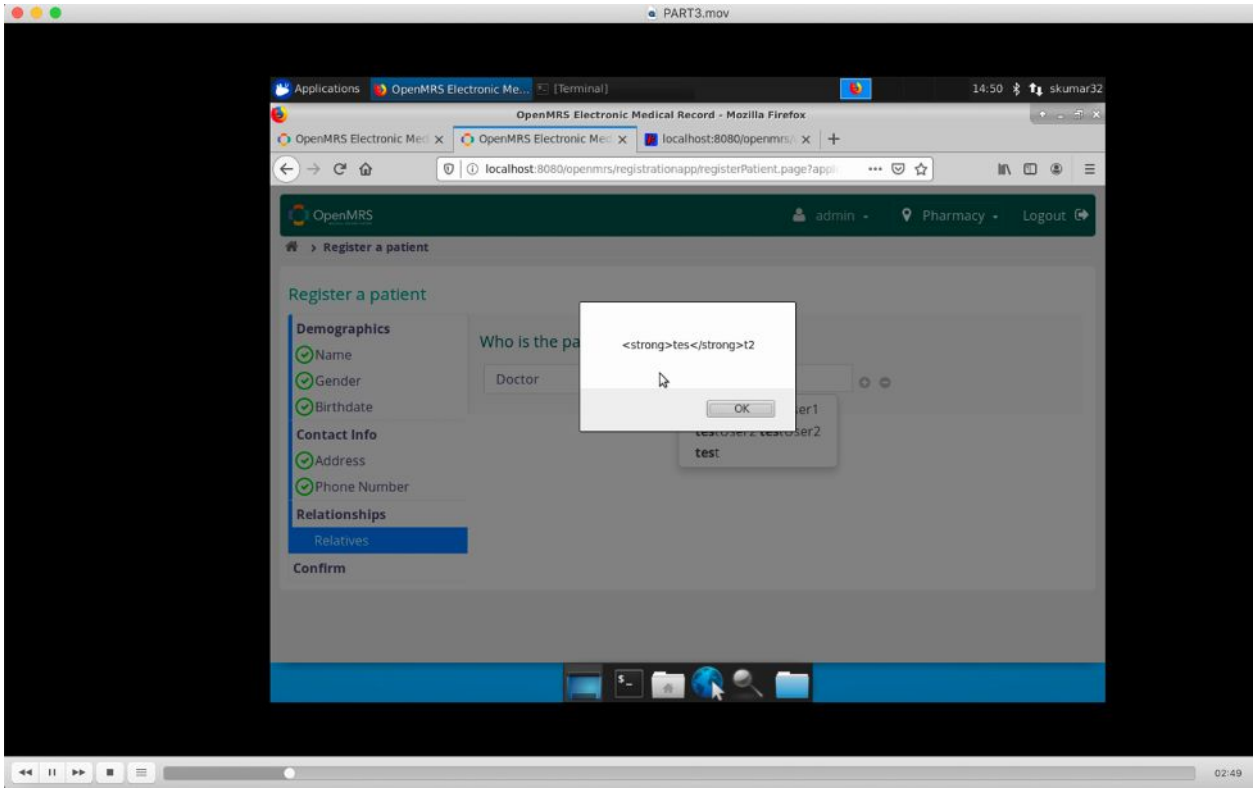
Repeatable Steps:

1. Navigate to - localhost:8080/openmrs/index.htm
2. Login with these credentials - Username: admin and Password - Admin123
3. Choose location as registration desk. Click on login.
4. Click on ‘Register a patient’.
   Enter all the details:
   Given name: </script><script>alert(‘test’);</script>
   Family name: test
   Gender: Male
   BirthDate: Day - 01 Month - Jan Year -1995 Address: 123, abc st
   City:Raleigh
   State:NC
   Country:USA
   Postal Code:12345
   Phone Number: 123456789
   Select Relationship type: Doctor; Person Name :test
5. Click on Confirm
6. Navigate back to the home page, by clicking on OpenMRS logo on left corner
7. Click on ‘Register a patient’.
   Enter all the details:
   Given name: testPatient
   Family name: testPatient
   Gender: Male
   BirthDate: Day - 01 Month - Jan Year -1995 Address: 123, abc st
   City:Raleigh
   State:NC
   Country:USA

Postal Code:12345
Phone Number: 123456789 Select Relationship type: Doctor
Click on Person name

Expected Results:
It should not execute script tag, input should be sanitized.

Actual Results:
Since there was a patient saves with script, it executes that while choosing the person name



Testcase	Unique ID	ASVS	CWE
10	5.2.5-10	5.2.5	94

5.2.5 Verify that the application protects against template injection attacks by ensuring that any user input being included is sanitized or sandboxed.

CWE-94: Improper Control of Generation of Code

Repeatable Steps:

1. Navigate to - localhost:8080/openmrs/index.htm
2. Login with these credentials - Username: admin and Password - Admin123
3. Choose location as registration desk. Click on login.
4. Click on ‘Find Patient Record’
5. Click on ‘John D Patient’
6. From ‘General Actions’ tab, click on Start visit > Choose outpatient > confirm
7. Click on Visit notes.
8. Enter the following details:

   Provider: Super User

   Location: Registration desk

   Date:04/16/2020

   Add presumed or confirmed diagnosis (required): </script><script>alert(‘test’);</script>

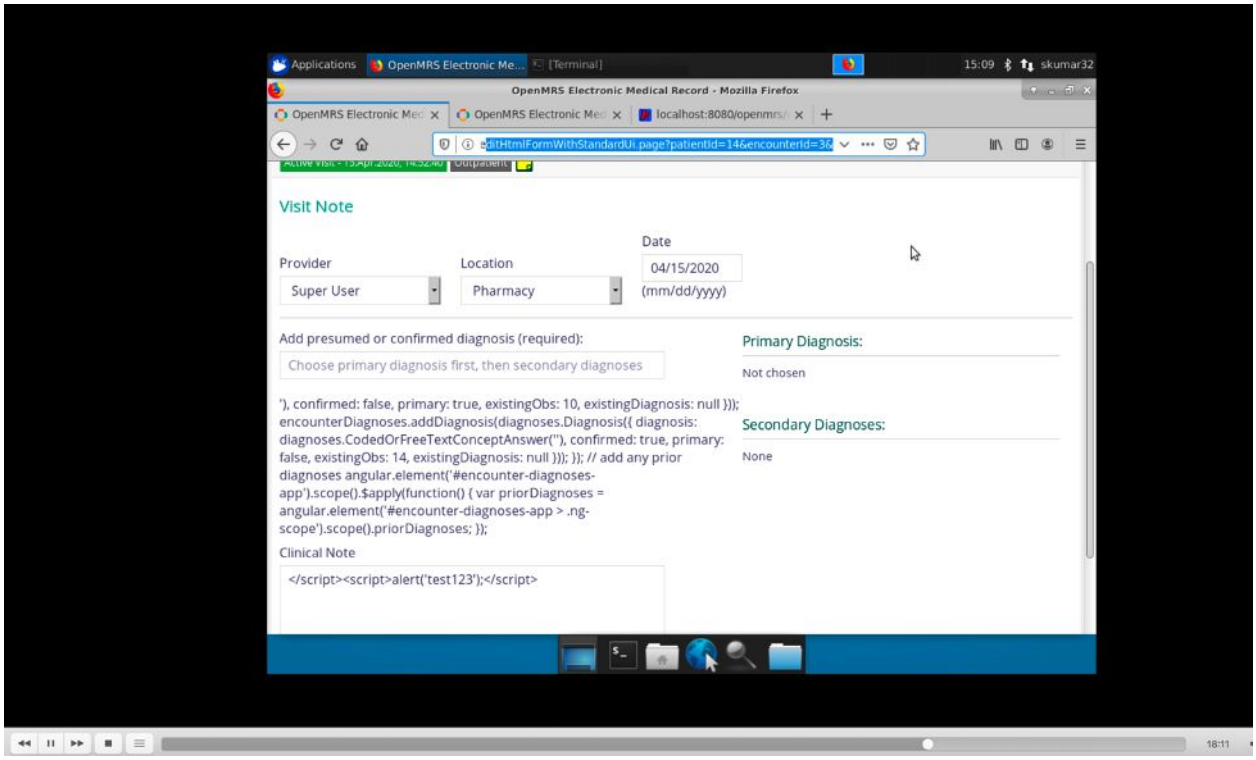
   Clinical Notes: </script><script>alert(‘test’);</script>
9. Click on save
10. Click on the edit button next to show details

Expected Results:

It should not accept input given by end user as code, it should be considered as any string data or should have some client-side validation.

Actual Results:

Input is considered as code and is not validated, thus displays part of the application code and doesn’t handle erroneous input.



Testcase	Unique ID	ASVS	CWE
11	**7.4.2-11**	**7.4.2**	**544**

ASVS:

7.4.2 Verify that exception handling (or a functional equivalent) is used across the codebase to account for expected and unexpected error conditions.

CWE-544: Missing Standardized Error Handling Mechanism

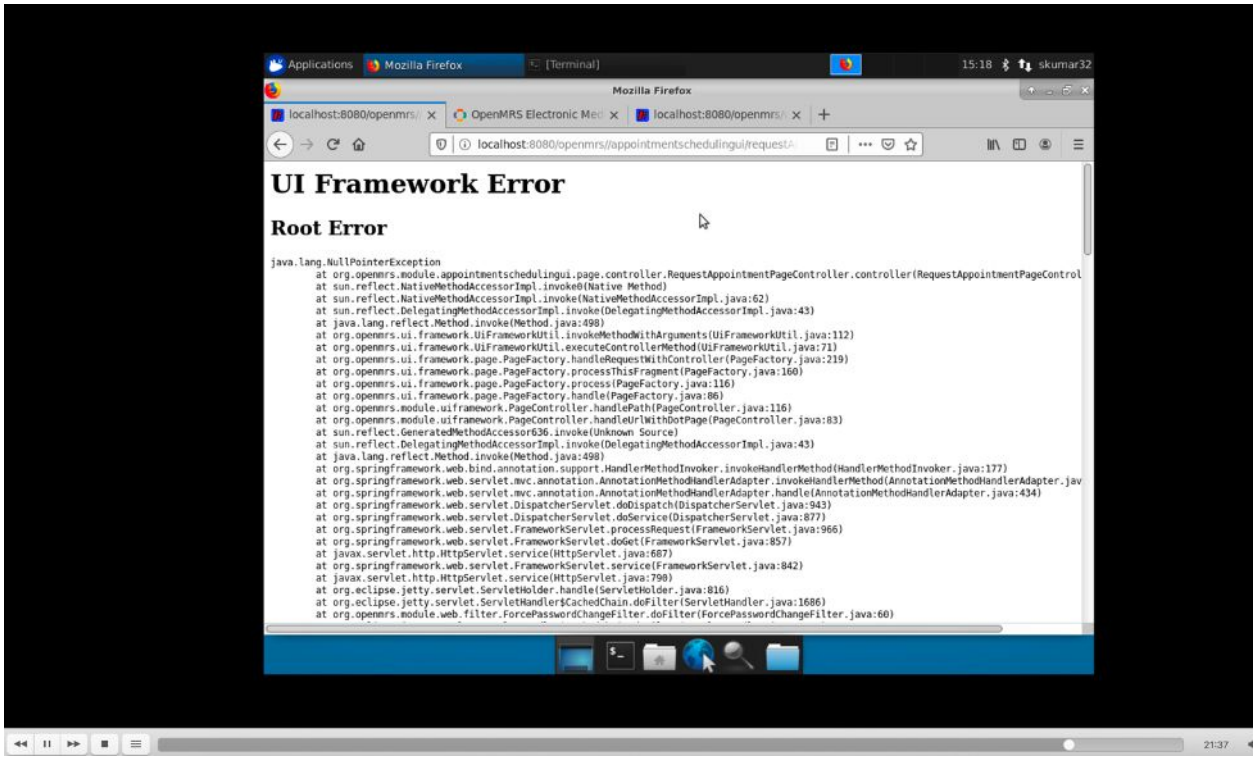
Repeatable Steps:

1. Navigate to - localhost:8080/openmrs/index.htm
2. Login with these credentials - Username: admin and Password - Admin123
3. Choose location as registration desk. Click on login.
4. Click on ‘Find Patient Record’
5. Click on ‘John D Patient’
6. From ‘General Actions’ tab, click on request appointment.
7. Update the URL in the browser tab as follows:

http://localhost:8080/openmrs//appointmentschedulingui/requestAppointment.page?patientId=</script><script>alert('test');</script>&returnUrl=%2Fopenmrs%2Fcoreapps%2Fclinicianfacing%2Fpatient.page%3FpatientId%3De117935d-2c04-11ea-a8c6-005056065e96%26

Expected Results:
If there is any error, should display the error message.

Actual Results:
Displays the complete stack trace



Testcase	Unique ID	ASVS	CWE
12	5.3.3	5.3.3-12	79

ASVS 5.3.3:

Verify that context-aware, preferably automated - or at worst, manual – output escaping protects against reflected, stored, and DOM based XSS.

CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')

Repeatable Steps:

1. Navigate to - localhost:8080/openmrs/index.htm
2. Login with these credentials - Username: admin and Password - Admin123
3. Choose location as registration desk. Click on login.
4. Click on ‘Find Patient Record’
5. Click on ‘John D Patient’
6. From ‘General Actions’ tab, click on request appointment.
7. Update the URL in the browser tab as follows:

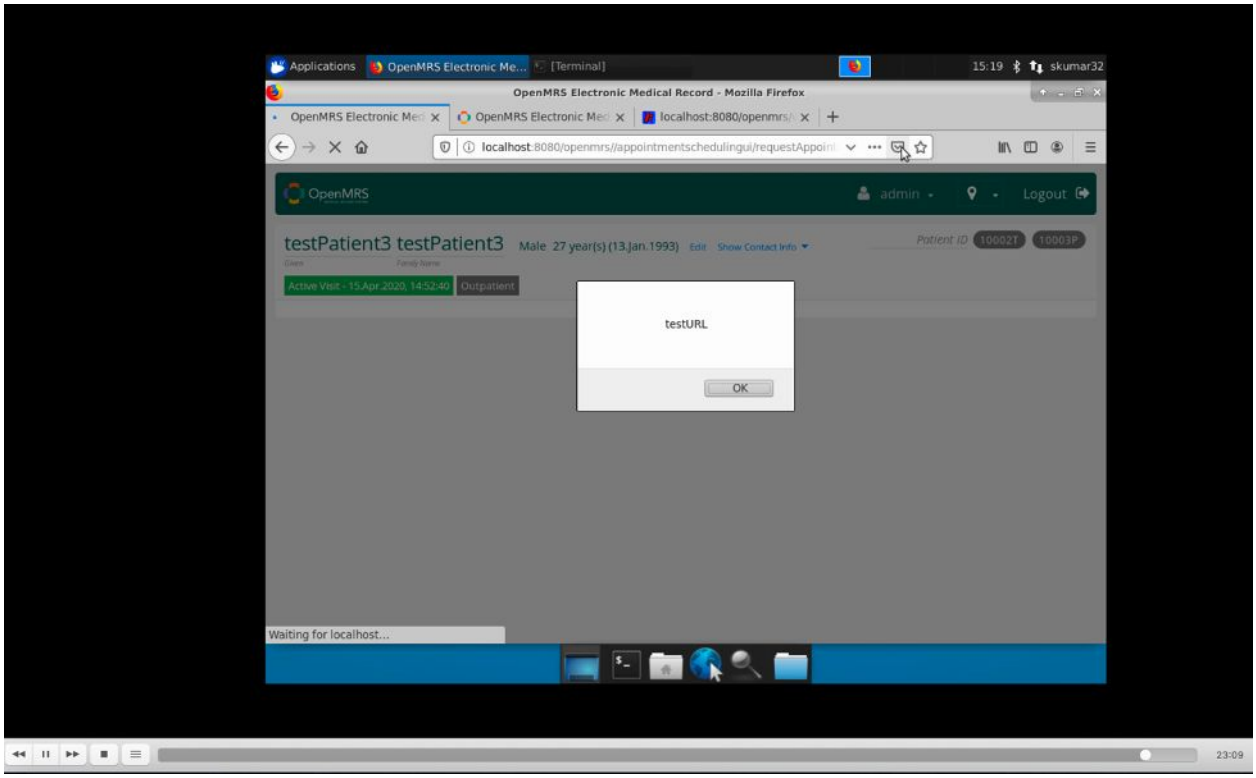
http://localhost:8080/openmrs//appointmentschedulingui/requestAppointment.page?patientId=e117935d-2c04-11ea-a8c6-005056065e96&returnUrl=”</script><script>alert(‘testURL’);</script>”

Expected Results:

It should not accept input given by end user as code, it should be considers as any string data or should have some client side validation.

Actual Results:

Input is considered as code and is not validated, thus displays part of the application code.



Testcase	Unique ID	ASVS	CWE
13	7.4.2	7.4.2-11	79

7.4.2 Verify that exception handling (or a functional equivalent) is used across the codebase to account for expected and unexpected error conditions.

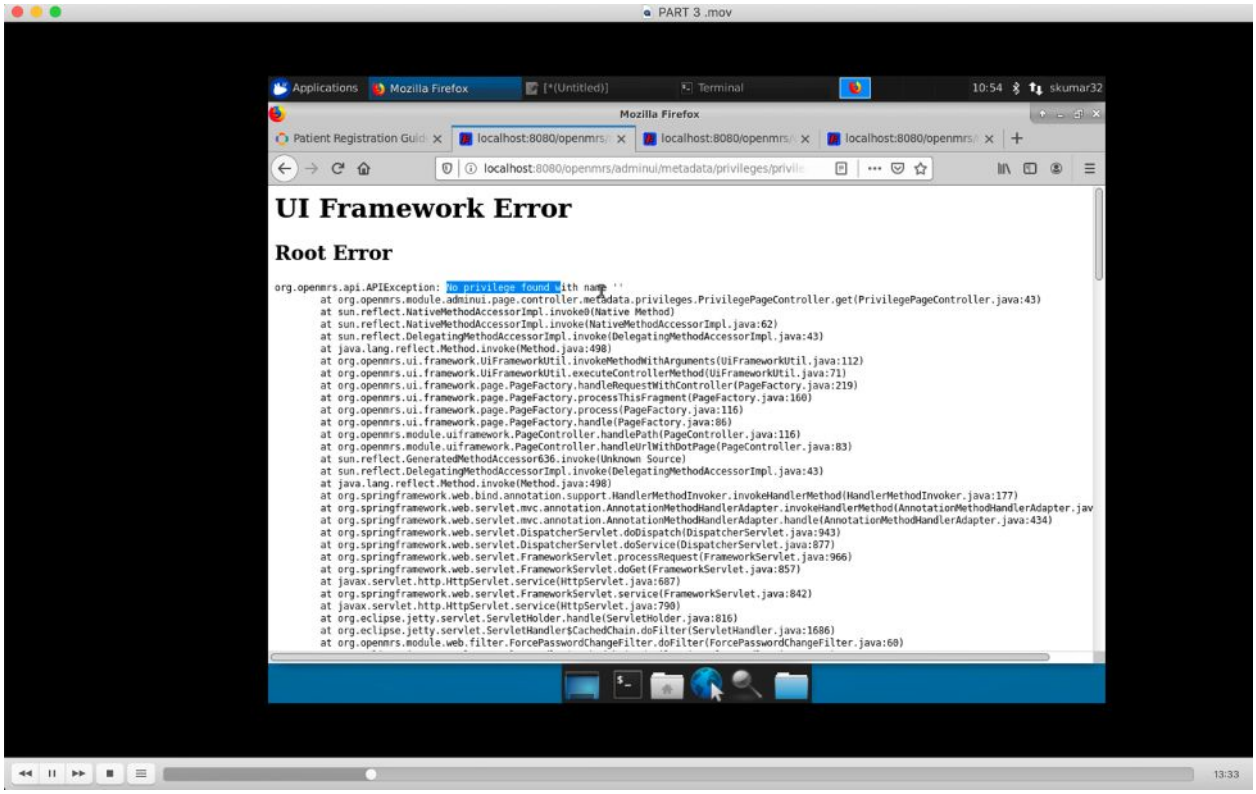
CWE-544: Missing Standardized Error Handling Mechanism

Repeatable Steps:

1. Navigate to - localhost:8080/openmrs/index.htm
2. Login with these credentials - Username: admin and Password - Admin123
3. Choose location as registration desk. Click on login.
4. Click on ‘Configure Metadata > manage privileges > Add new privilege’
5. Enter Name: </script><script>alert(‘testPrev’);</script> & Description: </script><script>alert(‘testPrevDesc’);</script>
6. Click on Save
7. Click on edit next to </script><script>alert(‘testPrevDesc’);</script>

Expected Results:
If there is any error, should display the error message.

Actual Results:
Displays the complete stack trace



Testcase	Unique ID	ASVS	CWE
14	5.3.3-14	5.3.3	79

ASVS 5.3.3
Verify that context-aware, preferably automated - or at worst, manual – output escaping protects against reflected, stored, and DOM based XSS.
CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')

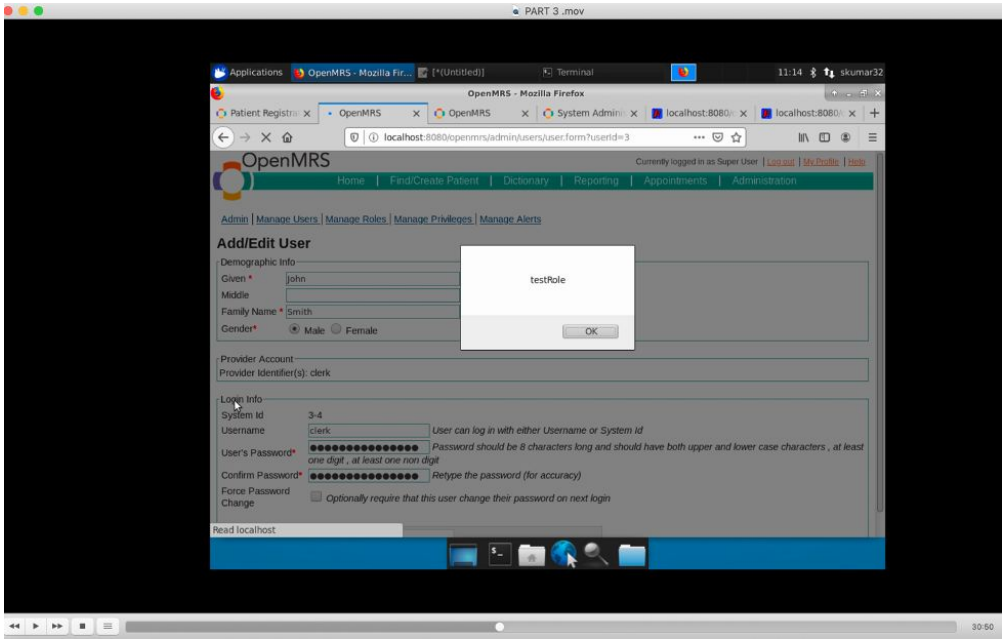
Repeatable Steps:

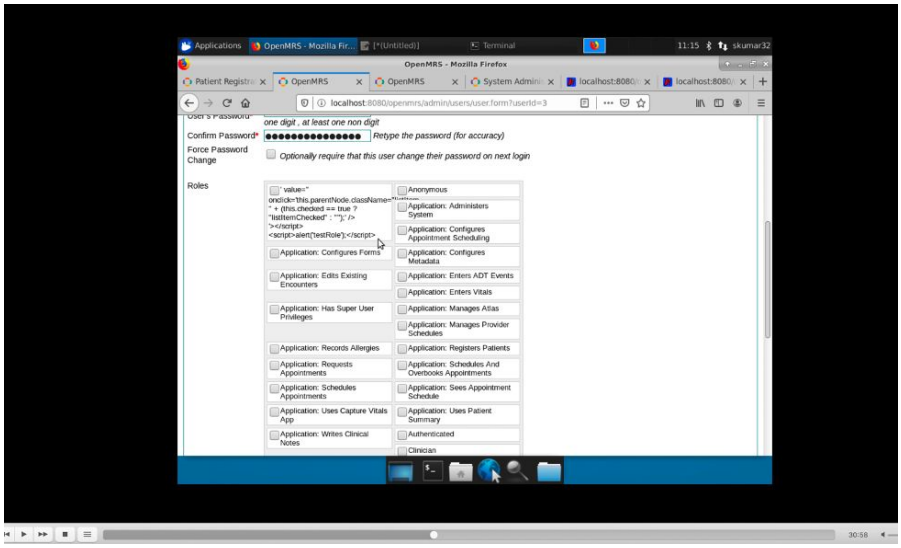
8. Navigate to - localhost:8080/openmrs/index.htm

9. Login with these credentials - Username: admin and Password - Admin123
10. Choose location as registration desk. Click on login.
11. Click on ‘Configure Metadata > manage privileges > Add new privilege’
12. Enter Name: </script><script>alert(‘testPrev’);</script> & Description: </script><script>alert(‘testPrevDesc’);</script>
13. Click on Save
14. Navigate back to homepage by clicking on OpenMRS logo.
15. Click on ‘Configure Metadata > manage roles > Add new role
16. Enter Role: </script><script>alert(‘testRole’);</script>
    Description: </script><script>alert(‘testRoleDesc’);</script>
    Privilege: </script><script>alert(‘testPrev’);</script>
10. Click on Save.
11. Navigate back to homepage by clicking on OpenMRS logo
12. Click on System Administration > Advanced Administration > manage Users
13. Find user on Name: enter John. Click on the row displayed.

Expected Results:
Input should be sanitized and should execute the script tag and display the alert

Actual Results:
Executes the script tag and displays the alert





Testcase	Unique ID	ASVS	CWE
15	4.1.3-14	4.1.3	285

ASVS:

4.1.3 Verify that the principle of least privilege exists - users should only be able to access functions, data files, URLs, controllers, services, and other resources, for which they possess specific authorization. This implies protection against spoofing and elevation of privilege.

CWE-285: Improper Authorization

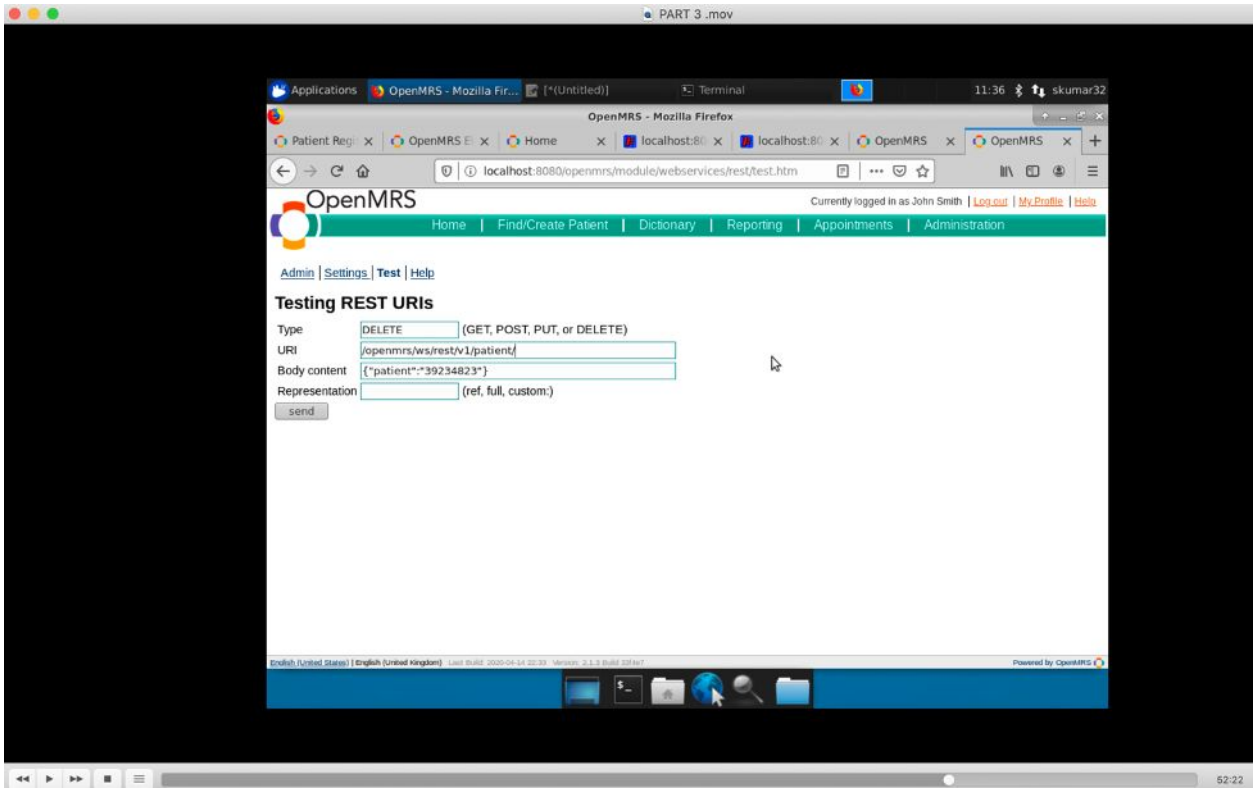
Repeatable steps:

1. Navigate to - localhost:8080/openmrs/index.htm
2. Login with these credentials - Username: admin and Password - Admin123
3. Choose location as registration desk. Click on System Administration > Advanced Administration > manage Users
4. Find user on Name: enter John. Click on the row displayed.
5. Change password to Clerk123 and click on save
6. Click on logout.
7. Login with these credentials - clerk: admin and Password - Clerk123
    1. Access the page - http://localhost:8080/openmrs/admin/index.htm
8. Under API setting, click on Test and change Type to DELET
   URI – enter a valid patient ID in the end - e117935d-2c04-11ea-a8c6-005056065e96

9. Click on send

Expected Results:
It should display an error saying the user is not authorized to perform this action.

Actual Results:
Deletes the patient record.



Testcase	Unique ID	ASVS	CWE
16	5.3.3-16	5.3.3	79

ASVS 5.3.3
Verify that context-aware, preferably automated - or at worst, manual – output escaping protects against reflected, stored, and DOM based XSS.
CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')

Repeatable Steps:

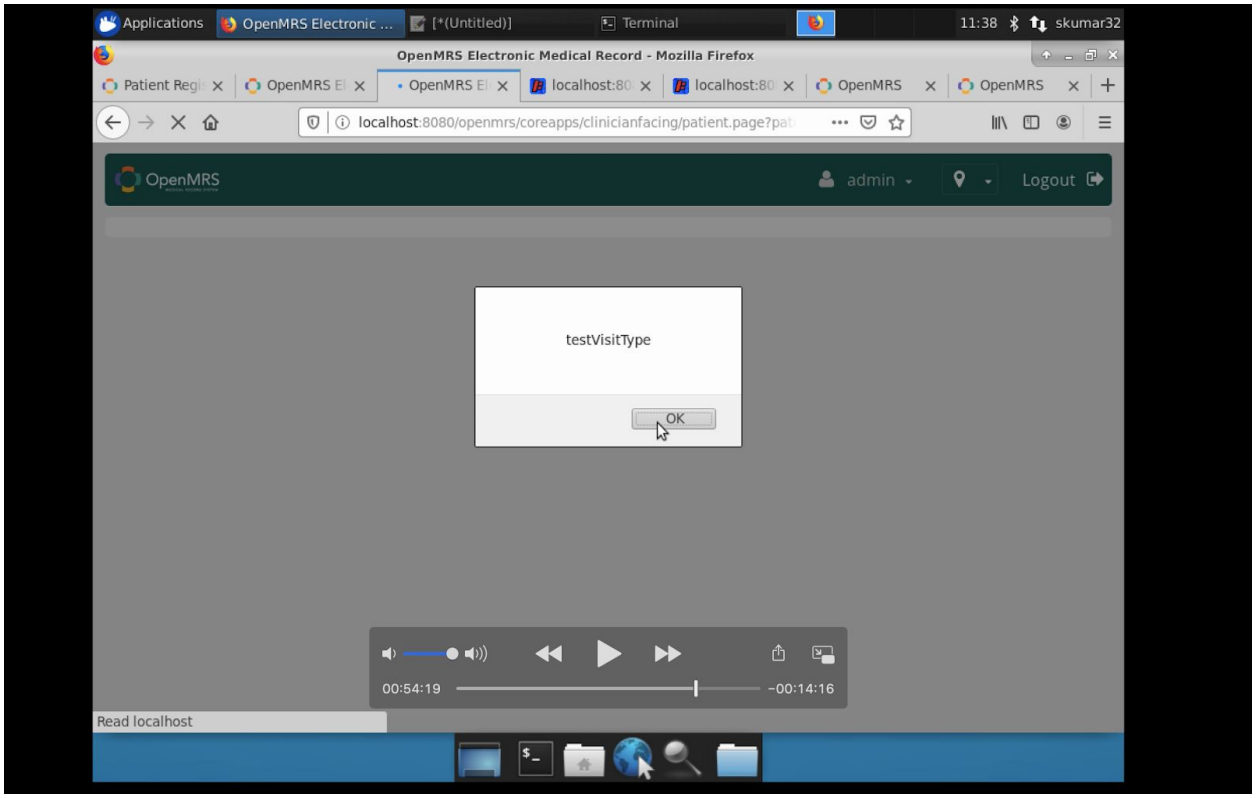
1. Navigate to - localhost:8080/openmrs/index.htm
2. Login with these credentials - Username: admin and Password - Admin123
3. Choose location as registration desk. Click on login.
4. Click on 'Configure Metadata > manage Visit Attribute Type'
5. Enter Name: </script><script>alert('testPrev');</script> & Description: </script><script>alert('testPrevDesc');</script>
6. Click on Save
7. Navigate back to homepage by clicking on OpenMRS logo.
8. Click on 'Configure Metadata > manage roles > Add new role
9. Enter Role: </script><script>alert('testVist');</script>
   Description: </script><script>alert('testVisitDesc');</script>
   DataType: float data type
10. Click on 'Configure Metadata > manage Visit Type
    Enter Role: </script><script>alert('testVistType');</script>
    Description: </script><script>alert('testVisitDesc');</script>
11. Click on Save.
12. Navigate back to the home page, by clicking on OpenMRS logo on left corner
13. Click on 'Register a patient'.
14. Enter all the details:
    Given name: testPatient5
    Family name: testPatient5
    Gender: Male
    BirthDate: Day - 01 Month - Jan Year -1995 Address: 123, abc st
    City:Raleigh
    State:NC
    Country:USA
    Postal Code:12345
    Phone Number: 123456789
    Select Relationship type: Doctor; Person Name :test
15. Click on Save

Expected Results:

Input should be sanitized and should execute the script tag and display the alert

Actual Results:

Executes the script tag and displays the alert



Video link 2:

https://drive.google.com/file/d/1JyDP-EUuEDgeMiRwwDrqFuX88-0qAp30/view?usp=sharing

Vulnerability #	Elapsed Time	Ref Info for Video Traceability	CWE	Commentary
1.	32 min	Video Link 2	20	User input not sanitized.
2.	34 min	Video Link 2	20	User input not sanitized.
3.	36 min	Video Link 2	20	Abnormal behavior observed on webpage.
4.	55 min	Video Link 2	601	Redirection to any URL.
5.	1 hr 5 min	Video Link 2	20	User input not sanitized.

6.	**2 hr 17 min**	**Video Link 2**	**20**	Unknown information displayed.
7.	**2 hr 20 min**	**Video Link 2**	**285**	Unauthorized access to the webpage.
8.	**2 hr 28 min**	**Video Link 2**	**285**	Unauthorized access to the webpage.
9.	**2 hr 44 min**	**Video Link 2**	**319**	Sensitive information sent in browser
10.	**3 hr**	**Video Link 2**	**20**	Stored XSS.

Vulnerability 1

Test Case	ASVS	Unique ID	CWE
1	**5.1.3**	**5.1.3-1**	**20**

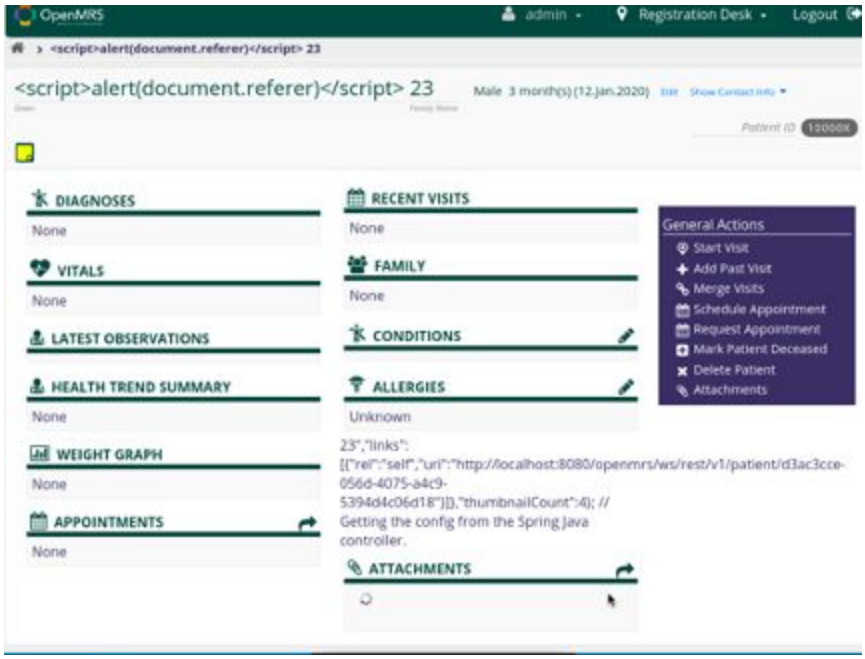
ASVS 5.1.3
Verify that all input (HTML form fields, REST requests, URL parameters, HTTP headers, cookies, batch files, RSS feeds, etc.) is validated using positive validation (whitelisting).

CWE 20
Improper Input Validation

Repeatable steps:

1) Navigate to http://localhost:8080/openmrs/.
2) Enter the username, “admin” and password as “Admin123”.
3) Select the location as Registration Desk.
4) Click on Login.
5) Click on Register a Patient.
6) Enter the given name as “Ram” and Family name as “Sharma”.
7) Click on Gender. Select Male.
8) Click on Birthdate. Enter the day as 12, and month as January and year 2020.
9) Click on Address. Enter the address as “1”.
10) Click on Phone Number and enter 1.
11) Click on Confirm.
12) Press on the Confirm button.

13) Click on the Edit option next to the name “Ram”.
14) Update the name to “<script>document.referer</script>”.
15) Click on Save Form. Click on Confirm button.



Actual Result: Input was not sanitized.

Expected Result: Input should have been sanitized.

Result: Fail

Vulnerability 2

Test Case	ASVS	Unique ID	CWE
2	5.1.3	5.1.3-1	20

ASVS 5.1.3
Verify that all input (HTML form fields, REST requests, URL parameters, HTTP headers, cookies, batch files, RSS feeds, etc.) is validated using positive validation (whitelisting).

CWE 20
Improper Input Validation

Repeatable steps:

1) Navigate to http://localhost:8080/openmrs/.
2) Enter the username, “admin” and password as “Admin123”.
3) Select the location as Registration Desk.
4) Click on Login.
5) Navigate to http://localhost:8080/openmrs/.
6) Enter the username, “admin” and password as “Admin123”.
7) Select the location as Registration Desk.
8) Click on Login.
9) Click on Register a Patient.
10) Enter the given name as “Ram” and Family name as “Sharma”.
11) Click on Gender. Select Male.
12) Click on Birthdate. Enter the day as 12, and month as January and year 2020.
13) Click on Address. Enter the address as “1”.
14) Click on Phone Number and enter 1.
15) Click on Confirm.
16) Press on the Confirm button.
17) Click on the Edit option next to the name “Ram”.
18) Update the name to “<script>document.referer</script>”.
19) Click on Save Form. Click on Confirm button.
20) Click on Attachments.

Actual Result: Input was not sanitized.

Expected Result: Input should have been sanitized.

Result: Fail

Vulnerability 3

Test Case	ASVS	Unique ID	CWE
3	**5.1.3**	**5.1.3-1**	**20**

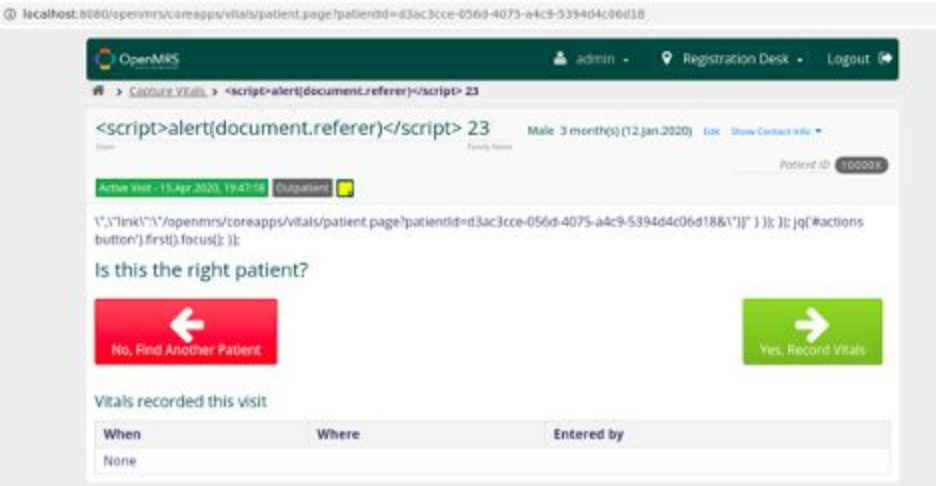
ASVS 5.1.3
Verify that all input (HTML form fields, REST requests, URL parameters, HTTP headers, cookies, batch files, RSS feeds, etc.) is validated using positive validation (whitelisting).

CWE 20
Improper Input Validation

Repeatable steps:

1) Navigate to http://localhost:8080/openmrs/.
2) Enter the username, “admin” and password as “Admin123”.
3) Select the location as Registration Desk.
4) Click on Login.
5) Navigate to http://localhost:8080/openmrs/.
6) Enter the username, “admin” and password as “Admin123”.
7) Select the location as Registration Desk.
8) Click on Login.
9) Click on Register a Patient.
10) Enter the given name as “Ram” and Family name as “Sharma”.
11) Click on Gender. Select Male.
12) Click on Birthdate. Enter the day as 12, and month as January and year 2020.
13) Click on Address. Enter the address as “1”.
14) Click on Phone Number and enter 1.
15) Click on Confirm.
16) Press on the Confirm button.
17) Click on the Edit option next to the name “Ram”.
18) Update the name to “<script>document.referer</script>”.
19) Click on Save Form. Click on Confirm button.
20) Go back to the home page by navigating to http://localhost:8080/openmrs/.

21) Click on Capture Vitals.
22) Select the patient with name “<script>document.referer</script>”.



Actual Result: Input was not sanitized.

Expected Result: Input should have been sanitized.

Result: Fail

Vulnerability 4

Test Case	ASVS	Unique ID	CWE
4	5.1.5	5.1.5-1	601

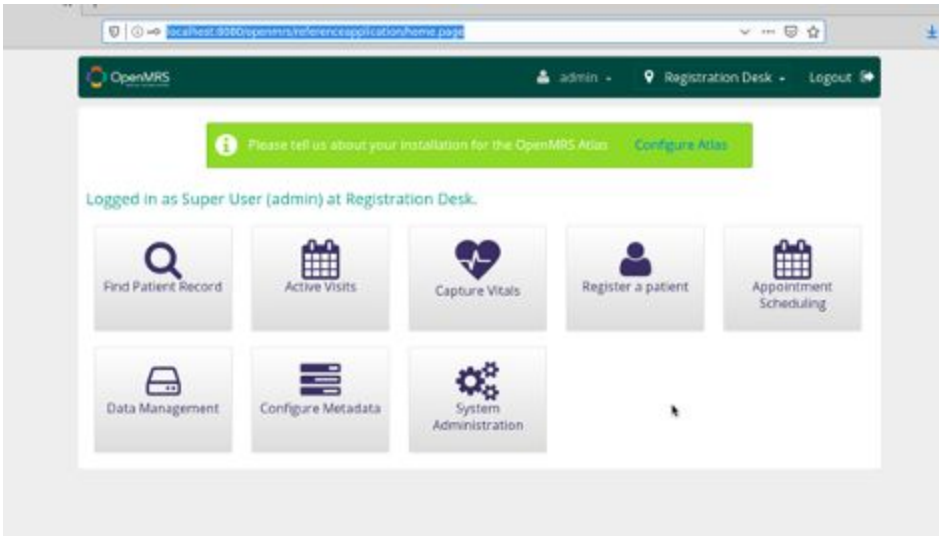
ASVS 5.1.5
Verify that URL redirects and forwards only allow whitelisted destinations, or show a warning when redirecting to potentially untrusted content.

CWE 601
URL Redirection to Untrusted Site ('Open Redirect').

Repeatable steps:

1) Navigate to http://localhost:8080/openmrs/.

2) Enter the username, “admin” and password as “Admin123”.
3) Select the location as Registration Desk.
4) Click on Login.
5) Click on Find Patient Record.
6) Click on the existing name, John D Patient.
7) Click on the edit option next to the patient birth date.
8) Update the URL to http://localhost:8080/openmrs/registrationapp/editSection.page?patientId=3&sectionId=demographics&appId=referenceapplication.registrationapp.registerPatient&returnUrl=http://localhost:8080/openmrs/referenceapplication/home.page and press enter.
9) Click on the Save Form button. Press Confirm.
10) A red popup is visible stating that it Failed to Save Changes.
11) Click on the name, John D Patient on the left hand side of the Demographics tag.
12) The user is redirected to http://localhost:8080/openmrs/referenceapplication/home.page.



Actual Result: The user can be redirected to any location.

Expected Result: Redirection only to whitelisted destinations should be allowed.

Result: Fail

Vulnerability 5

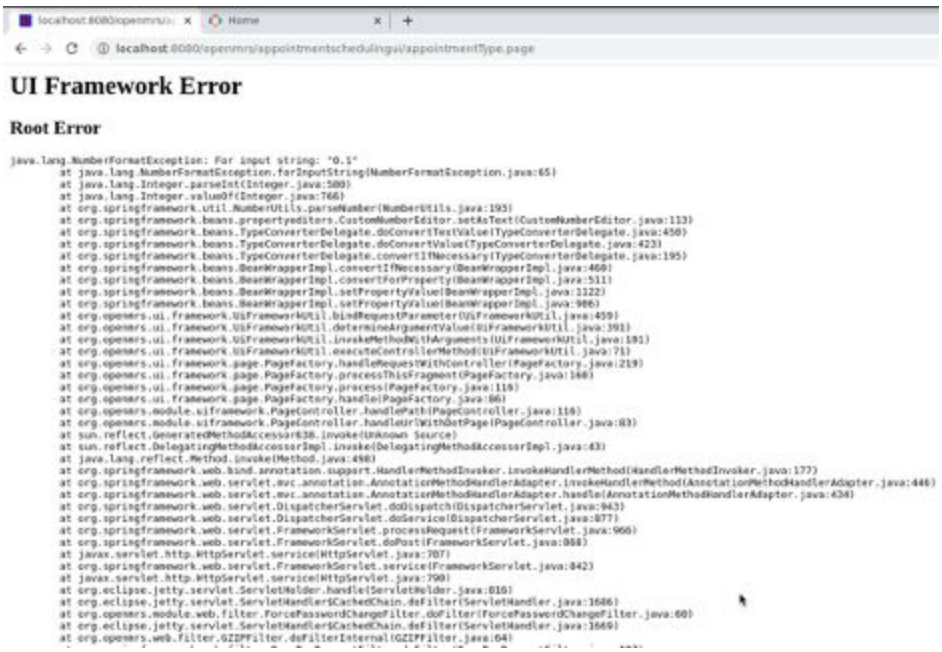
Test Case	ASVS	Unique ID	CWE
5	5.1.3	5.1.3-1	20

ASVS 5.1.3
Verify that all input (HTML form fields, REST requests, URL parameters, HTTP headers, cookies, batch files, RSS feeds, etc) is validated using positive validation (whitelisting).

CWE 20
Improper Input Validation

Repeatable Steps:

1) Navigate to http://localhost:8080/openmrs/.
2) Enter the username, “admin” and password as “Admin123”.
3) Select the location as Registration Desk.
4) Click on Login.
5) Click on Appointment Scheduling.
6) Click on Manage Service Types.
7) Click on the button “+ New Service Type”.
8) Remove any value present in Name and type “1”.
9) In the duration field, enter “0.1”.
10) Click on the Save Button.



