Logging

V7.1 Log Content

---

1. Test Case ID: 7.1.1-1
Description: Verify that the application does not log credentials or payment details. Session tokens should only be stored in logs in an irreversible, hashed form.
Repeatable Step:
1. Login as any user (Eg: username: admin_openemr password: admin_openemr)
2. In the top right corner, hover over the username, then select change password from the dropdown menu**.**
3. Enter the old password (admin_openemr) and repeat the new password twice in the form according to the requirement - test_OpenEmr@2 (should contain at least each of the following items: A number, a lowercase letter, an uppercase letter, a special character that is not a letter or number)
4. Click on Save Changes.

Expected Results:

- The credentials should not be logged anywhere in the openemr system. The user that has performed this operation and the timestamp to indicate when this was performed should be logged to support accountability and nonrepudiation.

CWE Info: 532: Insertion of Sensitive Information into Log File

2. Test Case ID: 7.1.4-1
Description: Verify that each log event includes necessary information that would allow for a detailed investigation of the timeline when an event happens.
Repeatable Step:
1. Login as any user (Eg: username: admin_openemr password: admin_openemr)
2. Hover over patient/client on the navigation bar and click on patients.
3. Click on add new patients, add relevant details like name, SSN, DOB, and sex. Click on create a new patient

Expected Results:

- All events like create and update should have the necessary information. The user that has performed this operation and the timestamp to indicate when this was performed should be logged to support accountability and nonrepudiation.