Actual Result: Input was not sanitized.

Expected Result: Input should have been sanitized.

Result: Fail

Vulnerability 6

Test Case	ASVS	Unique ID	CWE
6	**5.1.3**	**5.1.3-1**	**20**

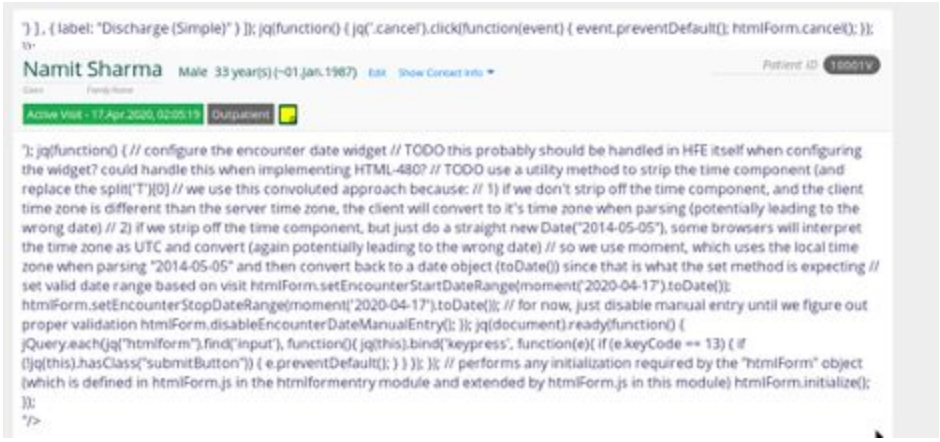
ASVS 5.1.3
Verify that all input (HTML form fields, REST requests, URL parameters, HTTP headers, cookies, batch files, RSS feeds, etc) is validated using positive validation (whitelisting).

CWE 20
Improper Input Validation

Repeatable Steps:

1) Navigate to http://localhost:8080/openmrs/.
2) Enter the username, “admin” and password as “Admin123”.
3) Select the location as Registration Desk.
4) Click on Login.
5) Click on System Administration.
6) Click on Manage Accounts.
7) Click on Add New Account.
8) Enter the family name as “Ayush” and given name as “Jain”.
9) Click on the checkbox, Add user account.
10) Enter the username as ayush and enter the password as “Hello123@”.
11) Confirm the same password, “Hello123@” in the Confirm Password field.
12) Set the privilege level as “High”.

13) Under the Capabilities, click on the checkboxes Administers System, Configures Forms, Configures Forms, Enters Vitals, Records Allergies, Requests Appointments, Schedules Appointments, Uses Capture Vitals App.
14) Click on the Save button.
15) Click on the Openmrs icon and go to the homepage or go update the URL to http://localhost:8080/openmrs/referenceapplication/home.page
16) Click on Register a Patient.
17) Enter the given name as “Namit” and family name as “Sharma”.
18) Click on Gender. Select the male option.
19) Click on Birthdate. Under Estimated years enter, 33.
20) Click on Address. Under Address, enter “1”. Under address2, enter “2”.
21) Click on phone number. Enter the number as “9191919919”.
22) Click on Confirm. Click on the Confirm button.
23) Click on start Visit.
24) Select Visit Type as “Hospitalization”.
25) Click on Confirm.
26) Select Capture Vitals.
27) Enter the height as 170.
28) Select weight, enter 70.
29) Select temperature and enter temperature as 40.
30) Select pulse and enter pulse as 40.
31) Select respiratory rate and enter 40.
32) Select Blood Pressure and enter 80/40.
33) Select Blood Oxygen Saturation and enter 40.
34) Click on the name, Namit Sharma.
35) Click on Admit to Inpatient.
36) Update the URL, and set the returnUrl parameter to “<script>alert(“hello”)</script>”. Click on enter.



Actual Result: The script triggered unknown information display on the webpage.

Expected Result: The script tag should have been sanitized and the website should have responded as expected.

Result: Fail

Vulnerability 7

Test Case	ASVS	Unique ID	CWE
7	4.1.3	4.1.3-1	285

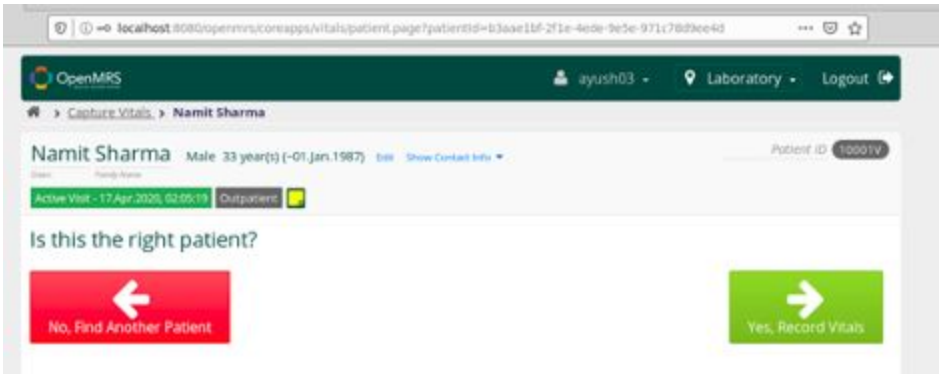
ASVS
Verify that the principle of least privilege exists - users should only be able to access functions, data files, URLs, controllers, services, and other resources, for which they possess specific authorization. This implies protection against spoofing and elevation of privilege.

CWE
Improper Authorization

Repeatable Steps:

1) Navigate to http://localhost:8080/openmrs/.
2) Enter the username, “admin” and password as “Admin123”.
3) Select the location as Registration Desk.

4) Click on Login.
5) Click on System Administration.
6) Click on Manage Accounts.
7) Click on Add New Account.
8) Enter the family name as “Ayush” and given name as “Jain”.
9) Click on the checkbox, Add user account.
10) Enter the username as “ayush03” and enter the password as “Admin123”.
11) Confirm the same password, “Admin123” in the Confirm Password field.
12) Set the privilege level as “High”.
13) Under the Capabilities, click on the checkbox Administers System.
14) Click on the Save button.
15) Click on Capture Vitals.
16) Copy the URL.
17) Click on Logout.
18) Enter the username as “ayush03” and password as “Admin123”. Click on Registration Desk and Click on Login.
19) Paste the copied URL in the URL field and press enter.
20) The below page is visible for which the person did not have access to.



Actual Result: The user not having the right access to view the page was able to access the page.

Expected Result: The user should not have been able to view the page.

Result: Fail

Vulnerability 8

Test Case	ASVS	Unique ID	CWE
8	**4.1.3**	**4.1.3-1**	**285**

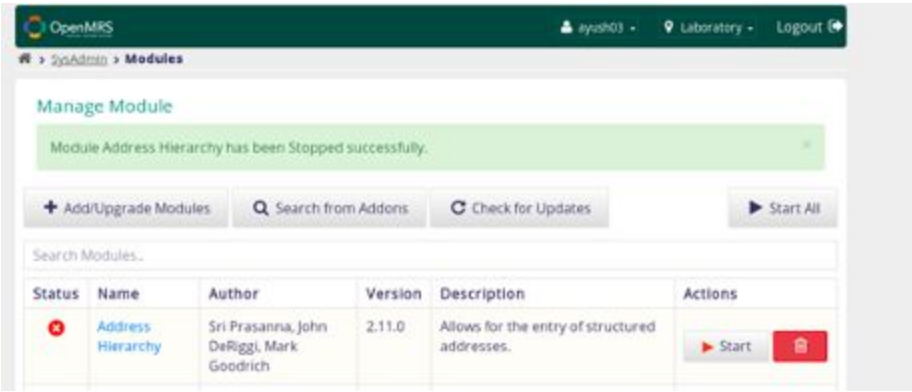
ASVS
Verify that the principle of least privilege exists - users should only be able to access functions, data files, URLs, controllers, services, and other resources, for which they possess specific authorization. This implies protection against spoofing and elevation of privilege.

CWE
Improper Authorization

Repeatable Steps:

1) Navigate to http://localhost:8080/openmrs/.
2) Enter the username, “admin” and password as “Admin123”.
3) Select the location as Registration Desk.
4) Click on Login.
5) Click on System Administration.
6) Click on Manage Accounts.
7) Click on Add New Account.
8) Enter the family name as “Ayush” and given name as “Jain”.
9) Click on the checkbox, Add user account.
10) Enter the username as “ayush03” and enter the password as “Admin123”.
11) Confirm the same password, “Hello123@” in the Confirm Password field.
12) Set the privilege level as “High”.
13) Under the Capabilities, click on the checkboxes Configures Forms and Writes Clinical Notes.
14) Click on the Save button.
15) Click on System Administration.
16) Click on System Information.
17) Copy the URL: http://localhost:8080/openmrs/owa/SystemAdministration/index.html#/system-info
18) Click on Logout.
19) Enter the username, “ayush03” and password as “Admin123”.
20) Select the location as Registration Desk. Click on Login.
21) Update the URL to http://localhost:8080/openmrs/owa/SystemAdministration/index.html#/system-info

22) Click on the tag, SysAdmin next to SytemInfo.
23) Click on Manage Modules.
24) Click on the Stop button corresponding to Address Hierarchy.
25) Click on the Yes button.



Actual Result: The user not having the right access to view the page was able to access the page.

Expected Result: The user should not have been able to view the page.

Result: Fail

Vulnerability 9

Test Case	ASVS	Unique ID	CWE
9	8.3.1	8.3.1-1	319

ASVS 8.3.1
Verify that sensitive data is sent to the server in the HTTP message body or headers, and that query string parameters from any HTTP verb do not contain sensitive data.
CWE 319
Cleartext Transmission of Sensitive Information

Repeatable Steps:

1) Navigate to http://localhost:8080/openmrs/.

2) Enter the username, “admin” and password as “Admin123”.

3) Select the location as Registration Desk.

4) Before clicking on login, open Developer tools or click on F12. Select the Network tab. Click on Login.

5) See the POST request made and view the Params field.

6) A password field is visible, “Admin123”.

Actual Result: Sensitive information sent in the browser in the params field.

Expected Result: No sensitive information should have been shared in the browser.

Result: Fail

Vulnerability 10

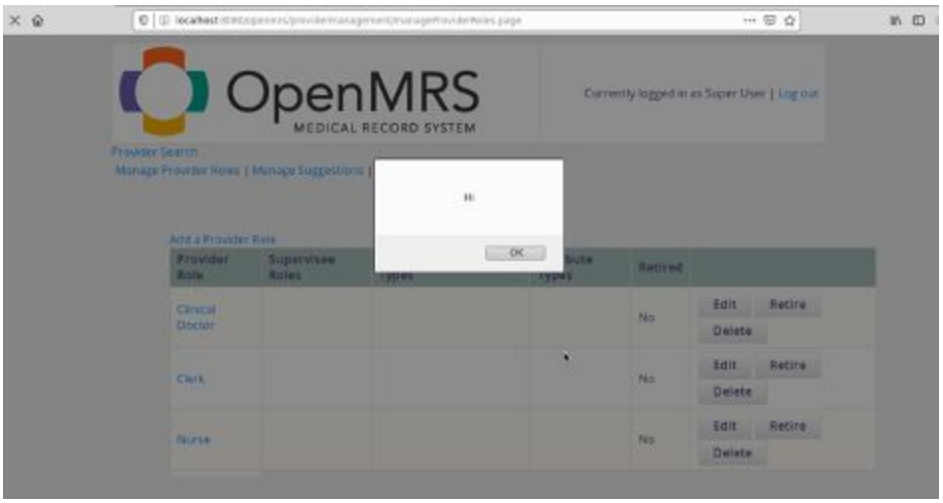
Test Case	ASVS	Unique ID	CWE
10	**5.1.3**	**5.1.3-1**	**20**

ASVS 5.1.3
Verify that all input (HTML form fields, REST requests, URL parameters, HTTP headers, cookies, batch files, RSS feeds, etc.) is validated using positive validation (whitelisting).

CWE 20
Improper Input Validation

Repeatable Steps:

1) Navigate to http://localhost:8080/openmrs/.
2) Enter the username, “admin” and password as “Admin123”.
3) Select the location as Registration Desk.
4) Click on Login.
5) Click on System Administration.
6) Click on Advanced Administration.
7) Under the Provider Management Module, select Manage Other Settings.
8) Click on Manage Provider Roles.
9) Click on Add a Provider Role.
10) Enter the provider role as, “<script>alert(“Hi”)</script>”.
11) Click on Submit.
12) An alert shows up.



Actual Result: Input was not sanitized.

Expected Result: Input should have been sanitized.

Result: Fail

Video link 3:
https://drive.google.com/file/d/1jdhoklNQ3ylcefaQtKeBdMiknjGxcdJX/view?usp=sharing

Alternate link : https://www.youtube.com/watch?v=eXdCfTlQAT8 (unlisted video)

Vulnerability number	Elapsed Time	CWE	Commentary
1	2:15	316	The server does not as HTTPS enabled
2	6:10	16	Default admin account is not disabeld
3	22:36	778	Insufficient logging
4	39:13	521	Does not follow strict password guidelines
5	57:50	284	Improper access to TEST user
6	1:06:00	79	Does not sanitise input in register patient page
7	1:33:18	79	Reflected XSS attack
8	1:48:26	778	Insufficient logging when the system app starts or stop
9	2:26:00	210	Information leakage through error message

Vulnerability 1 :

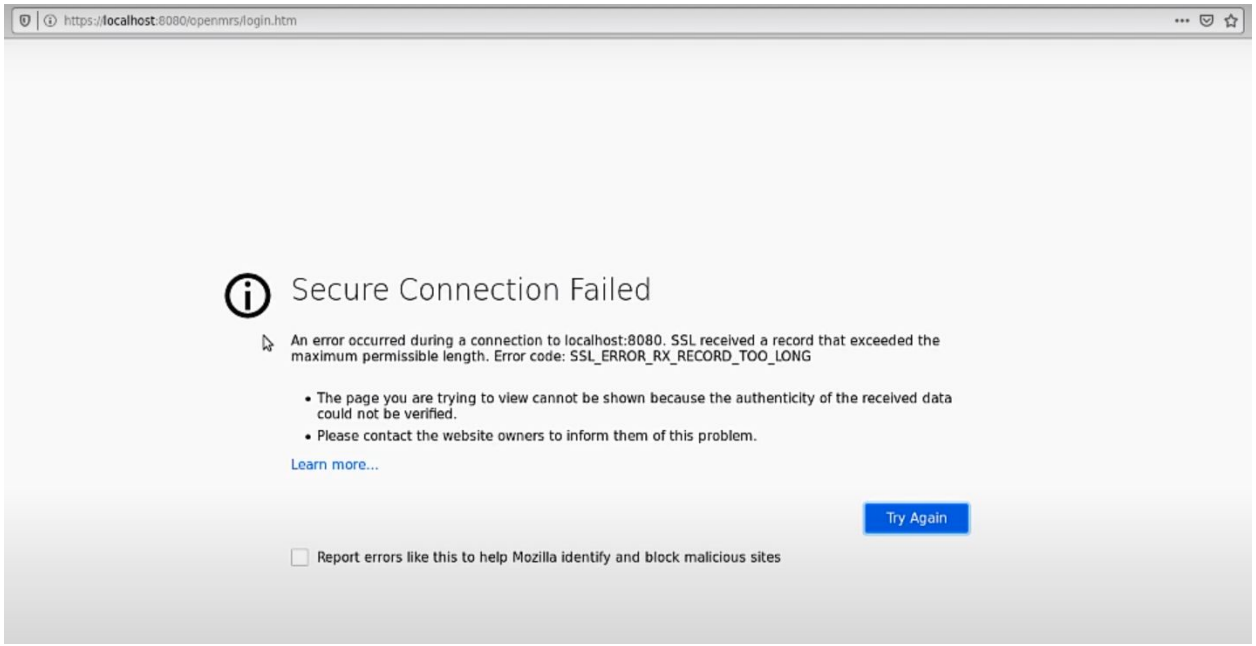
Test Case	ASVS	Unique ID	CWE
1	9.1.1	9.1.1-1	316

ASVS 9.1.1:
Verify that secured TLS is used for all client connectivity, and does not fall back to insecure or unencrypted protocols.

CWE 319: Cleartext Transmission of Sensitive Information
The software transmits sensitive or security-critical data in cleartext in a communication channel that can be sniffed by unauthorized actors.

Repeatable steps:
1. open the browser
2. go to the login page (http://localhost:8080/openmrs/login.htm)
3. Try to access the login page with https
4. Put the following address in address bar and press enter :
(https://localhost:8080/openmrs/login.htm)

Expected result : the server should establish a secure connection with the client
Actual result: the server is unable to establish a secure connection with the client
Result: Failed



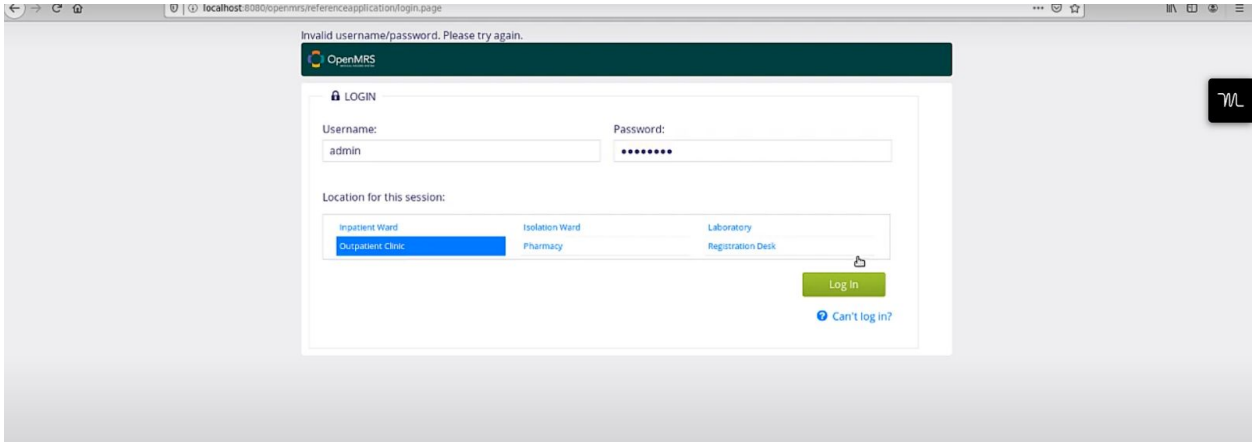
Vulnerability 2 :

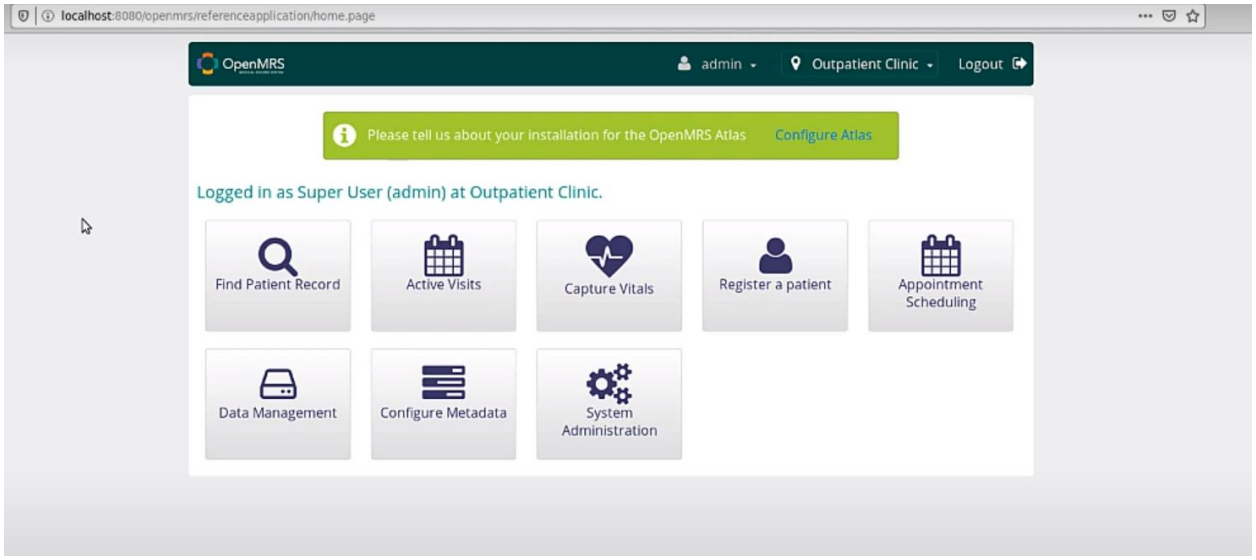
Test Case	ASVS	Unique ID	CWE
2	2.5.4	2.5.4-1	16

ASVS 2.5.4:
Verify shared or default accounts are not present (e.g. "root", "admin", or "sa").

CWE 16: Configuration
Weaknesses in this category are typically introduced during the configuration of the software.
Repeatable steps:
1. go to the login page (localhost:8080/openmrs/login.htm)
2. Enter the username as ‘admin’ and password as ‘Admin123’
3. Click on ‘outpatient clinic’ and then click on ‘Log In’

Expected result: The system doesn't let you log in.
Actual result : The system lets you log in
Result: Failed





Vulnerability 3 :

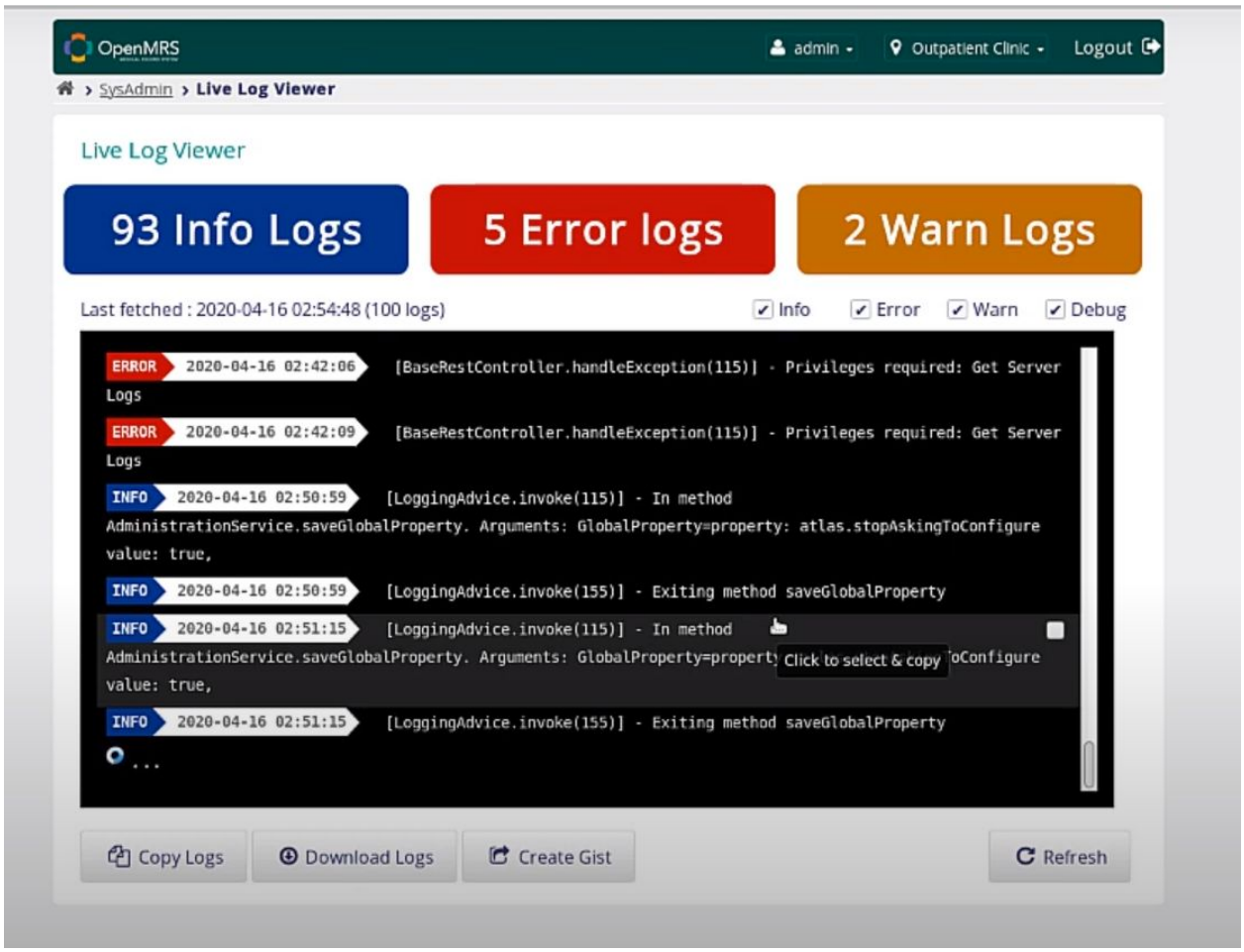
Test Case	ASVS	Unique ID	CWE
3	7.1.4	7.1.4-1	778

ASVS Name: Log Content Requirements

ASVS Description: Verify that each log event includes necessary information that would allow for a detailed investigation of the timeline when an event happens.

CWE Name: CWE-778: Insufficient Logging

Repeatable Steps:
1. Navigate to OpenMRS login page - http://localhost:8080/openmrs/login.htm
2. Login with Username - admin & Password - Admin123 and location : inpatient ward
3. Click on Logout button on the top right corner
4. Navigate to OpenMRS login page - http://localhost:8080/openmrs/login.htm
5. Login with Username - admin & Password - Admin1234 and location : inpatient ward
6. The login fails.
7. Login with Username - admin & Password - Admin123 and location : Inpatient ward
8. Check the logs at : /srv/.OpenMRS/openmrs.log or localhost:8080/openmrs/owa/SystemAdministration/index.html#/server-log



Expected result : All the authenticated users should be logged as well
Actual result: no logs pertaining to successful user login
Result : FAIL

Vulnerability 4 :

Test Case	ASVS	Unique ID	CWE
4	**2.1.1**	**2.1.1**	**521**

ASVS 2.1.1:
Verify that user set passwords are at least 12 characters in length.

CWE-521: Weak Password Requirements
The product does not require that users should have strong passwords, which makes it easier for attackers to compromise user accounts.

Repeatable Steps:
1. Navigate to OpenMRS login page - http://localhost:8080/openmrs/login.htm
2. Login with Username - admin & Password - Admin123 and location : inpatient ward
3. Click on ‘system administration’
4. Click on ‘advanced administration’
5. Click on ‘my profile’
6. Click on ‘change login info’
7. Type in ‘Admin123’ in Old password.
8. Type in ‘Admin123’ in ‘new password’ and ‘confirm new password’
9. Click on Save options

Expected result: The system should not accept the new password as it is less than 12 characters
Actual Result: The system accepts the password
Result - FAIL

Vulnerability 5 :

Test Case	ASVS	Unique ID	CWE
5	1.4.4.	1.4.4-1	284

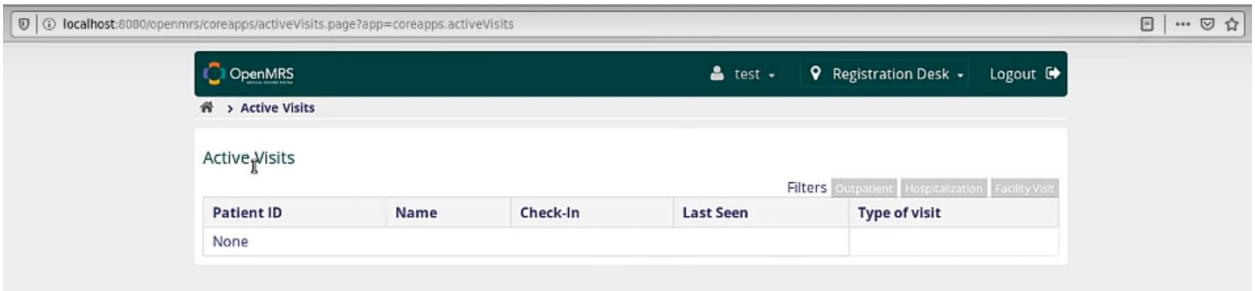
ASVS 1.4.4:
Verify the application uses a single and well-vetted access control mechanism for accessing protected data and resources. All requests must pass through this single mechanism to avoid copy and paste or insecure alternative paths.

CWE: Improper Access Control
The software does not restrict or incorrectly restricts access to a resource from an unauthorized actor.

Repeatable steps:
1. Navigate to OpenMRS login page - http://localhost:8080/openmrs/login.htm
2. Login with Username - admin & Password - Admin123 and location : inpatient ward
3. Click on ‘system administration’
4. Click on ‘Manage Account’
5. Click on ‘add new account’
6. Type the following details in the given fields : Family name : test , Given Name : test , Gender : Male , check on ‘ add user account’ , username : test , privilege level : Full , Password : testtest , Confirm Password : testtest, uncheck ‘force password change’

7. Click on save
8. Click on ‘Logout’
9.Navigate to OpenMRS login page - http://localhost:8080/openmrs/login.htm
10. Login with Username - test & Password - testtest and location : inpatient ward (the login location does not matter because we have not given the test user any privilege as of now)
11. On the URL bar , type the following address :
http://localhost:8080/openmrs/coreapps/activeVisits.page?app=coreapps.activeVisits

Expected result : the user ‘test’ should not be able to access this web page
Actual result: the user ‘test’ is able to access this web page
Result: FAIL



Vulnerability 6 :

Test Case	ASVS	Unique ID	CWE
6	5.3.3	5.3.3-1	79

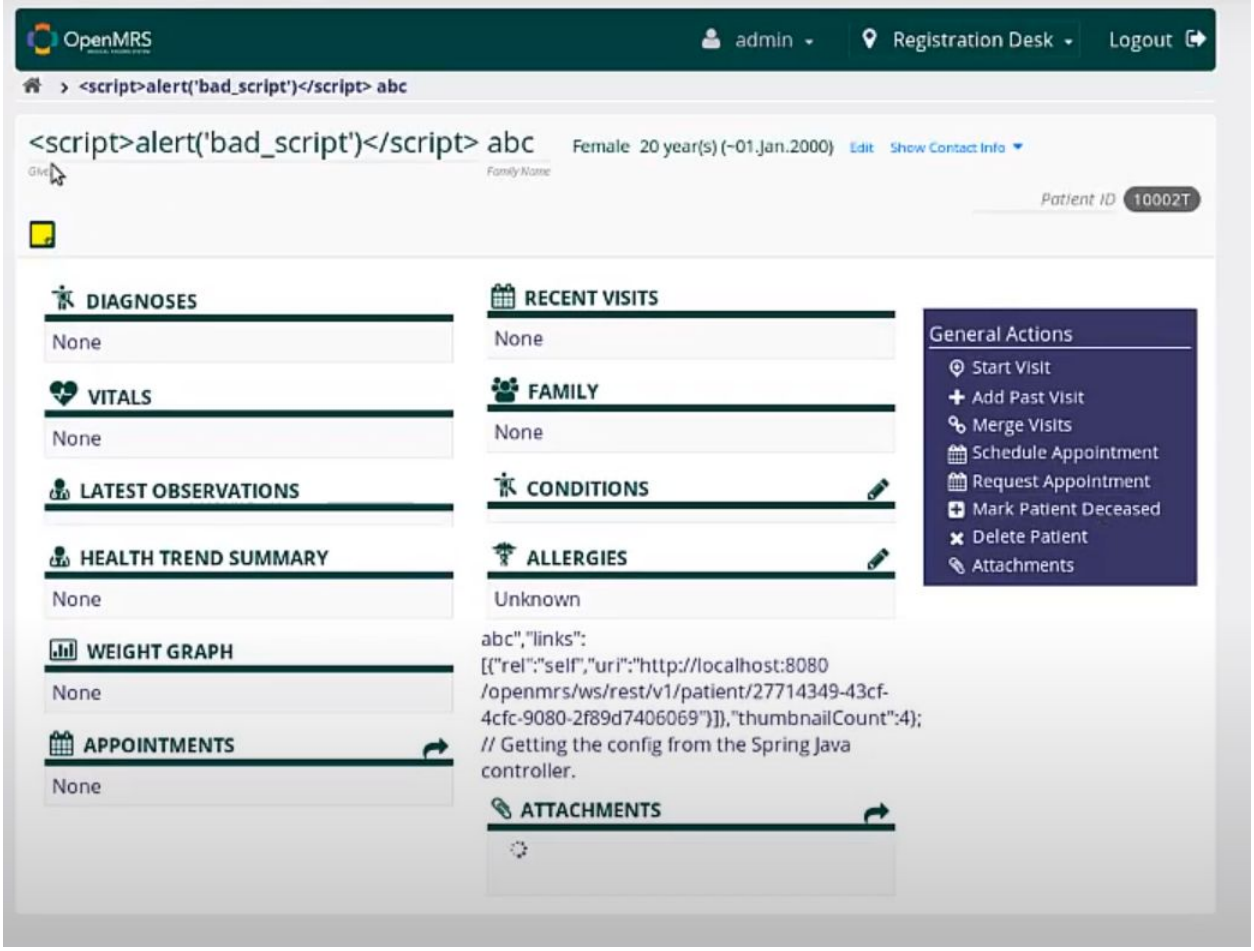
ASVS 5.3.3:
Verify that context-aware, preferably automated - or at worst, manual – output escaping protects against reflected, stored, and DOM based XSS.

CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
The software does not neutralize or incorrectly neutralizes user-controllable input before it is placed in output that is used as a web page that is served to other users.

Repeatable steps:
1. Navigate to OpenMRS login page - http://localhost:8080/openmrs/login.htm
2. Login with Username - admin & Password - Admin123 and location : Registration desk
3. Click on Register a patient and enter the following information : given name : <script>alert (‘bad_script’);</script>, family name : abc , gender : female , estimated years : 30 years, address : 123 , city : raleigh, state : NC , Country : US , phone number : 123 , relative : doctor - 111

4. Click on ‘confirm’

Expected result: the system should reject the request to create a new patient because the name can not contain <script> tags
Actual Result : The System accepts the inputs
Result : FAIL



Vulnerability 7 :

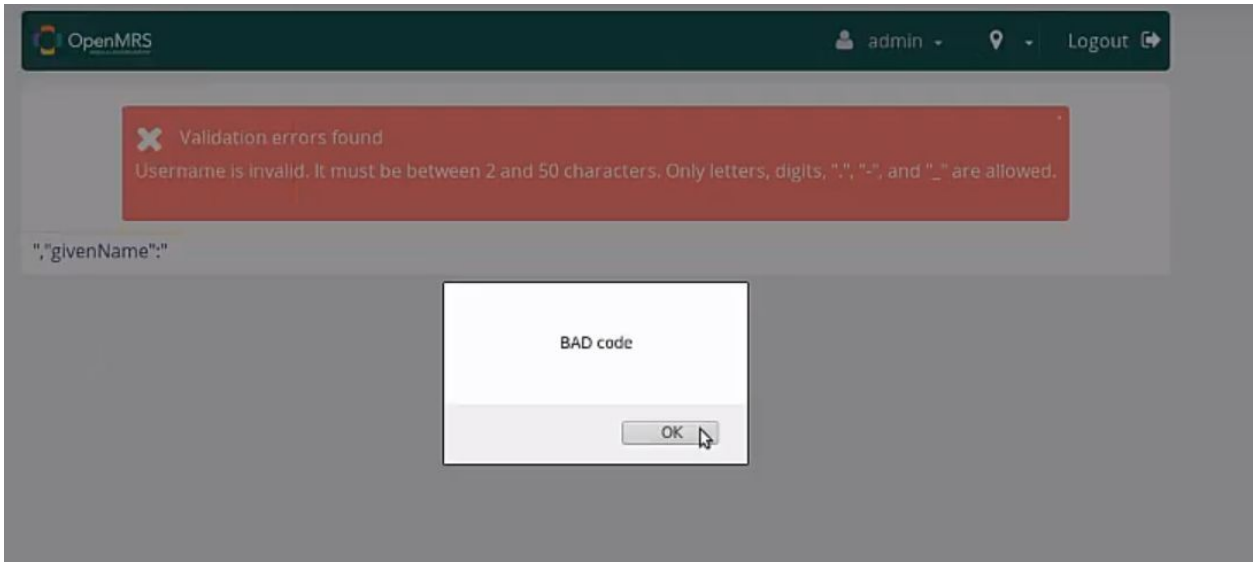
Test Case	ASVS	Unique ID	CWE
7	5.3.3	5.3.3-2	79

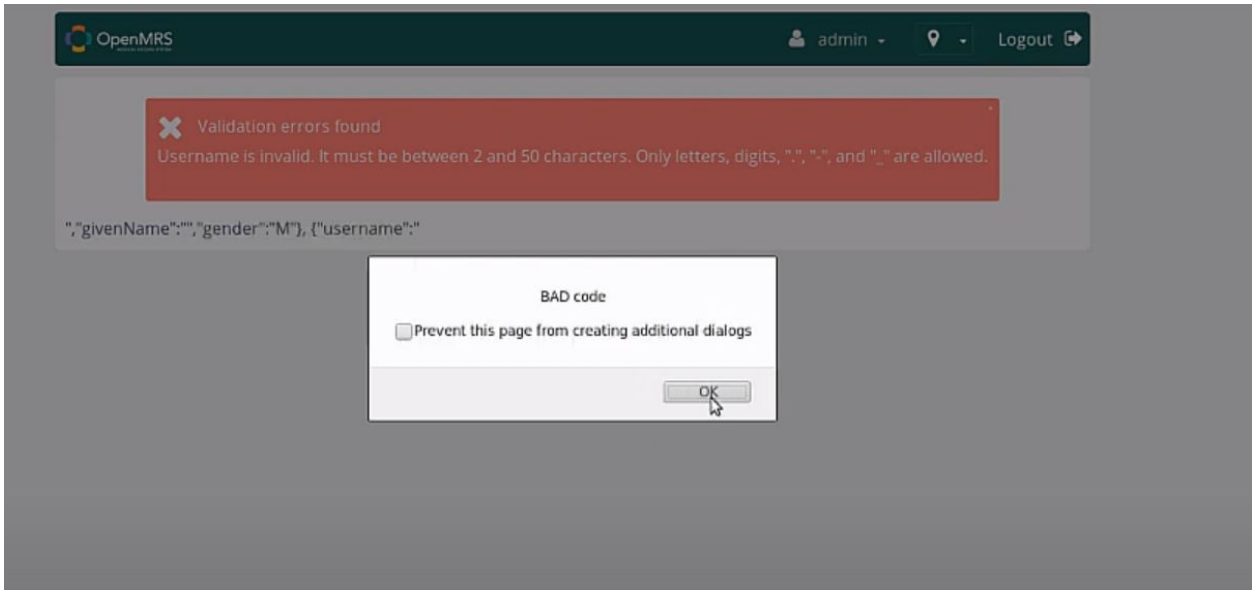
ASVS 5.3.3:
Verify that context-aware, preferably automated - or at worst, manual – output escaping protects against reflected, stored, and DOM based XSS.

CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
The software does not neutralize or incorrectly neutralizes user-controllable input before it is placed in output that is used as a web page that is served to other users.

Repeatable steps:
1. Navigate to OpenMRS login page - http://localhost:8080/openmrs/login.htm
2. Login with Username - admin & Password - Admin123 and location : Registration desk
3. Click on ‘system administration’
4. Click on ‘Manage Account’
5. Click on ‘add new account’
6. Type the following details in the given fields : Family name : <script>alert(‘BAd code’);</script>, Given Name : <script>alert(‘BAd code’);</script> , Gender : Male , check on ‘ add user account’ , username : <script>alert(‘BAd code’);</script> , privilege level : Full , Password : test1234, Confirm Password : testtest, uncheck ‘force password change’
7. Select all the capabilities (put a tick mark on the check boxes)
8. Click on save

Actual result: The system executes the code in the <script> tag
Expected result: The system should not execute the <script> tag.
Result : FAIL





Vulnerability 8 :

Test Case	ASVS	Unique ID	CWE
8	7.1.4	7.1.4-1	778

ASVS Name: Log Content Requirements

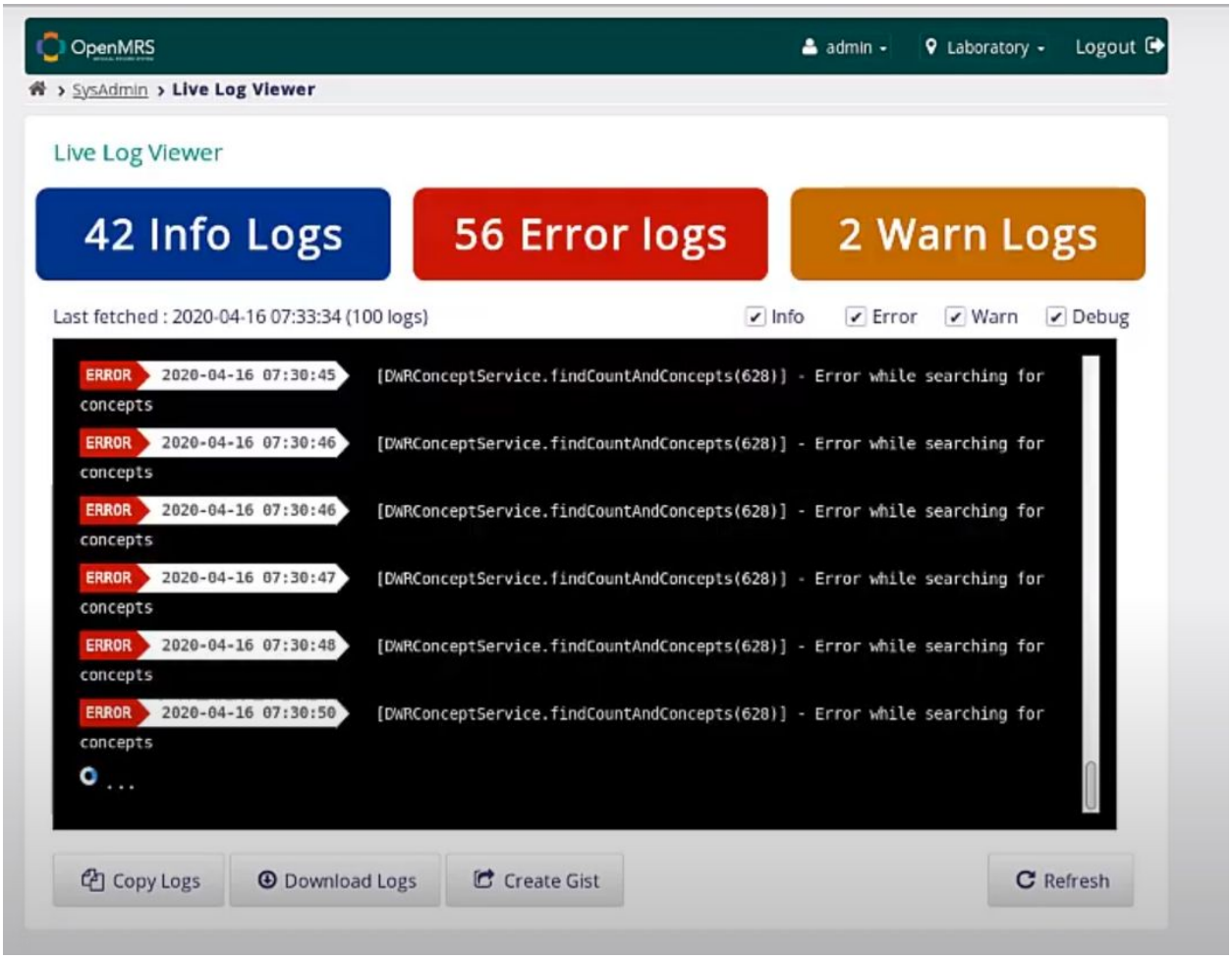
ASVS Description: Verify that each log event includes necessary information that would allow for a detailed investigation of the timeline when an event happens.

CWE Name: CWE-778: Insufficient Logging

Repeatable Steps:
1. Navigate to OpenMRS login page - http://localhost:8080/openmrs/login.htm
2. Login with Username - admin & Password - Admin123 and location : inpatient ward
3. Click on ‘system administration’
4. Click on ‘manage app’
5. Click on the small play button on the right hand side of the scree
6. Stop the next 3 apps by clicking on the stop button in the next 3 rows
7. Access the server logs by accessing this URL :
localhost:8080/openmrs/owa/SystemAdministration/index.html#/server-log

Expected result: The logs should mention that the application was stopped

Actual Result : The logs do not capture any of the information regarding app being stopped or started
Result : FAIL



Vulnerability 9 :

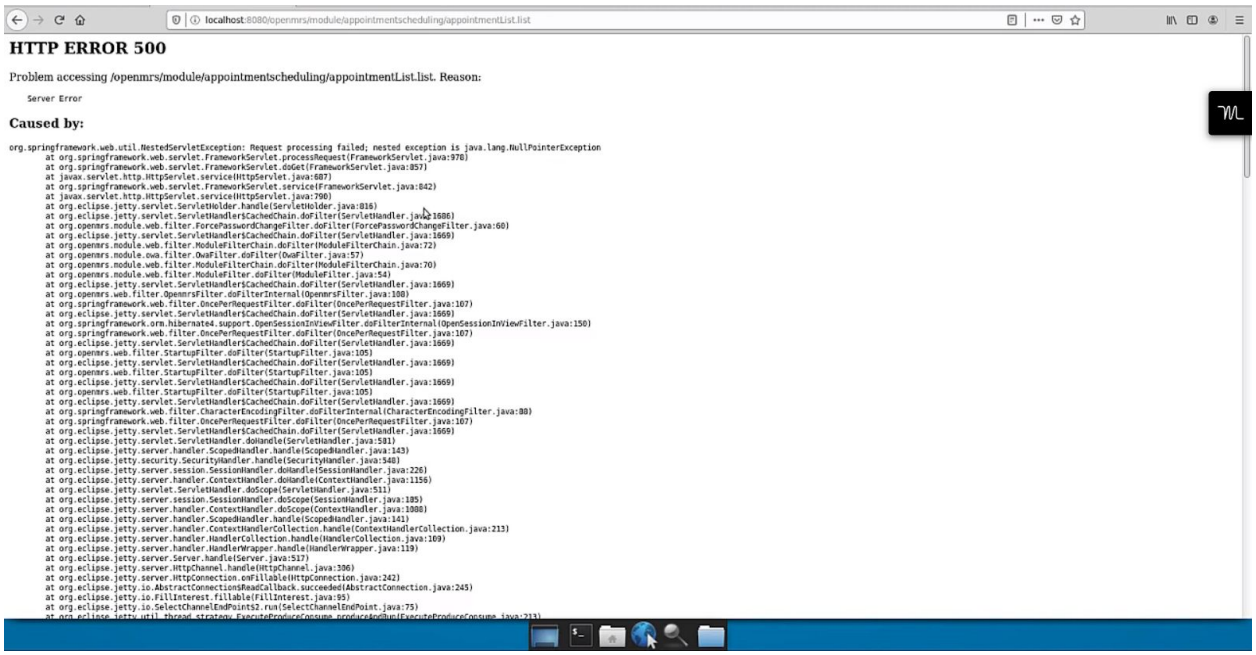
Test Case	ASVS	Unique ID	CWE
9	7.4.1	7.4.1-1	210

ASVS 7.4.1: Verify that a generic message is shown when an unexpected or security sensitive error occurs, potentially with a unique ID which support personnel can use to investigate

CWE 210 : The software identifies an error condition and creates its own diagnostic or error messages that contain sensitive information.

Repeatable Steps:
1. Navigate to OpenMRS login page - http://localhost:8080/openmrs/login.htm
2. Login with Username - admin & Password - Admin123 and location : inpatient ward
3. Click on ‘system administration’
4. Click on ‘Advanced Administration’
5. Click on ‘appointments’

Expected result: The system should give a generic error that the page cannot be displayed
Actual Result: The system gives a stack trace , with revela some of the internal file names / module name and line numbers as well
Result : Fail



Video link 4: https://drive.google.com/open?id=1vAxgTOyjLZBXZHA6XlyY8c0l5ltnmUDv

Vulnerability #	Elapsed Time	Ref Info for Video Traceability	CWE	Commentary

1.	**32 min**	**Part 1**	**598**	**Check to ensure sensitive data is not passed.**
2.	**34 min**	**Part 2**	**601**	**Untrusted re**
3.	**36 min**	**Part 2**	**598**	**Check to ensure sensitive data is not passed.**
4.	**55 min**	**Part 3**	**209**	**Untrusted redirection does not take place**
5.	**1 hr 5 min**	**Part 3**	**521**	**Password Requirements**
6.	**2 hr 17 min**	**Part 3**	**598**	**Check to ensure sensitive data is not passed.**
7.	**2 hr 20 min**	**Part 3**	**598**	**Check to ensure sensitive data is not passed.**

Vulnerability1:

Test Case	ASVS	Unique ID	CWE
1	**13.1.3**	**13.1.3-1**	598

ASVS-13.1.3: Verify API URLs do not expose sensitive information, such as the API key, session tokens etc.