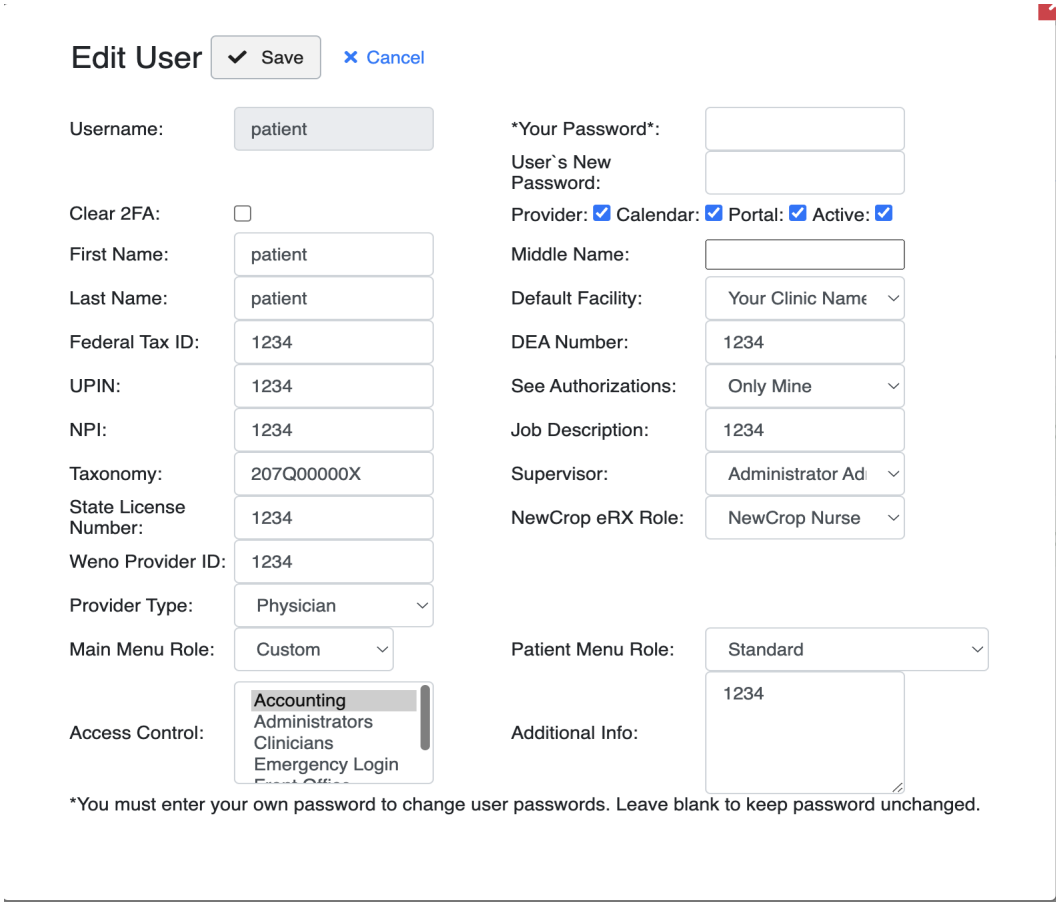
CWE Info: 778: Insufficient Logging

3. Test Case ID: 7.2.1-1
Description: Verify that all authentication decisions are logged, without storing sensitive session tokens or passwords. This should include requests with relevant metadata needed for security investigations.
Repeatable Step:
1. Login as any user (Eg: username: admin_openemr password: admin_openemr).
2. Connect to the VCL being used as an openemr host.
3. Open the terminal and enter the command “MySQL -u root -p” and enter the password as “root”.
4. Then use the openemr database by typing the command “use openemr;”.
5. Now enter the query “select * from log;” and check the last record(which is the latest in our case) that has the login details captured.
Expected Results:

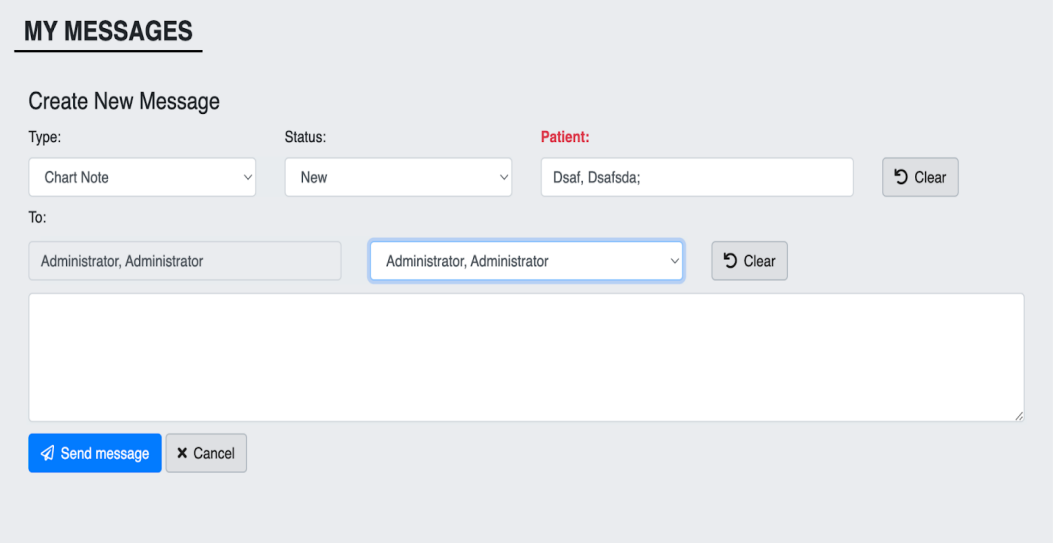
- All the events captured in the logs should not contain any sensitive information.

CWE Info: 778: Insufficient Logging

4. Test Case ID: 7.2.2-1
Description: Verify that all access control decisions can be logged and all failed decisions are logged. This should include requests with relevant metadata needed for security investigations.
Repeatable Step:
1. Login as any user (Eg: username: admin_openemr password: test).
2. Login failure popup will be displayed, as those are false login details.
3. Now login using credentials (username: admin_openemr password: admin_openemr).
4. Navigate to Administration -> Users and click on “+ Add User” button. And fill in the details as mentioned below. For Your Password: “admin_openemr” and for User’s New Password: “67984532@vV”.



5. Once all the details are filled click on the “Save” button.
6. Now log out and re-login using credentials (username: patient password: 67984532@vV).
7. Click on the “Message Center” tab and click on the “+ Add New” button.
8. Now fill in the details as mentioned in the below screenshot. Before this step, create a patient by navigating to Patient/Client -> Patients and by clicking on the “+Add New Patient” button with some dummy details. This will help us to select a patient in the “Patient:” blank mentioned below.



9. Once the details are filled click on the “Send Message” button.
10. Now you will not be able to see the status of the message sent to the administrator in the UI. Instead, this event gets recorded in the log table. Now let's check this in the log table in the backend.
11. Connect to the VCL being used as an openemr host.
12. Open the terminal and enter the command “MySQL -u root -p” and enter the password as “root”.
13. Then use the openemr database by typing the command “use openemr;”.
14. Now enter the query “select * from log;” and check the last few records which contain our failed login attempt and the message event between patient and administrator.
15. And in order to check the same event from UI, log out from the patient account and re-login to the administrator account (username: admin_openemr password: admin_openemr). Now navigate to “Message Center” and now you should be able to see a message from “Patient” in that list.