CWE-598: Information Exposure Through Query Strings in GET Request

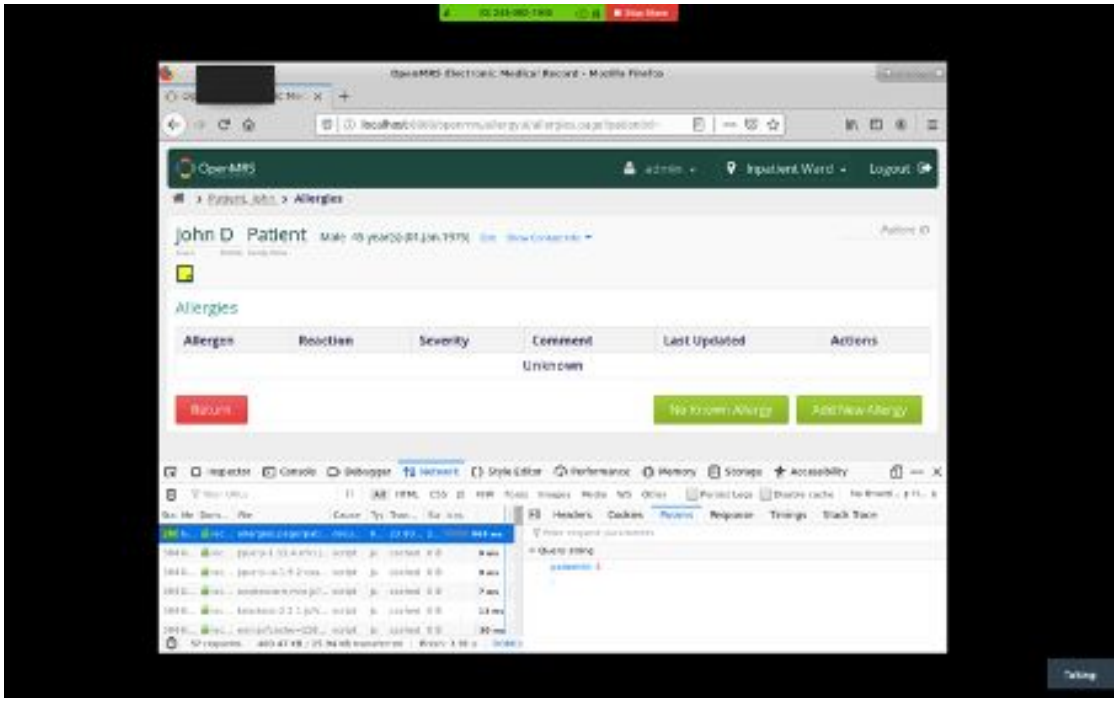
Repeatable Steps:

1. Open the OpenMRS application using the following link - http://localhost:8080/openmrs/login.htm
2. Enter the following admin credentials -
   i.Username: admin

ii.Password: Admin123

3. Select “Inpatient ward” as the location and Select “Login”.
4. Select ‘Find Patient Record’ tab.
5. Open the web developer console and navigate to the network tab.
6. Now select any patient (let’s say – patient with id - 10006H) and examine the http request response headers.
7. Click on the edit link to update allergy details for the patient.
8. Look for the corresponding URL request in the network tab and we see that the patient Id is exposed in the query string.

Actual Results: The request URL exposes the patient id and other sensitive information to the user. Screen shot of the same has been attached below-



Expected Results: If the web application uses HTTP GET method, it should not include sensitive information in the query string of that request.
Results: Fail

Vulnerability2:

Test Case	ASVS	Unique ID	CWE
2	5.1.5	5.1.5-1	601

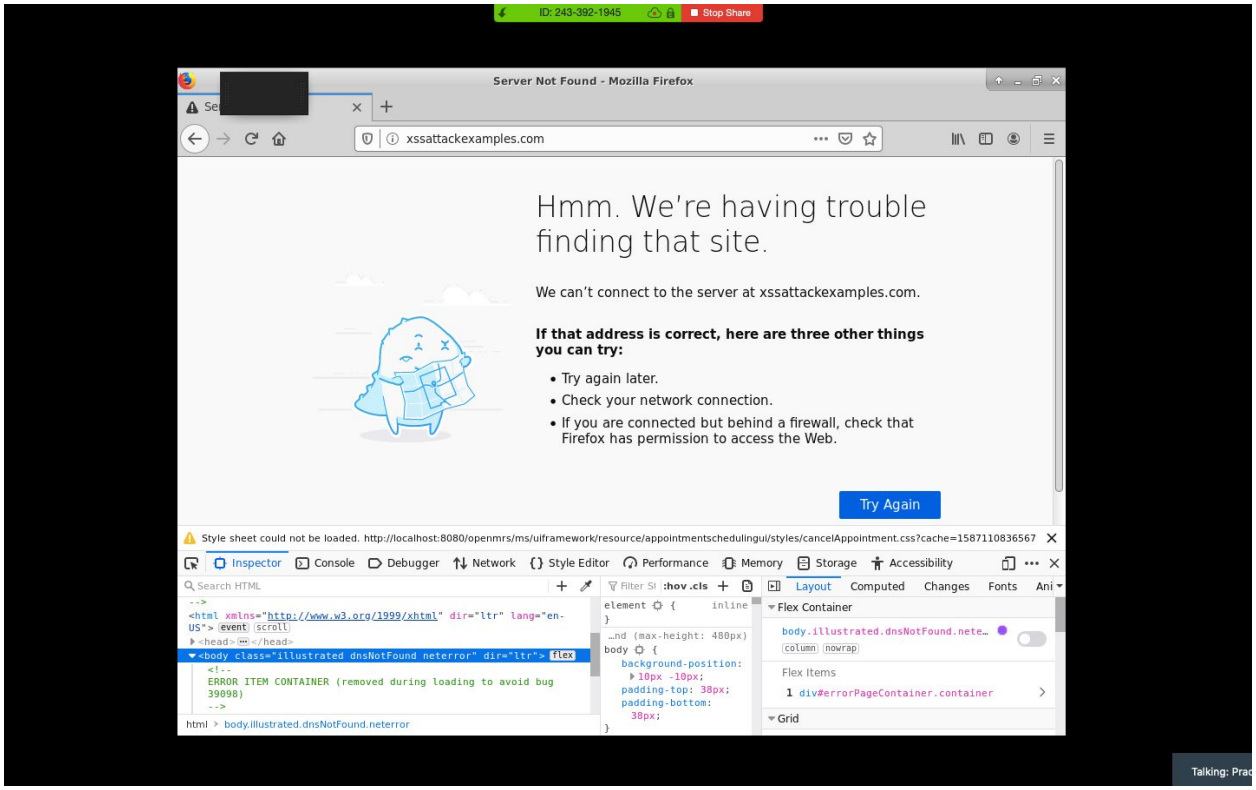
ASVS 5.1.5: Verify that URL redirects and forwards only allow whitelisted destinations, or show a warning when redirecting to potentially untrusted content

CWE-601: URL Redirection to Untrusted Site ('Open Redirect')

Repeatable Steps:

1. Open the OpenMRS application using the link - http://localhost:8080/openmrs/login.htm
2. Login as the admin using the following credentials-
3. Username: 'admin'
4. Passowrd:'Admin123'
5. Select 'Registration Desk' as the location and login.
6. Select 'Find a Patient tab' and look up for patient with id –10006H.
7. Select the 'Exit Form' button to inspect the element in the console.
8. Replace the href attribute contents with - http://xssattackexamples.com.
9. Click on the Exit Form**.**

Actual Results: We see that user gets redirected to http://xssattackexamples.com.
Screen shot of the same has been attached below-



Expected Results: A web application should not accept a user-controlled input that specifies a link to an external site, or use that link in a Redirect as this simplifies phishing attacks.

Results: Fail

Vulnerability3:

Test Case	ASVS	Unique ID	CWE
3	**13.1.3**	**13.1.3-1**	598

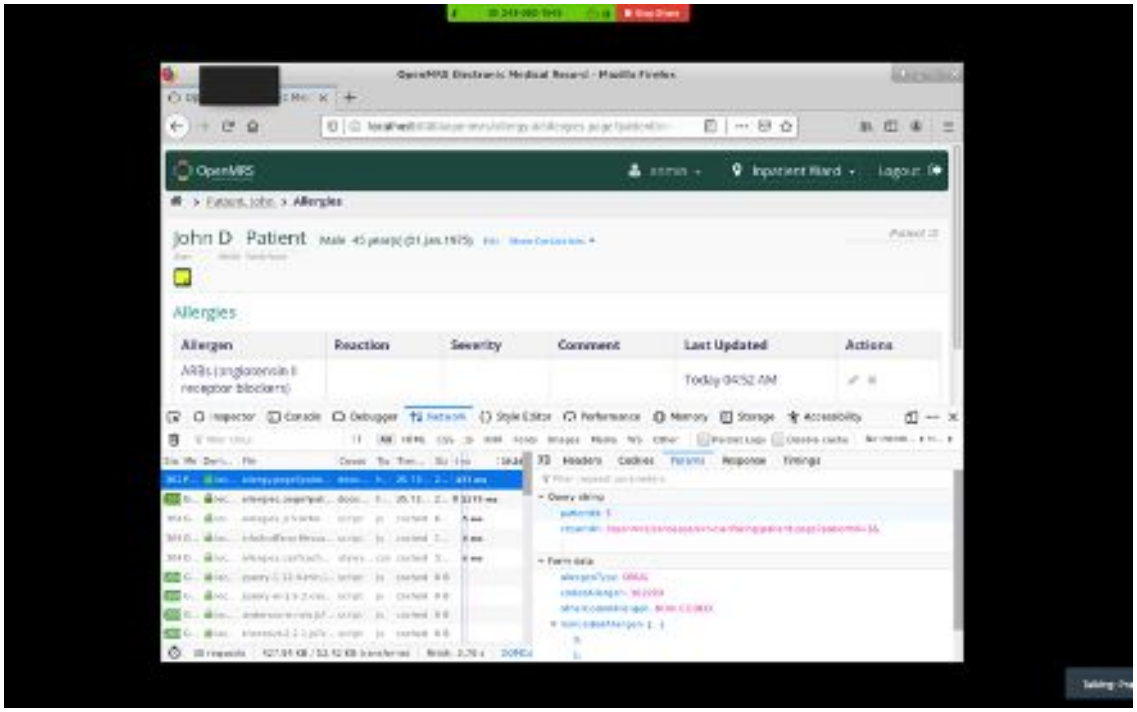
ASVS-13.1.3: Verify API URLs do not expose sensitive information, such as the API key, session tokens etc.

CWE-598: Information Exposure Through Query Strings in GET Request

Repeatable Steps:

1. Open the OpenMRS application using the following link - http://localhost:8080/openmrs/login.htm
2. Enter the following admin credentials -
   i.Username: admin
   ii.Password: Admin123
3. Select “Inpatient ward” as the location and Select “Login”.
4. Select ‘Find Patient Record’ tab.
5. Open the web developer console and navigate to the network tab.
6. Now select any patient (let’s say – patient with id - 10006H) and examine the http request response headers.
7. Click on the edit link to add allergy details for the patient.
8. Select the ARBs allergen from the menu and click save.
8. Look for the corresponding URL request in the network tab and we see that the patient Id is and other form data details exposed in the params

Actual Results: The request URL exposes the patient id and other sensitive information to the user. Screen shot of the same has been attached below-



Expected Results: If the web application uses HTTP GET method, it should not include sensitive information in the query string of that request.

Test Result : Fail

Vulnerability4:

Test Case	ASVS	Unique ID	CWE
4	**14.3.1**	**14.3.1-1**	209

Unique Id: 14.3.1-1

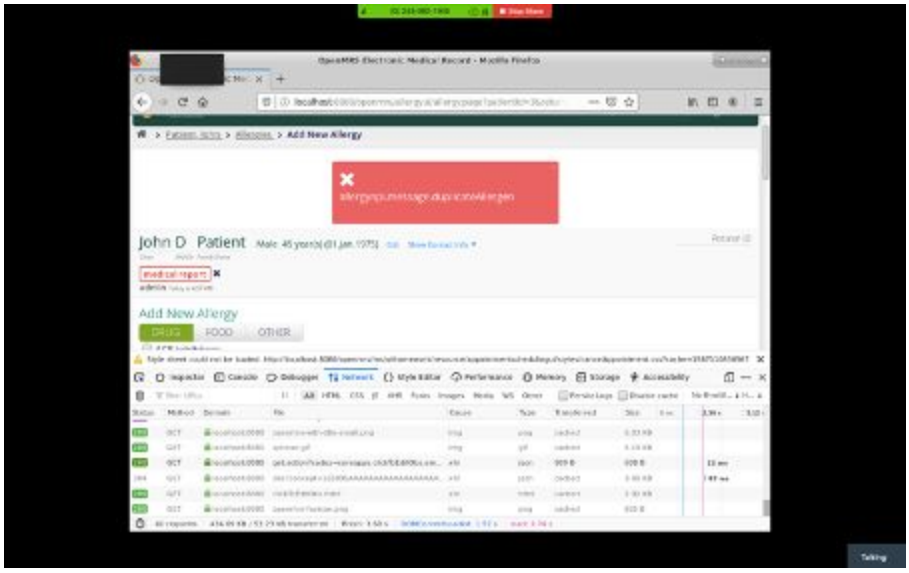
ASVS 14.3.1: Verify that web or application server and framework error messages are configured to deliver user actionable, customized responses to eliminate any unintended security disclosures.

CWE-209: Information Exposure Through an Error Message

Repeatable Steps:

1. Open the OpenMRS application using the link - http://localhost:8080/openmrs/login.htm
2. Login as the admin using the following credentials-
   i.Username: 'admin'
   ii.Passowrd:'Admin123'
3. Select 'Registration Desk' as the location and login.
4. Select 'Find a Patient tab' and look up for a patient, say,with id –10006H.
5. Click on the edit link to add allergies.
7. Now try adding the same allergy again.

Actual Results: The notification exposes the application api details. Screen shot of the same is attached below.



Expected Results: The software should not generate a message that includes sensitive information about application environment, users, or associated data.

Result: Fail

Vulnerability5:

Test Case	ASVS	Unique ID	CWE
5	2.1.1	2.1.1-1	521

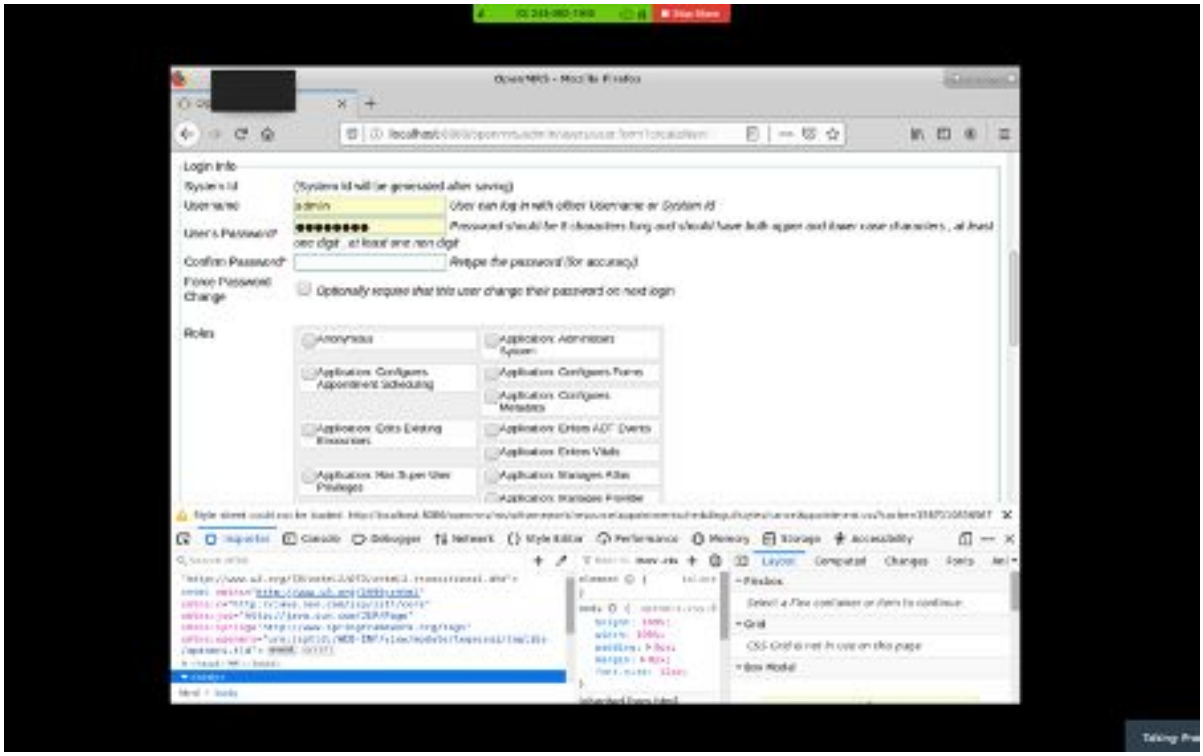
ASVS 2.1.1:Verify that user set passwords are at least 12 characters in length.
CWE-521: Weak Password Requirements

The product does not require that users should have strong passwords, which makes it easier for attackers to compromise user accounts.

Repeatable steps:

1. Login as admin by entering the username as “admin” and password as “Admin123”.
2. Click on System Administration
3. Click on advanced administration
4. Click on manage users
5. Click on Add user
6. Under ‘create a new person’ click on next.

Actual Result: We see that password length is set to 8 characters. Screen shot of the same has been attached below.



Expected result: As described in ASVS 2.1.1, minimum password length should be 12 characters.
Result: Fail

Vulnerability6:

Test Case	ASVS	Unique ID	CWE
6	13.1.3	13.1.3-1	598

Unique Id: 13.1.3-1

ASVS-13.1.3: Verify API URLs do not expose sensitive information, such as the API key, session tokens etc.

CWE-598: Information Exposure Through Query Strings in GET Request

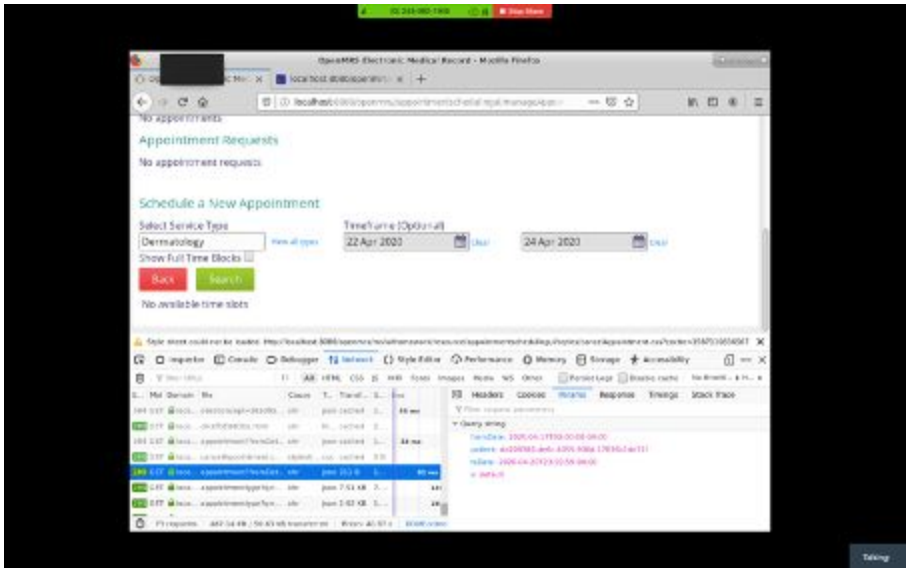
Repeatable Steps:

1. Open the OpenMRS application using the following link - http://localhost:8080/openmrs/login.htm
2. Enter the following admin credentials -
   i.Username: admin

ii.Password: Admin123

3. Select “Inpatient ward” as the location and Select “Login”.
4. Select ‘Find Patient Record’ tab.
5. Open the web developer console and navigate to the network tab.
6. Now select any patient (let’s say – patient with id - 10006H) and examine the http request response headers.
7. Click on the schedule appointment to set up an appointment for the patient.
8. Select the service type as Dermatology and choose appropriate dates.
9. Look for the corresponding URL request in the network tab and we see that the patient Id, to-Date and from-date are exposed in the query string.

Actual Results: The request URL exposes the patient id and other sensitive information to the user. Screen shot of the same has been attached below.



Expected Results: If the web application uses HTTP GET method, it should not include sensitive information in the query string of that request.

Test Result : Fail

Vulnerability7:

Test Case	ASVS	Unique ID	CWE
7	**13.1.3**	**13.1.3-1**	598

Unique Id: 13.1.3-1

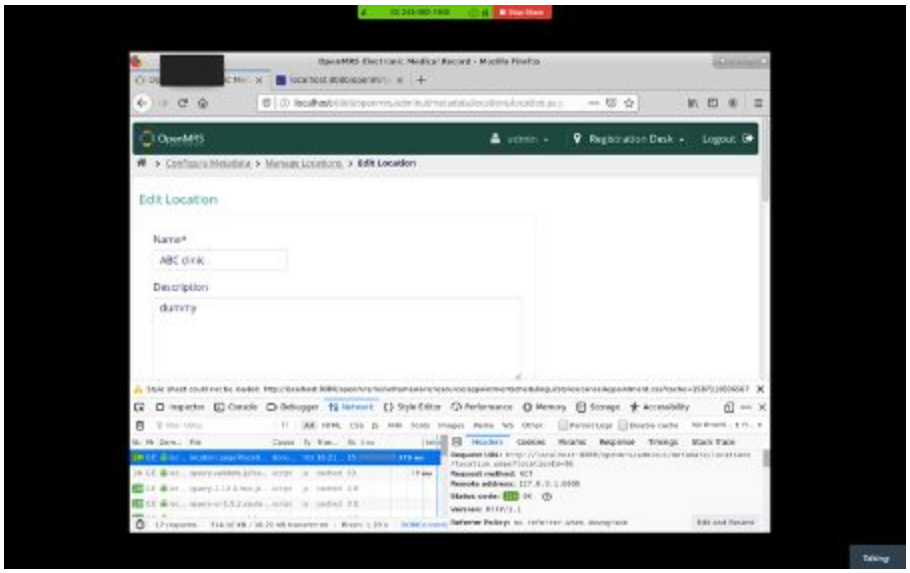
ASVS-13.1.3: Verify API URLs do not expose sensitive information, such as the API key, session tokens etc.

CWE-598: Information Exposure Through Query Strings in GET Request

Repeatable Steps:

1. Open the OpenMRS application using the following link - http://localhost:8080/openmrs/login.htm
2. Enter the following admin credentials -
   i.Username: admin
   ii.Password: Admin123
3. Select “Inpatient ward” as the location and Select “Login”.
4. Select the ‘Configure Metadata’ tab and go to Manage Location.
5. Open the web developer console and navigate to the network tab.
6. Try to edit a location- adding the name –‘abc’ and name –‘description’.
7. Now click on Save and see the param of the request url.
7. Look for the corresponding URL request in the network tab and we see that the location Id details exposed in the query string.

Actual Results: The request URL exposes sensitive information to the user. Screen shot of the same has been attached below.



Expected Results: If the web application uses HTTP GET method, it should not include sensitive information in the query string of that request.

Test Result : Fail

Overall Results:

Name	Vulnerability	Time	Efficiency
Name 1	16	3 hrs	5.33

Name 2	**10**	**3 hrs**	**3.33**
Name 3	**9**	**3 hrs**	**3**
Name 4	**7**	**3 hrs**	**2.33**
Total	**42**	**12**	**3.5**

Technique		**# of true positive vulnerabilities discovered**	**Total time (hours)**	**Efficiency: # vulnerabilities/ total. time**	**Detecting Exploitable vulnerabilities? (High/Med/Low)**	**Unique CWE numbers**
Manual black box	**Manual Testing**	10	11 hours 30 minutes	True Positive results : 10/11.5 = approximately 0.9 defects per hour	Medium	89,521,620, 284,544,79, 639,209,307,16,640,319,272,285,390
Static analysis	**Coverity**	10(out of 10 test cases)	6 hours	True Positive Results: 10/6 = approximately 2 defects/hour	High	470, 502, 79, 352, 601, 89, 22
	Fortify	6(out of 10 test cases)	5 hours	True Positive Results: 6/5 = approximately 1 defect/hour	Medium	20, 367, 173

				Total results: 10/5 = 2		
Dynamic Analysis	**Defensics**	5(out of 5 test cases)	10 hours	1 defect per 2 hrs	Medium	778,22,60 1,209,598
	ZAP	2(out of 10 test cases)	5	⅖ = approximately 2 defetcs/hour	Low	89, 79, 209, 120
Interactive testing	**Seeker**	5/5	5 hours	5/5 = 1 defect/hr	High	74, 89, 451, 613, 691, 918, 470, 319, 614, 113, 937
Penetration testing		42	12	42/12 = 3.5 approximately 3 defects per hour	High	**200, 209, 285, 613, 79, 94, 544, 20, 601, 319, 598, 601, 209, 521, 316, 16, 778, 521, 284 , 210**

Comment(Combined):

Penetration testing is a way to stress test the system. From the above table we can see the number of vulnerabilities that we were able to find a range of vulnerabilities.
Some of the errors include:

1. Input fields vulnerable to XSS attack

2. Stack trace errors - exceptions aren't handled correctly
3. Code display on UI

Also, we were able to find a few CWEs, which were not discovered before - 200,16 etc. It even covers a few which we had found earlier.
Some general observation on penetration testing:

--- you never know if the testing is over. For example, when we do static code analysis we get to know when the test has finished. In penetration testing , we do not know if we have covered all the areas of the system. A penetration tester does not know anything about the internal code of the system , and he might just keep on testing the same thing over and over again

--- Penetration testing takes more time to run as compared to static testing.
--- Penetration testing is more difficult to document as compared to static testing. The reason being , penetration testing can not find the exact code that is vulnerable, and hence it is difficult to document. However, in case of static analysis tools (such as fortify and coverity) , we can figure out the exact vulnerable code. These software generate automated reports that pinpoints the exact line of code in a specific file / module that might be a security risk

--- A static analysis tool can miss out some edge cases, while penetration testing is more focused on testing the edge cases.

--- From the above table (question 4) , we can see that the true positive rate for a vulnerability is more in penetration testing

Fortify Security Report

Jan 23, 2020

pgupta25

Executive Summary

Issues Overview

On Jan 23, 2020, a source code review was performed over the appointmentscheduling code base. 315 files, 31,941 LOC (Executable) were scanned and reviewed for defects that could lead to potential security vulnerabilities. A total of 114 reviewed findings were uncovered during the analysis.

Issues by Fortify Priority Order	
Critical	102
High	12

Recommendations and Conclusions

The Issues Category section provides Fortify recommendations for addressing issues at a generic level. The recommendations for specific fixes can be extrapolated from those generic recommendations by the development group.

Project Summary

Code Base Summary

Code location: /srv/openmrs_code/org/openmrs/module/appointmentscheduling
Number of Files: 315
Lines of Code: 31941
Build Label: <No Build Label>

Scan Information

Scan time: 29:30
SCA Engine version: 19.1.0.2241
Machine Name: vclv99-89.hpc.ncsu.edu
Username running scan: pgupta25

Results Certification

Results Certification Valid

Details:

Results Signature:

SCA Analysis Results has Valid signature

Rules Signature:

There were no custom rules used in this scan

Attack Surface

Attack Surface:
Private Information:
null.null.null

System Information:
null.null.null
java.lang.Throwable.getMessage

Web:
javax.servlet.http.HttpServletRequest.getMethod

Filter Set Summary

Current Enabled Filter Set:
Quick View

Filter Set Details:

Folder Filters:
If [fortify priority order] contains critical Then set folder to Critical
If [fortify priority order] contains high Then set folder to High

If [fortify priority order] contains medium Then set folder to Medium
If [fortify priority order] contains low Then set folder to Low
Visibility Filters:
If impact is not in range [2.5, 5.0] Then hide issue
If likelihood is not in range (1.0, 5.0] Then hide issue

Audit Guide Summary

Audit guide not enabled

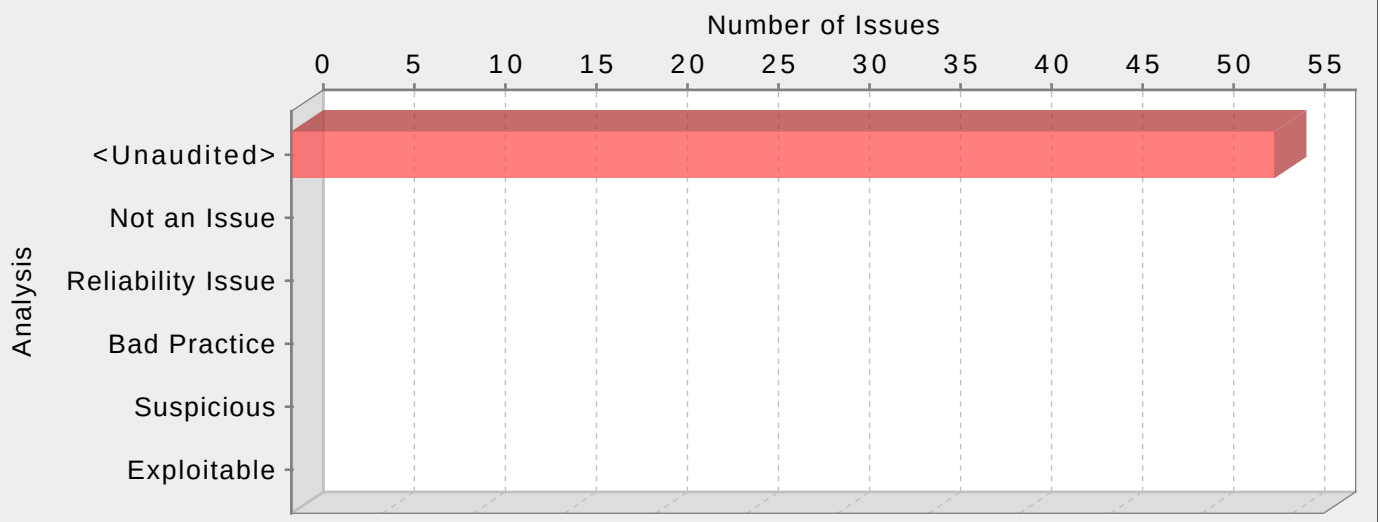
Results Outline

Overall number of results

The scan found 114 issues.

Vulnerability Examples by Category

Category: Cross-Site Scripting: DOM (54 Issues)



Abstract:

The method addNewAppointment() in appointments.jsp sends unvalidated data to a web browser on line 18, which can result in the browser executing malicious code.

Explanation:

Cross-site scripting (XSS) vulnerabilities occur when:

1. Data enters a web application through an untrusted source. In the case of DOM-based XSS, data is read from a URL parameter or other value within the browser and written back into the page with client-side code. In the case of reflected XSS, the untrusted source is typically a web request, while in the case of persisted (also known as stored) XSS it is typically a database or other back-end data store.

2. The data is included in dynamic content that is sent to a web user without being validated. In the case of DOM Based XSS, malicious content gets executed as part of DOM (Document Object Model) creation, whenever the victim's browser parses the HTML page.

The malicious content sent to the web browser often takes the form of a segment of JavaScript, but may also include HTML, Flash or any other type of code that the browser executes. The variety of attacks based on XSS is almost limitless, but they commonly include transmitting private data like cookies or other session information to the attacker, redirecting the victim to web content controlled by the attacker, or performing other malicious operations on the user's machine under the guise of the vulnerable site.

Example 1: The following JavaScript code segment reads an employee ID, eid, from a URL and displays it to the user.

```
<SCRIPT>
var pos=document.URL.indexOf("eid=")+4;
document.write(document.URL.substring(pos,document.URL.length));
</SCRIPT>
```

Example 2: Consider the HTML form:

```
<div id="myDiv">
Employee ID: <input type="text" id="eid"><br>
...
<button>Show results</button>
</div>
<div id="resultsDiv">
...
</div>
```

The following jQuery code segment reads an employee ID from the form, and displays it to the user.

```
$(document).ready(function(){
$("#myDiv").on("click", "button", function(){
var eid = $("#eid").val();
$("resultsDiv").append(eid);
...
});
});
```

These code examples operate correctly if the employee ID, from the text input with ID eid contains only standard alphanumeric text. If eid has a value that includes meta-characters or source code, then the code will be executed by the web browser as it displays the HTTP response.

Example 3: The following code shows an example of a DOM-based XSS within a React application:

```
let element = JSON.parse(getUntrustedInput());
ReactDOM.render(<App>
{element}
</App>);
```

In Example 3, if an attacker can control the entire JSON object retrieved from getUntrustedInput(), they may be able to make React render element as a component, and therefore can pass an object with dangerouslySetInnerHTML with their own controlled value, a typical cross-site scripting attack.

Initially these might not appear to be much of a vulnerability. After all, why would someone provide input containing malicious code to run on their own computer? The real danger is that an attacker will create the malicious URL, then use email or social engineering tricks to lure victims into visiting a link to the URL. When victims click the link, they unwittingly reflect the malicious content through the vulnerable web application back to their own computers. This mechanism of exploiting vulnerable web applications is known as Reflected XSS.

As the example demonstrates, XSS vulnerabilities are caused by code that includes unvalidated data in an HTTP response. There are three vectors by which an XSS attack can reach a victim:

- Data is read directly from the HTTP request and reflected back in the HTTP response. Reflected XSS exploits occur when an attacker causes a user to supply dangerous content to a vulnerable web application, which is then reflected back to the user and executed by the web browser. The most common mechanism for delivering malicious content is to include it as a parameter in a URL that is posted publicly or emailed directly to victims. URLs constructed in this manner constitute the core of many phishing schemes, whereby an attacker convinces victims to visit a URL that refers to a vulnerable site. After the site reflects the attacker's content back to the user, the content is executed and proceeds to transfer private information, such as cookies that may include session information, from the user's machine to the attacker or perform other nefarious activities.

- The application stores dangerous data in a database or other trusted data store. The dangerous data is subsequently read back into the application and included in dynamic content. Persistent XSS exploits occur when an attacker injects dangerous content into a data store that is later read and included in dynamic content. From an attacker's perspective, the optimal place to inject malicious content is in an area that is displayed to either many users or particularly interesting users. Interesting users typically have elevated privileges in the application or interact with sensitive data that is valuable to the attacker. If one of these users executes malicious content, the attacker may be able to perform privileged operations on behalf of the user or gain access to sensitive data belonging to the user.

- A source outside the application stores dangerous data in a database or other data store, and the dangerous data is subsequently read back into the application as trusted data and included in dynamic content.

Recommendations:

The solution to XSS is to ensure that validation occurs in the correct places and checks are made for the correct properties.

Since XSS vulnerabilities occur when an application includes malicious data in its output, one logical approach is to validate data immediately before it leaves the application. However, because web applications often have complex and intricate code for generating dynamic content, this method is prone to errors of omission (missing validation). An effective way to mitigate this risk is to also perform input validation for XSS.

Web applications must validate their input to prevent other vulnerabilities, such as SQL injection, so augmenting an application's existing input validation mechanism to include checks for XSS is generally relatively easy. Despite its value, input validation for XSS does not take the place of rigorous output validation. An application may accept input through a shared data store or other trusted source, and that data store may accept input from a source that does not perform adequate input validation. Therefore, the application cannot implicitly rely on the safety of this or any other data. This means the best way to prevent XSS vulnerabilities is to validate everything that enters the application and leaves the application destined for the user.

The most secure approach to validation for XSS is to create a whitelist of safe characters that are allowed to appear in HTTP content and accept input composed exclusively of characters in the approved set. For example, a valid username might only include alpha-numeric characters or a phone number might only include digits 0-9. However, this solution is often infeasible in web applications because many characters that have special meaning to the browser should still be considered valid input once they are encoded, such as a web design bulletin board that must accept HTML fragments from its users.

A more flexible, but less secure approach is known as blacklisting, which selectively rejects or escapes potentially dangerous characters before using the input. In order to form such a list, you first need to understand the set of characters that hold special meaning for web browsers. Although the HTML standard defines what characters have special meaning, many web browsers try to correct common mistakes in HTML and may treat other characters as special in certain contexts, which is why we do not encourage the use of blacklists as a means to prevent XSS. The CERT(R) Coordination Center at the Software Engineering Institute at Carnegie Mellon University provides the following details about special characters in various contexts [1]:

In the content of a block-level element (in the middle of a paragraph of text):

- "<" is special because it introduces a tag.

- "&" is special because it introduces a character entity.

- ">" is special because some browsers treat it as special, on the assumption that the author of the page intended to include an opening "<", but omitted it in error.

The following principles apply to attribute values:

- In attribute values enclosed with double quotes, the double quotes are special because they mark the end of the attribute value.

- In attribute values enclosed with single quote, the single quotes are special because they mark the end of the attribute value.

- In attribute values without any quotes, white-space characters, such as space and tab, are special.

- "&" is special when used with certain attributes, because it introduces a character entity.

In URLs, for example, a search engine might provide a link within the results page that the user can click to re-run the search. This can be implemented by encoding the search query inside the URL, which introduces additional special characters:

- Space, tab, and new line are special because they mark the end of the URL.

- "&" is special because it either introduces a character entity or separates CGI parameters.

- Non-ASCII characters (that is, everything greater than 127 in the ISO-8859-1 encoding) are not allowed in URLs, so they are considered to be special in this context.

- The "%" symbol must be filtered from input anywhere parameters encoded with HTTP escape sequences are decoded by server -side code. For example, "%" must be filtered if input such as "%68%65%6C%6C%6F" becomes "hello" when it appears on the web page in question.

Within the body of a <SCRIPT> </SCRIPT>:

- Semicolons, parentheses, curly braces, and new line characters should be filtered out in situations where text could be inserted directly into a pre-existing script tag.

Server-side scripts:

- Server-side scripts that convert any exclamation characters (!) in input to double-quote characters (") on output might require additional filtering.

Other possibilities:

- If an attacker submits a request in UTF-7, the special character '<' appears as '+ADw-' and may bypass filtering. If the output is included in a page that does not explicitly specify an encoding format, then some browsers try to intelligently identify the encoding based on the content (in this case, UTF-7).

After you identify the correct points in an application to perform validation for XSS attacks and what special characters the validation should consider, the next challenge is to identify how your validation handles special characters. If special characters are not considered valid input to the application, then you can reject any input that contains special characters as invalid. A second option in this situation is to remove special characters with filtering. However, filtering has the side effect of changing any visual representation of the filtered content and may be unacceptable in circumstances where the integrity of the input must be preserved for display.

If input containing special characters must be accepted and displayed accurately, validation must encode any special characters to remove their significance. A complete list of ISO 8859-1 encoded values for special characters is provided as part of the official HTML specification [2].

Many application servers attempt to limit an application's exposure to cross-site scripting vulnerabilities by providing implementations for the functions responsible for setting certain specific HTTP response content that perform validation for the characters essential to a cross-site scripting attack. Do not rely on the server running your application to make it secure. When an application is developed there are no guarantees about what application servers it will run on during its lifetime. As standards and known exploits evolve, there are no guarantees that application servers will also stay in sync.

Tips:

1. The Fortify Secure Coding Rulepacks warn about SQL Injection and Access Control: Database issues when untrusted data is written to a database and also treat the database as a source of untrusted data, which can lead to XSS vulnerabilities. If the database is a trusted resource in your environment, use custom filters to filter out dataflow issues that include the DATABASE taint flag or originate from database sources. Nonetheless, it is often still a good idea to validate everything read from the database.

2. Even though URL encoding untrusted data protects against many XSS attacks, some browsers (specifically, Internet Explorer 6 and 7 and possibly others) automatically decode content at certain locations within the Document Object Model (DOM) prior to passing it to the JavaScript interpreter. To reflect this danger, the rulepacks no longer treat URL encoding routines as sufficient to protect against cross-site scripting. Data values that are URL encoded and subsequently output will cause Fortify to report Cross-Site Scripting: Poor Validation vulnerabilities.

3. Older versions of React are more susceptible to cross-site scripting attacks by controlling an entire component. Newer versions use Symbols to identify a React component, which prevents the exploit, however older browsers that do not have Symbol support (natively, or through polyfills), such as all versions of Internet Explorer, are still vulnerable. Other types of cross-site scripting attacks are valid for all browsers and versions of React.

appointments.jsp, line 18 (Cross-Site Scripting: DOM)

Fortify Priority:	Critical	**Folder**	Critical
Kingdom:	Input Validation and Representation		
Abstract:	The method addNewAppointment() in appointments.jsp sends unvalidated data to a web browser on line 18, which can result in the browser executing malicious code.		

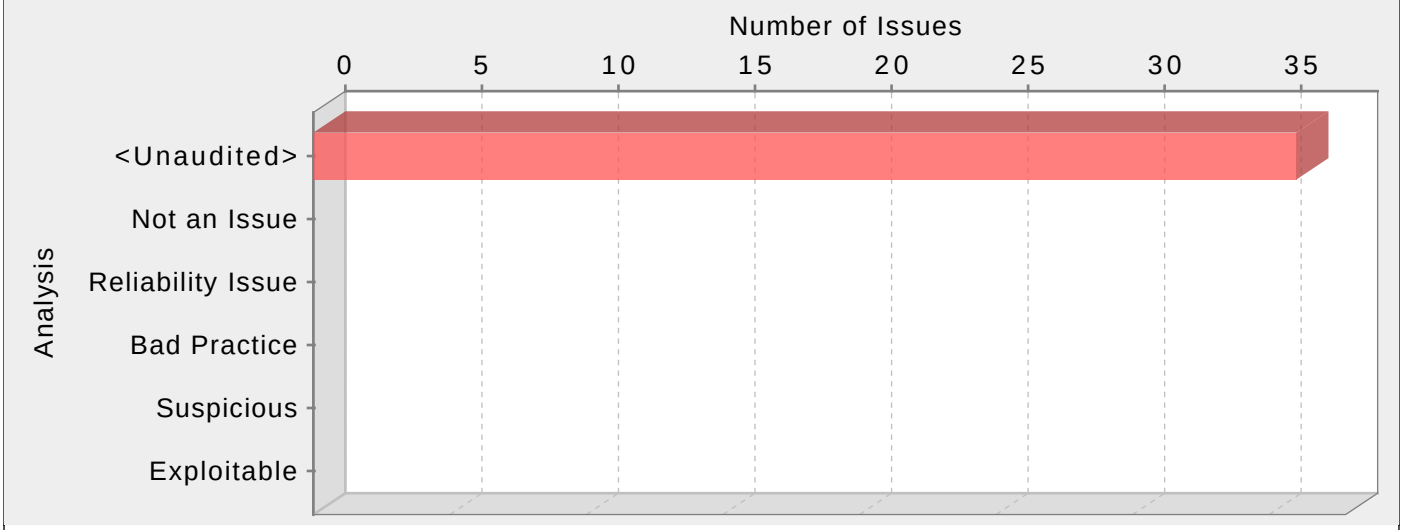
Source: appointments.jsp:17 Read value()

```
15    //Navigate to appointmentForm.form
16    function addNewAppointment(){
17     var patientId = document.getElementById("patientId").value;
18     window.location =
"module/appointmentscheduling/appointmentForm.form?patientId="+patientId;
19    }
```

Sink: appointments.jsp:18 Assignment to window.location()

```
16    function addNewAppointment(){
17     var patientId = document.getElementById("patientId").value;
18     window.location =
"module/appointmentscheduling/appointmentForm.form?patientId="+patientId;
19    }
20    //On the page load updates necessary stuff
```

Category: Cross-Site Scripting: Reflected (36 Issues)



Abstract:

The method _jspService() in appointmentForm.jsp sends unvalidated data to a web browser on line 113, which can result in the browser executing malicious code.

Explanation:

Cross-site scripting (XSS) vulnerabilities occur when:

1. Data enters a web application through an untrusted source. In the case of reflected XSS, the untrusted source is typically a web request, while in the case of persisted (also known as stored) XSS it is typically a database or other back-end data store.

2. The data is included in dynamic content that is sent to a web user without being validated.

The malicious content sent to the web browser often takes the form of a segment of JavaScript, but may also include HTML, Flash or any other type of code that the browser executes. The variety of attacks based on XSS is almost limitless, but they commonly include transmitting private data like cookies or other session information to the attacker, redirecting the victim to web content controlled by the attacker, or performing other malicious operations on the user's machine under the guise of the vulnerable site.

Example 1: The following JSP code segment reads an employee ID, eid, from an HTTP request and displays it to the user.

```
<% String eid = request.getParameter("eid"); %>
...
Employee ID: <%= eid %>
```

The code in this example operates correctly if eid contains only standard alphanumeric text. If eid has a value that includes meta-characters or source code, then the code will be executed by the web browser as it displays the HTTP response.

Initially this might not appear to be much of a vulnerability. After all, why would someone enter a URL which causes malicious code to run on their own computer? The real danger is that an attacker will create the malicious URL, then use email or social engineering tricks to lure victims into visiting a link to the URL. When victims click the link, they unwittingly reflect the malicious content through the vulnerable web application back to their own computers. This mechanism of exploiting vulnerable web applications is known as Reflected XSS.

Example 2: The following JSP code segment queries a database for an employee with a given ID and prints the corresponding employee's name.

```
<%...
Statement stmt = conn.createStatement();
ResultSet rs = stmt.executeQuery("select * from emp where id="+eid);
if (rs != null) {
rs.next();
String name = rs.getString("name");
}
%>
Employee Name: <%= name %>
```

As in Example 1, this code functions correctly when the values of name are well-behaved, but it does nothing to prevent exploits if they are not. Again, this code can appear less dangerous because the value of name is read from a database, whose contents are apparently managed by the application. However, if the value of name originates from user-supplied data, then the database can be a conduit for malicious content. Without proper input validation on all data stored in the database, an attacker may execute malicious commands in the user's web browser. This type of exploit, known as Persistent (or Stored) XSS, is particularly insidious because the indirection caused by the data store makes it more difficult to identify the threat and increases the possibility that the attack will affect multiple users. XSS got its start in this form with web sites that offered a "guestbook" to visitors. Attackers would include JavaScript in their guestbook entries, and all subsequent visitors to the guestbook page would execute the malicious code.

Some think that in the mobile world, classic web application vulnerabilities, such as cross-site scripting, do not make sense -- why would the user attack themself? However, keep in mind that the essence of mobile platforms is applications that are downloaded from various sources and run alongside each other on the same device. The likelihood of running a piece of malware next to a banking application is high, which necessitates expanding the attack surface of mobile applications to include inter-process communication.

Example 3: The following code enables JavaScript in Android's WebView (by default, JavaScript is disabled) and loads a page based on the value received from an Android intent.

```
...
WebView webview = (WebView) findViewById(R.id.webview);
webview.getSettings().setJavaScriptEnabled(true);
String url = this.getIntent().getExtras().getString("url");
webview.loadUrl(url);
...
```

If the value of url starts with javascript:, JavaScript code that follows will execute within the context of the web page inside WebView.

As the examples demonstrate, XSS vulnerabilities are caused by code that includes unvalidated data in an HTTP response. There are three vectors by which an XSS attack can reach a victim:

- As in Example 1, data is read directly from the HTTP request and reflected back in the HTTP response. Reflected XSS exploits occur when an attacker causes a user to supply dangerous content to a vulnerable web application, which is then reflected back to the user and executed by the web browser. The most common mechanism for delivering malicious content is to include it as a parameter in a URL that is posted publicly or emailed directly to victims. URLs constructed in this manner constitute the core of many phishing schemes, whereby an attacker convinces victims to visit a URL that refers to a vulnerable site. After the site reflects the attacker's content back to the user, the content is executed and proceeds to transfer private information, such as cookies that may include session information, from the user's machine to the attacker or perform other nefarious activities.

- As in Example 2, the application stores dangerous data in a database or other trusted data store. The dangerous data is subsequently read back into the application and included in dynamic content. Persistent XSS exploits occur when an attacker injects dangerous content into a data store that is later read and included in dynamic content. From an attacker's perspective, the optimal place to inject malicious content is in an area that is displayed to either many users or particularly interesting users. Interesting users typically have elevated privileges in the application or interact with sensitive data that is valuable to the attacker. If one of these users executes malicious content, the attacker may be able to perform privileged operations on behalf of the user or gain access to sensitive data belonging to the user.

- As in Example 3, a source outside the application stores dangerous data in a database or other data store, and the dangerous data is subsequently read back into the application as trusted data and included in dynamic content.

A number of modern web frameworks provide mechanisms to perform user input validation (including Struts and Spring MVC). To highlight the unvalidated sources of input, the rulepacks dynamically re-prioritize the issues reported by Fortify Static Code Analyzer by lowering their probability of exploit and providing pointers to the supporting evidence whenever the framework validation mechanism is in use. We refer to this feature as Context-Sensitive Ranking. To further assist the Fortify user with the auditing process, the Fortify Software Security Research group makes available the Data Validation project template that groups the issues into folders based on the validation mechanism applied to their source of input.

Recommendations:

The solution to XSS is to ensure that validation occurs in the correct places and checks are made for the correct properties.

Since XSS vulnerabilities occur when an application includes malicious data in its output, one logical approach is to validate data immediately before it leaves the application. However, because web applications often have complex and intricate code for generating dynamic content, this method is prone to errors of omission (missing validation). An effective way to mitigate this risk is to also perform input validation for XSS.

Web applications must validate their input to prevent other vulnerabilities, such as SQL injection, so augmenting an application's existing input validation mechanism to include checks for XSS is generally relatively easy. Despite its value, input validation for XSS does not take the place of rigorous output validation. An application may accept input through a shared data store or other trusted source, and that data store may accept input from a source that does not perform adequate input validation. Therefore, the application cannot implicitly rely on the safety of this or any other data. This means the best way to prevent XSS vulnerabilities is to validate everything that enters the application and leaves the application destined for the user.

The most secure approach to validation for XSS is to create a whitelist of safe characters that are allowed to appear in HTTP content and accept input composed exclusively of characters in the approved set. For example, a valid username might only include alpha-numeric characters or a phone number might only include digits 0-9. However, this solution is often infeasible in web applications because many characters that have special meaning to the browser should still be considered valid input once they are encoded, such as a web design bulletin board that must accept HTML fragments from its users.

A more flexible, but less secure approach is known as blacklisting, which selectively rejects or escapes potentially dangerous characters before using the input. In order to form such a list, you first need to understand the set of characters that hold special meaning for web browsers. Although the HTML standard defines what characters have special meaning, many web browsers try to correct common mistakes in HTML and may treat other characters as special in certain contexts, which is why we do not encourage the use of blacklists as a means to prevent XSS. The CERT(R) Coordination Center at the Software Engineering Institute at Carnegie Mellon University provides the following details about special characters in various contexts [1]:

In the content of a block-level element (in the middle of a paragraph of text):

- "<" is special because it introduces a tag.

- "&" is special because it introduces a character entity.

- ">" is special because some browsers treat it as special, on the assumption that the author of the page intended to include an opening "<", but omitted it in error.

The following principles apply to attribute values:

- In attribute values enclosed with double quotes, the double quotes are special because they mark the end of the attribute value.

- In attribute values enclosed with single quote, the single quotes are special because they mark the end of the attribute value.

- In attribute values without any quotes, white-space characters, such as space and tab, are special.

- "&" is special when used with certain attributes, because it introduces a character entity.

In URLs, for example, a search engine might provide a link within the results page that the user can click to re-run the search. This can be implemented by encoding the search query inside the URL, which introduces additional special characters:

- Space, tab, and new line are special because they mark the end of the URL.

- "&" is special because it either introduces a character entity or separates CGI parameters.

- Non-ASCII characters (that is, everything greater than 127 in the ISO-8859-1 encoding) are not allowed in URLs, so they are considered to be special in this context.

- The "%" symbol must be filtered from input anywhere parameters encoded with HTTP escape sequences are decoded by server -side code. For example, "%" must be filtered if input such as "%68%65%6C%6C%6F" becomes "hello" when it appears on the web page in question.

Within the body of a <SCRIPT> </SCRIPT>:

- Semicolons, parentheses, curly braces, and new line characters should be filtered out in situations where text could be inserted directly into a pre-existing script tag.

Server-side scripts:

- Server-side scripts that convert any exclamation characters (!) in input to double-quote characters (") on output might require additional filtering.

Other possibilities:

- If an attacker submits a request in UTF-7, the special character '<' appears as '+ADw-' and may bypass filtering. If the output is included in a page that does not explicitly specify an encoding format, then some browsers try to intelligently identify the encoding based on the content (in this case, UTF-7).

After you identify the correct points in an application to perform validation for XSS attacks and what special characters the validation should consider, the next challenge is to identify how your validation handles special characters. If special characters are not considered valid input to the application, then you can reject any input that contains special characters as invalid. A second option in this situation is to remove special characters with filtering. However, filtering has the side effect of changing any visual representation of the filtered content and may be unacceptable in circumstances where the integrity of the input must be preserved for display.

If input containing special characters must be accepted and displayed accurately, validation must encode any special characters to remove their significance. A complete list of ISO 8859-1 encoded values for special characters is provided as part of the official HTML specification [2].

Many application servers attempt to limit an application's exposure to cross-site scripting vulnerabilities by providing implementations for the functions responsible for setting certain specific HTTP response content that perform validation for the characters essential to a cross-site scripting attack. Do not rely on the server running your application to make it secure. When an application is developed there are no guarantees about what application servers it will run on during its lifetime. As standards and known exploits evolve, there are no guarantees that application servers will also stay in sync.

Tips:

1. The Fortify Secure Coding Rulepacks warn about SQL Injection and Access Control: Database issues when untrusted data is written to a database and also treat the database as a source of untrusted data, which can lead to XSS vulnerabilities. If the database is a trusted resource in your environment, use custom filters to filter out dataflow issues that include the DATABASE taint flag or originate from database sources. Nonetheless, it is often still a good idea to validate everything read from the database.

2. Even though URL encoding untrusted data protects against many XSS attacks, some browsers (specifically, Internet Explorer 6 and 7 and possibly others) automatically decode content at certain locations within the Document Object Model (DOM) prior to passing it to the JavaScript interpreter. To reflect this danger, the rulepacks no longer treat URL encoding routines as sufficient to protect against cross-site scripting. Data values that are URL encoded and subsequently output will cause Fortify to report Cross-Site Scripting: Poor Validation vulnerabilities.

3. Fortify RTA adds protection against this category.

appointmentForm.jsp, line 113 (Cross-Site Scripting: Reflected)

Fortify Priority:	Critical	**Folder**	Critical
Kingdom:	Input Validation and Representation		
Abstract:	The method _jspService() in appointmentForm.jsp sends unvalidated data to a web browser on line 113, which can result in the browser executing malicious code.		

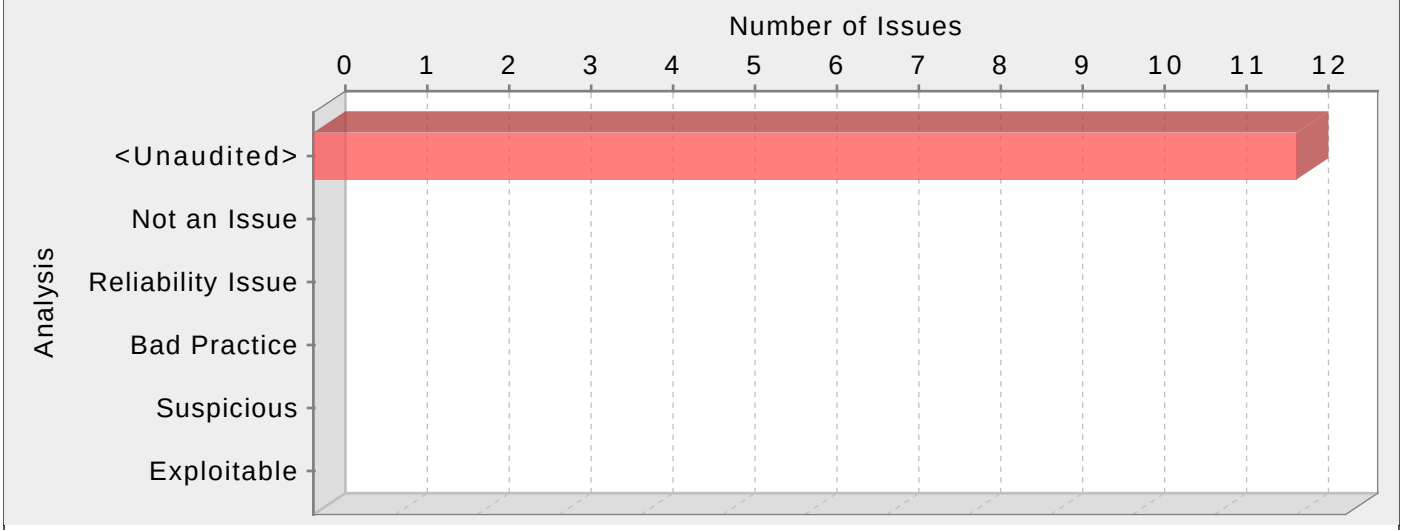
Source: appointmentForm.jsp:113 javax.servlet.ServletRequest.getParameter()

```
111                         "<input type=\"checkbox\" name=\"includeFull\" value=\"true\"
onchange='this.form.submit();' ${(param.includeFull=='true') ? 'checked' : ''}>"+
112                          "<spring:message code='appointmentscheduling.Appointment.create.label.showFu
113                          "<br/><c:if test='${param.includeFull==\'true\'}'>"+
114                          "<div id='slotIndex'> <img
src='${pageContext.request.contextPath}/moduleResources/appointmentscheduling/Images/i
ndex_fullTimeslot.png' alt='<spring:message
code='appointmentscheduling.Appointment.create.lbl.fullSlot'/>'/>"+
115                          " = <spring:message
code='appointmentscheduling.Appointment.create.lbl.fullSlot'/></div></c:if>"+
```

Sink: appointmentForm.jsp:113 javax.servlet.jsp.JspWriter.print()

```
111                         "<input type=\"checkbox\" name=\"includeFull\" value=\"true\"
onchange='this.form.submit();' ${(param.includeFull=='true') ? 'checked' : ''}>"+
112                          "<spring:message code='appointmentscheduling.Appointment.create.label.showFu
113                          "<br/><c:if test='${param.includeFull==\'true\'}'>"+
114                          "<div id='slotIndex'> <img
src='${pageContext.request.contextPath}/moduleResources/appointmentscheduling/Images/i
ndex_fullTimeslot.png' alt='<spring:message
code='appointmentscheduling.Appointment.create.lbl.fullSlot'/>'/>"+
115                          " = <spring:message
code='appointmentscheduling.Appointment.create.lbl.fullSlot'/></div></c:if>"+
```

Category: Header Manipulation: Cookies (12 Issues)



Abstract:

The method _fnCreateCookie() in jquery.dataTables.js includes unvalidated data in an HTTP cookie on line 4554. This enables Cookie manipulation attacks and can lead to other HTTP Response header manipulation attacks like: cache-poisoning, cross-site scripting, cross-user defacement, page hijacking or open redirect.

Explanation:

Cookie Manipulation vulnerabilities occur when:

1. Data enters a web application through an untrusted source, most frequently an HTTP request.

2. The data is included in an HTTP cookie sent to a web user without being validated.

As with many software security vulnerabilities, cookie manipulation is a means to an end, not an end in itself. At its root, the vulnerability is straightforward: an attacker passes malicious data to a vulnerable application, and the application includes the data in an HTTP cookie.

Cookie Manipulation: When combined with attacks like cross-site request forgery, attackers may change, add to, or even overwrite a legitimate user's cookies.

Being an HTTP Response header, Cookie manipulation attacks can also lead to other types of attacks like:

HTTP Response Splitting:

One of the most common Header Manipulation attacks is HTTP Response Splitting. To mount a successful HTTP Response Splitting exploit, the application must allow input that contains CR (carriage return, also given by %0d or \r) and LF (line feed, also given by %0a or \n)characters into the header. These characters not only give attackers control of the remaining headers and body of the response the application intends to send, but also allows them to create additional responses entirely under their control.

Many of today's modern application servers will prevent the injection of malicious characters into HTTP headers. For example, recent versions of Apache Tomcat will throw an IllegalArgumentException if you attempt to set a header with prohibited characters. If your application server prevents setting headers with new line characters, then your application is not vulnerable to HTTP Response Splitting. However, solely filtering for new line characters can leave an application vulnerable to Cookie Manipulation or Open Redirects, so care must still be taken when setting HTTP headers with user input.

Example: The following code segment reads the name of the author of a weblog entry, author, from an HTTP request and sets it in a cookie header of an HTTP response.

```
author = form.author.value;
...
document.cookie = "author=" + author + ";expires="+cookieExpiration;
...
```

Assuming a string consisting of standard alpha-numeric characters, such as "Jane Smith", is submitted in the request the HTTP response including this cookie might take the following form:

```
HTTP/1.1 200 OK
...
Set-Cookie: author=Jane Smith
...
```

However, because the value of the cookie is formed of unvalidated user input the response will only maintain this form if the value submitted for AUTHOR_PARAM does not contain any CR and LF characters. If an attacker submits a malicious string, such as "Wiley Hacker\r\nHTTP/1.1 200 OK\r\n...", then the HTTP response would be split into two responses of the following form:

HTTP/1.1 200 OK

...
Set-Cookie: author=Wiley Hacker

HTTP/1.1 200 OK

...

Clearly, the second response is completely controlled by the attacker and can be constructed with any header and body content desired. The ability of attacker to construct arbitrary HTTP responses permits a variety of resulting attacks, including: cross-user defacement, web and browser cache poisoning, cross-site scripting, and page hijacking.

Cross-User Defacement: An attacker will be able to make a single request to a vulnerable server that will cause the server to create two responses, the second of which may be misinterpreted as a response to a different request, possibly one made by another user sharing the same TCP connection with the server. This can be accomplished by convincing the user to submit the malicious request themselves, or remotely in situations where the attacker and the user share a common TCP connection to the server, such as a shared proxy server. In the best case, an attacker may leverage this ability to convince users that the application has been hacked, causing users to lose confidence in the security of the application. In the worst case, an attacker may provide specially crafted content designed to mimic the behavior of the application but redirect private information, such as account numbers and passwords, back to the attacker.

Cache Poisoning: The impact of a maliciously constructed response can be magnified if it is cached either by a web cache used by multiple users or even the browser cache of a single user. If a response is cached in a shared web cache, such as those commonly found in proxy servers, then all users of that cache will continue receive the malicious content until the cache entry is purged. Similarly, if the response is cached in the browser of an individual user, then that user will continue to receive the malicious content until the cache entry is purged, although only the user of the local browser instance will be affected.

Cross-Site Scripting: Once attackers have control of the responses sent by an application, they have a choice of a variety of malicious content to provide users. Cross-site scripting is common form of attack where malicious JavaScript or other code included in a response is executed in the user's browser. The variety of attacks based on XSS is almost limitless, but they commonly include transmitting private data like cookies or other session information to the attacker, redirecting the victim to web content controlled by the attacker, or performing other malicious operations on the user's machine under the guise of the vulnerable site. The most common and dangerous attack vector against users of a vulnerable application uses JavaScript to transmit session and authentication information back to the attacker who can then take complete control of the victim's account.

Page Hijacking: In addition to using a vulnerable application to send malicious content to a user, the same root vulnerability can also be leveraged to redirect sensitive content generated by the server and intended for the user to the attacker instead. By submitting a request that results in two responses, the intended response from the server and the response generated by the attacker, an attacker may cause an intermediate node, such as a shared proxy server, to misdirect a response generated by the server for the user to the attacker. Because the request made by the attacker generates two responses, the first is interpreted as a response to the attacker's request, while the second remains in limbo. When the user makes a legitimate request through the same TCP connection, the attacker's request is already waiting and is interpreted as a response to the victim's request. The attacker then sends a second request to the server, to which the proxy server responds with the server generated request intended for the victim, thereby compromising any sensitive information in the headers or body of the response intended for the victim.

Open Redirect: Allowing unvalidated input to control the URL used in a redirect can aid phishing attacks.

Recommendations:

The solution to cookie manipulation is to ensure that input validation occurs in the correct places and checks for the correct properties.

Since Header Manipulation vulnerabilities like cookie manipulation occur when an application includes malicious data in its output, one logical approach is to validate data immediately before it leaves the application. However, because web applications often have complex and intricate code for generating responses dynamically, this method is prone to errors of omission (missing validation). An effective way to mitigate this risk is to also perform input validation for Header Manipulation.

Web applications must validate their input to prevent other vulnerabilities, such as SQL injection, so augmenting an application's existing input validation mechanism to include checks for Header Manipulation is generally relatively easy. Despite its value, input validation for Header Manipulation does not take the place of rigorous output validation. An application may accept input through a shared data store or other trusted source, and that data store may accept input from a source that does not perform adequate input validation. Therefore, the application cannot implicitly rely on the safety of this or any other data. This means the best way to prevent Header Manipulation vulnerabilities is to validate everything that enters the application or leaves the application destined for the user.

The most secure approach to validation for Header Manipulation is to create a whitelist of safe characters that are allowed to appear in HTTP response headers and accept input composed exclusively of characters in the approved set. For example, a valid name might only include alpha-numeric characters or an account number might only include digits 0-9.

A more flexible, but less secure approach is known as blacklisting, which selectively rejects or escapes potentially dangerous characters before using the input. In order to form such a list, you first need to understand the set of characters that hold special meaning in HTTP response headers. Although the CR and LF characters are at the heart of an HTTP response splitting attack, other characters, such as ':' (colon) and '=' (equal), have special meaning in response headers as well.

After you identify the correct points in an application to perform validation for Header Manipulation attacks and what special characters the validation should consider, the next challenge is to identify how your validation handles special characters. The application should reject any input destined to be included in HTTP response headers that contains special characters, particularly CR and LF, as invalid.

Many application servers attempt to limit an application's exposure to HTTP response splitting vulnerabilities by providing implementations for the functions responsible for setting HTTP headers and cookies that perform validation for the characters essential to an HTTP response splitting attack. Do not rely on the server running your application to make it secure. When an application is developed there are no guarantees about what application servers it will run on during its lifetime. As standards and known exploits evolve, there are no guarantees that application servers will also stay in sync.

jquery.dataTables.js, line 4554 (Header Manipulation: Cookies)

Fortify Priority:	High	**Folder**	High
Kingdom:	Input Validation and Representation		
Abstract:	The method _fnCreateCookie() in jquery.dataTables.js includes unvalidated data in an HTTP cookie on line 4554. This enables Cookie manipulation attacks and can lead to other HTTP Response header manipulation attacks like: cache-poisoning, cross-site scripting, cross-user defacement, page hijacking or open redirect.		

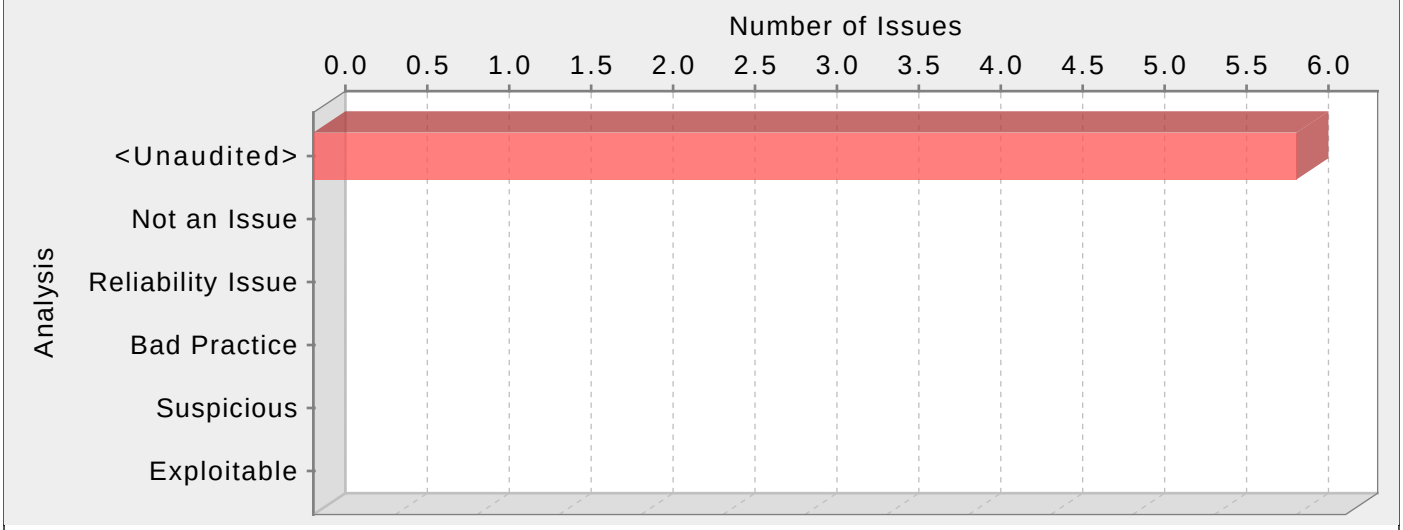
Source: jquery.dataTables.js:4493 Read window.location()

```
4491      * patch to use at least some of the path
4492      */
4493     var aParts = window.location.pathname.split('/');
4494     var sNameFile = sName + '_' + aParts.pop().replace(/[\/:]/g,"").toLowerCase();
4495     var sFullCookie, oData;
```

Sink: jquery.dataTables.js:4554 Assignment to document.cookie()

```
4552
4553       var old = aOldCookies.pop();
4554       document.cookie = old.name+"=; expires=Thu, 01-Jan-1970 00:00:01 GMT; path="+
4555        aParts.join('/') + "/";
4556      }
```

Category: Dynamic Code Evaluation: Code Injection (6 Issues)



Abstract:

The file jquery.dataTables.js interprets unvalidated user input as source code on line 4527. Interpreting user-controlled instructions at run-time can allow attackers to execute malicious code.

Explanation:

Many modern programming languages allow dynamic interpretation of source instructions. This capability allows programmers to perform dynamic instructions based on input received from the user. Code injection vulnerabilities occur when the programmer incorrectly assumes that instructions supplied directly from the user will perform only innocent operations, such as performing simple calculations on active user objects or otherwise modifying the user's state. However, without proper validation, a user might specify operations the programmer does not intend.

Example: In this classic code injection example, the application implements a basic calculator that allows the user to specify commands for execution.

...
userOp = form.operation.value;
calcResult = eval(userOp);
...

The program behaves correctly when the operation parameter is a benign value, such as "8 + 7 * 2", in which case the calcResult variable is assigned a value of 22. However, if an attacker specifies languages operations that are both valid and malicious, those operations would be executed with the full privilege of the parent process. Such attacks are even more dangerous when the underlying language provides access to system resources or allows execution of system commands. In the case of JavaScript, the attacker may utilize this vulnerability to perform a cross-site scripting attack.

Recommendations:

Avoid dynamic code interpretation whenever possible. If your program's functionality requires code to be interpreted dynamically, the likelihood of attack can be minimized by constraining the code your program will execute dynamically as much as possible, limiting it to an application- and context-specific subset of the base programming language.

If dynamic code execution is required, unvalidated user input should never be directly executed and interpreted by the application. Instead, use a level of indirection: create a list of legitimate operations and data objects that users are allowed to specify, and only allow users to select from the list. With this approach, input provided by users is never executed directly.

jquery.dataTables.js, line 4527 (Dynamic Code Evaluation: Code Injection)

Fortify Priority:	Critical	**Folder**	Critical
Kingdom:	Input Validation and Representation		
Abstract:	The file jquery.dataTables.js interprets unvalidated user input as source code on line 4527. Interpreting user-controlled instructions at run-time can allow attackers to execute malicious code.		

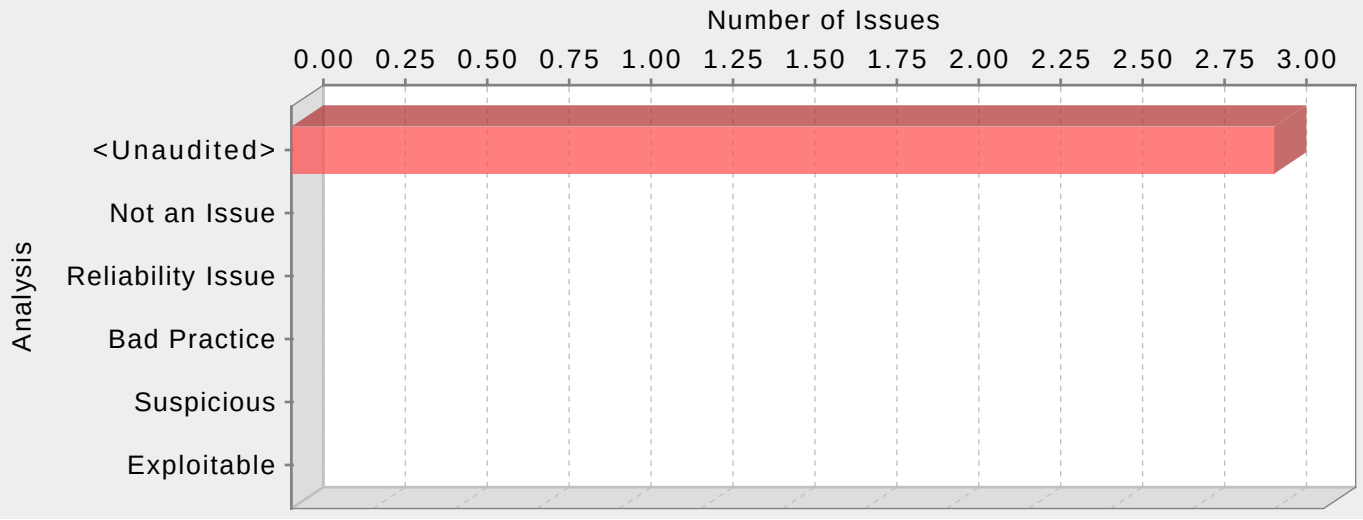
Source: jquery.dataTables.js:4514 Read document.cookie()

```
4512        */
4513       var
4514        aCookies =document.cookie.split(';'),
4515        iNewCookieLen = sFullCookie.split(';')[0].length,
4516        aOldCookies = [];
```

Sink: jquery.dataTables.js:4527 eval()

```
4525                        var aSplitCookie = aCookies[i].split('=');
4526                        try {
4527                         oData = eval( '('+decodeURIComponent(aSplitCookie[1])+')' );
4528
4529                         if ( oData && oData.iCreate )
```

Category: Key Management: Hardcoded Encryption Key (3 Issues)



Abstract:

Hardcoded encryption keys can compromise security in a way that cannot be easily remedied.

Explanation:

It is never a good idea to hardcode an encryption key because it allows all of the project's developers to view the encryption key, and makes fixing the problem extremely difficult. After the code is in production, a software patch is required to change the encryption key. If the account that is protected by the encryption key is compromised, the owners of the system must choose between security and availability.

Example 1: The following code uses a hardcoded encryption key:

```
...
var crypto = require('crypto');
var encryptionKey = "lakdsljkalkjlksdfkl";
var algorithm = 'aes-256-ctr';
var cipher = crypto.createCipher(algorithm, encryptionKey);
...
```

Anyone with access to the code has access to the encryption key. After the application has shipped, there is no way to change the encryption key unless the program is patched. An employee with access to this information can use it to break into the system. If attackers had access to the executable for the application, they could extract the encryption key value.

Recommendations:

Encryption keys should never be hardcoded and should be obfuscated and managed in an external source. Storing encryption keys in plain text anywhere on the system allows anyone with sufficient permissions to read and potentially misuse the encryption key.

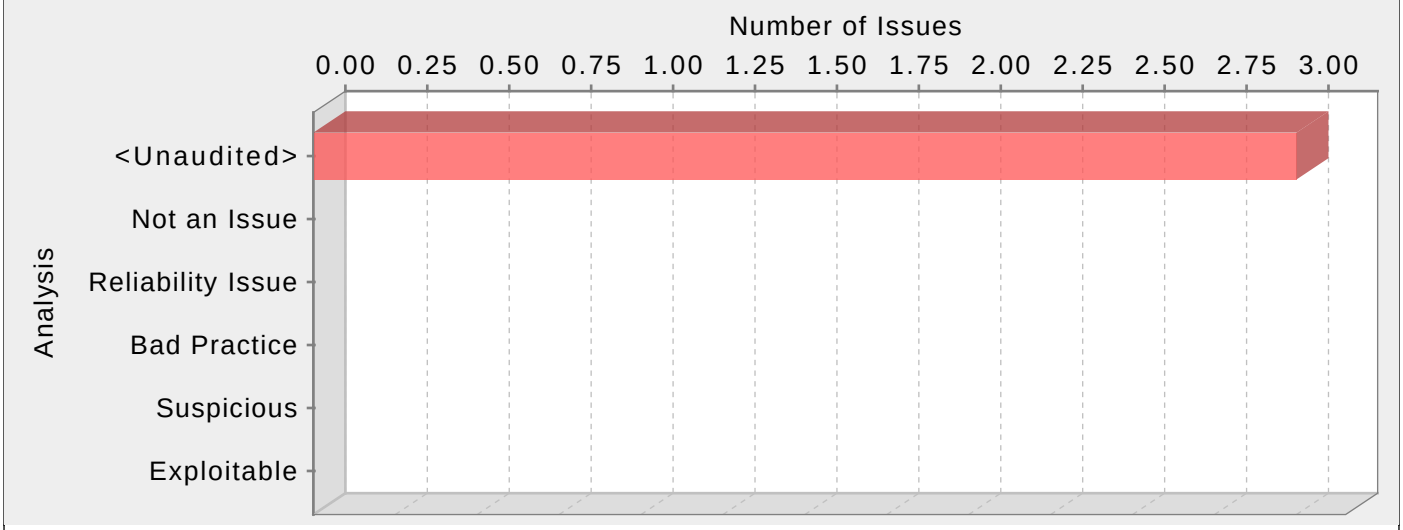
jquery.jeditable.js, line 515 (Key Management: Hardcoded Encryption Key)

Fortify Priority:	Critical	**Folder**	Critical
Kingdom:	Security Features		
Abstract:	Hardcoded encryption keys can compromise security in a way that cannot be easily remedied.		

Sink: jquery.jeditable.js:515 Operation()

```
513                         continue;
514                     }
515                     if ('selected' == key) {
516                         continue;
517                     }
```

Category: Open Redirect (3 Issues)



Abstract:

The file appointments.jsp passes unvalidated data to an HTTP redirect function on line 18. Allowing unvalidated input to control the URL used in a redirect can aid phishing attacks.

Explanation:

Redirects allow web applications to direct users to different pages within the same application or to external sites. Applications utilize redirects to aid in site navigation and, in some cases, to track how users exit the site. Open redirect vulnerabilities occur when a web application redirects clients to any arbitrary URL that can be controlled by an attacker.

Attackers may utilize open redirects to trick users into visiting a URL to a trusted site and redirecting them to a malicious site. By encoding the URL, an attacker is able to make it more difficult for end-users to notice the malicious destination of the redirect, even when it is passed as a URL parameter to the trusted site. Open redirects are often abused as part of phishing scams to harvest sensitive end-user data.

Example 1: The following JavaScript code instructs the user's browser to open a URL read from the dest request parameter when a user clicks the link.

```
...
strDest = form.dest.value;
window.open(strDest,"myresults");
...
```

If a victim received an email instructing them to follow a link to "http://trusted.example.com/ecommerce/redirect.asp?dest=www.wilyhacker.com", the user would likely click on the link believing they would be transferred to the trusted site. However, when the victim clicks the link, the code in Example 1 will redirect the browser to "http://www.wilyhacker.com".

Many users have been educated to always inspect URLs they receive in emails to make sure the link specifies a trusted site they know. However, if the attacker Hex encoded the destination url as follows:

"http://trusted.example.com/ecommerce/redirect.asp?dest=%77%69%6C%79%68%61%63%6B%65%72%2E%63%6F%6D"

then even a savvy end-user may be fooled into following the link.

Recommendations:

Unvalidated user input should not be allowed to control the destination URL in a redirect. Instead, use a level of indirection: create a list of legitimate URLs that users are allowed to specify, and only allow users to select from the list. With this approach, input provided by users is never used directly to specify a URL for redirects.

Example 2: The following code references an array populated with valid URLs. The link the user clicks passes in the array index that corresponds to the desired URL.

```
...
strDest = form.dest.value;
if((strDest.value != null)||(strDest.value.length!=0))
{
if((strDest >= 0) && (strDest <= strURLArray.length -1 ))
{
strFinalURL = strURLArray[strDest];
window.open(strFinalURL,"myresults");
```

```
}
}
...
```

In some situations this approach is impractical because the set of legitimate URLs is too large or too hard to keep track of. In such cases, use a similar approach to restrict the domains that users can be redirected to, which can at least prevent attackers from sending users to malicious external sites.

appointments.jsp, line 18 (Open Redirect)

Fortify Priority:	Critical	**Folder**	Critical
Kingdom:	Input Validation and Representation		
Abstract:	The file appointments.jsp passes unvalidated data to an HTTP redirect function on line 18. Allowing unvalidated input to control the URL used in a redirect can aid phishing attacks.		

Source: appointments.jsp:17 Read value()

```
15    //Navigate to appointmentForm.form
16    function addNewAppointment(){
17     var patientId = document.getElementById("patientId").value;
18     window.location =
"module/appointmentscheduling/appointmentForm.form?patientId="+patientId;
19    }
```

Sink: appointments.jsp:18 Assignment to window.location()

```
16    function addNewAppointment(){
17     var patientId = document.getElementById("patientId").value;
18     window.location =
"module/appointmentscheduling/appointmentForm.form?patientId="+patientId;
19    }
20    //On the page load updates necessary stuff
```

Detailed Project Summary

Files Scanned

Code base location: /srv/openmrs_code/org/openmrs/module/appointmentscheduling
Files Scanned:
.travis.yml yaml Dec 13, 2019 12:56:58 PM
OpenMRSFormatter.xml xml 27.9 KB Dec 13, 2019 12:56:58 PM
api/pom.xml xml 1.4 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/Appointment.java java 56 Lines 5.4 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/AppointmentActivator.java java 8 Lines 1.7 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/AppointmentBlock.java java 24 Lines 2.6 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/AppointmentRequest.java java 27 Lines 3.3 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/AppointmentSchedulingConstants.java java 2 Lines Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/AppointmentStatusHistory.java java 19 Lines 2.4 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/AppointmentType.java java 16 Lines 2 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/AppointmentUtils.java java 18 Lines 1.7 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/StudentT.java java 122 Lines 9.8 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/TimeFrameUnits.java java Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/TimeSlot.java java 18 Lines 2.1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/api/AppointmentService.java java 36.7 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/api/db/AppointmentBlockDAO.java java 1.1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/api/db/AppointmentDAO.java java 3.7 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/api/db/AppointmentRequestDAO.java java Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/api/db/AppointmentStatusHistoryDAO.java java 2.4 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/api/db/AppointmentTypeDAO.java java 1.4 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/api/db/SingleClassDAO.java java Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/api/db/TimeSlotDAO.java java 2 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/api/db/hibernate/HibernateAppointmentBlockDAO.java java 27 Lines 6.1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/api/db/hibernate/HibernateAppointmentDAO.java java 61 Lines 8.7 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/api/db/hibernate/HibernateAppointmentRequestDAO.java java 3 Lines 1.6 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/api/db/hibernate/HibernateAppointmentStatusHistoryDAO.java java 17 Lines 3.2 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/api/db/hibernate/HibernateAppointmentTypeDAO.java java 10 Lines 2.7 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/api/db/hibernate/HibernateSingleClassDAO.java java 18 Lines 3.9

KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/api/db/hibernate/HibernateTimeSlotDAO.java java 15 Lines 2.7 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/api/impl/AppointmentServiceImpl.java java 331 Lines 38.1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/exception/TimeSlotFullException.java java 3 Lines Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/context/AppointmentEvaluationContext.java java 12 Lines 2.6 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/AppointmentData.java java 1 Lines Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/AppointmentDataUtil.java java 16 Lines 2.6 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/EvaluatedAppointmentData.java java 8 Lines 1.5 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/definition/AppointmentCancelReasonDataDefinition.java java 4 Lines 1.1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/definition/AppointmentDataDefinition.java java Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/definition/AppointmentEndDateDataDefinition.java java 4 Lines 1.1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/definition/AppointmentLocationDataDefinition.java java 4 Lines 1.1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/definition/AppointmentProviderDataDefinition.java java 4 Lines 1.1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/definition/AppointmentReasonDataDefinition.java java 4 Lines 1.1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/definition/AppointmentStartDateDataDefinition.java java 4 Lines 1.1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/definition/AppointmentStatusDataDefinition.java java 4 Lines 1.2 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/definition/AppointmentTypeDataDefinition.java java 4 Lines 1.1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/definition/PatientToAppointmentDataDefinition.java java 4 Lines 1.1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/definition/PersonToAppointmentDataDefinition.java java 4 Lines 1.1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentCancelReasonDataEvaluator.java java 2 Lines Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentDataEvaluator.java java Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentEndDateDataEvaluator.java java 2 Lines Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentLocationDataEvaluator.java java 2 Lines Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentPropertyDataEvaluator.java java 8 Lines 1.8 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentProviderDataEvaluator.java java 2 Lines Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentReasonDataEvaluator.java java 2 Lines Dec 13, 2019 12:56:58 PM

api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentStartDateDataEvaluator.java java 2 Lines Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentStatusDataEvaluator.java java 2 Lines Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentTypeDataEvaluator.java java 2 Lines Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/PatientToAppointmentDataEvaluator.java java 21 Lines 4.4 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/PersonToAppointmentDataEvaluator.java java 22 Lines 4.7 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/service/AppointmentDataService.java java 1.5 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/data/service/AppointmentDataServiceImpl.java java 6 Lines 2.2 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/dataset/definition/AppointmentDataSetDefinition.java java 20 Lines 5.2 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/dataset/evaluator/AppointmentDataSetEvaluator.java java 25 Lines 5.1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/query/AppointmentIdSet.java java 5 Lines Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/query/AppointmentQueryResult.java java 8 Lines 1.4 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/query/definition/AppointmentQuery.java java Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/query/definition/BasicAppointmentQuery.java java 8 Lines 2 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/query/evaluator/AppointmentQueryEvaluator.java java Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/query/evaluator/BasicAppointmentQueryEvaluator.java java 12 Lines 2.3 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/query/service/AppointmentQueryService.java java 1.1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/reporting/query/service/AppointmentQueryServiceImpl.java java 4 Lines 1.9 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/serialize/AppointmentStatusSerializer.java java 4 Lines 1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/validator/AppointmentBlockValidator.java java 8 Lines 4 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/validator/AppointmentRequestValidator.java java 5 Lines 2 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/validator/AppointmentStatusHistoryValidator.java java 5 Lines 2.1 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/validator/AppointmentTypeValidator.java java 15 Lines 4.3 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/validator/AppointmentValidator.java java 7 Lines 3 KB Dec 13, 2019 12:56:58 PM
api/src/main/java/org/openmrs/module/appointmentscheduling/validator/TimeSlotValidator.java java 5 Lines 2.4 KB Dec 13, 2019 12:56:58 PM
api/src/main/resources/Appointment.hbm.xml xml 2.2 KB Dec 13, 2019 12:56:58 PM
api/src/main/resources/AppointmentBlock.hbm.xml xml 1.9 KB Dec 13, 2019 12:56:58 PM
api/src/main/resources/AppointmentRequest.hbm.xml xml 3.2 KB Dec 13, 2019 12:56:58 PM

api/src/main/resources/AppointmentStatusHistory.hbm.xml xml 1.2 KB Dec 13, 2019 12:56:58 PM
api/src/main/resources/AppointmentType.hbm.xml xml 1.7 KB Dec 13, 2019 12:56:58 PM
api/src/main/resources/TimeSlot.hbm.xml xml 1.6 KB Dec 13, 2019 12:56:58 PM
api/src/main/resources/liquibase.xml xml 19.2 KB Dec 13, 2019 12:56:58 PM
api/src/main/resources/messages.properties java_properties 21.7 KB Dec 13, 2019 12:56:58 PM
api/src/main/resources/moduleApplicationContext.xml xml 6.8 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/AppointmentTest.java java 25 Lines 2 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/api/AppointmentBlockServiceTest.java java 173 Lines 19.2 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/api/AppointmentRequestServiceTest.java java 106 Lines 13.4 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/api/AppointmentServiceTest.java java 259 Lines 24.5 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/api/AppointmentStatusHistoryServiceTest.java java 84 Lines 11.1 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/api/AppointmentTypeServiceTest.java java 82 Lines 13.4 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/api/AppointmentUtilityTest.java java 23 Lines 3.5 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/api/TimeSlotServiceTest.java java 165 Lines 18.5 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentCancelReasonDataEvaluatorTest.java java 4 Lines 1.6 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentEndDateDataEvaluatorTest.java java 5 Lines 1.6 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentLocationDataEvaluatorTest.java java 4 Lines 1.6 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentProviderDataEvaluatorTest.java java 4 Lines 1.6 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentReasonDataEvaluatorTest.java java 4 Lines 1.5 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentStartDateDataEvaluatorTest.java java 5 Lines 1.6 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentStatusDataEvaluatorTest.java java 4 Lines 1.6 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/AppointmentTypeDataEvaluatorTest.java java 5 Lines 1.6 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/PatientToAppointmentDataEvaluatorTest.java java 15 Lines 3.5 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/reporting/data/evaluator/PersonToAppointmentDataEvaluatorTest.java java 15 Lines 3.3 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/reporting/dataset/evaluator/AppointmentDataSetEvaluatorTest.java java 8 Lines 2.2 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/reporting/query/evaluator/BasicAppointmentQueryEvaluatorTest.java java 18 Lines 4.6 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/validator/AppointmentBlockValidatorComponentTest.java java 32 Lines 5.1 KB Dec 13, 2019 12:56:58 PM
api/src/test/java/org/openmrs/module/appointmentscheduling/validator/AppointmentTypeValidatorTest.java java 23 Lines 4 KB Dec 13, 2019 12:56:58 PM
api/src/test/resources/TestingApplicationContext.xml xml 1.1 KB Dec 13, 2019 12:56:58 PM
api/src/test/resources/standardAppointmentTestDataset.xml xml 13.1 KB Dec 13, 2019 12:56:58 PM

api/src/test/resources/test-hibernate.cfg.xml xml Dec 13, 2019 12:56:58 PM
api/target/classes/Appointment.hbm.xml xml 2.2 KB Dec 18, 2019 11:53:40 AM
api/target/classes/AppointmentBlock.hbm.xml xml 1.9 KB Dec 18, 2019 11:53:41 AM
api/target/classes/AppointmentRequest.hbm.xml xml 3.2 KB Dec 18, 2019 11:53:41 AM
api/target/classes/AppointmentStatusHistory.hbm.xml xml 1.2 KB Dec 18, 2019 11:53:41 AM
api/target/classes/AppointmentType.hbm.xml xml 1.7 KB Dec 18, 2019 11:53:40 AM
api/target/classes/TimeSlot.hbm.xml xml 1.6 KB Dec 18, 2019 11:53:40 AM
api/target/classes/liquibase.xml xml 19.2 KB Dec 18, 2019 11:53:41 AM
api/target/classes/messages.properties java_properties 20 KB Dec 18, 2019 11:53:41 AM
api/target/classes/moduleApplicationContext.xml xml 6.7 KB Dec 18, 2019 11:53:40 AM
api/target/maven-archiver/pom.properties java_properties Dec 18, 2019 11:54:27 AM
omod/pom.xml xml 6.1 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/extension/html/AdminList.java java 11 Lines 2.2 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/extension/html/AppointmentsHeaderLinkExt.java java 4 Lines Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/extension/html/PatientDashboardAppointmentExt.java java 24 Lines 5.6 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/extension/html/PatientDashboardAppointmentTabExt.java java 5 Lines Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/rest/controller/AppointmentRestController.java java 3 Lines Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/AppointmentAllowingOverbookResource1_9.java java 3 Lines Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/AppointmentBlockResource1_9.java java 37 Lines 6.9 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/AppointmentBlockWithTimeSlotResource1_9.java java 19 Lines 3.1 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/AppointmentRequestResource1_9.java java 39 Lines 8.2 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/AppointmentRequestStatusResource1_9.java java 8 Lines 1.8 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/AppointmentResource1_9.java java 66 Lines 10.1 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/AppointmentStatusResource1_9.java java 9 Lines 1.7 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/AppointmentStatusTypeResource1_9.java java 8 Lines 1.7 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/AppointmentTypeResource1_9.java java 18 Lines 4.5 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/TimeFrameUnitsResource1_9.java java 8 Lines 1.7 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/TimeSlotResource1_9.java java 43 Lines 7.7 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/util/AppointmentRestUtils.java java 8 Lines Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/AppointmentBlockData.java java 32 Lines 2.3 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/AppointmentBlockEditor.java java 14 Lines 2 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/AppointmentData.java java 20 Lines 1.4 KB Dec 13, 2019

12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/AppointmentEditor.java java 14 Lines 1.9 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/AppointmentTypeEditor.java java 14 Lines 1.9 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/DWRAppointmentService.java java 167 Lines 20 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/PatientData.java java 14 Lines 1.2 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/ProviderEditor.java java 14 Lines 2.1 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/TimeSlotEditor.java java 14 Lines 1.9 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/controller/AppointmentBlockCalendarController.java java 61 Lines 8.2 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/controller/AppointmentBlockFormController.java java 102 Lines 14.3 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/controller/AppointmentBlockListController.java java 88 Lines 12.7 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/controller/AppointmentFormController.java java 68 Lines 9.8 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/controller/AppointmentListController.java java 126 Lines 14.7 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/controller/AppointmentSettingsFormController.java java 64 Lines 7.8 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/controller/AppointmentStatisticsFormController.java java 3 Lines 1.6 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/controller/AppointmentTypeFormController.java java 24 Lines 4.9 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/controller/AppointmentTypeListController.java java 5 Lines 1.9 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/controller/AppointmentsPortletController.java java 9 Lines 2.2 KB Dec 13, 2019 12:56:58 PM
omod/src/main/java/org/openmrs/module/appointmentscheduling/web/controller/PatientDashboardAppointmentExtController.java java 14 Lines 2.1 KB Dec 13, 2019 12:56:58 PM
omod/src/main/resources/config.xml xml 9.4 KB Dec 13, 2019 12:56:58 PM
omod/src/main/resources/webModuleApplicationContext.xml xml 2.1 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/appointmentBlockCalendar.jsp jsp 133 Lines 13.7 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/appointmentBlockForm.jsp jsp 172 Lines 18.9 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/appointmentBlockList.jsp jsp 236 Lines 25 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/appointmentForm.jsp jsp 216 Lines 23.3 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/appointmentList.jsp jsp 45 Lines 22.8 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/appointmentSettingsForm.jsp jsp 35 Lines 10.9 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/appointmentStatisticsForm.jsp jsp 9 Lines 19.9 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/appointmentTypeForm.jsp jsp 20 Lines 4.5 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/appointmentTypeList.jsp jsp 36 Lines 3.6 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/localHeader.jsp jsp 8 Lines 1.7 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/portlets/appointments.jsp jsp 69 Lines 6.9 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/date.format.js typescript 68 Lines 3.8 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/fullcalendar.js typescript 2,639 Lines 125.4 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/fullcalendar.min.js typescript 2 Lines 48.2 KB Dec 13, 2019 12:56:58 PM

omod/src/main/webapp/resources/Scripts/gcal.js typescript 53 Lines 2.6 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/jqPlot-plugins/jqplot.barRenderer.min.js typescript 1 Lines 13.1 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/jqPlot-plugins/jqplot.categoryAxisRenderer.min.js typescript 1 Lines 9.5 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/jqPlot-plugins/jqplot.donutRenderer.min.js typescript 1 Lines 12.9 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/jqPlot-plugins/jqplot.highlighter.js typescript 179 Lines 21.4 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/jqPlot-plugins/jqplot.pieRenderer.min.js typescript 1 Lines 13.3 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/jqPlot-plugins/jqplot.pointLabels.min.js typescript 1 Lines 4.5 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/jquery-ui-1.10.2.custom.min.js typescript 1 Lines 47.7 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/jquery.dataTables.js typescript 2,644 Lines 368.7 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/jquery.jqplot.min.js typescript 1 Lines 168.4 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/jquery.maxlength.js typescript 38 Lines 2.9 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/json2.js typescript 109 Lines 17.1 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/opentip-jquery-excanvas.js typescript 1,541 Lines 85.4 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/queryParameters.js typescript 5 Lines Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/statusButtons.js typescript 65 Lines 2.2 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/Scripts/timepicker.js typescript 35 Lines 1.5 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/TableTools/media/ZeroClipboard/ZeroClipboard.as actionscript 32 Lines 3.2 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/TableTools/media/ZeroClipboard/ZeroClipboard.js typescript 148 Lines 9.7 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/TableTools/media/js/TableTools.js typescript 197 Lines 13.7 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/TableTools/media/support/jquery.dataTables.min.js typescript 442 Lines 53.4 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/resources/TableTools/media/support/jquery.jeditable.js typescript 231 Lines 23.5 KB Dec 13, 2019 12:56:58 PM
omod/src/main/webapp/template/localHeader.jsp jsp 4 Lines Dec 13, 2019 12:56:58 PM
omod/src/test/java/org/openmrs/module/appointmentscheduling/rest/controller/AppointmentAllowingOverbookResource1_9ControllerTest.java java 9 Lines 2.4 KB Dec 13, 2019 12:56:58 PM
omod/src/test/java/org/openmrs/module/appointmentscheduling/rest/controller/AppointmentBlockResource1_9ControllerTest.java java 58 Lines 12.3 KB Dec 13, 2019 12:56:58 PM
omod/src/test/java/org/openmrs/module/appointmentscheduling/rest/controller/AppointmentBlockWithTimeSlotResource1_9ControllerTest.java java 21 Lines 3.7 KB Dec 13, 2019 12:56:58 PM
omod/src/test/java/org/openmrs/module/appointmentscheduling/rest/controller/AppointmentRequestResource1_9ControllerTest.java java 45 Lines 10.6 KB Dec 13, 2019 12:56:58 PM
omod/src/test/java/org/openmrs/module/appointmentscheduling/rest/controller/AppointmentResource1_9ControllerTest.java java 78 Lines 10.8 KB Dec 13, 2019 12:56:58 PM
omod/src/test/java/org/openmrs/module/appointmentscheduling/rest/controller/AppointmentTypeResource1_9ControllerTest.java java 49 Lines 7.8 KB Dec 13, 2019 12:56:58 PM
omod/src/test/java/org/openmrs/module/appointmentscheduling/rest/controller/TimeSlotResource1_9ControllerTest.java java 67 Lines 12 KB Dec 13, 2019 12:56:58 PM
omod/src/test/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/AppointmentBlockResource1_9Test.java java 4 Lines 2.1 KB Dec 13, 2019 12:56:58 PM
omod/src/test/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/AppointmentBlockWithTimeSlotResource1_9Test.java java 4 Lines 2.1 KB Dec 13, 2019 12:56:58 PM
omod/src/test/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/AppointmentRequestResource1_9Test.