Expected Results:

- All the access-controlled events both failures and successes, are logged in the database. It should contain metadata that is useful for further investigation.

CWE Info: 285: Improper Authorization.

5. Test Case ID: 7.1.1-2
Description: Verify that the application does not log credentials or payment details. Session tokens should only be stored in logs in an irreversible, hashed form.
Repeatable Steps:
1. Open openemr application in vcl. Or navigate to http://localhost/openemr in firefox inside vcl.
2. Login as admin using admin credentials. (username: admin_openemr, password: admin_openemr).

3. On the top of the openemr application there is a tab called Administration. Hover over the Administration tab and click on Users.
4. Enter the following details to create a new user.
    i. Username: doctor_openemr
    ii. First Name: doctor
    iii. Last Name: doctor
    iv. Password: Doctor@123
    v. Your Password: admin_openemr
    vi. Access Control: Physicians
    vii. NewCrop eRX Role: NewCrop Doctor
5. And then click on save button.
6. Hover again on Administration tab on top of the page. Hover on System and click on Logs.
7. Click on Submit button.
8. Check the comments for each log.

Expected Results:

- The logs shouldn't contain the login credentials like Password. The actor who created the new user is displayed under User in the log table.

CWE Info: 532: Insertion of Sensitive Information into Log File

6. **Test Case ID:** 7.1.3-1
Description: Verify that the application logs security relevant events including successful and failed authentication events, access control failures, deserialization failures and input validation failures.
Repeatable Steps:
1. Open openemr application in vcl. Or navigate to http://localhost/openemr in firefox inside vcl.
2. Try to log in as admin using wrong password. (username: admin_openemr, password: admin).
3. Now log in using correct password. (username: admin_openemr, password: admin_openemr).
4. Hover Administration tab on top of the page. Hover on System and click on Logs.
7. Click on Submit button.
8. Check the comments for each log in log table.

Expected Results:

- The login attempt failure should be logged along with the actor who tried to login.

CWE Info: 778: Insufficient Logging

V7.3 Log Protection

---

7. Test Case ID: 7.3.4-1
Description: Verify that time sources are synchronized to the correct time and time zone. Strongly consider logging only in UTC if systems are global to assist with post-incident forensic analysis.
Repeatable Step:
1. Login as any user (Eg: username: admin_openemr password: admin_openemr)
2. Hover Administration tab on top of the page. Hover on System and click on Logs.
3. Click Submit and note the time of the log entry for your login

Expected Results:

- The time is displayed in the user's time zone. This is helpful when performing forensic investigations into security violations.

CWE Info: N/A - No CWE info in ASVS. Mainly helpful for forensic investigations

8. Test Case ID: 7.3.4-2
Description: Verify that time sources are synchronized to the correct time and time zone. Strongly consider logging only in UTC if systems are global to assist with post-incident forensic analysis.
Repeatable Step:
1. Login as any user (Eg: username: admin_openemr password: admin_openemr)
2. Open the terminal and enter the command "mysql -u root -p" and enter the password as "root".
3. Then use the openemr database by typing the command "use openemr;".
4. Now enter the query "select * from log where event='login' order by date desc;"
5. Note the time of the first entry

Expected Results:

- The time is saved in UTC. This is helpful when performing forensic investigations into security violations.

CWE Info: N/A - No CWE info in ASVS. Mainly helpful for forensic investigations

Comment on OpenEMR adequacy:
1. The openEMR logging is pretty good. Whenever a failed login or success login attempt happens, it logs the events which is the expectation from an application standpoint.
2. Also, only the administrator has access to take a look at the logs so that other users can't steal the sensitive data like usernames from the logs.

Time Spent: 6 hours
True Positives: 1
True Positives per hour: 0.167

Seeker

Black-box test cases ran in the Past:

1. Test Case ID: 4.3.2-1
Description: Verify that directory browsing is disabled unless deliberately desired. Additionally, applications should not allow discovery or disclosure of file or directory metadata, such as Thumbs.db, .DS_Store, .git or .svn folders
Repeatable Step:
1. Go to the openemr login page in firefox browser in vcl.
2. The login page will look like http://localhost/openemr/interface/login/login.php?site=default .
3. Edit the URL to http://localhost/openemr/interface.
4. You can view the list of all files in the directory.