java java 4 Lines 2.8 KB Dec 13, 2019 12:56:58 PM
omod/src/test/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/AppointmentResource1_9Test.java java 4 Lines 2 KB Dec 13, 2019 12:56:58 PM
omod/src/test/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/AppointmentStatusResource1_9Test.java java 8 Lines 1.2 KB Dec 13, 2019 12:56:58 PM
omod/src/test/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/AppointmentTypeResource1_9Test.java java 4 Lines 2 KB Dec 13, 2019 12:56:58 PM
omod/src/test/java/org/openmrs/module/appointmentscheduling/rest/resource/openmrs1_9/TimeSlotResource1_9Test.java java 4 Lines 2.1 KB Dec 13, 2019 12:56:58 PM
omod/src/test/java/org/openmrs/module/appointmentscheduling/rest/test/SameDatetimeMatcher.java java 9 Lines 1.3 KB Dec 13, 2019 12:56:58 PM
omod/src/test/resources/TestingApplicationContext.xml xml 1.1 KB Dec 13, 2019 12:56:58 PM
omod/src/test/resources/standardWebAppointmentTestDataset.xml xml 11.8 KB Dec 13, 2019 12:56:58 PM
omod/src/test/resources/test-hibernate.cfg.xml xml Dec 13, 2019 12:56:58 PM
omod/target/appointmentscheduling-1.10.0/Appointment.hbm.xml xml 2.2 KB Dec 18, 2019 11:55:36 AM
omod/target/appointmentscheduling-1.10.0/AppointmentBlock.hbm.xml xml 1.9 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/AppointmentRequest.hbm.xml xml 3.2 KB Dec 18, 2019 11:55:36 AM
omod/target/appointmentscheduling-1.10.0/AppointmentStatusHistory.hbm.xml xml 1.2 KB Dec 18, 2019 11:55:36 AM
omod/target/appointmentscheduling-1.10.0/AppointmentType.hbm.xml xml 1.7 KB Dec 18, 2019 11:55:36 AM
omod/target/appointmentscheduling-1.10.0/META-INF/maven/org.openmrs.module/appointmentscheduling-api/pom.properties java_properties Dec 18, 2019 11:55:36 AM
omod/target/appointmentscheduling-1.10.0/META-INF/maven/org.openmrs.module/appointmentscheduling-api/pom.xml xml 1.4 KB Dec 18, 2019 11:55:36 AM
omod/target/appointmentscheduling-1.10.0/TimeSlot.hbm.xml xml 1.6 KB Dec 18, 2019 11:55:36 AM
omod/target/appointmentscheduling-1.10.0/config.xml xml 9.2 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/liquibase.xml xml 19.2 KB Dec 18, 2019 11:55:36 AM
omod/target/appointmentscheduling-1.10.0/messages.properties java_properties 20 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/moduleApplicationContext.xml xml 6.7 KB Dec 18, 2019 11:55:36 AM
omod/target/appointmentscheduling-1.10.0/web/module/appointmentBlockCalendar.jsp jsp 133 Lines 13.7 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/appointmentBlockForm.jsp jsp 172 Lines 18.9 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/appointmentBlockList.jsp jsp 236 Lines 25 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/appointmentForm.jsp jsp 216 Lines 23.3 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/appointmentList.jsp jsp 45 Lines 22.8 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/appointmentSettingsForm.jsp jsp 35 Lines 10.9 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/appointmentStatisticsForm.jsp jsp 9 Lines 19.9 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/appointmentTypeForm.jsp jsp 20 Lines 4.5 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/appointmentTypeList.jsp jsp 36 Lines 3.6 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/localHeader.jsp jsp 8 Lines 1.7 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/portlets/appointments.jsp jsp 69 Lines 6.9 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/date.format.js typescript 68 Lines 3.8 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/fullcalendar.js typescript 2,639 Lines 125.4 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/fullcalendar.min.js typescript 2 Lines 48.2 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/gcal.js typescript 53 Lines 2.6 KB Dec 18, 2019 11:55:37 AM

omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/jqPlot-plugins/jqplot.barRenderer.min.js typescript 1 Lines 13.1 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/jqPlot-plugins/jqplot.categoryAxisRenderer.min.js typescript 1 Lines 9.5 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/jqPlot-plugins/jqplot.donutRenderer.min.js typescript 1 Lines 12.9 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/jqPlot-plugins/jqplot.highlighter.js typescript 179 Lines 21.4 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/jqPlot-plugins/jqplot.pieRenderer.min.js typescript 1 Lines 13.3 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/jqPlot-plugins/jqplot.pointLabels.min.js typescript 1 Lines 4.5 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/jquery-ui-1.10.2.custom.min.js typescript 1 Lines 47.7 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/jquery.dataTables.js typescript 2,644 Lines 368.7 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/jquery.jqplot.min.js typescript 1 Lines 168.4 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/jquery.maxlength.js typescript 38 Lines 2.9 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/json2.js typescript 109 Lines 17.1 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/opentip-jquery-excanvas.js typescript 1,541 Lines 85.4 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/queryParameters.js typescript 5 Lines Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/statusButtons.js typescript 65 Lines 2.2 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/Scripts/timepicker.js typescript 35 Lines 1.5 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/TableTools/media/ZeroClipboard/ZeroClipboard.as actionscript 32 Lines 3.2 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/TableTools/media/ZeroClipboard/ZeroClipboard.js typescript 148 Lines 9.7 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/TableTools/media/js/TableTools.js typescript 197 Lines 13.7 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/TableTools/media/support/jquery.dataTables.min.js typescript 442 Lines 53.4 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/resources/TableTools/media/support/jquery.jeditable.js typescript 231 Lines 23.5 KB Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/web/module/template/localHeader.jsp jsp 4 Lines Dec 18, 2019 11:55:37 AM
omod/target/appointmentscheduling-1.10.0/webModuleApplicationContext.xml xml 2.1 KB Dec 18, 2019 11:55:36 AM
omod/target/classes/Appointment.hbm.xml xml 2.2 KB Dec 18, 2019 11:53:40 AM
omod/target/classes/AppointmentBlock.hbm.xml xml 1.9 KB Dec 18, 2019 11:53:42 AM
omod/target/classes/AppointmentRequest.hbm.xml xml 3.2 KB Dec 18, 2019 11:53:42 AM
omod/target/classes/AppointmentStatusHistory.hbm.xml xml 1.2 KB Dec 18, 2019 11:53:42 AM
omod/target/classes/AppointmentType.hbm.xml xml 1.7 KB Dec 18, 2019 11:53:40 AM
omod/target/classes/META-INF/maven/org.openmrs.module/appointmentscheduling-api/pom.properties java_properties Dec 18, 2019 11:54:28 AM
omod/target/classes/META-INF/maven/org.openmrs.module/appointmentscheduling-api/pom.xml xml 1.4 KB Dec 13, 2019 12:56:58 PM

omod/target/classes/TimeSlot.hbm.xml xml 1.6 KB Dec 18, 2019 11:53:40 AM
omod/target/classes/config.xml xml 9.2 KB Dec 18, 2019 11:55:00 AM
omod/target/classes/liquibase.xml xml 19.2 KB Dec 18, 2019 11:53:42 AM
omod/target/classes/messages.properties java_properties 20 KB Dec 18, 2019 11:53:42 AM
omod/target/classes/moduleApplicationContext.xml xml 6.7 KB Dec 18, 2019 11:53:40 AM
omod/target/classes/web/module/appointmentBlockCalendar.jsp jsp 133 Lines 13.7 KB Dec 18, 2019 11:55:05 AM
omod/target/classes/web/module/appointmentBlockForm.jsp jsp 172 Lines 18.9 KB Dec 18, 2019 11:55:02 AM
omod/target/classes/web/module/appointmentBlockList.jsp jsp 236 Lines 25 KB Dec 18, 2019 11:55:05 AM
omod/target/classes/web/module/appointmentForm.jsp jsp 216 Lines 23.3 KB Dec 18, 2019 11:55:05 AM
omod/target/classes/web/module/appointmentList.jsp jsp 45 Lines 22.8 KB Dec 18, 2019 11:55:02 AM
omod/target/classes/web/module/appointmentSettingsForm.jsp jsp 35 Lines 10.9 KB Dec 18, 2019 11:55:05 AM
omod/target/classes/web/module/appointmentStatisticsForm.jsp jsp 9 Lines 19.9 KB Dec 18, 2019 11:55:05 AM
omod/target/classes/web/module/appointmentTypeForm.jsp jsp 20 Lines 4.5 KB Dec 18, 2019 11:55:02 AM
omod/target/classes/web/module/appointmentTypeList.jsp jsp 36 Lines 3.6 KB Dec 18, 2019 11:55:02 AM
omod/target/classes/web/module/localHeader.jsp jsp 8 Lines 1.7 KB Dec 18, 2019 11:55:02 AM
omod/target/classes/web/module/portlets/appointments.jsp jsp 69 Lines 6.9 KB Dec 18, 2019 11:55:02 AM
omod/target/classes/web/module/resources/Scripts/date.format.js typescript 68 Lines 3.8 KB Dec 18, 2019 11:55:04 AM
omod/target/classes/web/module/resources/Scripts/fullcalendar.js typescript 2,639 Lines 125.4 KB Dec 18, 2019 11:55:03 AM
omod/target/classes/web/module/resources/Scripts/fullcalendar.min.js typescript 2 Lines 48.2 KB Dec 18, 2019 11:55:04 AM
omod/target/classes/web/module/resources/Scripts/gcal.js typescript 53 Lines 2.6 KB Dec 18, 2019 11:55:04 AM
omod/target/classes/web/module/resources/Scripts/jqPlot-plugins/jqplot.barRenderer.min.js typescript 1 Lines 13.1 KB Dec 18, 2019 11:55:04 AM
omod/target/classes/web/module/resources/Scripts/jqPlot-plugins/jqplot.categoryAxisRenderer.min.js typescript 1 Lines 9.5 KB Dec 18, 2019 11:55:04 AM
omod/target/classes/web/module/resources/Scripts/jqPlot-plugins/jqplot.donutRenderer.min.js typescript 1 Lines 12.9 KB Dec 18, 2019 11:55:04 AM
omod/target/classes/web/module/resources/Scripts/jqPlot-plugins/jqplot.highlighter.js typescript 179 Lines 21.4 KB Dec 18, 2019 11:55:04 AM
omod/target/classes/web/module/resources/Scripts/jqPlot-plugins/jqplot.pieRenderer.min.js typescript 1 Lines 13.3 KB Dec 18, 2019 11:55:04 AM
omod/target/classes/web/module/resources/Scripts/jqPlot-plugins/jqplot.pointLabels.min.js typescript 1 Lines 4.5 KB Dec 18, 2019 11:55:04 AM
omod/target/classes/web/module/resources/Scripts/jquery-ui-1.10.2.custom.min.js typescript 1 Lines 47.7 KB Dec 18, 2019 11:55:04 AM
omod/target/classes/web/module/resources/Scripts/jquery.dataTables.js typescript 2,644 Lines 368.7 KB Dec 18, 2019 11:55:04 AM
omod/target/classes/web/module/resources/Scripts/jquery.jqplot.min.js typescript 1 Lines 168.4 KB Dec 18, 2019 11:55:03 AM
omod/target/classes/web/module/resources/Scripts/jquery.maxlength.js typescript 38 Lines 2.9 KB Dec 18, 2019 11:55:04 AM
omod/target/classes/web/module/resources/Scripts/json2.js typescript 109 Lines 17.1 KB Dec 18, 2019 11:55:03 AM
omod/target/classes/web/module/resources/Scripts/opentip-jquery-excanvas.js typescript 1,541 Lines 85.4 KB Dec 18, 2019 11:55:04 AM
omod/target/classes/web/module/resources/Scripts/queryParameters.js typescript 5 Lines Dec 18, 2019 11:55:04 AM
omod/target/classes/web/module/resources/Scripts/statusButtons.js typescript 65 Lines 2.2 KB Dec 18, 2019 11:55:03 AM
omod/target/classes/web/module/resources/Scripts/timepicker.js typescript 35 Lines 1.5 KB Dec 18, 2019 11:55:03 AM
omod/target/classes/web/module/resources/TableTools/media/ZeroClipboard/ZeroClipboard.as actionscript 32 Lines 3.2 KB Dec 18, 2019 11:55:05 AM
omod/target/classes/web/module/resources/TableTools/media/ZeroClipboard/ZeroClipboard.js typescript 148 Lines 9.7 KB Dec 18, 2019 11:55:05 AM
omod/target/classes/web/module/resources/TableTools/media/js/TableTools.js typescript 197 Lines 13.7 KB Dec 18, 2019 11:55:05 AM
omod/target/classes/web/module/resources/TableTools/media/support/jquery.dataTables.min.js typescript 442 Lines 53.4 KB Dec 18, 2019 11:55:04 AM

omod/target/classes/web/module/resources/TableTools/media/support/jquery.jeditable.js typescript 231 Lines 23.5 KB Dec 18, 2019 11:55:04 AM
omod/target/classes/web/module/template/localHeader.jsp jsp 4 Lines Dec 18, 2019 11:55:02 AM
omod/target/classes/webModuleApplicationContext.xml xml 2.1 KB Dec 18, 2019 11:55:00 AM
omod/target/maven-archiver/pom.properties java_properties Dec 18, 2019 11:55:31 AM
pom.xml xml 6.5 KB Dec 13, 2019 12:56:58 PM

Reference Elements

Classpath:

No classpath specified during translation

Libdirs:

No libdirs specified during translation

Rulepacks

Valid Rulepacks:

Name: Fortify Secure Coding Rules, Core, Java
Version: 2019.4.0.0009
ID: 06A6CC97-8C3F-4E73-9093-3E74C64A2AAF
SKU: RUL13003

Name: Fortify Secure Coding Rules, Core, Annotations
Version: 2019.4.0.0009
ID: 14EE50EB-FA1C-4AE8-8B59-39F952E21E3B
SKU: RUL13078

Name: Fortify Secure Coding Rules, Core, ActionScript 3.0
Version: 2019.4.0.0009
ID: 92127AA2-E666-4F28-B1C1-C0F6A939A089
SKU: RUL13094

Name: Fortify Secure Coding Rules, Core, JavaScript
Version: 2019.4.0.0009
ID: BD292C4E-4216-4DB8-96C7-9B607BFD9584
SKU: RUL13059

Name: Fortify Secure Coding Rules, Core, Android
Version: 2019.4.0.0009
ID: FF9890E6-D119-4EE8-A591-83DCF4CA6952
SKU: RUL13093

Name: Fortify Secure Coding Rules, Extended, JavaScript
Version: 2019.4.0.0009
ID: C4D1969E-B734-47D3-87D4-73962C1D32E2
SKU: RUL13141

Name: Fortify Secure Coding Rules, Extended, Configuration

Version: 2019.4.0.0009
ID: CD6959FC-0C37-45BE-9637-BAA43C3A4D56
SKU: RUL13005

Name: Fortify Secure Coding Rules, Extended, Java
Version: 2019.4.0.0009
ID: AAAC0B10-79E7-4FE5-9921-F4903A79D317
SKU: RUL13007

Name: Fortify Secure Coding Rules, Extended, Content
Version: 2019.4.0.0009
ID: 9C48678C-09B6-474D-B86D-97EE94D38F17
SKU: RUL13067

Name: Fortify Secure Coding Rules, Core, Golang
Version: 2019.4.0.0009
ID: 1DCE79F8-AF6B-474D-A05A-5BFFC8B13DCD
SKU: RUL13218

Name: Fortify Secure Coding Rules, Extended, JSP
Version: 2019.4.0.0009
ID: 00403342-15D0-48C9-8E67-4B1CFBDEFCD2
SKU: RUL13026

External Metadata:
Version: 2019.4.0.0009

Name: CWE
ID: 3ADB9EE4-5761-4289-8BD3-CBFCC593EBBC
The Common Weakness Enumeration (CWE), co-sponsored and maintained by MITRE, is international in scope and free for public use. CWE provides a unified, measurable set of software weaknesses that is enabling more effective discussion, description, selection, and use of software security tools and services that can find these weaknesses in source code and operational systems as well as better understanding and management of software weaknesses related to architecture and design.

Name: CWE Top 25 2019
ID: 7AF935C9-15AA-45B2-8EEC-0EAE4194ACDE
The 2019 CWE Top 25 Most Dangerous Software Errors lists the most widespread and critical weaknesses that can lead to serious vulnerabilities in software (as demonstrated by the National Vulnerability Database). These weaknesses occur frequently, are often easy to find, and easy to exploit. They are dangerous because they will frequently enable attackers to completely take over the software, steal data, or prevent the software from working at all. The list is the result of heuristic formula that the CWE Team used with a data-driven approach that leveraged the Common Vulnerabilities and Exposure (CVE), National Vulnerability Database (NVD), and Common Vulnerability Scoring System (CVSS). Due to the hiearchical nature of the CWE taxonomy, Fortify considers all CWE IDs which are children of a Top 25 entry, as included within the context of the entry due to the "CHILD-OF" relationship within the hierarchy. Exercise caution if using only this Top 25 list to prioritize auditing efforts because the software under analysis might not align with the assumptions of the heuristic used to define the Top 25. For example, many of these weaknesses are related to C-like langauges and the software under analysis might not be within the C-family of languages - thus, many CWEs would not be in scope.

Name: DISA CCI 2
ID: 7F037130-41E5-40F0-B653-7819A4B3E241
The purpose of a Defense Information Systems Agency (DISA) Control Correlation Identifier (CCI) is to provide a standard

identifier for policy based requirements which connect high-level policy expressions and low-level technical implementations. Associated with each CCI is a description for each of the singular, actionable, statements compromising an information assurance (IA) control or IA best practice. Using CCI allows high-level policy framework security requirements to be decomposed and explicitly associated with low-level implementations, thus enabling the assessment of related compliance assessment results spanning heterogeneous technologies. The current IA controls and best practices associated with each CCI, that are specified in NIST SP 800-53 Revision 4, can be viewed using the DISA STIG Viewer.
The following table summarizes the number of issues identified across the different CCIs broken down by Fortify Priority Order. The status of a CCI is considered "In Place" when there are no issues reported for a given CCI.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, CCI-003187 is not considered "In Place". Similarly, if the project is missing a Micro Focus Fortify WebInspect scan, or the scan contains any critical findings, CCI-000366 and CCI-000256 are not considered "In Place".

Name: FISMA
ID: B40F9EE0-3824-4879-B9FE-7A789C89307C
The Federal Information Processing Standard (FIPS) 200 document is part of the official series of publications, issued by the National Institute of Standards and Technology (NIST), relating to standards and guidelines adopted and promulgated under the provisions of the Federal Information Security Management Act (FISMA). Specifically, FIPS Publication 200 specifies the "Minimum Security Requirements for Federal Information and Information Systems."

Name: GDPR
ID: 771C470C-9274-4580-8556-C12F5E4BEC51
The EU General Data Protection Regulation (GDPR) replaces the Data Protection Directive 95/46/EC and was designed to harmonize data privacy laws across Europe, to protect and empower all EU citizens data privacy and to reshape the way organizations across the region approach data privacy. Going into effect on May 25, 2018, GDPR provides a framework for organizations on how to handle personal data. According to GDPR regulation personal data "means any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person." GDPR articles that pertain to application security and require businesses to protect personal data during design and development of its product and services are:

- Article 25, Data protection by design and by default - which requires "The controller shall implement appropriate technical and organisational measures for ensuring that, by default, only personal data which are necessary for each specific purpose of the processing are processed."
- Article 32, Security of processing - which requires businesses to protect its systems and applications "from accidental or unlawful destruction, loss, alteration, unauthorized disclosure of, or access to personal data". This report may be used by organizations as a framework to help identify and protect personal data as it relates to application security.

Name: MISRA C 2012
ID: 555A3A66-A0E1-47AF-910C-3F19A6FB2506
Now in its third edition, the Motor Industry Software Reliability Association (MISRA) C Guidelines describe a subset of the C programming language in which there is reduced risk of introducing mistakes in critical systems. While the MISRA C Guidelines focus upon safety-related software development, a subset of the rules also reflect security properties. Fortify interprets the MISRA C Guidelines under the context of security and provides correlation of security vulnerability categories to the rules defined by MISRA. Fortify provides these security focused detection mechanism with the standard rulepacks, however, further support of the MISRA C Guidelines related to safety can be added through the use of custom rules. The results in this report can assist in the creation of a compliance matrix for MISRA.

Name: MISRA C++ 2008
ID: 5D4B75A1-FC91-4B4B-BD4D-C81BBE9604FA

The Motor Industry Software Reliability Association (MISRA) C++ Guidelines describe a subset of the C++ programming language in which there is reduced risk of introducing mistakes in critical systems. While the MISRA C++ Guidelines focus upon safety-related software development, a subset of the rules also reflect security properties. Fortify interprets the MISRA C++ Guidelines under the context of security and provides correlation of security vulnerability categories to the rules defined by MISRA. Fortify provides these security focused detection mechanism with the standard rulepacks, however, further support of the MISRA C++ Guidelines related to safety can be added through the use of custom rules. The results in this report can assist in the creation of a compliance matrix for MISRA.

Name: NIST SP 800-53 Rev.4
ID: 1114583B-EA24-45BE-B7F8-B61201BACDD0
NIST Special Publication 800-53 Revision 4 provides a list of security and privacy controls designed to protect federal organizations and information systems from security threats. The following table summarizes the number of issues identified across the different controls and broken down by Fortify Priority Order.

Name: OWASP Mobile 2014
ID: EEE3F9E7-28D6-4456-8761-3DA56C36F4EE
The OWASP Mobile Top 10 Risks 2014 provides a powerful awareness document for mobile application security. The OWASP Mobile Top 10 represents a broad consensus about what the most critical mobile application security flaws are. Project members include a variety of security experts from around the world who have shared their expertise to produce this list.

Name: OWASP Top 10 2004
ID: 771C470C-9274-4580-8556-C023E4D3ADB4
The OWASP Top Ten 2004 provides a powerful awareness document for web application security. The OWASP Top Ten represents a broad consensus about what the most critical web application security flaws are. Project members include a variety of security experts from around the world who have shared their expertise to produce this list.

Name: OWASP Top 10 2007
ID: 1EB1EC0E-74E6-49A0-BCE5-E6603802987A
The OWASP Top Ten 2007 provides a powerful awareness document for web application security. The OWASP Top Ten represents a broad consensus about what the most critical web application security flaws are. Project members include a variety of security experts from around the world who have shared their expertise to produce this list.

Name: OWASP Top 10 2010
ID: FDCECA5E-C2A8-4BE8-BB26-76A8ECD0ED59
The OWASP Top Ten 2010 provides a powerful awareness document for web application security. The OWASP Top Ten represents a broad consensus about what the most critical web application security flaws are. Project members include a variety of security experts from around the world who have shared their expertise to produce this list.

Name: OWASP Top 10 2013
ID: 1A2B4C7E-93B0-4502-878A-9BE40D2A25C4
The OWASP Top Ten 2013 provides a powerful awareness document for web application security. The OWASP Top Ten represents a broad consensus about what the most critical web application security flaws are. Project members include a variety of security experts from around the world who have shared their expertise to produce this list.

Name: OWASP Top 10 2017
ID: 3C6ECB67-BBD9-4259-A8DB-B49328927248
The OWASP Top Ten 2017 provides a powerful awareness document for web application security focused on informing the community about the consequences of the most common and most important web application security weaknesses. The OWASP Top Ten represents a broad agreement about what the most critical web application security flaws are with consensus being drawn from data collection and survey results. Project members include a variety of security experts from around the world who have shared their expertise to produce this list.

Name: PCI 1.1
ID: CBDB9D4D-FC20-4C04-AD58-575901CAB531
The Payment Card Industry (PCI) Data Security Standard (DSS) 1.1 compliance standard describes 12 requirements which are organized into 6 logically related groups, which are "control objectives". PCI DSS requirements are applicable if Primary Account Number (PAN) is stored, processed, or transmitted by the system.

Name: PCI 1.2
ID: 57940BDB-99F0-48BF-BF2E-CFC42BA035E5
Payment Card Industry Data Security Standard Version 1.2 description

Name: PCI 2.0
ID: 8970556D-7F9F-4EA7-8033-9DF39D68FF3E
The PCI DSS 2.0 compliance standard, particularly sections 6.3, 6.5, and 6.6, references the OWASP Top 10 vulnerability categories as the core categories that must be tested for and remediated. The following table summarizes the number of issues identified across the different PCI DSS requirements and broken down by Fortify Priority Order.

Name: PCI 3.0
ID: E2FB0D38-0192-4F03-8E01-FE2A12680CA3
The following is a summary of the application security portions of Payment Card Industry (PCI) Data Security Standard (DSS) v3.0. Fortify tests for 32 application security related requirements across sections 1, 2, 3, 4, 6, 7, 8, and 10 of PCI DSS and reports whether each requirement is In Place or Not In Place to indicate whether requirements are satisfied or not. This report is intended to measure the level of adherence the specific application(s) possess when compared to PCI DSS 3.0 compliance and is not intended to serve as a comprehensive Report on Compliance (ROC). The information contained in this report is targeted at project managers, security auditors, and compliance auditors.

Name: PCI 3.1
ID: AC0D18CF-C1DA-47CF-9F1A-E8EC0A4A717E
The following is a summary of the application security portions of Payment Card Industry (PCI) Data Security Standard (DSS) v3.1. Fortify tests for 31 application security related requirements across sections 1, 2, 3, 4, 6, 7, 8, and 10 of PCI DSS and reports whether each requirement is In Place or Not In Place to indicate whether requirements are satisfied or not. This report is intended to measure the level of adherence the specific application(s) possess when compared to PCI DSS 3.1 compliance and is not intended to serve as a comprehensive Report on Compliance (ROC). The information contained in this report is targeted at project managers, security auditors, and compliance auditors.

Name: PCI 3.2
ID: 4E8431F9-1BA1-41A8-BDBD-087D5826751A
The following is a summary of the application security portions of Payment Card Industry (PCI) Data Security Standard (DSS) v3.2. Fortify tests for 31 application security related requirements across sections 1, 2, 3, 4, 6, 7, 8, and 10 of PCI DSS and reports whether each requirement is In Place or Not In Place to indicate whether requirements are satisfied or not. This report is intended to measure the level of adherence the specific application(s) possess when compared to PCI DSS 3.2 compliance and is not intended to serve as a comprehensive Report on Compliance (ROC). The information contained in this report is targeted at project managers, security auditors, and compliance auditors.

Name: PCI 3.2.1
ID: EADE255F-6561-4EFE-AD31-2914F6BFA329
The following is a summary of the application security portions of Payment Card Industry (PCI) Data Security Standard (DSS) v3.2.1. Fortify tests for 31 application security related requirements across sections 1, 2, 3, 4, 6, 7, 8, and 10 of PCI DSS and reports whether each requirement is In Place or Not In Place to indicate whether requirements are satisfied or not. This report is intended to measure the level of adherence the specific application(s) possess when compared to PCI DSS 3.2.1 compliance and is not intended to serve as a comprehensive Report on Compliance (ROC). The information contained in this report is targeted at

project managers, security auditors, and compliance auditors.

Name: PCI SSF 1.0
ID: 0F551543-AF0E-4334-BEDF-1DDCD5F4BF74
The following is a summary of the application security portions of the Secure Software Requirements and Assessment Procedures defined in the Payment Card Industry (PCI) Software Security Framework (SSF) v1.0. Fortify tests for 23 application security related control objectives across Control Objective sections 2, 3, 4, 5, 6, 7, 8, and A.2 of PCI SSF and reports whether each control objective is In Place or Not In Place to indicate whether requirements are satisfied or not. This report is intended to measure the level of adherence the specific application(s) possess when compared to PCI SSF 1.0 compliance and is not intended to serve as a comprehensive Report on Compliance (ROC). The information contained in this report is targeted at project managers, security auditors, and compliance auditors.

Name: SANS Top 25 2009
ID: 939EF193-507A-44E2-ABB7-C00B2168B6D8
The 2009 CWE/SANS Top 25 Programming Errors lists the most significant programming errors that can lead to serious software vulnerabilities. They occur frequently, are often easy to find, and easy to exploit. They are dangerous because they will frequently allow attackers to completely take over the software, steal data, or prevent the software from working at all. The list is the result of collaboration between the SANS Institute, MITRE, and many top software security experts.

Name: SANS Top 25 2010
ID: 72688795-4F7B-484C-88A6-D4757A6121CA
SANS Top 25 2010 Most Dangerous Software Errors provides an enumeration of the most widespread and critical errors, categorized by Common Weakness Enumeration (CWE) identifiers, that lead to serious vulnerabilities in software (http://cwe.mitre.org/). These software errors are often easy to find and exploit. The inherent danger in these errors is that they can allow an attacker to completely take over the software, steal data, or prevent the software from working at all.

Name: SANS Top 25 2011
ID: 92EB4481-1FD9-4165-8E16-F2DE6CB0BD63
SANS Top 25 2011 Most Dangerous Software Errors provides an enumeration of the most widespread and critical errors, categorized by Common Weakness Enumeration (CWE) identifiers, that lead to serious vulnerabilities in software (http://cwe.mitre.org/). These software errors are often easy to find and exploit. The inherent danger in these errors is that they can allow an attacker to completely take over the software, steal data, or prevent the software from working at all.

Name: STIG 3.1
ID: F2FA57EA-5AAA-4DDE-90A5-480BE65CE7E7
Security Technical Implementation Guide Version 3.1 description

Name: STIG 3.10
ID: 788A87FE-C9F9-4533-9095-0379A9B35B12
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APP<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden

or suppressed, STIGID APP5080: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APP5100: CAT II is not considered "In Place".

Name: STIG 3.4
ID: 58E2C21D-C70F-4314-8994-B859E24CF855
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APP<I>ID</I>: CAT <I>SEV</I>]. DISA STIG identifies several severities with respect to vulnerabilities:

<LI>CAT I: allow an attacker immediate access into a machine, allow super user access, or bypass a firewall.</LI>
<LI>CAT II: provide information that have a high potential of giving access to an intruder.</LI>
<LI>CAT III: provide information that potentially could lead to compromise.</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

Name: STIG 3.5
ID: DD18E81F-3507-41FA-9DFA-2A9A15B5479F
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APP<I>ID</I>: CAT <I>SEV</I>]. DISA STIG identifies several severities with respect to vulnerabilities:

<LI>CAT I: allow an attacker immediate access into a machine, allow super user access, or bypass a firewall.</LI>
<LI>CAT II: provide information that have a high potential of giving access to an intruder.</LI>
<LI>CAT III: provide information that potentially could lead to compromise.</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

Name: STIG 3.6
ID: 000CA760-0FED-4374-8AA2-6FA3968A07B1
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APP<I>ID</I>: CAT <I>SEV</I>]. DISA STIG identifies several severities with respect to vulnerabilities:

<LI>CAT I: allow an attacker immediate access into a machine, allow super user access, or bypass a firewall.</LI>
<LI>CAT II: provide information that have a high potential of giving access to an intruder.</LI>
<LI>CAT III: provide information that potentially could lead to compromise.</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APP5080: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APP5100: CAT II is not considered "In Place".

Name: STIG 3.7
ID: E69C07C0-81D8-4B04-9233-F3E74167C3D2
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APP<I>ID</I>: CAT <I>SEV</I>]. DISA STIG identifies several severities with respect to vulnerabilities:

<LI>CAT I: allow an attacker immediate access into a machine, allow super user access, or bypass a firewall.</LI>
<LI>CAT II: provide information that have a high potential of giving access to an intruder.</LI>
<LI>CAT III: provide information that potentially could lead to compromise.</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APP5080: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APP5100: CAT II is not considered "In Place".

Name: STIG 3.9
ID: 1A9D736B-2D4A-49D1-88CA-DF464B40D732
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APP<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APP5080: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APP5100: CAT II is not considered "In Place".

Name: STIG 4.1
ID: 95227C50-A9E4-4C9D-A8AF-FD98ABAE1F3C
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.10
ID: EF1FF442-1673-4CF1-B7C4-920F1A96A8150
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>].

DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.2
ID: 672C15F8-8822-4E05-8C9E-1A4BAAA7A373
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>].
DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.3
ID: A0B313F0-29BD-430B-9E34-6D10F1178506
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>].
DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.4

ID: ECEC5CA2-7ACA-4B70-BF44-3248B9C6F4F8
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.5
ID: E6010E0A-7F71-4388-B8B7-EE9A02143474
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.6
ID: EFB9B012-44D6-456D-B197-03D2FD7C7AD6
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930:

CAT II are not considered "In Place".

Name: STIG 4.7
ID: B04A1E01-F1C1-48D3-A827-0F70872182D7
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.8
ID: E6805D9F-D5B5-4192-962C-46828FF68507
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.9
ID: 7B9F7B3B-07FC-4B61-99A1-70E3BB23A6A0
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: WASC 2.00
ID: 74f8081d-dd49-49da-880f-6830cebe9777
The Web Application Security Consortium (WASC) was created as a cooperative effort to standardize, clarify, and organize the threats to the security of a web site. Version 2.00 of their Threat Classification outlines the attacks and weaknesses that can commonly lead to a website being compromised.

Name: WASC 24 + 2
ID: 9DC61E7F-1A48-4711-BBFD-E9DFF537871F
The Web Application Security Consortium (WASC) was created as a cooperative effort to standardize, clarify, and organize the threats to the security of a web site.

Properties

WinForms.CollectionMutationMonitor.Label=WinFormsDataSource
awt.toolkit=sun.awt.X11.XToolkit
com.fortify.AuthenticationKey=/home/pgupta25/.fortify/config/tools
com.fortify.Core=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core
com.fortify.InstallRoot=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0
com.fortify.InstallationUserName=pgupta25
com.fortify.SCAExecutablePath=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/bin/sourceanalyzer
com.fortify.TotalPhysicalMemory=8363917312
com.fortify.VS.RequireASPPrecompilation=true
com.fortify.WorkingDirectory=/home/pgupta25/.fortify
com.fortify.locale=en
com.fortify.sca.AddImpliedMethods=true
com.fortify.sca.AntCompilerClass=com.fortify.dev.ant.SCACompiler
com.fortify.sca.AppendLogFile=true
com.fortify.sca.BuildID=appointmentscheduling
com.fortify.sca.BundleControlflowIssues=true
com.fortify.sca.BytecodePreview=true
com.fortify.sca.CollectPerformanceData=true
com.fortify.sca.CustomRulesDir=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/config/customrules
com.fortify.sca.DaemonCompilers=com.fortify.sca.util.compilers.GppCompiler,com.fortify.sca.util.compilers.GccCompiler,com.fortify.sca.util.compilers.AppleGppCompiler,com.fortify.sca.util.compilers.AppleGccCompiler,com.fortify.sca.util.compilers.MicrosoftCompiler,com.fortify.sca.util.compilers.MicrosoftLinker,com.fortify.sca.util.compilers.LdCompiler,com.fortify.sca.util.compilers.ArUtil,com.fortify.sca.util.compilers.SunCCompiler,com.fortify.sca.util.compilers.SunCppCompiler,com.fortify.sca.util.compilers.IntelCompiler,com.fortify.sca.util.compilers.ExternalCppAdapter,com.fortify.sca.util.compilers.ClangCompiler
com.fortify.sca.DeadCodeFilter=true
com.fortify.sca.DeadCodeIgnoreTrivialPredicates=true
com.fortify.sca.DefaultAnalyzers=semantic:dataflow:controlflow:nullptr:configuration:content:structural:buffer
com.fortify.sca.DefaultFileTypes=java,rb,erb,jsp,jspx,jspf,tag,tagx,tld,sql,cfm,php,phtml,ctp,pks,pkh,pkb,xml,config,Config,settings,properties,dll,exe,winmd,cs,vb,asax,ascx,ashx,asmx,aspx,master,Master,xaml,baml,cshtml,vbhtml,inc,asp,vbscript,js,ini,bas,cls,vbs,frm,ctl,html,htm,xsd,wsdd,xmi,py,cfml,cfc,abap,xhtml,cpx,xcfg,jsff,as,mxml,cbl,cscfg,csdef,wadcfg,wadcfgx,appxmanifest,wsdl,plist,bsp,ABAP,BSP,swift,page,trigger,scala,ts,conf,json,yaml,yml
com.fortify.sca.DefaultJarsDirs=default_jars

com.fortify.sca.DefaultRulesDir=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/config/rules
com.fortify.sca.DisableDeadCodeElimination=false
com.fortify.sca.DisableFunctionPointers=false
com.fortify.sca.DisableGlobals=false
com.fortify.sca.DisableInferredConstants=false
com.fortify.sca.EnableInterproceduralConstantResolution=true
com.fortify.sca.EnableNestedWrappers=true
com.fortify.sca.EnableStructuralMatchCache=true
com.fortify.sca.EnableWrapperDetection=true
com.fortify.sca.FVDLDisableDescriptions=false
com.fortify.sca.FVDLDisableProgramData=false
com.fortify.sca.FVDLDisableSnippets=false
com.fortify.sca.FVDLStylesheet=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/resources/sca/fvdl2html.xsl
com.fortify.sca.IndirectCallGraphBuilders=WinFormsAdHocFunctionBuilder,VirtualCGBuilder,J2EEIndirectCGBuilder,JNICGBuilder,StoredProcedureResolver,JavaWSCGBuilder,StrutsCGBuilder,DotNetWSCGBuilder,SqlServerSPResolver,ASPCGBuilder,ScriptedCGBuilder,NewJspCustomTagCGBuilder,DotNetCABCGBuilder,StateInjectionCGBuilder,SqlServerSPResolver2,PHPLambdaResolver,JavaWebCGBuilder
com.fortify.sca.JVMArgs=-XX:SoftRefLRUPolicyMSPerMB=3000 -Xmx4096M -Xss16M
com.fortify.sca.JavaSourcepathSearch=true
com.fortify.sca.JdkVersion=1.8
com.fortify.sca.LogFileDir=/home/pgupta25/.fortify/sca19.1/log
com.fortify.sca.LogFileExt=.log
com.fortify.sca.LogFileName=sca.log
com.fortify.sca.LogFileNameNoExt=sca
com.fortify.sca.LogFilePath=/home/pgupta25/.fortify/sca19.1/log/sca.log
com.fortify.sca.LogLevel=INFO
com.fortify.sca.LowSeverityCutoff=1.0
com.fortify.sca.MachineOutputMode=
com.fortify.sca.MultithreadedAnalysis=true
com.fortify.sca.NoNestedOutTagOutput=org.apache.taglibs.standard.tag.rt.core.RemoveTag,org.apache.taglibs.standard.tag.rt.core.SetTag
com.fortify.sca.OldVbNetExcludeFileTypes=vb,asax,ascx,ashx,asmx,aspx,xaml,cshtml,vbhtml
com.fortify.sca.PID=30642
com.fortify.sca.Phase0HigherOrder.Languages=python,ruby,swift,javascript,typescript
com.fortify.sca.Phase0HigherOrder.Level=1
com.fortify.sca.PrintPerformanceDataAfterScan=false
com.fortify.sca.ProjectRoot=/home/pgupta25/.fortify
com.fortify.sca.ProjectRoot=/home/pgupta25/.fortify
com.fortify.sca.Renderer=fpr
com.fortify.sca.RequireMapKeys=classrule
com.fortify.sca.ResultsFile=/srv/openmrs_code/org/openmrs/module/appointmentscheduling/appointmentscheduling_scan.fpr
com.fortify.sca.SolverTimeout=15
com.fortify.sca.SqlLanguage=PLSQL
com.fortify.sca.SuppressLowSeverity=true
com.fortify.sca.ThreadCount.NameTableLoading=1
com.fortify.sca.TypeInferenceFunctionTimeout=60
com.fortify.sca.TypeInferenceLanguages=javascript,typescript,python,ruby
com.fortify.sca.TypeInferencePhase0Timeout=300
com.fortify.sca.UnicodeInputFile=true
com.fortify.sca.UniversalBlacklist=.*yyparse.*
com.fortify.sca.alias.mode.csharp=fs

com.fortify.sca.alias.mode.javascript=fi
com.fortify.sca.alias.mode.scala=fi
com.fortify.sca.alias.mode.swift=fi
com.fortify.sca.alias.mode.typescript=fi
com.fortify.sca.alias.mode.vb=fs
com.fortify.sca.analyzer.controlflow.EnableLivenessOptimization=false
com.fortify.sca.analyzer.controlflow.EnableMachineFiltering=false
com.fortify.sca.analyzer.controlflow.EnableRefRuleOptimization=false
com.fortify.sca.analyzer.controlflow.EnableTimeOut=true
com.fortify.sca.compilers.ant=com.fortify.sca.util.compilers.AntAdapter
com.fortify.sca.compilers.ar=com.fortify.sca.util.compilers.ArUtil
com.fortify.sca.compilers.armcc=com.fortify.sca.util.compilers.ArmCcCompiler
com.fortify.sca.compilers.armcpp=com.fortify.sca.util.compilers.ArmCppCompiler
com.fortify.sca.compilers.c++=com.fortify.sca.util.compilers.GppCompiler
com.fortify.sca.compilers.cc=com.fortify.sca.util.compilers.GccCompiler
com.fortify.sca.compilers.clearmake=com.fortify.sca.util.compilers.TouchlessCompiler
com.fortify.sca.compilers.fortify=com.fortify.sca.util.compilers.FortifyCompiler
com.fortify.sca.compilers.g++=com.fortify.sca.util.compilers.GppCompiler
com.fortify.sca.compilers.g++-*=com.fortify.sca.util.compilers.GppCompiler
com.fortify.sca.compilers.g++2*=com.fortify.sca.util.compilers.GppCompiler
com.fortify.sca.compilers.g++3*=com.fortify.sca.util.compilers.GppCompiler
com.fortify.sca.compilers.g++4*=com.fortify.sca.util.compilers.GppCompiler
com.fortify.sca.compilers.gcc=com.fortify.sca.util.compilers.GccCompiler
com.fortify.sca.compilers.gcc-*=com.fortify.sca.util.compilers.GccCompiler
com.fortify.sca.compilers.gcc2*=com.fortify.sca.util.compilers.GccCompiler
com.fortify.sca.compilers.gcc3*=com.fortify.sca.util.compilers.GccCompiler
com.fortify.sca.compilers.gcc4*=com.fortify.sca.util.compilers.GccCompiler
com.fortify.sca.compilers.gmake=com.fortify.sca.util.compilers.TouchlessCompiler
com.fortify.sca.compilers.gradle=com.fortify.sca.util.compilers.GradleAdapter
com.fortify.sca.compilers.gradlew=com.fortify.sca.util.compilers.GradleAdapter
com.fortify.sca.compilers.icc=com.fortify.sca.util.compilers.IntelCompiler
com.fortify.sca.compilers.icpc=com.fortify.sca.util.compilers.IntelCompiler
com.fortify.sca.compilers.jam=com.fortify.sca.util.compilers.TouchlessCompiler
com.fortify.sca.compilers.javac=com.fortify.sca.util.compilers.JavacCompiler
com.fortify.sca.compilers.ld=com.fortify.sca.util.compilers.LdCompiler
com.fortify.sca.compilers.make=com.fortify.sca.util.compilers.TouchlessCompiler
com.fortify.sca.compilers.mvn=com.fortify.sca.util.compilers.MavenAdapter
com.fortify.sca.compilers.scalac=com.fortify.sca.util.compilers.ScalacCompiler
com.fortify.sca.compilers.tcc=com.fortify.sca.util.compilers.ArmCcCompiler
com.fortify.sca.compilers.tcpp=com.fortify.sca.util.compilers.ArmCppCompiler
com.fortify.sca.compilers.touchless=com.fortify.sca.util.compilers.FortifyCompiler
com.fortify.sca.cpfe.441.command=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/private-bin/sca/cpfe441.rfct
com.fortify.sca.cpfe.command=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/private-bin/sca/cpfe48
com.fortify.sca.cpfe.file.option=--gen_c_file_name
com.fortify.sca.cpfe.options=--remove_unneeded_entities --suppress_vtbl -tused
com.fortify.sca.cpfe.options=--remove_unneeded_entities --suppress_vtbl -tused
com.fortify.sca.env.exesearchpath=/sbin:/bin:/usr/bin:/usr/local/bin
com.fortify.sca.fileextensions.ABAP=ABAP
com.fortify.sca.fileextensions.BSP=ABAP
com.fortify.sca.fileextensions.Config=XML
com.fortify.sca.fileextensions.abap=ABAP