Expected Results:
1. No one should have access to look at the files in the directory.
2.The directory should not disclose the file or directory metadata, such as Thumbs.db, .DS_Store, .git or .svn folders.

CWE Info: 548: Exposure of Information Through Directory Listing

2. Test Case ID: 5.2.2-1
Description: Verify that unstructured data is sanitized to enforce safety measures such as allowed characters and length.
Repeatable Step:
1. Go to the openemr login page in firefox browser in vcl.
2. Login as administrator (Eg: username: admin_openemr , password: admin_openemr)
2. Under the administration tab, click on Users
3. Click on the username (Eg: admin_openemr)
4. Enter the Last Name as **<iframe src="javascript:alert(document.cookie)">**
5. Log out and log back as administrator again to view the user's name in the top right corner

Expected Results: The input should be sanitized to allow only letters and whitespace.

CWE Info: 138: Improper Neutralization of Special Elements

3. Test Case id: 5.3.4-2
Description: 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.
Repeatable Steps:
1. Open ZAP in vcl.
2. Clock on Manual Explore.
3. Enter the openemr url "**http://localhost/openemr**" in the URL dialogue box, and then click on launch browser with Firefox as default browser.
4. Login as administrator in firefox browser with username: **admin_openemr** and password: **admin_openemr**
5. On the top headers, hover over **Patient/Client** and click on **Patients**.
6. Under Patient finder, click on Add New Patient.
7. Create a new patient. Select Mr. from pronouns drop down. Type **Abc** as first name in first box, leave out second box and typedef as last name in third box.
8. Select DOB as March 07 2022.
9. Select Sex as **Male**.
10.Scroll down and click on **Create New Patient**.
11. Note: Sometimes you'll not be able to see Create New Patient in the firefox browser. Click on the green icon on the bottom right corner to hide history details.
12. In ZAP, there is a tab towards the left hand side. Under Sites, expand http://localhost, openemr, interface, new
13.Under new directory, you can find a POST:new_comprehensive_save.php(csrf token....)
14.The URL in request will be
http://localhost/openemr/interface/new/new_comprehensive_save.php
15.Right click on POST:new_comprehensive_save.php file and select Break, and click on save.
16.Go back to the firefox browser and repeat steps 5,6,7,8,9,10.
17.When you click on Create New Patient in step 10, you'll be able to see a HTTP request dialogue box.
18.Come back to the ZAP application to find that the application stopped at the break point. 19.In the dialogue box below (Break tab), you'll be able to see the **form_DOB** in the payload. Update form_DOB=2022-03-07 to **form_DOB=' or 1=1–**
20.Click on the play icon to forward through on top of the ZAP application.
21.Go back to firefox browser to see the HTTP response dialogue box.
22.Click on continue in the firefox browser HTTP window.
23.If any API error window pops up, close the error window.
24.On the top headers, hover over **Patient/Client** and click on **Patients** to see the new Patient details created just now.

Expected Results: The new patient should not be created.

CWE: 89, Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')

4. Test Case id: 4.1.1-1

Description: Verify that the application enforces access control rules on a trusted service layer, especially if client-side access control is present and could be bypassed.

Repeatable steps:
1. Open ZAP in vcl.
2. Clock on Manual Explore.
3. Enter the openemr URL “http://localhost/openemr” in the URL dialogue box, and then click on launch browser with Firefox as the default browser.
4. log in as administrator in firefox browser with username: admin_openemr and password: admin_openemr
5. On the top headers, hover over Patient/Client and click on Patients.
6. Under Patient finder, click on Add New Patient.
7. Create a new patient. Select Mr. from pronouns drop down. Type Abc as the first name in the first box, leave out the second box, and type def as the last name in the third box.
8. Select DOB as March 07, 2022.
9. Select Sex as Male.
10. Scroll down and click on Create New Patient.
11. Note: Sometimes you’ll not be able to see Create New Patient in the firefox browser. Click on the green icon on the bottom right corner to hide history details.
12. On the top headers click on Messages.
13. Under Messages, Reminders, Recalls click on Add New.
14. Select Chart Note under Type.
15. Click on the Patient box and select Def,abc from the Patient name drop down.
16. Click on Add to list, and click on ok.
17. Under To: select Administrator, Administrator
18. Type a message as hello in the message field.
19. Click on send message.
20. In ZAP, there is a tab towards the left-hand side. Under Sites, expand http://localhost, openemr, interface, main, messages.
21. Under the orders directory, you can find a POST:messages.php(begin, form_active, showall, sort by, sort order)
22. The URL in request will be http://localhost/openemr/interface/main/messages/messages.php?showall=no&sortby=&sortorder=&begin=&form_active=1
23. Right-click on POST:messages.php file and select Break, and click on save.
24. Go back to the firefox browser and repeat steps 12,13,14,15,16,17,18,19.
25. When you click on send message in step 19, you’ll be able to see a HTTP request dialogue box.

26. Come back to the ZAP application to find that the application stopped at the break point.
27. In the dialogue box below (Break tab), you'll be able to see the form_note_type in the payload. Update form_note_type=Chart+Note to form_note_type=<script>alert('xss')</script>
28. Click on the play icon to forward through on top of the ZAP application.
29. Go back to the firefox browser to see the HTTP response dialogue box.
30. Click on continue in the firefox browser HTTP window.
31. If any API error window pops up, close the error window.
32. Close the messages dialog box and on top of the web page click on messages again to refresh the messages window.

Expected Results: Since the new message form_note_type contains malicious or invalid characters, the new message should not be stored and sent.

CWE: 602, Client-Side Enforcement of Server-Side Security

5. **Test Case id:** 5.1.3-1

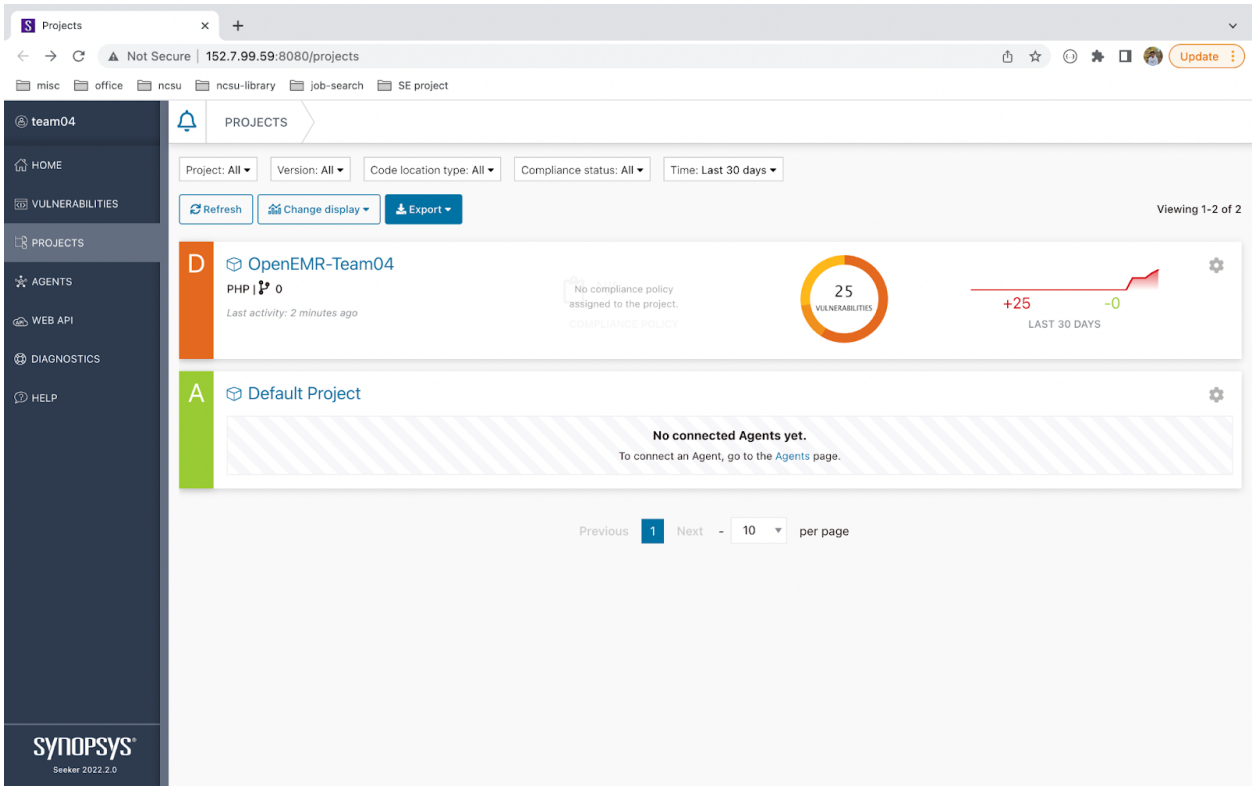
Description: Verify that all input (HTML form fields, REST requests, URL parameters, HTTP headers, cookies, batch files, RSS feeds, etc) is validated using positive validation (allow lists).