com.fortify.sca.fileextensions.appxmanifest=XML
com.fortify.sca.fileextensions.as=ACTIONSCRIPT
com.fortify.sca.fileextensions.asp=ASP
com.fortify.sca.fileextensions.bas=VB6
com.fortify.sca.fileextensions.bsp=ABAP
com.fortify.sca.fileextensions.cfc=CFML
com.fortify.sca.fileextensions.cfm=CFML
com.fortify.sca.fileextensions.cfml=CFML
com.fortify.sca.fileextensions.cls=VB6
com.fortify.sca.fileextensions.conf=HOCON
com.fortify.sca.fileextensions.config=XML
com.fortify.sca.fileextensions.cpx=XML
com.fortify.sca.fileextensions.cscfg=XML
com.fortify.sca.fileextensions.csdef=XML
com.fortify.sca.fileextensions.ctl=VB6
com.fortify.sca.fileextensions.ctp=PHP
com.fortify.sca.fileextensions.erb=RUBY_ERB
com.fortify.sca.fileextensions.faces=JSPX
com.fortify.sca.fileextensions.frm=VB6
com.fortify.sca.fileextensions.htm=HTML
com.fortify.sca.fileextensions.html=HTML
com.fortify.sca.fileextensions.ini=JAVA_PROPERTIES
com.fortify.sca.fileextensions.java=JAVA
com.fortify.sca.fileextensions.js=TYPESCRIPT
com.fortify.sca.fileextensions.jsff=JSPX
com.fortify.sca.fileextensions.json=JSON
com.fortify.sca.fileextensions.jsp=JSP
com.fortify.sca.fileextensions.jspf=JSP
com.fortify.sca.fileextensions.jspx=JSPX
com.fortify.sca.fileextensions.jsx=TYPESCRIPT
com.fortify.sca.fileextensions.mxml=MXML
com.fortify.sca.fileextensions.page=VISUAL_FORCE
com.fortify.sca.fileextensions.php=PHP
com.fortify.sca.fileextensions.phtml=PHP
com.fortify.sca.fileextensions.pkb=PLSQL
com.fortify.sca.fileextensions.pkh=PLSQL
com.fortify.sca.fileextensions.pks=PLSQL
com.fortify.sca.fileextensions.plist=XML
com.fortify.sca.fileextensions.properties=JAVA_PROPERTIES
com.fortify.sca.fileextensions.py=PYTHON
com.fortify.sca.fileextensions.rb=RUBY
com.fortify.sca.fileextensions.scala=SCALA
com.fortify.sca.fileextensions.settings=XML
com.fortify.sca.fileextensions.sql=SQL
com.fortify.sca.fileextensions.swift=SWIFT
com.fortify.sca.fileextensions.tag=JSP
com.fortify.sca.fileextensions.tagx=JSP
com.fortify.sca.fileextensions.tld=TLD
com.fortify.sca.fileextensions.trigger=APEX_TRIGGER
com.fortify.sca.fileextensions.ts=TYPESCRIPT
com.fortify.sca.fileextensions.tsx=TYPESCRIPT

com.fortify.sca.fileextensions.vbs=VBSCRIPT
com.fortify.sca.fileextensions.vbscript=VBSCRIPT
com.fortify.sca.fileextensions.wadcfg=XML
com.fortify.sca.fileextensions.wadcfgx=XML
com.fortify.sca.fileextensions.wsdd=XML
com.fortify.sca.fileextensions.wsdl=XML
com.fortify.sca.fileextensions.xcfg=XML
com.fortify.sca.fileextensions.xhtml=JSPX
com.fortify.sca.fileextensions.xmi=XML
com.fortify.sca.fileextensions.xml=XML
com.fortify.sca.fileextensions.xsd=XML
com.fortify.sca.fileextensions.yaml=YAML
com.fortify.sca.fileextensions.yml=YAML
com.fortify.sca.jsp.UseNativeParser=true
com.fortify.sca.parser.python.ignore.module.1=test.badsyntax_future3
com.fortify.sca.parser.python.ignore.module.2=test.badsyntax_future4
com.fortify.sca.parser.python.ignore.module.3=test.badsyntax_future5
com.fortify.sca.parser.python.ignore.module.4=test.badsyntax_future6
com.fortify.sca.parser.python.ignore.module.5=test.badsyntax_future7
com.fortify.sca.parser.python.ignore.module.6=test.badsyntax_future8
com.fortify.sca.parser.python.ignore.module.7=test.badsyntax_future9
com.fortify.sca.parser.python.ignore.module.8=test.badsyntax_nocaret
com.fortify.sca.skip.libraries.AngularJS=angular.js,angular.min.js,angular-animate.js,angular-aria.js,angular_1_router.js,angular-cookies.js,angular-message-format.js,angular-messages.js,angular-mocks.js,angular-parse-ext.js,angular-resource.js,angular-route.js,angular-sanitize.js,angular-touch.js
com.fortify.sca.skip.libraries.ES6=es6-shim.min.js,system-polyfills.js,shims_for_IE.js
com.fortify.sca.skip.libraries.jQuery=jquery.js,jquery.min.js,jquery-migrate.js,jquery-migrate.min.js,jquery-ui.js,jquery-ui.min.js,jquery.mobile.js,jquery.mobile.min.js,jquery.color.js,jquery.color.min.js,jquery.color.svg-names.js,jquery.color.svg-names.min.js,jquery.color.plus-names.js,jquery.color.plus-names.min.js,jquery.tools.min.js
com.fortify.sca.skip.libraries.javascript=bootstrap.js,bootstrap.min.js,typescript.js,typescriptServices.js
com.fortify.sca.skip.libraries.typescript=typescript.d.ts,typescriptServices.d.ts
com.fortify.search.defaultSyntaxVer=2
com.sun.management.jmxremote=true
file.encoding=UTF-8
file.encoding.pkg=sun.io
file.separator=/
java.awt.graphicsenv=sun.awt.X11GraphicsEnvironment
java.awt.headless=true
java.awt.printerjob=sun.print.PSPrinterJob
java.class.path=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/lib/exe/sca-exe.jar
java.class.version=52.0
java.endorsed.dirs=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/endorsed
java.ext.dirs=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/ext:/usr/java/packages/lib/ext
java.home=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre
java.io.tmpdir=/tmp
java.library.path=/usr/java/packages/lib/amd64:/usr/lib64:/lib64:/lib:/usr/lib
java.rmi.server.randomIDs=true
java.runtime.name=OpenJDK Runtime Environment
java.runtime.version=1.8.0_181-b02
java.specification.name=Java Platform API Specification
java.specification.vendor=Oracle Corporation

java.specification.version=1.8
java.vendor=Azul Systems, Inc.
java.vendor.url=http://www.azulsystems.com/
java.vendor.url.bug=http://www.azulsystems.com/support/
java.version=1.8.0_181
java.vm.info=mixed mode
java.vm.name=OpenJDK 64-Bit Server VM
java.vm.specification.name=Java Virtual Machine Specification
java.vm.specification.vendor=Oracle Corporation
java.vm.specification.version=1.8
java.vm.vendor=Azul Systems, Inc.
java.vm.version=25.181-b02
line.separator=

log4j.configurationFile=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/config/log4j2.xml
log4j.isThreadContextMapInheritable=true
max.file.path.length=255
os.arch=amd64
os.name=Linux
os.version=4.15.0-58-generic
path.separator=:
stderr.isatty=false
stdout.isatty=false
sun.arch.data.model=64
sun.boot.class.path=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/resources.jar:/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/rt.jar:/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/sunrsasign.jar:/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/jsse.jar:/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/jce.jar:/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/charsets.jar:/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/jfr.jar:/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/classes
sun.boot.library.path=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/amd64
sun.cpu.endian=little
sun.cpu.isalist=
sun.io.unicode.encoding=UnicodeLittle
sun.java.command=sourceanalyzer -Djava.awt.headless=true -Dcom.sun.management.jmxremote=true -XX:SoftRefLRUPolicyMSPerMB=3000 -Dcom.fortify.sca.env.exesearchpath=/sbin:/bin:/usr/bin:/usr/local/bin -Dcom.fortify.sca.ProjectRoot=/home/pgupta25/.fortify -Dstdout.isatty=false -Dstderr.isatty=false -Dcom.fortify.sca.PID=30642 -Xmx4096M -Dcom.fortify.TotalPhysicalMemory=8363917312 -Xss16M -Dcom.fortify.sca.JVMArgs=-XX:SoftRefLRUPolicyMSPerMB=3000 -Xmx4096M -Xss16M -Djava.class.path=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/lib/exe/sca-exe.jar -scan @/home/pgupta25/.fortify/Eclipse.Plugin-19.1.0/appointmentscheduling/appointmentschedulingScan.txt
sun.jnu.encoding=UTF-8
sun.management.compiler=HotSpot 64-Bit Tiered Compilers
sun.os.patch.level=unknown
user.country=US
user.dir=/home/pgupta25
user.home=/home/pgupta25
user.language=en
user.name=pgupta25
user.timezone=America/New_York

Commandline Arguments

-scan
-b
appointmentscheduling
-format
fpr
-machine-output
-f
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/appointmentscheduling_scan.fpr

Warnings

[12002] Could not locate the deployment descriptor (web.xml) for your web application. Please build your web application and try again. File: /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentBlockCalendar.jsp
[12003] Assuming Java source level to be 1.8 as it was not specified. Note that the default value may change in future versions.
[12004] The Java frontend was unable to resolve the following include: /WEB-INF/template/include.jsp at /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/portlets/appointments.jsp:1. /WEB-INF/template/footer.jsp at /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentBlockList.jsp:530. /WEB-INF/template/header.jsp at /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentBlockList.jsp:2.
[12004] The ActionScript frontend was unable to resolve the following import: flash.display at /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/resources/TableTools/media/ZeroClipboard/ZeroClipboard.as:2. flash.events at /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/resources/TableTools/media/ZeroClipboard/ZeroClipboard.as:6. flash.net at /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/resources/TableTools/media/ZeroClipboard/ZeroClipboard.as:13. flash.external at /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/resources/TableTools/media/ZeroClipboard/ZeroClipboard.as:9. flash.system at /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/resources/TableTools/media/ZeroClipboard/ZeroClipboard.as:10. flash.utils at /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/resources/TableTools/media/ZeroClipboard/ZeroClipboard.as:11.
[12010] You may need to specify additional SWC or SWF Flex libraries (-flex-libraries option, or com.fortify.sca.FlexLibraries property)
[12022] The class "javax.servlet.http.HttpServlet" could not be found on the classpath, but it was found in the JAR file provided by Fortify in "/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/default_jars/javax.servlet-api-3.0.1.jar" as a convenience. To ensure consistent translation behavior add the JAR file that contains "javax.servlet.http.HttpServlet" to the classpath given to the translation step. Refer to the documentation about "default JARs" in the SCA User Guide for more information.
[12022] The class "javax.servlet.jsp.PageContext" could not be found on the classpath, but it was found in the JAR file provided by Fortify in "/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/default_jars/javax.servlet.jsp-api.jar" as a convenience. To ensure consistent translation behavior add the JAR file that contains "javax.servlet.jsp.PageContext" to the classpath given to the translation step. Refer to the documentation about "default JARs" in the SCA User Guide for more information.
[1214] Multiple definitions found for class /appointmentBlockCalendar.jsp (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentBlockCalendar.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentBlockCalendar.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentBlockCalendar_jsp$ftfy_frameworkVisibleObjects

(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentBlockCalendar.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentBlockCalendar.jsp).
[1214] Multiple definitions found for class /appointmentBlockForm.jsp (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentBlockForm.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentBlockForm.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentBlockForm_jsp$ftfy_frameworkVisibleObjects (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentBlockForm.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentBlockForm.jsp).
[1214] Multiple definitions found for class /appointmentBlockList.jsp (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentBlockList.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentBlockList.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentBlockList_jsp$ftfy_frameworkVisibleObjects (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentBlockList.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentBlockList.jsp).
[1214] Multiple definitions found for class /appointmentForm.jsp (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentForm.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentForm.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentForm_jsp$ftfy_frameworkVisibleObjects (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentForm.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentForm.jsp).
[1214] Multiple definitions found for class /appointmentList.jsp (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentList.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentList.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentList_jsp$ftfy_frameworkVisibleObjects (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentList.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentList.jsp).
[1214] Multiple definitions found for class /appointmentSettingsForm.jsp (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentSettingsForm.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentSettingsForm.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentSettingsForm_jsp$ftfy_frameworkVisibleObjects (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentSettingsForm.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentSettingsForm.jsp).
[1214] Multiple definitions found for class /appointmentStatisticsForm.jsp (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentStatisticsForm.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentStatisticsForm.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentStatisticsForm_jsp$ftfy_frameworkVisibleObjects (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentStatisticsForm.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentStatisticsForm.jsp).

[1214] Multiple definitions found for class /appointmentTypeForm.jsp (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentTypeForm.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentTypeForm.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentTypeForm_jsp$ftfy_frameworkVisibleObjects (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentTypeForm.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentTypeForm.jsp).
[1214] Multiple definitions found for class /appointmentTypeList.jsp (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentTypeList.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentTypeList.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentTypeList_jsp$ftfy_frameworkVisibleObjects (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentTypeList.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/appointmentTypeList.jsp).
[1214] Multiple definitions found for class /appointments.jsp (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/portlets/appointments.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/portlets/appointments.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointments_jsp$ftfy_frameworkVisibleObjects (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/portlets/appointments.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/portlets/appointments.jsp).
[1214] Multiple definitions found for class /localHeader.jsp (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/template/localHeader.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/template/localHeader.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jsplocalHeader_jsp$ftfy_frameworkVisibleObjects (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/template/localHeader.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/appointmentscheduling-1.10.0/web/module/template/localHeader.jsp).
[1214] Multiple definitions found for class /appointmentBlockCalendar.jsp (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentBlockCalendar.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentBlockCalendar.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentBlockCalendar_jsp$ftfy_frameworkVisibleObjects (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentBlockCalendar.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentBlockCalendar.jsp).
[1214] Multiple definitions found for class /appointmentBlockForm.jsp (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentBlockForm.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentBlockForm.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentBlockForm_jsp$ftfy_frameworkVisibleObjects (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentBlockForm.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentBlockForm.jsp).
[1214] Multiple definitions found for class /appointmentBlockList.jsp (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentBlockList.jsp and /srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentBlockList.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentBlockList_jsp$ftfy_frameworkVisibleObjects (/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentBlockList.jsp and

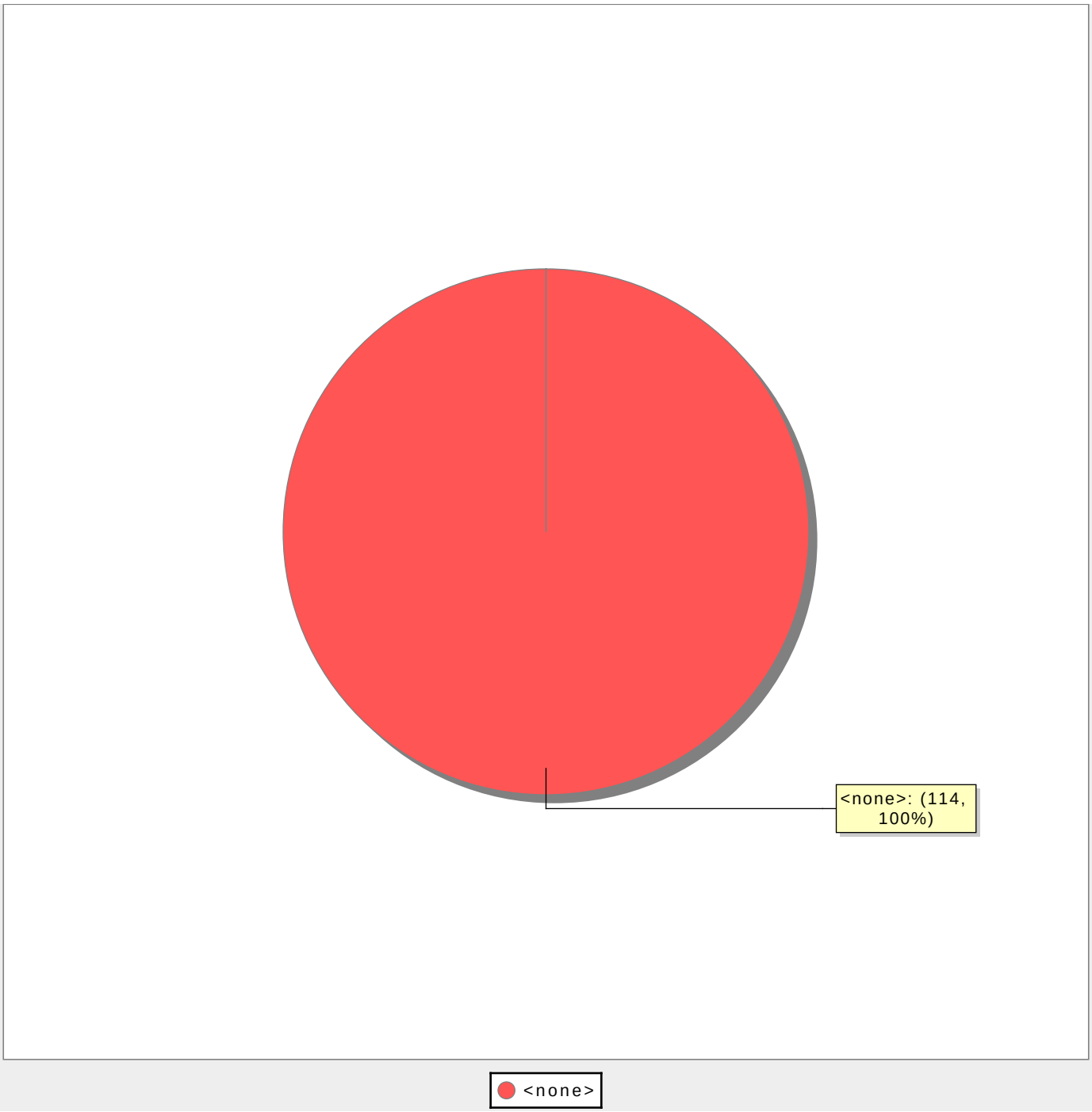
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentBlockList.jsp).
[1214] Multiple definitions found for class /appointmentForm.jsp
(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentForm.jsp and
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentForm.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentForm_jsp$ftfy_frameworkVisibleObjects
(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentForm.jsp and
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentForm.jsp).
[1214] Multiple definitions found for class /appointmentList.jsp
(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentList.jsp and
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentList.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentList_jsp$ftfy_frameworkVisibleObjects
(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentList.jsp and
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentList.jsp).
[1214] Multiple definitions found for class /appointmentSettingsForm.jsp
(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentSettingsForm.jsp and
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentSettingsForm.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentSettingsForm_jsp$ftfy_frameworkVisibleObjects
(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentSettingsForm.jsp and
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentSettingsForm.jsp).
[1214] Multiple definitions found for class /appointmentStatisticsForm.jsp
(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentStatisticsForm.jsp and
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentStatisticsForm.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentStatisticsForm_jsp$ftfy_frameworkVisibleObjects
(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentStatisticsForm.jsp and
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentStatisticsForm.jsp).
[1214] Multiple definitions found for class /appointmentTypeForm.jsp
(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentTypeForm.jsp and
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentTypeForm.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentTypeForm_jsp$ftfy_frameworkVisibleObjects
(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentTypeForm.jsp and
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentTypeForm.jsp).
[1214] Multiple definitions found for class /appointmentTypeList.jsp
(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentTypeList.jsp and
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentTypeList.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointmentTypeList_jsp$ftfy_frameworkVisibleObjects
(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/appointmentTypeList.jsp and
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/appointmentTypeList.jsp).
[1214] Multiple definitions found for class /appointments.jsp
(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/portlets/appointments.jsp and
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/portlets/appointments.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspappointments_jsp$ftfy_frameworkVisibleObjects
(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/portlets/appointments.jsp and
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/portlets/appointments.jsp).
[1214] Multiple definitions found for class /localHeader.jsp
(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/template/localHeader.jsp and
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/template/localHeader.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jsplocalHeader_jsp$ftfy_frameworkVisibleObjects
(/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/src/main/webapp/template/localHeader.jsp and
/srv/openmrs_code/org/openmrs/module/appointmentscheduling/omod/target/classes/web/module/template/localHeader.jsp).
[1215] Could not locate the root (WEB-INF) of the web application. Please build your web application and try again.

Issue Count by Category

Issues by Category	
Cross-Site Scripting: DOM	54
Cross-Site Scripting: Reflected	36
Header Manipulation: Cookies	12
Dynamic Code Evaluation: Code Injection	6
Key Management: Hardcoded Encryption Key	3
Open Redirect	3

Issue Breakdown by Analysis

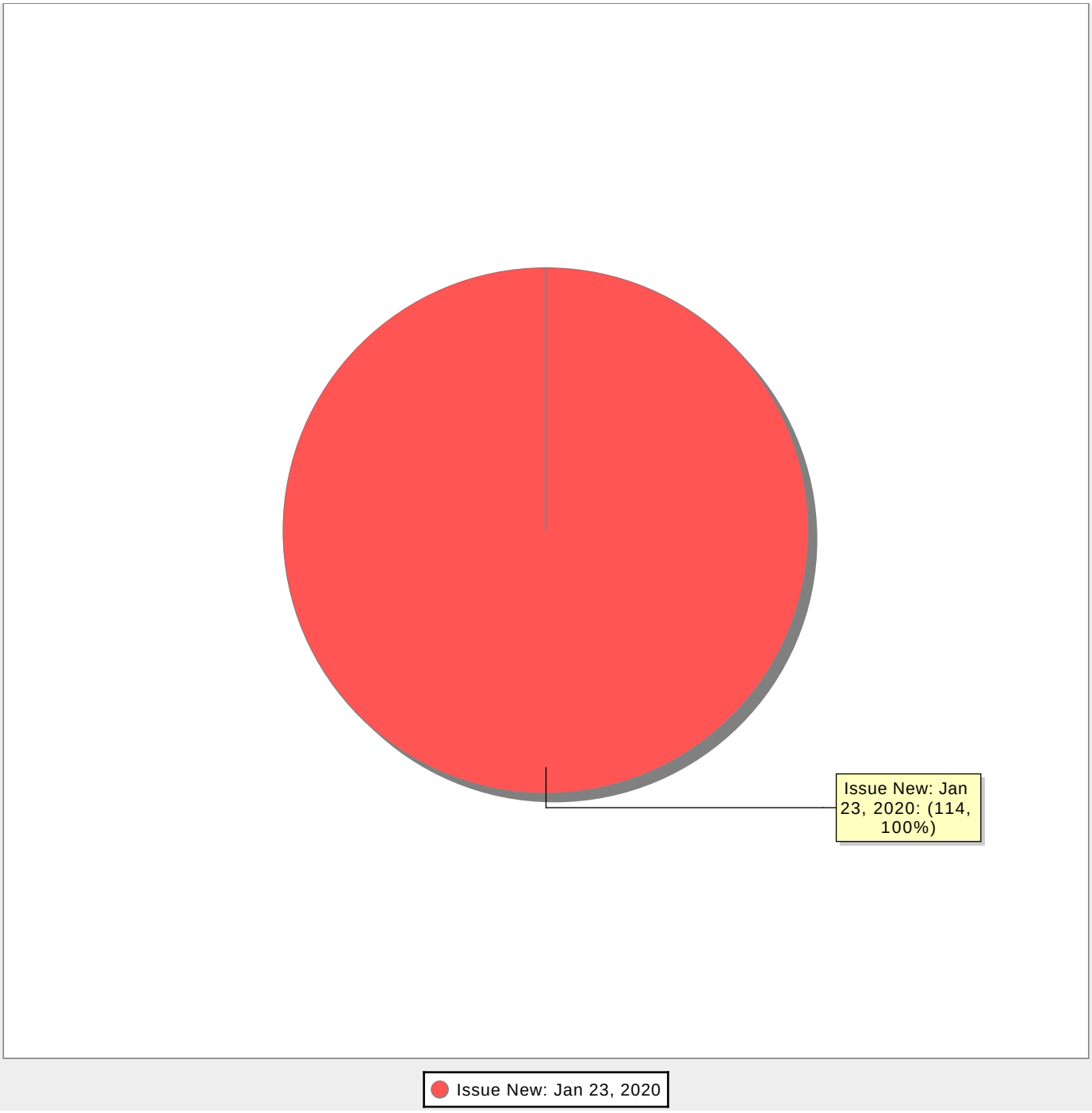
Issues by Analysis



New Issues

Issues by New Issue

The following issues have been discovered since the last scan.



Fortify Security Report

Apr 17, 2020

agautam6

Executive Summary

Issues Overview

On Apr 17, 2020, a source code review was performed over the owa code base. 100 files, 3,913 LOC (Executable) were scanned and reviewed for defects that could lead to potential security vulnerabilities. A total of 14 reviewed findings were uncovered during the analysis.

Issues by Fortify Priority Order	
Critical	8
High	6

Recommendations and Conclusions

The Issues Category section provides Fortify recommendations for addressing issues at a generic level. The recommendations for specific fixes can be extrapolated from those generic recommendations by the development group.

Project Summary

Code Base Summary

Code location: /srv/openmrs_code/org/openmrs/module/owa
Number of Files: 100
Lines of Code: 3913
Build Label: <No Build Label>

Scan Information

Scan time: 15:13
SCA Engine version: 19.1.0.2241
Machine Name: vm18-85.vcl.ncsu.edu
Username running scan: agautam6

Results Certification

Results Certification Valid

Details:

Results Signature:

SCA Analysis Results has Valid signature

Rules Signature:

There were no custom rules used in this scan

Attack Surface

Attack Surface:
File System:
java.io.FileInputStream.FileInputStream
java.io.FileInputStream.FileInputStream
java.util.zip.ZipFile.getEntry

Stream:
java.io.RandomAccessFile.read

System Information:
null.null.null
null.null.e
null.null.error
null.null.k
null.null.lambda
null.null.q
java.io.File.listFiles
java.lang.Throwable.getMessage
javax.servlet.ServletContext.getRealPath

Web:

javax.servlet.ServletRequest.getScheme

Filter Set Summary

Current Enabled Filter Set:
Quick View

Filter Set Details:

Folder Filters:
If [fortify priority order] contains critical Then set folder to Critical
If [fortify priority order] contains high Then set folder to High
If [fortify priority order] contains medium Then set folder to Medium
If [fortify priority order] contains low Then set folder to Low
Visibility Filters:
If impact is not in range [2.5, 5.0] Then hide issue
If likelihood is not in range (1.0, 5.0] Then hide issue

Audit Guide Summary

Audit guide not enabled

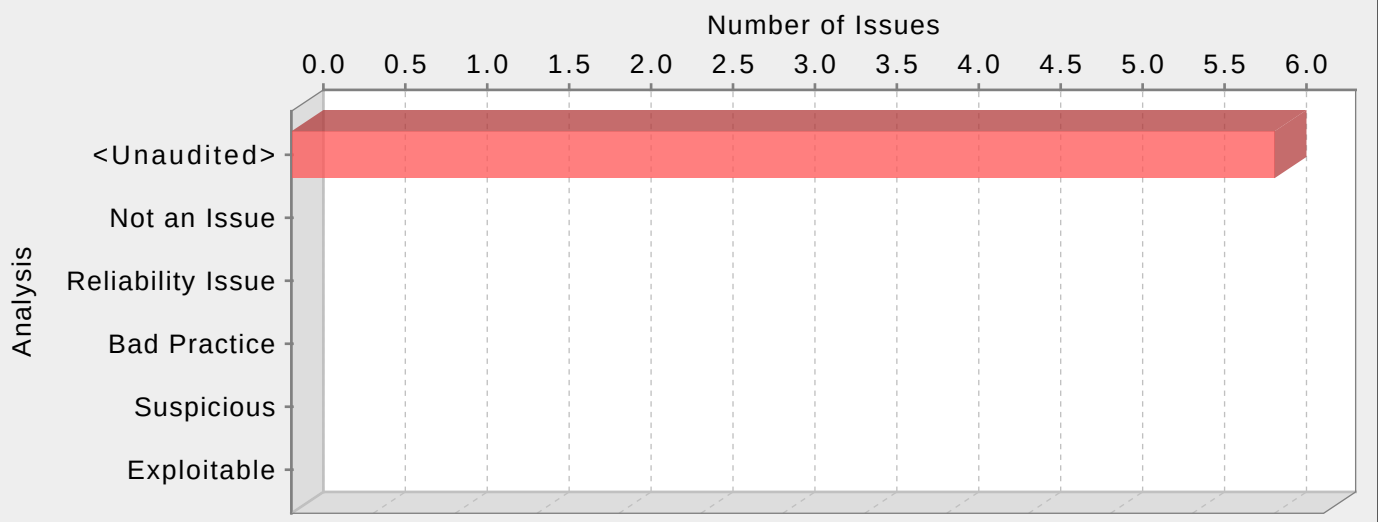
Results Outline

Overall number of results

The scan found 14 issues.

Vulnerability Examples by Category

Category: Cross-Site Scripting: DOM (6 Issues)



Abstract:

The method Aa() in angular-1.4.5.min.js sends unvalidated data to a web browser on line 146, which can result in the browser executing malicious code.

Explanation:

Cross-site scripting (XSS) vulnerabilities occur when:

1. Data enters a web application through an untrusted source. In the case of DOM-based XSS, data is read from a URL parameter or other value within the browser and written back into the page with client-side code. In the case of reflected XSS, the untrusted source is typically a web request, while in the case of persisted (also known as stored) XSS it is typically a database or other back-end data store.

2. The data is included in dynamic content that is sent to a web user without being validated. In the case of DOM Based XSS, malicious content gets executed as part of DOM (Document Object Model) creation, whenever the victim's browser parses the HTML page.

The malicious content sent to the web browser often takes the form of a segment of JavaScript, but may also include HTML, Flash or any other type of code that the browser executes. The variety of attacks based on XSS is almost limitless, but they commonly include transmitting private data like cookies or other session information to the attacker, redirecting the victim to web content controlled by the attacker, or performing other malicious operations on the user's machine under the guise of the vulnerable site.

Example 1: The following JavaScript code segment reads an employee ID, eid, from a URL and displays it to the user.

```
<SCRIPT>
var pos=document.URL.indexOf("eid=")+4;
document.write(document.URL.substring(pos,document.URL.length));
</SCRIPT>
```

Example 2: Consider the HTML form:

```
<div id="myDiv">
Employee ID: <input type="text" id="eid"><br>
...
<button>Show results</button>
</div>
<div id="resultsDiv">
...
</div>
```

The following jQuery code segment reads an employee ID from the form, and displays it to the user.

```
$(document).ready(function(){
$("#myDiv").on("click", "button", function(){
var eid = $("#eid").val();
$("resultsDiv").append(eid);
...
});
});
```

These code examples operate correctly if the employee ID, from the text input with ID eid contains only standard alphanumeric text. If eid has a value that includes meta-characters or source code, then the code will be executed by the web browser as it displays the HTTP response.

Example 3: The following code shows an example of a DOM-based XSS within a React application:

```
let element = JSON.parse(getUntrustedInput());
ReactDOM.render(<App>
{element}
</App>);
```

In Example 3, if an attacker can control the entire JSON object retrieved from getUntrustedInput(), they may be able to make React render element as a component, and therefore can pass an object with dangerouslySetInnerHTML with their own controlled value, a typical cross-site scripting attack.

Initially these might not appear to be much of a vulnerability. After all, why would someone provide input containing malicious code to run on their own computer? The real danger is that an attacker will create the malicious URL, then use email or social engineering tricks to lure victims into visiting a link to the URL. When victims click the link, they unwittingly reflect the malicious content through the vulnerable web application back to their own computers. This mechanism of exploiting vulnerable web applications is known as Reflected XSS.

As the example demonstrates, XSS vulnerabilities are caused by code that includes unvalidated data in an HTTP response. There are three vectors by which an XSS attack can reach a victim:

- Data is read directly from the HTTP request and reflected back in the HTTP response. Reflected XSS exploits occur when an attacker causes a user to supply dangerous content to a vulnerable web application, which is then reflected back to the user and executed by the web browser. The most common mechanism for delivering malicious content is to include it as a parameter in a URL that is posted publicly or emailed directly to victims. URLs constructed in this manner constitute the core of many phishing schemes, whereby an attacker convinces victims to visit a URL that refers to a vulnerable site. After the site reflects the attacker's content back to the user, the content is executed and proceeds to transfer private information, such as cookies that may include session information, from the user's machine to the attacker or perform other nefarious activities.

- The application stores dangerous data in a database or other trusted data store. The dangerous data is subsequently read back into the application and included in dynamic content. Persistent XSS exploits occur when an attacker injects dangerous content into a data store that is later read and included in dynamic content. From an attacker's perspective, the optimal place to inject malicious content is in an area that is displayed to either many users or particularly interesting users. Interesting users typically have elevated privileges in the application or interact with sensitive data that is valuable to the attacker. If one of these users executes malicious content, the attacker may be able to perform privileged operations on behalf of the user or gain access to sensitive data belonging to the user.

- A source outside the application stores dangerous data in a database or other data store, and the dangerous data is subsequently read back into the application as trusted data and included in dynamic content.

Recommendations:

The solution to XSS is to ensure that validation occurs in the correct places and checks are made for the correct properties.

Since XSS vulnerabilities occur when an application includes malicious data in its output, one logical approach is to validate data immediately before it leaves the application. However, because web applications often have complex and intricate code for generating dynamic content, this method is prone to errors of omission (missing validation). An effective way to mitigate this risk is to also perform input validation for XSS.

Web applications must validate their input to prevent other vulnerabilities, such as SQL injection, so augmenting an application's existing input validation mechanism to include checks for XSS is generally relatively easy. Despite its value, input validation for XSS does not take the place of rigorous output validation. An application may accept input through a shared data store or other trusted source, and that data store may accept input from a source that does not perform adequate input validation. Therefore, the application cannot implicitly rely on the safety of this or any other data. This means the best way to prevent XSS vulnerabilities is to validate everything that enters the application and leaves the application destined for the user.

The most secure approach to validation for XSS is to create a whitelist of safe characters that are allowed to appear in HTTP content and accept input composed exclusively of characters in the approved set. For example, a valid username might only include alpha-numeric characters or a phone number might only include digits 0-9. However, this solution is often infeasible in web applications because many characters that have special meaning to the browser should still be considered valid input once they are encoded, such as a web design bulletin board that must accept HTML fragments from its users.

A more flexible, but less secure approach is known as blacklisting, which selectively rejects or escapes potentially dangerous characters before using the input. In order to form such a list, you first need to understand the set of characters that hold special meaning for web browsers. Although the HTML standard defines what characters have special meaning, many web browsers try to correct common mistakes in HTML and may treat other characters as special in certain contexts, which is why we do not encourage the use of blacklists as a means to prevent XSS. The CERT(R) Coordination Center at the Software Engineering Institute at Carnegie Mellon University provides the following details about special characters in various contexts [1]:

In the content of a block-level element (in the middle of a paragraph of text):

- "<" is special because it introduces a tag.

- "&" is special because it introduces a character entity.

- ">" is special because some browsers treat it as special, on the assumption that the author of the page intended to include an opening "<", but omitted it in error.

The following principles apply to attribute values:

- In attribute values enclosed with double quotes, the double quotes are special because they mark the end of the attribute value.

- In attribute values enclosed with single quote, the single quotes are special because they mark the end of the attribute value.

- In attribute values without any quotes, white-space characters, such as space and tab, are special.

- "&" is special when used with certain attributes, because it introduces a character entity.

In URLs, for example, a search engine might provide a link within the results page that the user can click to re-run the search. This can be implemented by encoding the search query inside the URL, which introduces additional special characters:

- Space, tab, and new line are special because they mark the end of the URL.

- "&" is special because it either introduces a character entity or separates CGI parameters.

- Non-ASCII characters (that is, everything greater than 127 in the ISO-8859-1 encoding) are not allowed in URLs, so they are considered to be special in this context.

- The "%" symbol must be filtered from input anywhere parameters encoded with HTTP escape sequences are decoded by server -side code. For example, "%" must be filtered if input such as "%68%65%6C%6C%6F" becomes "hello" when it appears on the web page in question.

Within the body of a <SCRIPT> </SCRIPT>:

- Semicolons, parentheses, curly braces, and new line characters should be filtered out in situations where text could be inserted directly into a pre-existing script tag.

Server-side scripts:

- Server-side scripts that convert any exclamation characters (!) in input to double-quote characters (") on output might require additional filtering.

Other possibilities:

- If an attacker submits a request in UTF-7, the special character '<' appears as '+ADw-' and may bypass filtering. If the output is included in a page that does not explicitly specify an encoding format, then some browsers try to intelligently identify the encoding based on the content (in this case, UTF-7).

After you identify the correct points in an application to perform validation for XSS attacks and what special characters the validation should consider, the next challenge is to identify how your validation handles special characters. If special characters are not considered valid input to the application, then you can reject any input that contains special characters as invalid. A second option in this situation is to remove special characters with filtering. However, filtering has the side effect of changing any visual representation of the filtered content and may be unacceptable in circumstances where the integrity of the input must be preserved for display.

If input containing special characters must be accepted and displayed accurately, validation must encode any special characters to remove their significance. A complete list of ISO 8859-1 encoded values for special characters is provided as part of the official HTML specification [2].

Many application servers attempt to limit an application's exposure to cross-site scripting vulnerabilities by providing implementations for the functions responsible for setting certain specific HTTP response content that perform validation for the characters essential to a cross-site scripting attack. Do not rely on the server running your application to make it secure. When an application is developed there are no guarantees about what application servers it will run on during its lifetime. As standards and known exploits evolve, there are no guarantees that application servers will also stay in sync.

Tips:

1. The Fortify Secure Coding Rulepacks warn about SQL Injection and Access Control: Database issues when untrusted data is written to a database and also treat the database as a source of untrusted data, which can lead to XSS vulnerabilities. If the database is a trusted resource in your environment, use custom filters to filter out dataflow issues that include the DATABASE taint flag or originate from database sources. Nonetheless, it is often still a good idea to validate everything read from the database.

2. Even though URL encoding untrusted data protects against many XSS attacks, some browsers (specifically, Internet Explorer 6 and 7 and possibly others) automatically decode content at certain locations within the Document Object Model (DOM) prior to passing it to the JavaScript interpreter. To reflect this danger, the rulepacks no longer treat URL encoding routines as sufficient to protect against cross-site scripting. Data values that are URL encoded and subsequently output will cause Fortify to report Cross-Site Scripting: Poor Validation vulnerabilities.

3. Older versions of React are more susceptible to cross-site scripting attacks by controlling an entire component. Newer versions use Symbols to identify a React component, which prevents the exploit, however older browsers that do not have Symbol support (natively, or through polyfills), such as all versions of Internet Explorer, are still vulnerable. Other types of cross-site scripting attacks are valid for all browsers and versions of React.

angular-1.4.5.min.js, line 146 (Cross-Site Scripting: DOM)

Fortify Priority:	Critical	**Folder**	Critical
Kingdom:	Input Validation and Representation		
Abstract:	The method Aa() in angular-1.4.5.min.js sends unvalidated data to a web browser on line 146, which can result in the browser executing malicious code.		

Source: angular-1.4.5.min.js:146 Read Z.href()

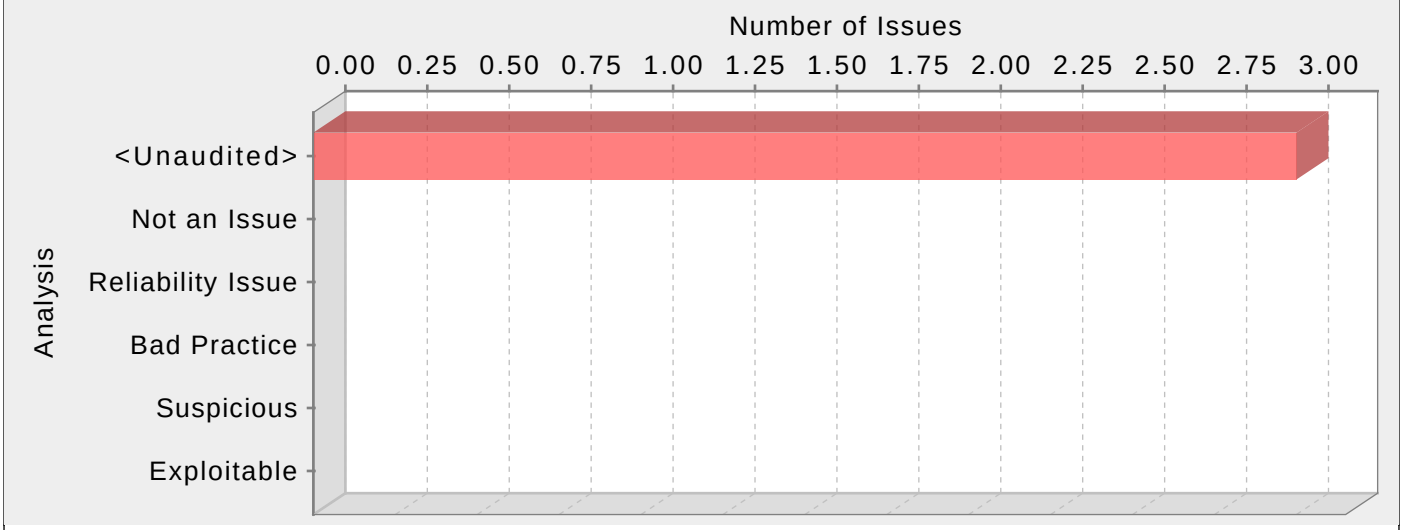
```
144  return e}]}function
     sf(){this.$get=["$rootScope","$browser","$location",function(b,a,c){return{findBinding
     s:function(a,b,c){a=a.getElementsByClassName("ng-binding");var
     g=[];n(a,function(a){var d=aa.element(a).data("$binding");d&&n(d,function(d){c?(new
     RegExp("(^|\\s)"+ud(b)+"(\\s|\\||$)")).test(d)&&g.push(a):-
     1!=d.indexOf(b)&&g.push(a)})});return g},findModels:function(a,b,c){for(var g=["ng-
     ","data-ng-","ng\\:"],h=0;h<g.length;++h){var
     l=a.querySelectorAll("["+g[h]+"model"+(c?"=":"*=")+'"'+b+'"...
145  if(l.length)return l}},getLocation:function(){return
     c.url()},setLocation:function(a){a!==c.url()&&(c.url(a),b.$digest())},whenStable:funct
     ion(b){a.notifyWhenNoOutstandingRequests(b)}}]}function
     tf(){this.$get=["$rootScope","$browser","$q","$$q","$exceptionHandler",function(b,a,c,
     d,e){function f(f,l,k){B(f)||(k=l,l=f,f=v);var
     m=xa.call(arguments,3),q=x(k)&&!k,s=(q?d:c).defer(),t=s.promise,n;n=a.defer(function()
     {try{s.resolve(f.apply(null,m))}catch(a){s.reject(a),e(a)}finally{delete
     g[t.$$timeou...
146  b.$apply()},l);t.$$timeoutId=n;g[n]=s;return t}var g={};f.cancel=function(b){return
     b&&b.$$timeoutId in g?(g[b.$$timeoutId].reject("canceled"),delete
     g[b.$$timeoutId],a.defer.cancel(b.$$timeoutId)):!1};return f}]}function
     Aa(b){Va&&(Z.setAttribute("href",b),b=Z.href);Z.setAttribute("href",b);return{href:Z.h
     ref,protocol:Z.protocol?Z.protocol.replace(/:$/,""):"",host:Z.host,search:Z.search?Z.s
     earch.replace(/^\?/,""):"",hash:Z.hash?Z.hash.replace(/^#/,""):"",hostname:Z.hostname,
     port:Z.port,pathname...
147  Z.pathname.charAt(0)?Z.pathname:"/"+Z.pathname}}function gd(b){b=H(b)?Aa(b):b;return
     b.protocol===wd.protocol&&b.host===wd.host}function uf(){this.$get=qa(N)}function
     xd(b){function a(a){try{return decodeURIComponent(a)}catch(b){return a}}var
     c=b[0]||{},d={},e="";return function(){var
     b,g,h,l,k;b=c.cookie||"";if(b!==e)for(e=b,b=e.split("
     "),d={},h=0;h<b.length;h++)g=b[h],l=g.indexOf("="),0<l&&(k=a(g.substring(0,l)),d[k]===
     u&&(d[k]=a(g.substring(l+1))));return d}}function yf(){this.$get=xd}funct...
148  d){if(D(c)){var e={};n(c,function(b,c){e[c]=a(c,b)});return e}return
     b.factory(c+"Filter",d)}this.register=a;this.$get=["$injector",function(a){return
     function(b){return
     a.get(b+"Filter")}}];a("currency",yd);a("date",zd);a("filter",$f);a("json",ag);a("limi
     tTo",bg);a("lowercase",cg);a("number",Ad);a("orderBy",Bd);a("uppercase",dg)}function
     $f(){return function(b,a,c){if(!Da(b)){if(null==b)return b;throw
     G("filter")("notarray",b);}var d;switch(gc(a)){case "function":break;case
     "boolean":case "null...
```

Sink: angular-1.4.5.min.js:146 ~JS_Generic.setAttribute()

```
144  return e}]}function
     sf(){this.$get=["$rootScope","$browser","$location",function(b,a,c){return{findBinding
     s:function(a,b,c){a=a.getElementsByClassName("ng-binding");var
     g=[];n(a,function(a){var d=aa.element(a).data("$binding");d&&n(d,function(d){c?(new
     RegExp("(^|\\s)"+ud(b)+"(\\s|\\||$)")).test(d)&&g.push(a):-
     1!=d.indexOf(b)&&g.push(a)})});return g},findModels:function(a,b,c){for(var g=["ng-
     ","data-ng-","ng\\:"],h=0;h<g.length;++h){var
     l=a.querySelectorAll("["+g[h]+"model"+(c?"=":"*=")+'"'+b+'"...
145  if(l.length)return l}},getLocation:function(){return
     c.url()},setLocation:function(a){a!==c.url()&&(c.url(a),b.$digest())},whenStable:funct
     ion(b){a.notifyWhenNoOutstandingRequests(b)}}]}function
     tf(){this.$get=["$rootScope","$browser","$q","$$q","$exceptionHandler",function(b,a,c,
     d,e){function f(f,l,k){B(f)||(k=l,l=f,f=v);var
     m=xa.call(arguments,3),q=x(k)&&!k,s=(q?d:c).defer(),t=s.promise,n;n=a.defer(function()
     {try{s.resolve(f.apply(null,m))}catch(a){s.reject(a),e(a)}finally{delete
     g[t.$$timeou...
146  b.$apply()},l);t.$$timeoutId=n;g[n]=s;return t}var g={};f.cancel=function(b){return
     b&&b.$$timeoutId in g?(g[b.$$timeoutId].reject("canceled"),delete
     g[b.$$timeoutId],a.defer.cancel(b.$$timeoutId)):!1};return f}]}function
     Aa(b){Va&&(Z.setAttribute("href",b),b=Z.href);Z.setAttribute("href",b);return{href:Z.h
     ref,protocol:Z.protocol?Z.protocol.replace(/:$/,""):"",host:Z.host,search:Z.search?Z.s
     earch.replace(/^\?/,""):"",hash:Z.hash?Z.hash.replace(/^#/,""):"",hostname:Z.hostname,
     port:Z.port,pathname...
```

```
147                 Z.pathname.charAt(0)?Z.pathname:"/"+Z.pathname}}function gd(b){b=H(b)?Aa(b):b;return
                    b.protocol===wd.protocol&&b.host===wd.host}function uf(){this.$get=qa(N)}function
                    xd(b){function a(a){try{return decodeURIComponent(a)}catch(b){return a}}var
                    c=b[0]||{},d={},e="";return function(){var
                    b,g,h,l,k;b=c.cookie||"";if(b!==e)for(e=b,b=e.split(";
                    "),d={},h=0;h<b.length;h++)g=b[h],l=g.indexOf("="),0<l&&(k=a(g.substring(0,l)),d[k]===
                    u&&(d[k]=a(g.substring(l+1))));return d}}function yf(){this.$get=xd}funct...
148                 d){if(D(c)){var e={};n(c,function(b,c){e[c]=a(c,b)});return e}return
                    b.factory(c+"Filter",d)}this.register=a;this.$get=["$injector",function(a){return
                    function(b){return
                    a.get(b+"Filter")}}];a("currency",yd);a("date",zd);a("filter",$f);a("json",ag);a("limi
                    tTo",bg);a("lowercase",cg);a("number",Ad);a("orderBy",Bd);a("uppercase",dg)}function
                    $f(){return function(b,a,c){if(!Da(b)){if(null==b)return b;throw
                    G("filter")("notarray",b);}var d;switch(gc(a)){case "function":break;case
                    "boolean":case "null...
```

Category: Header Manipulation (3 Issues)



Abstract:

The method processRequest() in FileServlet.java includes unvalidated data in an HTTP response header on line 159. This enables attacks such as cache-poisoning, cross-site scripting, cross-user defacement, page hijacking, cookie manipulation or open redirect.

Explanation:

Header Manipulation vulnerabilities occur when:

1. Data enters a web application through an untrusted source, most frequently an HTTP request.

2. The data is included in an HTTP response header sent to a web user without being validated.

As with many software security vulnerabilities, Header Manipulation is a means to an end, not an end in itself. At its root, the vulnerability is straightforward: an attacker passes malicious data to a vulnerable application, and the application includes the data in an HTTP response header.

One of the most common Header Manipulation attacks is HTTP Response Splitting. To mount a successful HTTP Response Splitting exploit, the application must allow input that contains CR (carriage return, also given by %0d or \r) and LF (line feed, also given by %0a or \n)characters into the header. These characters not only give attackers control of the remaining headers and body of the response the application intends to send, but also allows them to create additional responses entirely under their control.