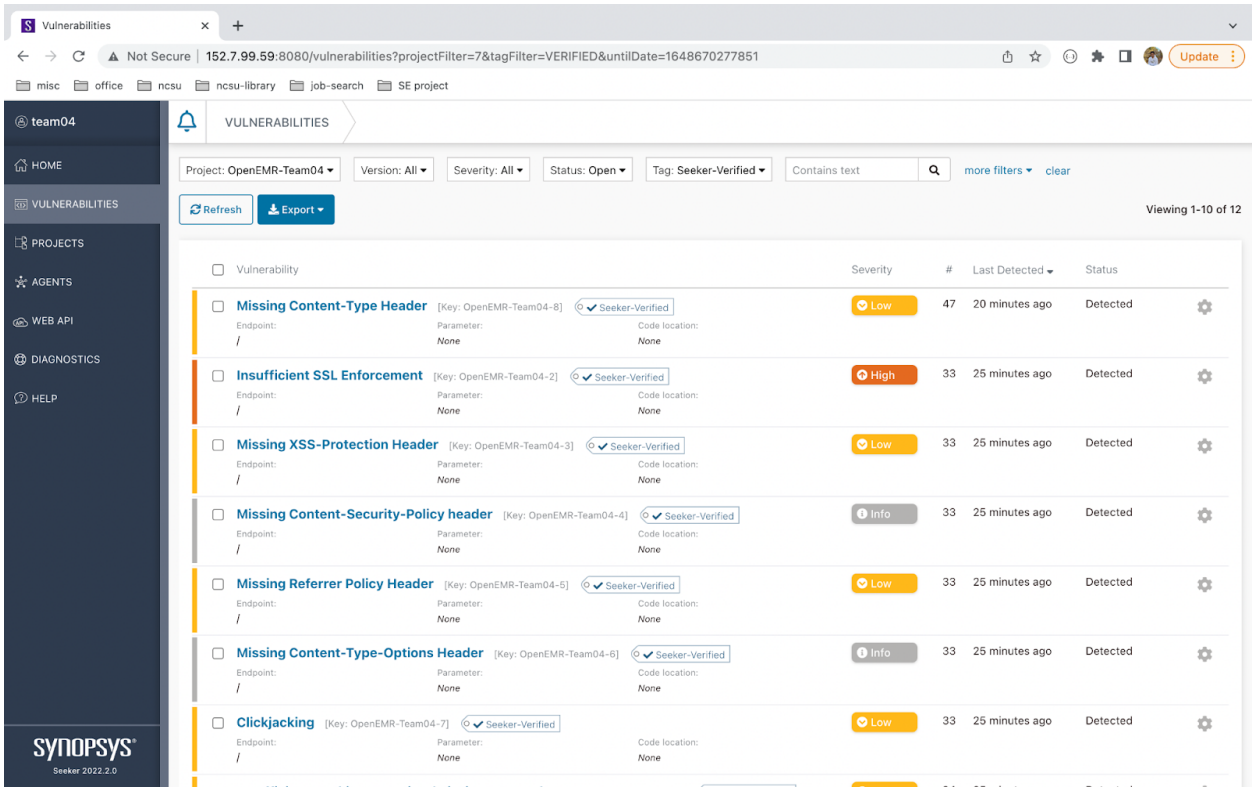
Repeatable Steps:
1. Open ZAP in vcl.
2. Clock on Manual Explore.
3. Enter the openemr URL "http://localhost/openemr" in the URL dialogue box, and then click on launch browser with Firefox as the default browser.
4. log in as administrator in firefox browser with username: admin_openemr and password: admin_openemr
5. On the top headers, hover over Patient/Client and click on Patients.
6. Under Patient finder, click on Add New Patients.
7. Create a new patient. Select Mr. from pronouns drop down. Type test as the first name in the first box, leave out the second box, and type test as the last name in the third box
8. Select DOB as June 5th, 1975.
9. Select Sex as Male.
10. Scroll down and click on Create New Patient.
11. Close the success patient created dialogue box.
12. Click on patient finder and the patient we just created ,
(test, test)
13. On bottom right, Under Recurrent Appointments, click on Edit Allergies