Many of today's modern application servers will prevent the injection of malicious characters into HTTP headers. For example, recent versions of Apache Tomcat will throw an IllegalArgumentException if you attempt to set a header with prohibited characters. If your application server prevents setting headers with new line characters, then your application is not vulnerable to HTTP Response Splitting. However, solely filtering for new line characters can leave an application vulnerable to Cookie Manipulation or Open Redirects, so care must still be taken when setting HTTP headers with user input.

Example: The following code segment reads the name of the author of a weblog entry, author, from an HTTP request and sets it in a cookie header of an HTTP response.

```
String author = request.getParameter(AUTHOR_PARAM);
...
Cookie cookie = new Cookie("author", author);
cookie.setMaxAge(cookieExpiration);
response.addCookie(cookie);
```

Assuming a string consisting of standard alpha-numeric characters, such as "Jane Smith", is submitted in the request the HTTP response including this cookie might take the following form:

```
HTTP/1.1 200 OK
...
Set-Cookie: author=Jane Smith
...
```

However, because the value of the cookie is formed of unvalidated user input the response will only maintain this form if the value submitted for AUTHOR_PARAM does not contain any CR and LF characters. If an attacker submits a malicious string, such as "Wiley Hacker\r\nHTTP/1.1 200 OK\r\n...", then the HTTP response would be split into two responses of the following form:

```
HTTP/1.1 200 OK
```

...
Set-Cookie: author=Wiley Hacker

HTTP/1.1 200 OK

...

Clearly, the second response is completely controlled by the attacker and can be constructed with any header and body content desired. The ability of attacker to construct arbitrary HTTP responses permits a variety of resulting attacks, including: cross-user defacement, web and browser cache poisoning, cross-site scripting, and page hijacking.

Cross-User Defacement: An attacker will be able to make a single request to a vulnerable server that will cause the server to create two responses, the second of which may be misinterpreted as a response to a different request, possibly one made by another user sharing the same TCP connection with the server. This can be accomplished by convincing the user to submit the malicious request themselves, or remotely in situations where the attacker and the user share a common TCP connection to the server, such as a shared proxy server. In the best case, an attacker may leverage this ability to convince users that the application has been hacked, causing users to lose confidence in the security of the application. In the worst case, an attacker may provide specially crafted content designed to mimic the behavior of the application but redirect private information, such as account numbers and passwords, back to the attacker.

Cache Poisoning: The impact of a maliciously constructed response can be magnified if it is cached either by a web cache used by multiple users or even the browser cache of a single user. If a response is cached in a shared web cache, such as those commonly found in proxy servers, then all users of that cache will continue receive the malicious content until the cache entry is purged. Similarly, if the response is cached in the browser of an individual user, then that user will continue to receive the malicious content until the cache entry is purged, although only the user of the local browser instance will be affected.

Cross-Site Scripting: Once attackers have control of the responses sent by an application, they have a choice of a variety of malicious content to provide users. Cross-site scripting is common form of attack where malicious JavaScript or other code included in a response is executed in the user's browser. The variety of attacks based on XSS is almost limitless, but they commonly include transmitting private data like cookies or other session information to the attacker, redirecting the victim to web content controlled by the attacker, or performing other malicious operations on the user's machine under the guise of the vulnerable site. The most common and dangerous attack vector against users of a vulnerable application uses JavaScript to transmit session and authentication information back to the attacker who can then take complete control of the victim's account.

Page Hijacking: In addition to using a vulnerable application to send malicious content to a user, the same root vulnerability can also be leveraged to redirect sensitive content generated by the server and intended for the user to the attacker instead. By submitting a request that results in two responses, the intended response from the server and the response generated by the attacker, an attacker may cause an intermediate node, such as a shared proxy server, to misdirect a response generated by the server for the user to the attacker. Because the request made by the attacker generates two responses, the first is interpreted as a response to the attacker's request, while the second remains in limbo. When the user makes a legitimate request through the same TCP connection, the attacker's request is already waiting and is interpreted as a response to the victim's request. The attacker then sends a second request to the server, to which the proxy server responds with the server generated request intended for the victim, thereby compromising any sensitive information in the headers or body of the response intended for the victim.

Cookie Manipulation: When combined with attacks like Cross-Site Request Forgery, attackers may change, add to, or even overwrite a legitimate user's cookies.

Open Redirect: Allowing unvalidated input to control the URL used in a redirect can aid phishing attacks.

Recommendations:

The solution to Header Manipulation is to ensure that input validation occurs in the correct places and checks for the correct properties.

Since Header Manipulation vulnerabilities occur when an application includes malicious data in its output, one logical approach is to validate data immediately before it leaves the application. However, because web applications often have complex and intricate code for generating responses dynamically, this method is prone to errors of omission (missing validation). An effective way to mitigate this risk is to also perform input validation for Header Manipulation.

Web applications must validate their input to prevent other vulnerabilities, such as SQL injection, so augmenting an application's existing input validation mechanism to include checks for Header Manipulation is generally relatively easy. Despite its value, input validation for Header Manipulation does not take the place of rigorous output validation. An application may accept input through a shared data store or other trusted source, and that data store may accept input from a source that does not perform adequate input validation. Therefore, the application cannot implicitly rely on the safety of this or any other data. This means the best way to prevent Header Manipulation vulnerabilities is to validate everything that enters the application or leaves the application destined for the user.

The most secure approach to validation for Header Manipulation is to create a whitelist of safe characters that are allowed to appear in HTTP response headers and accept input composed exclusively of characters in the approved set. For example, a valid name might only include alpha-numeric characters or an account number might only include digits 0-9.

A more flexible, but less secure approach is known as blacklisting, which selectively rejects or escapes potentially dangerous characters before using the input. In order to form such a list, you first need to understand the set of characters that hold special meaning in HTTP response headers. Although the CR and LF characters are at the heart of an HTTP response splitting attack, other characters, such as ':' (colon) and '=' (equal), have special meaning in response headers as well.

After you identify the correct points in an application to perform validation for Header Manipulation attacks and what special characters the validation should consider, the next challenge is to identify how your validation handles special characters. The application should reject any input destined to be included in HTTP response headers that contains special characters, particularly CR and LF, as invalid.

Many application servers attempt to limit an application's exposure to HTTP response splitting vulnerabilities by providing implementations for the functions responsible for setting HTTP headers and cookies that perform validation for the characters essential to an HTTP response splitting attack. Do not rely on the server running your application to make it secure. When an application is developed there are no guarantees about what application servers it will run on during its lifetime. As standards and known exploits evolve, there are no guarantees that application servers will also stay in sync.

Tips:

1. Many HttpServletRequest implementations return a URL-encoded string from getHeader(), will not cause a HTTP response splitting issue unless it is decoded first because the CR and LF characters will not carry a meta-meaning in their encoded form. However, this behavior is not specified in the J2EE standard and varies by implementation. Furthermore, even encoded user input returned from getHeader() can lead to other vulnerabilities, including open redirects and other HTTP header tampering.

2. A number of modern web frameworks provide mechanisms to perform user input validation (including Struts and Spring MVC). To highlight the unvalidated sources of input, the Fortify Secure Coding Rulepacks dynamically re-prioritize the issues reported by Fortify Static Code Analyzer by lowering their probability of exploit and providing pointers to the supporting evidence whenever the framework validation mechanism is in use. We refer to this feature as Context-Sensitive Ranking. To further assist the Fortify user with the auditing process, the Fortify Software Security Research group makes available the Data Validation project template that groups the issues into folders based on the validation mechanism applied to their source of input.

3. Fortify RTA adds protection against this category.

FileServlet.java, line 159 (Header Manipulation)

Fortify Priority:	High	**Folder**	High
Kingdom:	Input Validation and Representation		
Abstract:	The method processRequest() in FileServlet.java includes unvalidated data in an HTTP response header on line 159. This enables attacks such as cache-poisoning, cross-site scripting, cross-user defacement, page hijacking, cookie manipulation or open redirect.		

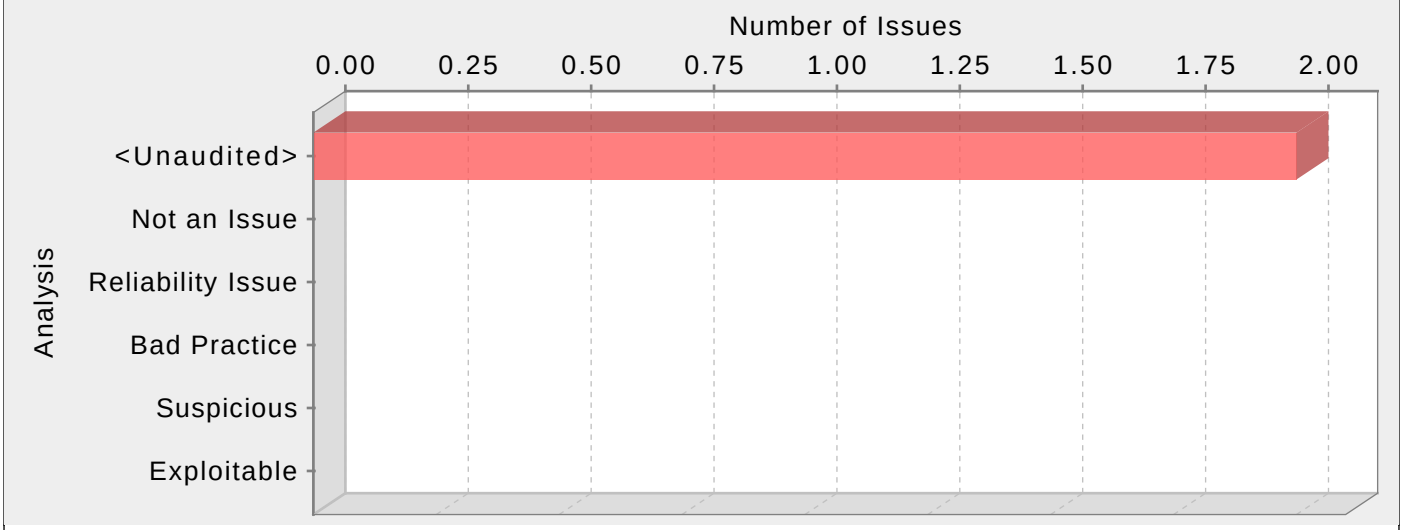
Source: FileServlet.java:126 javax.servlet.http.HttpServletRequest.getPathInfo()

```
124
125   // Get requested file by path info.
126   String requestedFile = request.getPathInfo().replace("/owa/fileServlet", "");
127
128   // Check if file is actually supplied to the request URL.
```

Sink: FileServlet.java:159 javax.servlet.http.HttpServletResponse.setHeader()

```
157   if (ifNoneMatch != null && matches(ifNoneMatch, eTag)) {
158    response.setStatus(HttpServletResponse.SC_NOT_MODIFIED);
159    response.setHeader("ETag", eTag); // Required in 304.
160    response.setDateHeader("Expires", expires); // Postpone cache with 1 week.
161    return;
```

Category: File Disclosure: J2EE (2 Issues)



Abstract:

On line 55 of OwaFilter.java, the method doFilter() invokes a server side forward using a path built with unvalidated input. This could allow an attacker to download application binaries or view arbitrary files within protected directories.

Explanation:

A file disclosure occur when:

1. Data enters a program from an untrusted source.

2. The data is used to dynamically construct a path.

Example 1: The following code takes untrusted data and uses it to build a path which is used in a server side forward.

```
...
String returnURL = request.getParameter("returnURL");
RequestDispatcher rd = request.getRequestDispatcher(returnURL);
rd.forward();
...
```

Example 2: The following code takes untrusted data and uses it to build a path which is used in a server side forward.

```
...
<% String returnURL = request.getParameter("returnURL"); %>
<jsp:include page="<%=returnURL%>" />
...
```

If an attacker provided a URL with the request parameter matching a sensitive file location, they would be able to view that file. For example, "http://www.yourcorp.com/webApp/logic?returnURL=WEB-INF/applicationContext.xml" would allow them to view the applicationContext.xml of the application.

After the attacker had the applicationContext.xml, they could locate and download other configuration files referenced in the applicationContext.xml or even class or jar files. This would allow attackers to gain sensitive infomation about an application and target it for other types of attack.

Recommendations:

Do not use untrusted data to direct requests to server-side resources. Instead, use a level of indirection between locations and paths.

Instead of the following:

```
< a href="http://www.yourcorp.com/webApp/logic?nextPage=WEB-INF/signup.jsp">New Customer</a>
```

Use the following:

```
< a href="http://www.yourcorp.com/webApp/logic?nextPage=newCustomer">New Customer</a>
```

The server-side logic would have a map keyed by logical names to server-side paths such that the path stored under the key "newCustomer" would be "/WEB-INF/signup.jsp".

OwaFilter.java, line 55 (File Disclosure: J2EE)

Fortify Priority:	High	**Folder**	High

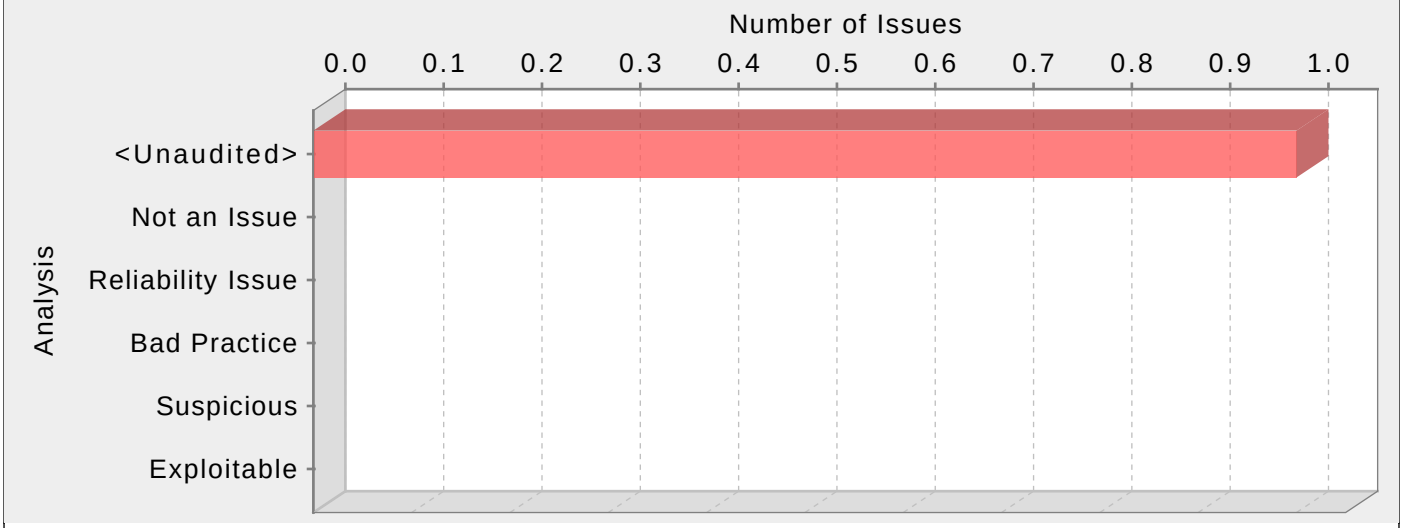
Kingdom:	API Abuse
Abstract:	On line 55 of OwaFilter.java, the method doFilter() invokes a server side forward using a path built with unvalidated input. This could allow an attacker to download application binaries or view arbitrary files within protected directories.
Source:	OwaFilter.java:47 javax.servlet.http.HttpServletRequest.getRequestURL()

```
45    String requestURL = null;
46    if (isFullBasePath(owaBasePath)) {
47     requestURL = request.getRequestURL().toString();
48    } else {
49     requestURL = request.getServletPath();
```

Sink:	OwaFilter.java:55 javax.servlet.ServletRequest.getRequestDispatcher()

```
53     if (requestURL.startsWith(owaBasePath)) {
54      String newURL = requestURL.replace(owaBasePath, "/ms/owa/fileServlet");
55      req.getRequestDispatcher(newURL).forward(req, res);
56     } else {
57      chain.doFilter(req, res);
```

Category: Open Redirect (1 Issues)



Abstract:

The file RedirectServlet.java passes unvalidated data to an HTTP redirect function on line 57. Allowing unvalidated input to control the URL used in a redirect can aid phishing attacks.

Explanation:

Redirects allow web applications to direct users to different pages within the same application or to external sites. Applications utilize redirects to aid in site navigation and, in some cases, to track how users exit the site. Open redirect vulnerabilities occur when a web application redirects clients to any arbitrary URL that can be controlled by an attacker.

Attackers may utilize open redirects to trick users into visiting a URL to a trusted site and redirecting them to a malicious site. By encoding the URL, an attacker is able to make it more difficult for end-users to notice the malicious destination of the redirect, even when it is passed as a URL parameter to the trusted site. Open redirects are often abused as part of phishing scams to harvest sensitive end-user data.

Example 1: The following JSP code instructs the user's browser to open a URL parsed from the dest request parameter when a user clicks the link.

```
<%
...
String strDest = request.getParameter("dest");
pageContext.forward(strDest);
...
%>
```

If a victim received an email instructing them to follow a link to "http://trusted.example.com/ecommerce/redirect.asp?dest=www.wilyhacker.com", the user would likely click on the link believing they would be transferred to the trusted site. However, when the victim clicks the link, the code in Example 1 will redirect the browser to "http://www.wilyhacker.com".

Many users have been educated to always inspect URLs they receive in emails to make sure the link specifies a trusted site they know. However, if the attacker Hex encoded the destination url as follows:

"http://trusted.example.com/ecommerce/redirect.asp?dest=%77%69%6C%79%68%61%63%6B%65%72%2E%63%6F%6D"

then even a savvy end-user may be fooled into following the link.

Recommendations:

Unvalidated user input should not be allowed to control the destination URL in a redirect. Instead, use a level of indirection: create a list of legitimate URLs that users are allowed to specify and only allow users to select from the list. With this approach, input provided by users is never used directly to specify a URL for redirects.

Example 2: The following code references an array populated with valid URLs. The link the user clicks passes in the array index that corresponds to the desired URL.

```
<%
...
try {
int strDest = Integer.parseInt(request.getParameter("dest"));
if((strDest >= 0) && (strDest <= strURLArray.length -1 ))
{
```

```
strFinalURL = strURLArray[strDest];
pageContext.forward(strFinalURL);
}
}
catch (NumberFormatException nfe) {
// Handle exception
...
}
...
%>
```

In some situations this approach is impractical because the set of legitimate URLs is too large or too hard to keep track of. In such cases, use a similar approach to restrict the domains that users can be redirected to, which can at least prevent attackers from sending users to malicious external sites.

Tips:

1. A number of modern web frameworks provide mechanisms to perform user input validation (including Struts and Spring MVC). To highlight the unvalidated sources of input, the Fortify Secure Coding Rulepacks dynamically re-prioritize the issues reported by Fortify Static Code Analyzer by lowering their probability of exploit and providing pointers to the supporting evidence whenever the framework validation mechanism is in use. We refer to this feature as Context-Sensitive Ranking. To further assist the Fortify user with the auditing process, the Fortify Software Security Research group makes available the Data Validation project template that groups the issues into folders based on the validation mechanism applied to their source of input.

2. Fortify RTA adds protection against this category.

RedirectServlet.java, line 57 (Open Redirect)

Fortify Priority:	Critical	**Folder**	Critical
Kingdom:	Input Validation and Representation		
Abstract:	The file RedirectServlet.java passes unvalidated data to an HTTP redirect function on line 57. Allowing unvalidated input to control the URL used in a redirect can aid phishing attacks.		

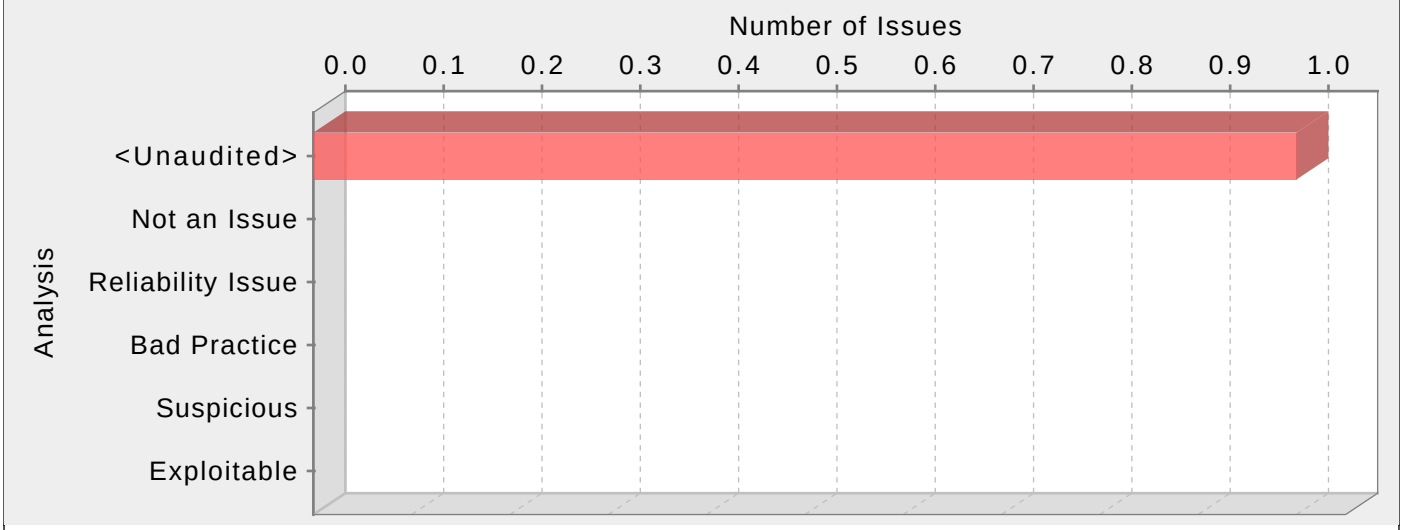
Source: RedirectServlet.java:54 javax.servlet.http.HttpServletRequest.getRequestURL()

```
52    private void processRequest(HttpServletRequest request, HttpServletResponse response,
      boolean content)
53            throws IOException {
54      String url = request.getRequestURL().toString().replace("/ms/owa/redirectServlet",
      "/owa");
55      //@TODO redirecting to original url after login in openmrs.
56      String loginUrl = Context.getAdministrationService().getGlobalProperty("login.url",
      "login.htm");
```

Sink: RedirectServlet.java:57 javax.servlet.http.HttpServletResponse.sendRedirect()

```
55      //@TODO redirecting to original url after login in openmrs.
56      String loginUrl = Context.getAdministrationService().getGlobalProperty("login.url",
      "login.htm");
57      response.sendRedirect(request.getContextPath() + "/" + loginUrl + "?redirect=" + url);
58    }
59  }
```

Category: Path Manipulation (1 Issues)



Abstract:

Attackers are able to control the file system path argument to File() at FileServlet.java line 137, which allows them to access or modify otherwise protected files.

Explanation:

Path manipulation errors occur when the following two conditions are met:

1. An attacker is able to specify a path used in an operation on the file system.

2. By specifying the resource, the attacker gains a capability that would not otherwise be permitted.

For example, the program may give the attacker the ability to overwrite the specified file or run with a configuration controlled by the attacker.

Example 1: The following code uses input from an HTTP request to create a file name. The programmer has not considered the possibility that an attacker could provide a file name such as "../../tomcat/conf/server.xml", which causes the application to delete one of its own configuration files.

```
String rName = request.getParameter("reportName");
File rFile = new File("/usr/local/apfr/reports/" + rName);
...
rFile.delete();
```

Example 2: The following code uses input from a configuration file to determine which file to open and echo back to the user. If the program runs with adequate privileges and malicious users can change the configuration file, they can use the program to read any file on the system that ends with the extension .txt.

```
fis = new FileInputStream(cfg.getProperty("sub")+".txt");
amt = fis.read(arr);
out.println(arr);
```

Some think that in the mobile world, classic vulnerabilities, such as path manipulation, do not make sense -- why would the user attack themself? However, keep in mind that the essence of mobile platforms is applications that are downloaded from various sources and run alongside each other on the same device. The likelihood of running a piece of malware next to a banking application is high, which necessitates expanding the attack surface of mobile applications to include inter-process communication.

Example 3: The following code adapts Example 1 to the Android platform.

```
...
String rName = this.getIntent().getExtras().getString("reportName");
File rFile = getBaseContext().getFileStreamPath(rName);
...
rFile.delete();
...
```

Recommendations:

The best way to prevent path manipulation is with a level of indirection: create a list of legitimate resource names that a user is allowed to specify, and only allow the user to select from the list. With this approach the input provided by the user is never used directly to specify the resource name.

In some situations this approach is impractical because the set of legitimate resource names is too large or too hard to keep track of. Programmers often resort to blacklisting in these situations. Blacklisting selectively rejects or escapes potentially dangerous characters before using the input. However, any such list of unsafe characters is likely to be incomplete and will almost certainly become out of date. A better approach is to create a whitelist of characters that are allowed to appear in the resource name and accept input composed exclusively of characters in the approved set.

Tips:

1. If the program is performing custom input validation you are satisfied with, use the Fortify Custom Rules Editor to create a cleanse rule for the validation routine.

2. Implementation of an effective blacklist is notoriously difficult. One should be skeptical if validation logic requires blacklisting. Consider different types of input encoding and different sets of meta-characters that might have special meaning when interpreted by different operating systems, databases, or other resources. Determine whether or not the blacklist can be updated easily, correctly, and completely if these requirements ever change.

3. A number of modern web frameworks provide mechanisms to perform user input validation (including Struts and Spring MVC). To highlight the unvalidated sources of input, the Fortify Secure Coding Rulepacks dynamically re-prioritize the issues reported by Fortify Static Code Analyzer by lowering their probability of exploit and providing pointers to the supporting evidence whenever the framework validation mechanism is in use. We refer to this feature as Context-Sensitive Ranking. To further assist the Fortify user with the auditing process, the Fortify Software Security Research group makes available the Data Validation project template that groups the issues into folders based on the validation mechanism applied to their source of input.

FileServlet.java, line 137 (Path Manipulation)

Fortify Priority:	Critical	**Folder**	Critical
Kingdom:	Input Validation and Representation		
Abstract:	Attackers are able to control the file system path argument to File() at FileServlet.java line 137, which allows them to access or modify otherwise protected files.		

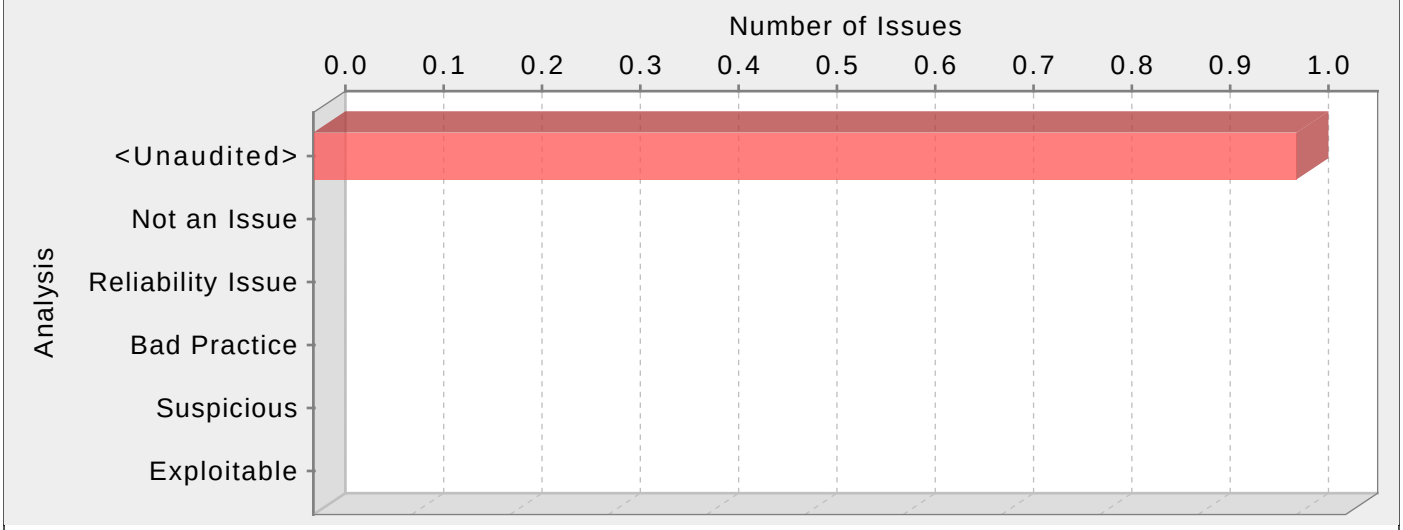
Source: FileServlet.java:126 javax.servlet.http.HttpServletRequest.getPathInfo()

```
124
125                 // Get requested file by path info.
126                 String requestedFile = request.getPathInfo().replace("/owa/fileServlet", "");
127
128                 // Check if file is actually supplied to the request URL.
```

Sink: FileServlet.java:137 java.io.File.File()

```
135
136                 // URL-decode the file name (might contain spaces and on) and prepare file object.
137                 File file = new File(basePath, URLDecoder.decode(requestedFile, "UTF-8"));
138
139                 // Check if file actually exists in filesystem.
```

Category: Race Condition: Singleton Member Field (1 Issues)



Abstract:

The class DefaultAppManager is a singleton, so the member field apps is shared between users. The result is that one user could see another user's data.

Explanation:

Many Servlet developers do not understand that a Servlet is a singleton. There is only one instance of the Servlet, and that single instance is used and re-used to handle multiple requests that are processed simultaneously by different threads.

A common result of this misunderstanding is that developers use Servlet member fields in such a way that one user may inadvertently see another user's data. In other words, storing user data in Servlet member fields introduces a data access race condition.

Example 1: The following Servlet stores the value of a request parameter in a member field and then later echoes the parameter value to the response output stream.

```
public class GuestBook extends HttpServlet {

String name;

protected void doPost (HttpServletRequest req, HttpServletResponse res) {
name = req.getParameter("name");
...
out.println(name + ", thanks for visiting!");
}
}
```

While this code will work perfectly in a single-user environment, if two users access the Servlet at approximately the same time, it is possible for the two request handler threads to interleave in the following way:

Thread 1: assign "Dick" to name
Thread 2: assign "Jane" to name
Thread 1: print "Jane, thanks for visiting!"
Thread 2: print "Jane, thanks for visiting!"

Thereby showing the first user the second user's name.

Recommendations:

Do not use Servlet member fields for anything but constants. (i.e. make all member fields static final).

Developers are often tempted to use Servlet member fields for user data when they need to transport data from one region of code to another. If this is your aim, consider declaring a separate class and using the Servlet only to "wrap" this new class.

Example 2: The bug in Example 1 can be corrected in the following way:

```
public class GuestBook extends HttpServlet {

protected void doPost (HttpServletRequest req, HttpServletResponse res) {
GBRequestHandler handler = new GBRequestHandler();
handler.handle(req, res);
}
```

```
}
public class GBRequestHandler {
String name;
public void handle(HttpServletRequest req, HttpServletResponse res) {
name = req.getParameter("name");
...
out.println(name + ", thanks for visiting!");
}
}
```

Alternatively, a Servlet can utilize synchronized blocks to access servlet instance variables but using synchronized blocks may cause significant performance problems.

Please notice that wrapping the field access within a synchronized block will only prevent the issue if all read and write operations on that member are performed within the same synchronized block or method.

Example 3: Wrapping the Example 1 write operation (assignment) in a synchronized block will not fix the problem since the threads will have to get a lock to modify name field, but they will release the lock afterwards, allowing a second thread to change the value again. If, after changing the name value, the first thread resumes execution, the value printed will be the one assigned by the second thread:

```
public class GuestBook extends HttpServlet {
String name;
protected void doPost (HttpServletRequest req, HttpServletResponse res) {
synchronized(name) {
name = req.getParameter("name");
}
...
out.println(name + ", thanks for visiting!");
}
}
```

In order to fix the race condition, all the write and read operations on the shared member field should be run atomically within the same synchronized block:

```
public class GuestBook extends HttpServlet {
String name;
protected void doPost (HttpServletRequest req, HttpServletResponse res) {
synchronized(name) {
name = req.getParameter("name");
...
out.println(name + ", thanks for visiting!");
}
}
}
```

DefaultAppManager.java, line 299 (Race Condition: Singleton Member Field)

Fortify Priority:	High	**Folder**	High
Kingdom:	Time and State		
Abstract:	The class DefaultAppManager is a singleton, so the member field apps is shared between users. The result is that one user could see another user's data.		

Sink: DefaultAppManager.java:299 AssignmentStatement()

```
297    }
298
299    this.apps = appList;
300    log.info("Detected apps: " + apps);
301    if (owaListeners != null) {
```

Detailed Project Summary

Files Scanned

Code base location: /srv/openmrs_code/org/openmrs/module/owa
Files Scanned:
.travis.yml yaml Dec 13, 2019 12:57:08 PM
/home/agautam6/.fortify/sca19.1/build/owa/extracted/angularjs/sca.frontend.angularjs.aux.js secondary 5 Lines Apr 17, 2020 4:53:01 AM
/home/agautam6/.fortify/sca19.1/build/owa/extracted/javascript/srv/openmrs_code/org/openmrs/module/owa/omod/src/main/webapp/manage.html.js secondary Apr 17, 2020 4:53:01 AM
/home/agautam6/.fortify/sca19.1/build/owa/extracted/javascript/srv/openmrs_code/org/openmrs/module/owa/omod/src/main/webapp/settings.html.js secondary Apr 17, 2020 4:53:01 AM
/home/agautam6/.fortify/sca19.1/build/owa/extracted/javascript/srv/openmrs_code/org/openmrs/module/owa/omod/target/classes/web/module/manage.html.js secondary Apr 17, 2020 4:53:01 AM
/home/agautam6/.fortify/sca19.1/build/owa/extracted/javascript/srv/openmrs_code/org/openmrs/module/owa/omod/target/classes/web/module/settings.html.js secondary Apr 17, 2020 4:53:01 AM
/home/agautam6/.fortify/sca19.1/build/owa/extracted/javascript/srv/openmrs_code/org/openmrs/module/owa/omod/target/owa-1.10.0/web/module/manage.html.js secondary Apr 17, 2020 4:53:01 AM
/home/agautam6/.fortify/sca19.1/build/owa/extracted/javascript/srv/openmrs_code/org/openmrs/module/owa/omod/target/owa-1.10.0/web/module/settings.html.js secondary Apr 17, 2020 4:53:01 AM
OpenMRSFormatter.xml xml 27.9 KB Dec 13, 2019 12:57:08 PM
api/pom.xml xml 2.6 KB Dec 13, 2019 12:57:08 PM
api/src/main/java/org/openmrs/module/owa/App.java java 41 Lines 5.7 KB Dec 17, 2019 7:06:51 PM
api/src/main/java/org/openmrs/module/owa/AppActivities.java java 4 Lines 2.1 KB Dec 17, 2019 7:06:49 PM
api/src/main/java/org/openmrs/module/owa/AppDeveloper.java java 10 Lines 2.8 KB Dec 17, 2019 7:06:48 PM
api/src/main/java/org/openmrs/module/owa/AppIcons.java java 8 Lines 2.5 KB Dec 17, 2019 7:06:50 PM
api/src/main/java/org/openmrs/module/owa/AppManager.java java 5 Lines 3.8 KB Dec 17, 2019 7:06:49 PM
api/src/main/java/org/openmrs/module/owa/AppOpenmrs.java java 4 Lines 2.1 KB Dec 17, 2019 7:06:51 PM
api/src/main/java/org/openmrs/module/owa/OwaListener.java java Dec 17, 2019 7:06:48 PM
api/src/main/resources/messages.properties java_properties 1.6 KB Dec 13, 2019 12:57:08 PM
api/src/main/resources/messages_es.properties java_properties Dec 13, 2019 12:57:08 PM
api/src/main/resources/messages_fr.properties java_properties Dec 13, 2019 12:57:08 PM
api/src/main/resources/moduleApplicationContext.xml xml 1.6 KB Dec 13, 2019 12:57:08 PM
api/target/classes/messages.properties java_properties 1.1 KB Dec 18, 2019 12:01:29 PM
api/target/classes/messages_es.properties java_properties Dec 18, 2019 12:01:29 PM
api/target/classes/messages_fr.properties java_properties Dec 18, 2019 12:01:29 PM
api/target/classes/moduleApplicationContext.xml xml 1.6 KB Dec 18, 2019 12:01:29 PM
api/target/maven-archiver/pom.properties java_properties Dec 18, 2019 12:01:33 PM
api/target/maven-java-formatter-cache.properties java_properties Dec 18, 2019 12:01:29 PM
omod/pom.xml xml 8.3 KB Dec 13, 2019 12:57:08 PM
omod/src/main/java/org/openmrs/module/owa/activator/OwaActivator.java java 40 Lines 5.3 KB Dec 17, 2019 7:09:22 PM
omod/src/main/java/org/openmrs/module/owa/extension/html/AdminList.java java 7 Lines 1.2 KB Dec 17, 2019 7:09:39 PM
omod/src/main/java/org/openmrs/module/owa/filter/OwaFilter.java java 19 Lines 2.5 KB Dec 13, 2019 12:57:08 PM
omod/src/main/java/org/openmrs/module/owa/impl/DefaultAppManager.java java 88 Lines 9.9 KB Dec 17, 2019 7:09:30 PM
omod/src/main/java/org/openmrs/module/owa/servlet/FileServlet.java java 142 Lines 17.9 KB Dec 17, 2019 7:09:33 PM
omod/src/main/java/org/openmrs/module/owa/servlet/RedirectServlet.java java 6 Lines 2.1 KB Dec 17, 2019 7:09:31 PM
omod/src/main/java/org/openmrs/module/owa/utils/OwaUtils.java java 10 Lines 1.5 KB Dec 17, 2019 7:09:26 PM
omod/src/main/java/org/openmrs/module/owa/web/controller/AddAppController.java java 28 Lines 5.5 KB Dec 17, 2019 7:09:39 PM
omod/src/main/java/org/openmrs/module/owa/web/controller/InstallAppRequestObject.java java 4 Lines Dec 17, 2019 7:09:37 PM
omod/src/main/java/org/openmrs/module/owa/web/controller/OwaManageController.java java 29 Lines 4 KB Dec 17, 2019 7:09:34

PM
omod/src/main/java/org/openmrs/module/owa/web/controller/OwaManagerRestController.java java 3 Lines Dec 17, 2019 7:09:35 PM
omod/src/main/java/org/openmrs/module/owa/web/controller/OwaRestController.java java 103 Lines 9.5 KB Dec 17, 2019 7:09:34 PM
omod/src/main/java/org/openmrs/module/owa/web/controller/SettingsFormController.java java 25 Lines 4.8 KB Dec 17, 2019 7:09:34 PM
omod/src/main/resources/config.xml xml 3.2 KB Dec 13, 2019 12:57:08 PM
omod/src/main/resources/webModuleApplicationContext.xml xml 1.8 KB Dec 13, 2019 12:57:08 PM
omod/src/main/webapp/manage.html html 9 Lines 4.5 KB Dec 13, 2019 12:57:08 PM
omod/src/main/webapp/manage.jsp jsp 24 Lines 6.6 KB Dec 13, 2019 12:57:08 PM
omod/src/main/webapp/resources/javascript/angular-1.4.5.min.js typescript 287 Lines 143.4 KB Dec 13, 2019 12:57:08 PM
omod/src/main/webapp/resources/javascript/bootstrap-filestyle.min.js typescript 1 Lines 7 KB Dec 13, 2019 12:57:08 PM
omod/src/main/webapp/resources/javascript/deleteApp.js typescript 4 Lines Dec 13, 2019 12:57:08 PM
omod/src/main/webapp/resources/javascript/jquery.dataTables.min.js typescript 160 Lines 79.8 KB Dec 13, 2019 12:57:08 PM
omod/src/main/webapp/resources/javascript/jquery.form.js typescript 525 Lines 40.1 KB Dec 13, 2019 12:57:08 PM
omod/src/main/webapp/resources/javascript/uploadApp.js typescript 27 Lines 1.6 KB Dec 13, 2019 12:57:08 PM
omod/src/main/webapp/settings.html html 1 Lines 2.2 KB Dec 13, 2019 12:57:08 PM
omod/src/main/webapp/settings.jsp jsp 7 Lines 2.9 KB Dec 13, 2019 12:57:08 PM
omod/src/main/webapp/template/localHeader.jsp jsp 6 Lines Dec 13, 2019 12:57:08 PM
omod/src/test/java/org/openmrs/module/owa/filter/OwaFilterTest.java java 27 Lines 4 KB Dec 17, 2019 7:09:42 PM
omod/src/test/java/org/openmrs/module/owa/impl/DefaultAppManagerTest.java java 37 Lines 5.2 KB Dec 17, 2019 7:09:44 PM
omod/src/test/java/org/openmrs/module/owa/utils/OwaUtilsTest.java java 7 Lines Dec 17, 2019 7:09:41 PM
omod/src/test/java/org/openmrs/module/owa/web/controller/AddAppControllerTest.java java 32 Lines 4.6 KB Dec 13, 2019 12:57:08 PM
omod/src/test/java/org/openmrs/module/owa/web/controller/OwaRestControllerTest.java java 76 Lines 9.3 KB Dec 17, 2019 7:09:45 PM
omod/target/classes/META-INF/maven/org.openmrs.module/owa-api/pom.properties java_properties Dec 18, 2019 12:01:34 PM
omod/target/classes/META-INF/maven/org.openmrs.module/owa-api/pom.xml xml 2.6 KB Dec 13, 2019 12:57:08 PM
omod/target/classes/config.xml xml 3 KB Dec 18, 2019 12:01:43 PM
omod/target/classes/messages.properties java_properties 1.1 KB Dec 18, 2019 12:01:30 PM
omod/target/classes/messages_es.properties java_properties Dec 18, 2019 12:01:30 PM
omod/target/classes/messages_fr.properties java_properties Dec 18, 2019 12:01:30 PM
omod/target/classes/moduleApplicationContext.xml xml 1.6 KB Dec 18, 2019 12:01:30 PM
omod/target/classes/web/module/manage.html html 9 Lines 4.5 KB Dec 18, 2019 12:01:43 PM
omod/target/classes/web/module/manage.jsp jsp 24 Lines 6.6 KB Dec 18, 2019 12:01:43 PM
omod/target/classes/web/module/resources/javascript/angular-1.4.5.min.js typescript 287 Lines 143.4 KB Dec 18, 2019 12:01:43 PM
omod/target/classes/web/module/resources/javascript/bootstrap-filestyle.min.js typescript 1 Lines 7 KB Dec 18, 2019 12:01:43 PM
omod/target/classes/web/module/resources/javascript/deleteApp.js typescript 4 Lines Dec 18, 2019 12:01:43 PM
omod/target/classes/web/module/resources/javascript/jquery.dataTables.min.js typescript 160 Lines 79.8 KB Dec 18, 2019 12:01:43 PM
omod/target/classes/web/module/resources/javascript/jquery.form.js typescript 525 Lines 40.1 KB Dec 18, 2019 12:01:43 PM
omod/target/classes/web/module/resources/javascript/uploadApp.js typescript 27 Lines 1.6 KB Dec 18, 2019 12:01:43 PM
omod/target/classes/web/module/settings.html html 1 Lines 2.2 KB Dec 18, 2019 12:01:43 PM
omod/target/classes/web/module/settings.jsp jsp 7 Lines 2.9 KB Dec 18, 2019 12:01:43 PM
omod/target/classes/web/module/template/localHeader.jsp jsp 6 Lines Dec 18, 2019 12:01:43 PM
omod/target/classes/webModuleApplicationContext.xml xml 1.8 KB Dec 18, 2019 12:01:43 PM
omod/target/maven-archiver/pom.properties java_properties Dec 18, 2019 12:01:46 PM
omod/target/maven-java-formatter-cache.properties java_properties 1.9 KB Dec 18, 2019 12:01:41 PM
omod/target/owa-1.10.0/META-INF/maven/org.openmrs.module/owa-api/pom.properties java_properties Dec 18, 2019 12:01:46

PM
omod/target/owa-1.10.0/META-INF/maven/org.openmrs.module/owa-api/pom.xml xml 2.6 KB Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/config.xml xml 3 KB Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/messages.properties java_properties 1.1 KB Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/messages_es.properties java_properties Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/messages_fr.properties java_properties Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/moduleApplicationContext.xml xml 1.6 KB Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/web/module/manage.html html 9 Lines 4.5 KB Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/web/module/manage.jsp jsp 24 Lines 6.6 KB Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/web/module/resources/javascript/angular-1.4.5.min.js typescript 287 Lines 143.4 KB Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/web/module/resources/javascript/bootstrap-filestyle.min.js typescript 1 Lines 7 KB Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/web/module/resources/javascript/deleteApp.js typescript 4 Lines Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/web/module/resources/javascript/jquery.dataTables.min.js typescript 160 Lines 79.8 KB Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/web/module/resources/javascript/jquery.form.js typescript 525 Lines 40.1 KB Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/web/module/resources/javascript/uploadApp.js typescript 27 Lines 1.6 KB Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/web/module/settings.html html 1 Lines 2.2 KB Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/web/module/settings.jsp jsp 7 Lines 2.9 KB Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/web/module/template/localHeader.jsp jsp 6 Lines Dec 18, 2019 12:01:46 PM
omod/target/owa-1.10.0/webModuleApplicationContext.xml xml 1.8 KB Dec 18, 2019 12:01:46 PM
pom.xml xml 7.2 KB Dec 13, 2019 12:57:08 PM

Reference Elements

Classpath:

No classpath specified during translation

Libdirs:

No libdirs specified during translation

Rulepacks

Valid Rulepacks:

Name: Fortify Secure Coding Rules, Core, Java
Version: 2019.4.0.0009
ID: 06A6CC97-8C3F-4E73-9093-3E74C64A2AAF
SKU: RUL13003

Name: Fortify Secure Coding Rules, Core, Annotations
Version: 2019.4.0.0009
ID: 14EE50EB-FA1C-4AE8-8B59-39F952E21E3B
SKU: RUL13078

Name: Fortify Secure Coding Rules, Core, JavaScript
Version: 2019.4.0.0009
ID: BD292C4E-4216-4DB8-96C7-9B607BFD9584
SKU: RUL13059

Name: Fortify Secure Coding Rules, Core, Android
Version: 2019.4.0.0009
ID: FF9890E6-D119-4EE8-A591-83DCF4CA6952
SKU: RUL13093

Name: Fortify Secure Coding Rules, Extended, JavaScript
Version: 2019.4.0.0009
ID: C4D1969E-B734-47D3-87D4-73962C1D32E2
SKU: RUL13141

Name: Fortify Secure Coding Rules, Extended, Configuration
Version: 2019.4.0.0009
ID: CD6959FC-0C37-45BE-9637-BAA43C3A4D56
SKU: RUL13005

Name: Fortify Secure Coding Rules, Extended, Java
Version: 2019.4.0.0009
ID: AAAC0B10-79E7-4FE5-9921-F4903A79D317
SKU: RUL13007

Name: Fortify Secure Coding Rules, Extended, Content
Version: 2019.4.0.0009
ID: 9C48678C-09B6-474D-B86D-97EE94D38F17
SKU: RUL13067

Name: Fortify Secure Coding Rules, Core, Golang
Version: 2019.4.0.0009
ID: 1DCE79F8-AF6B-474D-A05A-5BFFC8B13DCD
SKU: RUL13218

Name: Fortify Secure Coding Rules, Extended, JSP
Version: 2019.4.0.0009
ID: 00403342-15D0-48C9-8E67-4B1CFBDEFCD2
SKU: RUL13026

External Metadata:
Version: 2019.4.0.0009

Name: CWE
ID: 3ADB9EE4-5761-4289-8BD3-CBFCC593EBBC
The Common Weakness Enumeration (CWE), co-sponsored and maintained by MITRE, is international in scope and free for public use. CWE provides a unified, measurable set of software weaknesses that is enabling more effective discussion, description, selection, and use of software security tools and services that can find these weaknesses in source code and operational systems as well as better understanding and management of software weaknesses related to architecture and design.

Name: CWE Top 25 2019
ID: 7AF935C9-15AA-45B2-8EEC-0EAE4194ACDE
The 2019 CWE Top 25 Most Dangerous Software Errors lists the most widespread and critical weaknesses that can lead to serious vulnerabilities in software (as demonstrated by the National Vulnerability Database). These weaknesses occur frequently, are often easy to find, and easy to exploit. They are dangerous because they will frequently enable attackers to completely take over

the software, steal data, or prevent the software from working at all. The list is the result of heuristic formula that the CWE Team used with a data-driven approach that leveraged the Common Vulnerabilities and Exposure (CVE), National Vulnerability Database (NVD), and Common Vulnerability Scoring System (CVSS). Due to the hiearchical nature of the CWE taxonomy, Fortify considers all CWE IDs which are children of a Top 25 entry, as included within the context of the entry due to the "CHILD-OF" relationship within the hierarchy. Exercise caution if using only this Top 25 list to prioritize auditing efforts because the software under analysis might not align with the assumptions of the heuristic used to define the Top 25. For example, many of these weaknesses are related to C-like langauges and the software under analysis might not be within the C-family of languages - thus, many CWEs would not be in scope.

Name: DISA CCI 2
ID: 7F037130-41E5-40F0-B653-7819A4B3E241
The purpose of a Defense Information Systems Agency (DISA) Control Correlation Identifier (CCI) is to provide a standard identifier for policy based requirements which connect high-level policy expressions and low-level technical implementations. Associated with each CCI is a description for each of the singular, actionable, statements compromising an information assurance (IA) control or IA best practice. Using CCI allows high-level policy framework security requirements to be decomposed and explicitly associated with low-level implementations, thus enabling the assessment of related compliance assessment results spanning heterogeneous technologies. The current IA controls and best practices associated with each CCI, that are specified in NIST SP 800-53 Revision 4, can be viewed using the DISA STIG Viewer.
The following table summarizes the number of issues identified across the different CCIs broken down by Fortify Priority Order. The status of a CCI is considered "In Place" when there are no issues reported for a given CCI.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, CCI-003187 is not considered "In Place". Similarly, if the project is missing a Micro Focus Fortify WebInspect scan, or the scan contains any critical findings, CCI-000366 and CCI-000256 are not considered "In Place".

Name: FISMA
ID: B40F9EE0-3824-4879-B9FE-7A789C89307C
The Federal Information Processing Standard (FIPS) 200 document is part of the official series of publications, issued by the National Institute of Standards and Technology (NIST), relating to standards and guidelines adopted and promulgated under the provisions of the Federal Information Security Management Act (FISMA). Specifically, FIPS Publication 200 specifies the "Minimum Security Requirements for Federal Information and Information Systems."

Name: GDPR
ID: 771C470C-9274-4580-8556-C12F5E4BEC51
The EU General Data Protection Regulation (GDPR) replaces the Data Protection Directive 95/46/EC and was designed to harmonize data privacy laws across Europe, to protect and empower all EU citizens data privacy and to reshape the way organizations across the region approach data privacy. Going into effect on May 25, 2018, GDPR provides a framework for organizations on how to handle personal data. According to GDPR regulation personal data "means any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person." GDPR articles that pertain to application security and require businesses to protect personal data during design and development of its product and services are:

- Article 25, Data protection by design and by default - which requires "The controller shall implement appropriate technical and organisational measures for ensuring that, by default, only personal data which are necessary for each specific purpose of the processing are processed."
- Article 32, Security of processing - which requires businesses to protect its systems and applications "from accidental or unlawful destruction, loss, alteration, unauthorized disclosure of, or access to personal data". This report may be used by organizations as a framework to help identify and protect personal data as it relates to application security.

Name: MISRA C 2012
ID: 555A3A66-A0E1-47AF-910C-3F19A6FB2506
Now in its third edition, the Motor Industry Software Reliability Association (MISRA) C Guidelines describe a subset of the C programming language in which there is reduced risk of introducing mistakes in critical systems. While the MISRA C Guidelines focus upon safety-related software development, a subset of the rules also reflect security properties. Fortify interprets the MISRA C Guidelines under the context of security and provides correlation of security vulnerability categories to the rules defined by MISRA. Fortify provides these security focused detection mechanism with the standard rulepacks, however, further support of the MISRA C Guidelines related to safety can be added through the use of custom rules. The results in this report can assist in the creation of a compliance matrix for MISRA.

Name: MISRA C++ 2008
ID: 5D4B75A1-FC91-4B4B-BD4D-C81BBE9604FA
The Motor Industry Software Reliability Association (MISRA) C++ Guidelines describe a subset of the C++ programming language in which there is reduced risk of introducing mistakes in critical systems. While the MISRA C++ Guidelines focus upon safety-related software development, a subset of the rules also reflect security properties. Fortify interprets the MISRA C++ Guidelines under the context of security and provides correlation of security vulnerability categories to the rules defined by MISRA. Fortify provides these security focused detection mechanism with the standard rulepacks, however, further support of the MISRA C++ Guidelines related to safety can be added through the use of custom rules. The results in this report can assist in the creation of a compliance matrix for MISRA.

Name: NIST SP 800-53 Rev.4
ID: 1114583B-EA24-45BE-B7F8-B61201BACDD0
NIST Special Publication 800-53 Revision 4 provides a list of security and privacy controls designed to protect federal organizations and information systems from security threats. The following table summarizes the number of issues identified across the different controls and broken down by Fortify Priority Order.

Name: OWASP Mobile 2014
ID: EEE3F9E7-28D6-4456-8761-3DA56C36F4EE
The OWASP Mobile Top 10 Risks 2014 provides a powerful awareness document for mobile application security. The OWASP Mobile Top 10 represents a broad consensus about what the most critical mobile application security flaws are. Project members include a variety of security experts from around the world who have shared their expertise to produce this list.

Name: OWASP Top 10 2004
ID: 771C470C-9274-4580-8556-C023E4D3ADB4
The OWASP Top Ten 2004 provides a powerful awareness document for web application security. The OWASP Top Ten represents a broad consensus about what the most critical web application security flaws are. Project members include a variety of security experts from around the world who have shared their expertise to produce this list.

Name: OWASP Top 10 2007
ID: 1EB1EC0E-74E6-49A0-BCE5-E6603802987A
The OWASP Top Ten 2007 provides a powerful awareness document for web application security. The OWASP Top Ten represents a broad consensus about what the most critical web application security flaws are. Project members include a variety of security experts from around the world who have shared their expertise to produce this list.

Name: OWASP Top 10 2010
ID: FDCECA5E-C2A8-4BE8-BB26-76A8ECD0ED59
The OWASP Top Ten 2010 provides a powerful awareness document for web application security. The OWASP Top Ten represents a broad consensus about what the most critical web application security flaws are. Project members include a variety of security experts from around the world who have shared their expertise to produce this list.

Name: OWASP Top 10 2013

ID: 1A2B4C7E-93B0-4502-878A-9BE40D2A25C4
The OWASP Top Ten 2013 provides a powerful awareness document for web application security. The OWASP Top Ten represents a broad consensus about what the most critical web application security flaws are. Project members include a variety of security experts from around the world who have shared their expertise to produce this list.

Name: OWASP Top 10 2017
ID: 3C6ECB67-BBD9-4259-A8DB-B49328927248
The OWASP Top Ten 2017 provides a powerful awareness document for web application security focused on informing the community about the consequences of the most common and most important web application security weaknesses. The OWASP Top Ten represents a broad agreement about what the most critical web application security flaws are with consensus being drawn from data collection and survey results. Project members include a variety of security experts from around the world who have shared their expertise to produce this list.

Name: PCI 1.1
ID: CBDB9D4D-FC20-4C04-AD58-575901CAB531
The Payment Card Industry (PCI) Data Security Standard (DSS) 1.1 compliance standard describes 12 requirements which are organized into 6 logically related groups, which are "control objectives". PCI DSS requirements are applicable if Primary Account Number (PAN) is stored, processed, or transmitted by the system.

Name: PCI 1.2
ID: 57940BDB-99F0-48BF-BF2E-CFC42BA035E5
Payment Card Industry Data Security Standard Version 1.2 description

Name: PCI 2.0
ID: 8970556D-7F9F-4EA7-8033-9DF39D68FF3E
The PCI DSS 2.0 compliance standard, particularly sections 6.3, 6.5, and 6.6, references the OWASP Top 10 vulnerability categories as the core categories that must be tested for and remediated. The following table summarizes the number of issues identified across the different PCI DSS requirements and broken down by Fortify Priority Order.

Name: PCI 3.0
ID: E2FB0D38-0192-4F03-8E01-FE2A12680CA3
The following is a summary of the application security portions of Payment Card Industry (PCI) Data Security Standard (DSS) v3.0. Fortify tests for 32 application security related requirements across sections 1, 2, 3, 4, 6, 7, 8, and 10 of PCI DSS and reports whether each requirement is In Place or Not In Place to indicate whether requirements are satisfied or not. This report is intended to measure the level of adherence the specific application(s) possess when compared to PCI DSS 3.0 compliance and is not intended to serve as a comprehensive Report on Compliance (ROC). The information contained in this report is targeted at project managers, security auditors, and compliance auditors.

Name: PCI 3.1
ID: AC0D18CF-C1DA-47CF-9F1A-E8EC0A4A717E
The following is a summary of the application security portions of Payment Card Industry (PCI) Data Security Standard (DSS) v3.1. Fortify tests for 31 application security related requirements across sections 1, 2, 3, 4, 6, 7, 8, and 10 of PCI DSS and reports whether each requirement is In Place or Not In Place to indicate whether requirements are satisfied or not. This report is intended to measure the level of adherence the specific application(s) possess when compared to PCI DSS 3.1 compliance and is not intended to serve as a comprehensive Report on Compliance (ROC). The information contained in this report is targeted at project managers, security auditors, and compliance auditors.

Name: PCI 3.2
ID: 4E8431F9-1BA1-41A8-BDBD-087D5826751A
The following is a summary of the application security portions of Payment Card Industry (PCI) Data Security Standard (DSS) v3.2. Fortify tests for 31 application security related requirements across sections 1, 2, 3, 4, 6, 7, 8, and 10 of PCI DSS and reports

whether each requirement is In Place or Not In Place to indicate whether requirements are satisfied or not. This report is intended to measure the level of adherence the specific application(s) possess when compared to PCI DSS 3.2 compliance and is not intended to serve as a comprehensive Report on Compliance (ROC). The information contained in this report is targeted at project managers, security auditors, and compliance auditors.

Name: PCI 3.2.1
ID: EADE255F-6561-4EFE-AD31-2914F6BFA329
The following is a summary of the application security portions of Payment Card Industry (PCI) Data Security Standard (DSS) v3.2.1. Fortify tests for 31 application security related requirements across sections 1, 2, 3, 4, 6, 7, 8, and 10 of PCI DSS and reports whether each requirement is In Place or Not In Place to indicate whether requirements are satisfied or not. This report is intended to measure the level of adherence the specific application(s) possess when compared to PCI DSS 3.2.1 compliance and is not intended to serve as a comprehensive Report on Compliance (ROC). The information contained in this report is targeted at project managers, security auditors, and compliance auditors.

Name: PCI SSF 1.0
ID: 0F551543-AF0E-4334-BEDF-1DDCD5F4BF74
The following is a summary of the application security portions of the Secure Software Requirements and Assessment Procedures defined in the Payment Card Industry (PCI) Software Security Framework (SSF) v1.0. Fortify tests for 23 application security related control objectives across Control Objective sections 2, 3, 4, 5, 6, 7, 8, and A.2 of PCI SSF and reports whether each control objective is In Place or Not In Place to indicate whether requirements are satisfied or not. This report is intended to measure the level of adherence the specific application(s) possess when compared to PCI SSF 1.0 compliance and is not intended to serve as a comprehensive Report on Compliance (ROC). The information contained in this report is targeted at project managers, security auditors, and compliance auditors.

Name: SANS Top 25 2009
ID: 939EF193-507A-44E2-ABB7-C00B2168B6D8
The 2009 CWE/SANS Top 25 Programming Errors lists the most significant programming errors that can lead to serious software vulnerabilities. They occur frequently, are often easy to find, and easy to exploit. They are dangerous because they will frequently allow attackers to completely take over the software, steal data, or prevent the software from working at all. The list is the result of collaboration between the SANS Institute, MITRE, and many top software security experts.

Name: SANS Top 25 2010
ID: 72688795-4F7B-484C-88A6-D4757A6121CA
SANS Top 25 2010 Most Dangerous Software Errors provides an enumeration of the most widespread and critical errors, categorized by Common Weakness Enumeration (CWE) identifiers, that lead to serious vulnerabilities in software (http://cwe.mitre.org/). These software errors are often easy to find and exploit. The inherent danger in these errors is that they can allow an attacker to completely take over the software, steal data, or prevent the software from working at all.

Name: SANS Top 25 2011
ID: 92EB4481-1FD9-4165-8E16-F2DE6CB0BD63
SANS Top 25 2011 Most Dangerous Software Errors provides an enumeration of the most widespread and critical errors, categorized by Common Weakness Enumeration (CWE) identifiers, that lead to serious vulnerabilities in software (http://cwe.mitre.org/). These software errors are often easy to find and exploit. The inherent danger in these errors is that they can allow an attacker to completely take over the software, steal data, or prevent the software from working at all.

Name: STIG 3.1
ID: F2FA57EA-5AAA-4DDE-90A5-480BE65CE7E7
Security Technical Implementation Guide Version 3.1 description

Name: STIG 3.10
ID: 788A87FE-C9F9-4533-9095-0379A9B35B12

Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APP<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APP5080: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APP5100: CAT II is not considered "In Place".

Name: STIG 3.4
ID: 58E2C21D-C70F-4314-8994-B859E24CF855
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APP<I>ID</I>: CAT <I>SEV</I>]. DISA STIG identifies several severities with respect to vulnerabilities:

<LI>CAT I: allow an attacker immediate access into a machine, allow super user access, or bypass a firewall.</LI>
<LI>CAT II: provide information that have a high potential of giving access to an intruder.</LI>
<LI>CAT III: provide information that potentially could lead to compromise.</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

Name: STIG 3.5
ID: DD18E81F-3507-41FA-9DFA-2A9A15B5479F
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APP<I>ID</I>: CAT <I>SEV</I>]. DISA STIG identifies several severities with respect to vulnerabilities:

<LI>CAT I: allow an attacker immediate access into a machine, allow super user access, or bypass a firewall.</LI>
<LI>CAT II: provide information that have a high potential of giving access to an intruder.</LI>
<LI>CAT III: provide information that potentially could lead to compromise.</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

Name: STIG 3.6
ID: 000CA760-0FED-4374-8AA2-6FA3968A07B1
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APP<I>ID</I>: CAT <I>SEV</I>]. DISA STIG identifies several severities with respect to vulnerabilities:

<LI>CAT I: allow an attacker immediate access into a machine, allow super user access, or bypass a firewall.</LI>
<LI>CAT II: provide information that have a high potential of giving access to an intruder.</LI>
<LI>CAT III: provide information that potentially could lead to compromise.</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APP5080: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APP5100: CAT II is not considered "In Place".

Name: STIG 3.7
ID: E69C07C0-81D8-4B04-9233-F3E74167C3D2
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APP<I>ID</I>: CAT <I>SEV</I>]. DISA STIG identifies several severities with respect to vulnerabilities:

<LI>CAT I: allow an attacker immediate access into a machine, allow super user access, or bypass a firewall.</LI>
<LI>CAT II: provide information that have a high potential of giving access to an intruder.</LI>
<LI>CAT III: provide information that potentially could lead to compromise.</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APP5080: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APP5100: CAT II is not considered "In Place".

Name: STIG 3.9
ID: 1A9D736B-2D4A-49D1-88CA-DF464B40D732
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APP<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APP5080: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APP5100: CAT II is not considered "In Place".

Name: STIG 4.1
ID: 95227C50-A9E4-4C9D-A8AF-FD98ABAE1F3C
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.10
ID: EF1FF442-1673-4CF1-B7C4-920F1A96A8150
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.2
ID: 672C15F8-8822-4E05-8C9E-1A4BAAA7A373
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.3
ID: A0B313F0-29BD-430B-9E34-6D10F1178506
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>

<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.4
ID: ECEC5CA2-7ACA-4B70-BF44-3248B9C6F4F8
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.5
ID: E6010E0A-7F71-4388-B8B7-EE9A02143474
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.6
ID: EFB9B012-44D6-456D-B197-03D2FD7C7AD6
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>].

DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.7
ID: B04A1E01-F1C1-48D3-A827-0F70872182D7
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>].
DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.8
ID: E6805D9F-D5B5-4192-962C-46828FF68507
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>].
DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: STIG 4.9

ID: 7B9F7B3B-07FC-4B61-99A1-70E3BB23A6A0
Each requirement or recommendation identified by the Defense Information Systems Agency (DISA) STIG is represented by a STIG identifier (STIGID), which corresponds to a checklist item and a severity code [APSC-DV-<I>ID</I>: CAT <I>SEV</I>]. DISA STIG defines three severities with respect to vulnerabilities where their:

<LI>exploitation leads to direct and immediate loss of Confidentiality, Availability, or Integrity (CAT I).</LI>
<LI>exploitation potentially results in loss of Confidentiality, Availability, or Integrity (CAT II).</LI>
<LI>existence degrades protections against loss of Confidentiality, Availability, or Integrity (CAT III).</LI> </UL>

The following table summarizes the number of issues identified across the different STIGIDs broken down by Fortify Priority Order. The status of a STIGID is considered "In Place" when there are no issues reported for a given STIGID.

If the project is missing a Fortify Static Code Analyzer (SCA) scan, or the scan contains findings that have not been fixed, hidden or suppressed, STIGID APSC-DV-003170: CAT II is not considered "In Place". Similarly, if the project is missing a Fortify WebInspect scan, or the scan contains any critical findings, STIGID APSC-DV-001460: CAT II and STIGID APSC-DV-002930: CAT II are not considered "In Place".

Name: WASC 2.00
ID: 74f8081d-dd49-49da-880f-6830cebe9777
The Web Application Security Consortium (WASC) was created as a cooperative effort to standardize, clarify, and organize the threats to the security of a web site. Version 2.00 of their Threat Classification outlines the attacks and weaknesses that can commonly lead to a website being compromised.

Name: WASC 24 + 2
ID: 9DC61E7F-1A48-4711-BBFD-E9DFF537871F
The Web Application Security Consortium (WASC) was created as a cooperative effort to standardize, clarify, and organize the threats to the security of a web site.

Properties

WinForms.CollectionMutationMonitor.Label=WinFormsDataSource
awt.toolkit=sun.awt.X11.XToolkit
com.fortify.AuthenticationKey=/home/agautam6/.fortify/config/tools
com.fortify.Core=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core
com.fortify.InstallRoot=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0
com.fortify.InstallationUserName=agautam6
com.fortify.SCAExecutablePath=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/bin/sourceanalyzer
com.fortify.TotalPhysicalMemory=8339922944
com.fortify.VS.RequireASPPrecompilation=true
com.fortify.WorkingDirectory=/home/agautam6/.fortify
com.fortify.locale=en
com.fortify.sca.AddImpliedMethods=true
com.fortify.sca.AntCompilerClass=com.fortify.dev.ant.SCACompiler
com.fortify.sca.AppendLogFile=true
com.fortify.sca.BuildID=owa
com.fortify.sca.BundleControlflowIssues=true
com.fortify.sca.BytecodePreview=true
com.fortify.sca.CollectPerformanceData=true
com.fortify.sca.CustomRulesDir=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/config/customrules
com.fortify.sca.DaemonCompilers=com.fortify.sca.util.compilers.GppCompiler,com.fortify.sca.util.compilers.GccCompiler,com.f

ortify.sca.util.compilers.AppleGppCompiler,com.fortify.sca.util.compilers.AppleGccCompiler,com.fortify.sca.util.compilers.Micr
osoftCompiler,com.fortify.sca.util.compilers.MicrosoftLinker,com.fortify.sca.util.compilers.LdCompiler,com.fortify.sca.util.com
pilers.ArUtil,com.fortify.sca.util.compilers.SunCCompiler,com.fortify.sca.util.compilers.SunCppCompiler,com.fortify.sca.util.co
mpilers.IntelCompiler,com.fortify.sca.util.compilers.ExternalCppAdapter,com.fortify.sca.util.compilers.ClangCompiler
com.fortify.sca.DeadCodeFilter=true
com.fortify.sca.DeadCodeIgnoreTrivialPredicates=true
com.fortify.sca.DefaultAnalyzers=semantic:dataflow:controlflow:nullptr:configuration:content:structural:buffer
com.fortify.sca.DefaultFileTypes=java,rb,erb,jsp,jspx,jspf,tag,tagx,tld,sql,cfm,php,phtml,ctp,pks,pkh,pkb,xml,config,Config,settin
gs,properties,dll,exe,winmd,cs,vb,asax,ascx,ashx,asmx,aspx,master,Master,xaml,baml,cshtml,vbhtml,inc,asp,vbscript,js,ini,bas,cls
,vbs,frm,ctl,html,htm,xsd,wsdd,xmi,py,cfml,cfc,abap,xhtml,cpx,xcfg,jsff,as,mxml,cbl,cscfg,csdef,wadcfg,wadcfgx,appxmanifest,
wsdl,plist,bsp,ABAP,BSP,swift,page,trigger,scala,ts,conf,json,yaml,yml
com.fortify.sca.DefaultJarsDirs=default_jars
com.fortify.sca.DefaultRulesDir=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/config/rules
com.fortify.sca.DisableDeadCodeElimination=false
com.fortify.sca.DisableFunctionPointers=false
com.fortify.sca.DisableGlobals=false
com.fortify.sca.DisableInferredConstants=false
com.fortify.sca.EnableInterproceduralConstantResolution=true
com.fortify.sca.EnableNestedWrappers=true
com.fortify.sca.EnableStructuralMatchCache=true
com.fortify.sca.EnableWrapperDetection=true
com.fortify.sca.FVDLDisableDescriptions=false
com.fortify.sca.FVDLDisableProgramData=false
com.fortify.sca.FVDLDisableSnippets=false
com.fortify.sca.FVDLStylesheet=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/resources/sca/fvdl2html.xsl
com.fortify.sca.IndirectCallGraphBuilders=WinFormsAdHocFunctionBuilder,VirtualCGBuilder,J2EEIndirectCGBuilder,JNICG
Builder,StoredProcedureResolver,JavaWSCGBuilder,StrutsCGBuilder,DotNetWSCGBuilder,SqlServerSPResolver,ASPCGBuild
er,ScriptedCGBuilder,NewJspCustomTagCGBuilder,DotNetCABCGBuilder,StateInjectionCGBuilder,SqlServerSPResolver2,PH
PLambdaResolver,JavaWebCGBuilder
com.fortify.sca.JVMArgs=-XX:SoftRefLRUPolicyMSPerMB=3000 -Xmx6192439296 -Xss16M
com.fortify.sca.JavaSourcepathSearch=true
com.fortify.sca.JdkVersion=1.8
com.fortify.sca.LogFileDir=/home/agautam6/.fortify/sca19.1/log
com.fortify.sca.LogFileExt=.log
com.fortify.sca.LogFileName=sca.log
com.fortify.sca.LogFileNameNoExt=sca
com.fortify.sca.LogFilePath=/home/agautam6/.fortify/sca19.1/log/sca.log
com.fortify.sca.LogLevel=INFO
com.fortify.sca.LowSeverityCutoff=1.0
com.fortify.sca.MachineOutputMode=
com.fortify.sca.MultithreadedAnalysis=true
com.fortify.sca.NoNestedOutTagOutput=org.apache.taglibs.standard.tag.rt.core.RemoveTag,org.apache.taglibs.standard.tag.rt.cor
e.SetTag
com.fortify.sca.OldVbNetExcludeFileTypes=vb,asax,ascx,ashx,asmx,aspx,xaml,cshtml,vbhtml
com.fortify.sca.PID=19576
com.fortify.sca.Phase0HigherOrder.Languages=python,ruby,swift,javascript,typescript
com.fortify.sca.Phase0HigherOrder.Level=1
com.fortify.sca.PrintPerformanceDataAfterScan=false
com.fortify.sca.ProjectRoot=/home/agautam6/.fortify
com.fortify.sca.ProjectRoot=/home/agautam6/.fortify
com.fortify.sca.Renderer=fpr

com.fortify.sca.RequireMapKeys=classrule
com.fortify.sca.ResultsFile=/srv/openmrs_code/org/openmrs/module/owa/owa_scan.fpr
com.fortify.sca.SolverTimeout=15
com.fortify.sca.SqlLanguage=PLSQL
com.fortify.sca.SuppressLowSeverity=true
com.fortify.sca.ThreadCount.NameTableLoading=1
com.fortify.sca.TypeInferenceFunctionTimeout=60
com.fortify.sca.TypeInferenceLanguages=javascript,typescript,python,ruby
com.fortify.sca.TypeInferencePhase0Timeout=300
com.fortify.sca.UnicodeInputFile=true
com.fortify.sca.UniversalBlacklist=.*yyparse.*
com.fortify.sca.alias.mode.csharp=fs
com.fortify.sca.alias.mode.javascript=fi
com.fortify.sca.alias.mode.scala=fi
com.fortify.sca.alias.mode.swift=fi
com.fortify.sca.alias.mode.typescript=fi
com.fortify.sca.alias.mode.vb=fs
com.fortify.sca.analyzer.controlflow.EnableLivenessOptimization=false
com.fortify.sca.analyzer.controlflow.EnableMachineFiltering=false
com.fortify.sca.analyzer.controlflow.EnableRefRuleOptimization=false
com.fortify.sca.analyzer.controlflow.EnableTimeOut=true
com.fortify.sca.compilers.ant=com.fortify.sca.util.compilers.AntAdapter
com.fortify.sca.compilers.ar=com.fortify.sca.util.compilers.ArUtil
com.fortify.sca.compilers.armcc=com.fortify.sca.util.compilers.ArmCcCompiler
com.fortify.sca.compilers.armcpp=com.fortify.sca.util.compilers.ArmCppCompiler
com.fortify.sca.compilers.c++=com.fortify.sca.util.compilers.GppCompiler
com.fortify.sca.compilers.cc=com.fortify.sca.util.compilers.GccCompiler
com.fortify.sca.compilers.clearmake=com.fortify.sca.util.compilers.TouchlessCompiler
com.fortify.sca.compilers.fortify=com.fortify.sca.util.compilers.FortifyCompiler
com.fortify.sca.compilers.g++=com.fortify.sca.util.compilers.GppCompiler
com.fortify.sca.compilers.g++-*=com.fortify.sca.util.compilers.GppCompiler
com.fortify.sca.compilers.g++2*=com.fortify.sca.util.compilers.GppCompiler
com.fortify.sca.compilers.g++3*=com.fortify.sca.util.compilers.GppCompiler
com.fortify.sca.compilers.g++4*=com.fortify.sca.util.compilers.GppCompiler
com.fortify.sca.compilers.gcc=com.fortify.sca.util.compilers.GccCompiler
com.fortify.sca.compilers.gcc-*=com.fortify.sca.util.compilers.GccCompiler
com.fortify.sca.compilers.gcc2*=com.fortify.sca.util.compilers.GccCompiler
com.fortify.sca.compilers.gcc3*=com.fortify.sca.util.compilers.GccCompiler
com.fortify.sca.compilers.gcc4*=com.fortify.sca.util.compilers.GccCompiler
com.fortify.sca.compilers.gmake=com.fortify.sca.util.compilers.TouchlessCompiler
com.fortify.sca.compilers.gradle=com.fortify.sca.util.compilers.GradleAdapter
com.fortify.sca.compilers.gradlew=com.fortify.sca.util.compilers.GradleAdapter
com.fortify.sca.compilers.icc=com.fortify.sca.util.compilers.IntelCompiler
com.fortify.sca.compilers.icpc=com.fortify.sca.util.compilers.IntelCompiler
com.fortify.sca.compilers.jam=com.fortify.sca.util.compilers.TouchlessCompiler
com.fortify.sca.compilers.javac=com.fortify.sca.util.compilers.JavacCompiler
com.fortify.sca.compilers.ld=com.fortify.sca.util.compilers.LdCompiler
com.fortify.sca.compilers.make=com.fortify.sca.util.compilers.TouchlessCompiler
com.fortify.sca.compilers.mvn=com.fortify.sca.util.compilers.MavenAdapter
com.fortify.sca.compilers.scalac=com.fortify.sca.util.compilers.ScalacCompiler
com.fortify.sca.compilers.tcc=com.fortify.sca.util.compilers.ArmCcCompiler

com.fortify.sca.compilers.tcpp=com.fortify.sca.util.compilers.ArmCppCompiler
com.fortify.sca.compilers.touchless=com.fortify.sca.util.compilers.FortifyCompiler
com.fortify.sca.cpfe.441.command=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/private-bin/sca/cpfe441.rfct
com.fortify.sca.cpfe.command=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/private-bin/sca/cpfe48
com.fortify.sca.cpfe.file.option=--gen_c_file_name
com.fortify.sca.cpfe.options=--remove_unneeded_entities --suppress_vtbl -tused
com.fortify.sca.cpfe.options=--remove_unneeded_entities --suppress_vtbl -tused
com.fortify.sca.env.exesearchpath=/sbin:/bin:/usr/bin:/usr/local/bin
com.fortify.sca.fileextensions.ABAP=ABAP
com.fortify.sca.fileextensions.BSP=ABAP
com.fortify.sca.fileextensions.Config=XML
com.fortify.sca.fileextensions.abap=ABAP
com.fortify.sca.fileextensions.appxmanifest=XML
com.fortify.sca.fileextensions.as=ACTIONSCRIPT
com.fortify.sca.fileextensions.asp=ASP
com.fortify.sca.fileextensions.bas=VB6
com.fortify.sca.fileextensions.bsp=ABAP
com.fortify.sca.fileextensions.cfc=CFML
com.fortify.sca.fileextensions.cfm=CFML
com.fortify.sca.fileextensions.cfml=CFML
com.fortify.sca.fileextensions.cls=VB6
com.fortify.sca.fileextensions.conf=HOCON
com.fortify.sca.fileextensions.config=XML
com.fortify.sca.fileextensions.cpx=XML
com.fortify.sca.fileextensions.cscfg=XML
com.fortify.sca.fileextensions.csdef=XML
com.fortify.sca.fileextensions.ctl=VB6
com.fortify.sca.fileextensions.ctp=PHP
com.fortify.sca.fileextensions.erb=RUBY_ERB
com.fortify.sca.fileextensions.faces=JSPX
com.fortify.sca.fileextensions.frm=VB6
com.fortify.sca.fileextensions.htm=HTML
com.fortify.sca.fileextensions.html=HTML
com.fortify.sca.fileextensions.ini=JAVA_PROPERTIES
com.fortify.sca.fileextensions.java=JAVA
com.fortify.sca.fileextensions.js=TYPESCRIPT
com.fortify.sca.fileextensions.jsff=JSPX
com.fortify.sca.fileextensions.json=JSON
com.fortify.sca.fileextensions.jsp=JSP
com.fortify.sca.fileextensions.jspf=JSP
com.fortify.sca.fileextensions.jspx=JSPX
com.fortify.sca.fileextensions.jsx=TYPESCRIPT
com.fortify.sca.fileextensions.mxml=MXML
com.fortify.sca.fileextensions.page=VISUAL_FORCE
com.fortify.sca.fileextensions.php=PHP
com.fortify.sca.fileextensions.phtml=PHP
com.fortify.sca.fileextensions.pkb=PLSQL
com.fortify.sca.fileextensions.pkh=PLSQL
com.fortify.sca.fileextensions.pks=PLSQL
com.fortify.sca.fileextensions.plist=XML
com.fortify.sca.fileextensions.properties=JAVA_PROPERTIES

com.fortify.sca.fileextensions.py=PYTHON
com.fortify.sca.fileextensions.rb=RUBY
com.fortify.sca.fileextensions.scala=SCALA
com.fortify.sca.fileextensions.settings=XML
com.fortify.sca.fileextensions.sql=SQL
com.fortify.sca.fileextensions.swift=SWIFT
com.fortify.sca.fileextensions.tag=JSP
com.fortify.sca.fileextensions.tagx=JSP
com.fortify.sca.fileextensions.tld=TLD
com.fortify.sca.fileextensions.trigger=APEX_TRIGGER
com.fortify.sca.fileextensions.ts=TYPESCRIPT
com.fortify.sca.fileextensions.tsx=TYPESCRIPT
com.fortify.sca.fileextensions.vbs=VBSCRIPT
com.fortify.sca.fileextensions.vbscript=VBSCRIPT
com.fortify.sca.fileextensions.wadcfg=XML
com.fortify.sca.fileextensions.wadcfgx=XML
com.fortify.sca.fileextensions.wsdd=XML
com.fortify.sca.fileextensions.wsdl=XML
com.fortify.sca.fileextensions.xcfg=XML
com.fortify.sca.fileextensions.xhtml=JSPX
com.fortify.sca.fileextensions.xmi=XML
com.fortify.sca.fileextensions.xml=XML
com.fortify.sca.fileextensions.xsd=XML
com.fortify.sca.fileextensions.yaml=YAML
com.fortify.sca.fileextensions.yml=YAML
com.fortify.sca.jsp.UseNativeParser=true
com.fortify.sca.parser.python.ignore.module.1=test.badsyntax_future3
com.fortify.sca.parser.python.ignore.module.2=test.badsyntax_future4
com.fortify.sca.parser.python.ignore.module.3=test.badsyntax_future5
com.fortify.sca.parser.python.ignore.module.4=test.badsyntax_future6
com.fortify.sca.parser.python.ignore.module.5=test.badsyntax_future7
com.fortify.sca.parser.python.ignore.module.6=test.badsyntax_future8
com.fortify.sca.parser.python.ignore.module.7=test.badsyntax_future9
com.fortify.sca.parser.python.ignore.module.8=test.badsyntax_nocaret
com.fortify.sca.skip.libraries.AngularJS=angular.js,angular.min.js,angular-animate.js,angular-aria.js,angular_1_router.js,angular-cookies.js,angular-message-format.js,angular-messages.js,angular-mocks.js,angular-parse-ext.js,angular-resource.js,angular-route.js,angular-sanitize.js,angular-touch.js
com.fortify.sca.skip.libraries.ES6=es6-shim.min.js,system-polyfills.js,shims_for_IE.js
com.fortify.sca.skip.libraries.jQuery=jquery.js,jquery.min.js,jquery-migrate.js,jquery-migrate.min.js,jquery-ui.js,jquery-ui.min.js,jquery.mobile.js,jquery.mobile.min.js,jquery.color.js,jquery.color.min.js,jquery.color.svg-names.js,jquery.color.svg-names.min.js,jquery.color.plus-names.js,jquery.color.plus-names.min.js,jquery.tools.min.js
com.fortify.sca.skip.libraries.javascript=bootstrap.js,bootstrap.min.js,typescript.js,typescriptServices.js
com.fortify.sca.skip.libraries.typescript=typescript.d.ts,typescriptServices.d.ts
com.fortify.search.defaultSyntaxVer=2
com.sun.management.jmxremote=true
file.encoding=UTF-8
file.encoding.pkg=sun.io
file.separator=/
java.awt.graphicsenv=sun.awt.X11GraphicsEnvironment
java.awt.headless=true
java.awt.printerjob=sun.print.PSPrinterJob

java.class.path=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/lib/exe/sca-exe.jar
java.class.version=52.0
java.endorsed.dirs=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/endorsed
java.ext.dirs=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/ext:/usr/java/packages/lib/ext
java.home=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre
java.io.tmpdir=/tmp
java.library.path=/usr/java/packages/lib/amd64:/usr/lib64:/lib64:/lib:/usr/lib
java.rmi.server.randomIDs=true
java.runtime.name=OpenJDK Runtime Environment
java.runtime.version=1.8.0_181-b02
java.specification.name=Java Platform API Specification
java.specification.vendor=Oracle Corporation
java.specification.version=1.8
java.vendor=Azul Systems, Inc.
java.vendor.url=http://www.azulsystems.com/
java.vendor.url.bug=http://www.azulsystems.com/support/
java.version=1.8.0_181
java.vm.info=mixed mode
java.vm.name=OpenJDK 64-Bit Server VM
java.vm.specification.name=Java Virtual Machine Specification
java.vm.specification.vendor=Oracle Corporation
java.vm.specification.version=1.8
java.vm.vendor=Azul Systems, Inc.
java.vm.version=25.181-b02
line.separator=

log4j.configurationFile=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/config/log4j2.xml
log4j.isThreadContextMapInheritable=true
max.file.path.length=255
os.arch=amd64
os.name=Linux
os.version=4.15.0-58-generic
path.separator=:
stderr.isatty=false
stdout.isatty=false
sun.arch.data.model=64
sun.boot.class.path=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/resources.jar:/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/rt.jar:/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/sunrsasign.jar:/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/jsse.jar:/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/jce.jar:/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/charsets.jar:/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/jfr.jar:/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/classes
sun.boot.library.path=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/jre/lib/amd64
sun.cpu.endian=little
sun.cpu.isalist=
sun.io.unicode.encoding=UnicodeLittle
sun.java.command=sourceanalyzer -Djava.awt.headless=true -Dcom.sun.management.jmxremote=true -XX:SoftRefLRUPolicyMSPerMB=3000 -Dcom.fortify.sca.env.exesearchpath=/sbin:/bin:/usr/bin:/usr/local/bin -Dcom.fortify.sca.ProjectRoot=/home/agautam6/.fortify -Dstdout.isatty=false -Dstderr.isatty=false -Dcom.fortify.sca.PID=19576 -Xmx6192439296 -Dcom.fortify.TotalPhysicalMemory=8339922944 -Xss16M -Dcom.fortify.sca.JVMArgs=-XX:SoftRefLRUPolicyMSPerMB=3000 -Xmx6192439296 -Xss16M -Djava.class.path=/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/lib/exe/sca-exe.jar -scan @/home/agautam6/.fortify/Eclipse.Plugin-19.1.0/owa/owaScan.txt

sun.jnu.encoding=UTF-8
sun.management.compiler=HotSpot 64-Bit Tiered Compilers
sun.os.patch.level=unknown
user.country=US
user.dir=/home/agautam6
user.home=/home/agautam6
user.language=en
user.name=agautam6
user.timezone=America/New_York

Commandline Arguments

-scan
-b
owa
-format
fpr
-machine-output
-f
/srv/openmrs_code/org/openmrs/module/owa/owa_scan.fpr

Warnings

[1001] Unexpected exception while parsing file (javascript) /srv/openmrs_code/org/openmrs/module/owa/omod/target/classes/web/module/resources/javascript/main.js: Parse error at line 69, column 1. Encountered: var
[1001] Unexpected exception while parsing file (javascript) /srv/openmrs_code/org/openmrs/module/owa/omod/target/owa-1.10.0/web/module/resources/javascript/main.js: Parse error at line 69, column 1. Encountered: var
[1001] Unexpected exception while parsing file (javascript) /srv/openmrs_code/org/openmrs/module/owa/omod/src/main/webapp/resources/javascript/main.js: Parse error at line 69, column 1. Encountered: var
[12002] Could not locate the deployment descriptor (web.xml) for your web application. Please build your web application and try again. File: /srv/openmrs_code/org/openmrs/module/owa/omod/src/main/webapp/manage.jsp
[12003] Assuming Java source level to be 1.8 as it was not specified. Note that the default value may change in future versions.
[12004] The Java frontend was unable to resolve the following include:
/WEB-INF/template/include.jsp at /srv/openmrs_code/org/openmrs/module/owa/omod/target/owa-1.10.0/web/module/settings.jsp:1.
/WEB-INF/template/footer.jsp at /srv/openmrs_code/org/openmrs/module/owa/omod/target/owa-1.10.0/web/module/settings.jsp:58.
/WEB-INF/template/header.jsp at /srv/openmrs_code/org/openmrs/module/owa/omod/target/owa-1.10.0/web/module/settings.jsp:3.
[12019] The following references to Java functions could not be resolved. These functions may be part of classes that could not be found, or there may be a type error at the call site of the given function relative to the function declaration. Please ensure the Java source code can be compiled by a Java compiler.
javax.servlet.http.HttpSession.getAttribute
javax.servlet.http.HttpSession.removeAttribute

[12022] The class "javax.servlet.http.HttpServlet" could not be found on the classpath, but it was found in the JAR file provided by Fortify in "/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/default_jars/javax.servlet-api-3.0.1.jar" as a convenience. To ensure consistent translation behavior add the JAR file that contains "javax.servlet.http.HttpServlet" to the classpath given to the translation step. Refer to the documentation about "default JARs" in the SCA User Guide for more information.

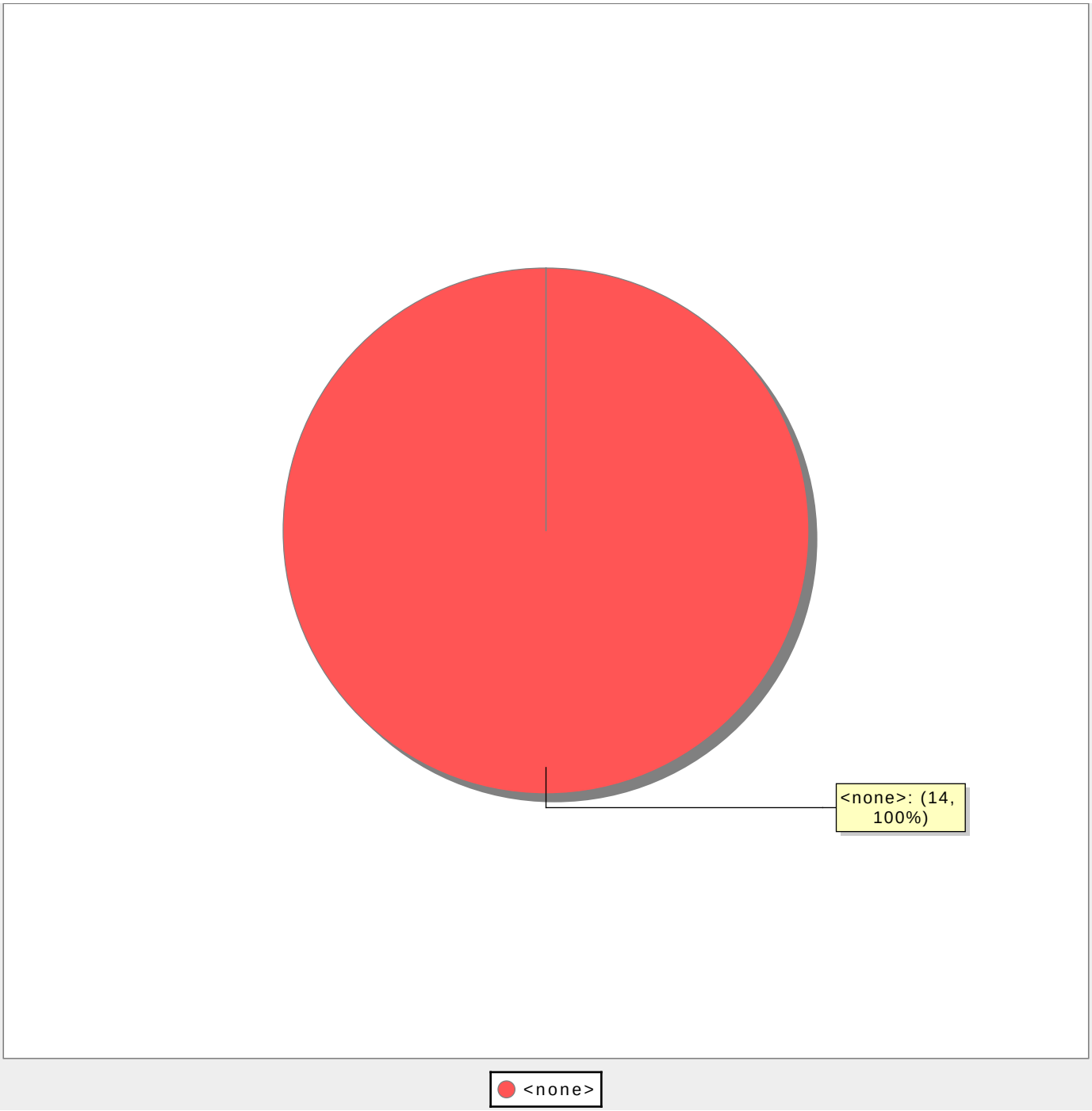
[12022] The class "javax.servlet.jsp.PageContext" could not be found on the classpath, but it was found in the JAR file provided by Fortify in "/opt/Fortify/Fortify_SCA_and_Apps_19.1.0/Core/default_jars/javax.servlet.jsp-api.jar" as a convenience. To ensure consistent translation behavior add the JAR file that contains "javax.servlet.jsp.PageContext" to the classpath given to the translation step. Refer to the documentation about "default JARs" in the SCA User Guide for more information.
[1214] Multiple definitions found for class /manage.jsp
(/srv/openmrs_code/org/openmrs/module/owa/omod/src/main/webapp/manage.jsp and
/srv/openmrs_code/org/openmrs/module/owa/omod/target/classes/web/module/manage.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspmanage_jsp$ftfy_frameworkVisibleObjects
(/srv/openmrs_code/org/openmrs/module/owa/omod/src/main/webapp/manage.jsp and
/srv/openmrs_code/org/openmrs/module/owa/omod/target/classes/web/module/manage.jsp).
[1214] Multiple definitions found for class /settings.jsp
(/srv/openmrs_code/org/openmrs/module/owa/omod/src/main/webapp/settings.jsp and
/srv/openmrs_code/org/openmrs/module/owa/omod/target/classes/web/module/settings.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspsettings_jsp$ftfy_frameworkVisibleObjects
(/srv/openmrs_code/org/openmrs/module/owa/omod/src/main/webapp/settings.jsp and
/srv/openmrs_code/org/openmrs/module/owa/omod/target/classes/web/module/settings.jsp).
[1214] Multiple definitions found for class /localHeader.jsp
(/srv/openmrs_code/org/openmrs/module/owa/omod/src/main/webapp/template/localHeader.jsp and
/srv/openmrs_code/org/openmrs/module/owa/omod/target/classes/web/module/template/localHeader.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jsplocalHeader_jsp$ftfy_frameworkVisibleObjects
(/srv/openmrs_code/org/openmrs/module/owa/omod/src/main/webapp/template/localHeader.jsp and
/srv/openmrs_code/org/openmrs/module/owa/omod/target/classes/web/module/template/localHeader.jsp).
[1214] Multiple definitions found for class /manage.jsp
(/srv/openmrs_code/org/openmrs/module/owa/omod/src/main/webapp/manage.jsp and
/srv/openmrs_code/org/openmrs/module/owa/omod/target/owa-1.10.0/web/module/manage.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspmanage_jsp$ftfy_frameworkVisibleObjects
(/srv/openmrs_code/org/openmrs/module/owa/omod/src/main/webapp/manage.jsp and
/srv/openmrs_code/org/openmrs/module/owa/omod/target/owa-1.10.0/web/module/manage.jsp).
[1214] Multiple definitions found for class /settings.jsp
(/srv/openmrs_code/org/openmrs/module/owa/omod/src/main/webapp/settings.jsp and
/srv/openmrs_code/org/openmrs/module/owa/omod/target/owa-1.10.0/web/module/settings.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jspsettings_jsp$ftfy_frameworkVisibleObjects
(/srv/openmrs_code/org/openmrs/module/owa/omod/src/main/webapp/settings.jsp and
/srv/openmrs_code/org/openmrs/module/owa/omod/target/owa-1.10.0/web/module/settings.jsp).
[1214] Multiple definitions found for class /localHeader.jsp
(/srv/openmrs_code/org/openmrs/module/owa/omod/src/main/webapp/template/localHeader.jsp and
/srv/openmrs_code/org/openmrs/module/owa/omod/target/owa-1.10.0/web/module/template/localHeader.jsp).
[1214] Multiple definitions found for class JSPPAGE._/_jsplocalHeader_jsp$ftfy_frameworkVisibleObjects
(/srv/openmrs_code/org/openmrs/module/owa/omod/src/main/webapp/template/localHeader.jsp and
/srv/openmrs_code/org/openmrs/module/owa/omod/target/owa-1.10.0/web/module/template/localHeader.jsp).
[1215] Could not locate the root (WEB-INF) of the web application. Please build your web application and try again.
[1216] Unable to locate a class for import org.openmrs.web.WebConstants
[1237] The following references to Java symbols could not be resolved. Please make sure to supply all the required JAR files that contain these symbols to SCA.
java.lang.String.OPENMRS_ERROR_ATTR
java.lang.String.OPENMRS_MSG_ATTR

Issue Count by Category

Issues by Category	
Cross-Site Scripting: DOM	6
Header Manipulation	3
File Disclosure: J2EE	2
Open Redirect	1
Path Manipulation	1
Race Condition: Singleton Member Field	1

Issue Breakdown by Analysis

Issues by Analysis



New Issues

Issues by New Issue

The following issues have been discovered since the last scan.