14. Click on Add allergies, select penicillin as the title and select the begin date as 2022-03-08 and Reaction as Nausea click on save allergy
15. Note: Sometimes you'll not be able to see Create New allergy for a patient in firefox browser. Click on the green icon on the bottom right corner to hide history details.
16. In ZAP, there is a tab towards the left hand side. Under Sites, expand http://localhost, openemr, interface, patient_file, summary
17. Under this directory you can find POST:add_edit_issue.php entry, right click that and click on break and save it.
18. Under Request tab, the URL will be http://localhost/openemr/interface/patient_file/summary/add_edit_issue.php?issue=0&thistype=allergy
19. Repeat the steps from 12-14 after click on save allergy, we can change the actual input (In break tab) for the following input form_reaction in the response tab to series of character 'a' of length 1500 (Buffer Overflow)
20. Click on the play icon to forward through on top of the ZAP application.
21. Go back to the firefox browser to see the HTTP response dialogue box.
22. Click on continue in the firefox browser HTTP window.
23. On the patient Issues tab, we can see the list of allergies, in the allergy we created we can check that Reaction is a series of character 'a' which is not part of our options in the select element.

Expected Results: The Allergy for the corresponding patient **test**, should **not** be created.

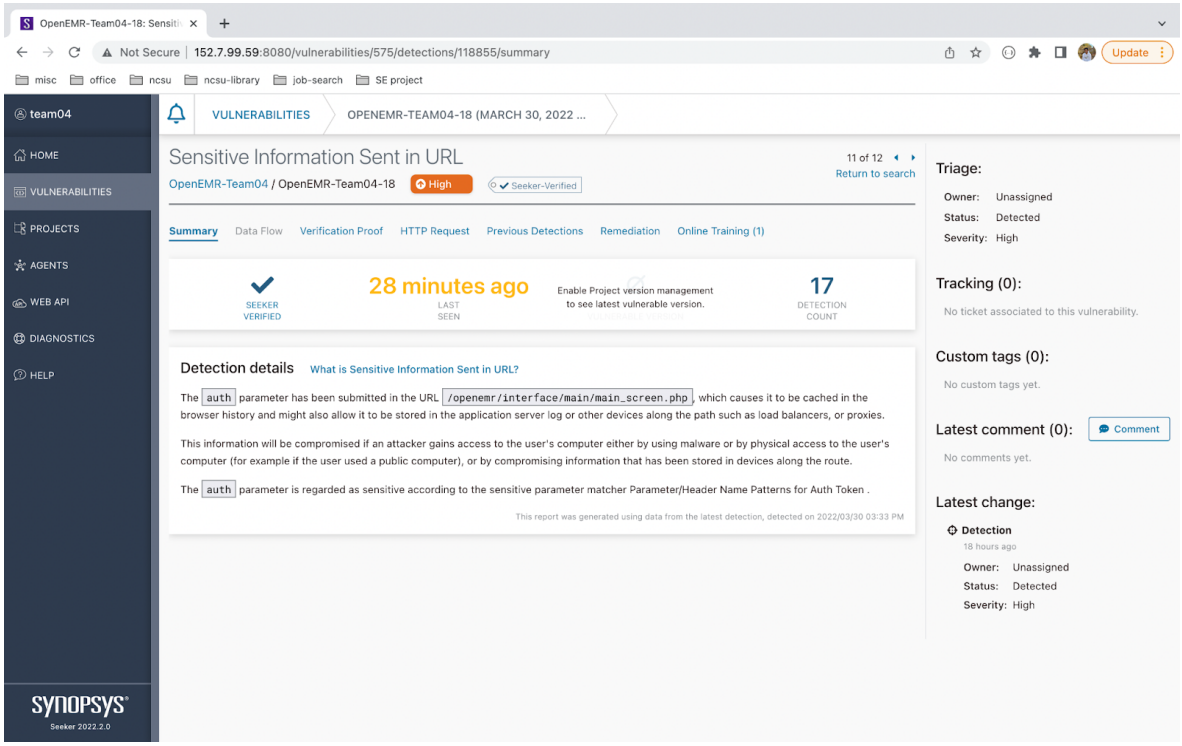
CWE: 20, Improper Input Validation


Projects
Not Secure | 152.7.99.59:8080/projects
Update
misc
office
ncsu
ncsu-library
job-search
SE project
team04
HOME
VULNERABILITIES
PROJECTS
AGENTS
WEB API
DIAGNOSTICS
HELP
PROJECTS
Project: All
Version: All
Code location type: All
Compliance status: All
Time: Last 30 days
Refresh
Change display
Export
Viewing 1-2 of 2
D
OpenEMR-Team04
PHP | 0
Last activity: 2 minutes ago
No compliance policy assigned to the project.
COMPLIANCE POLICY
25
VULNERABILITIES
+25
-0
LAST 30 DAYS
A
Default Project
No connected Agents yet.
To connect an Agent, go to the Agents page.
Previous
1
Next
10
per page
SYNOPSYS
Seeker 2022.2.0

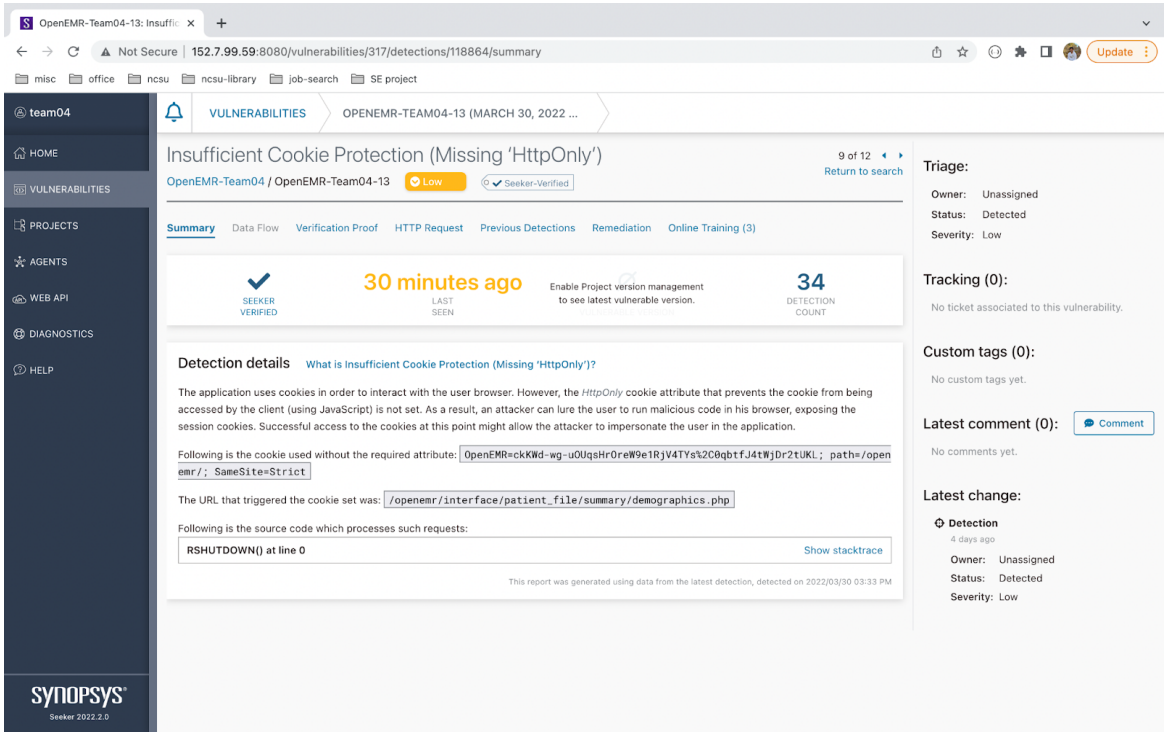

Vulnerabilities
Not Secure | 152.7.99.59:8080/vulnerabilities?projectFilter=7&tagFilter=VERIFIED&untilDate=1648670277851
Update
misc
office
ncsu
ncsu-library
job-search
SE project
team04
HOME
VULNERABILITIES
PROJECTS
AGENTS
WEB API
DIAGNOSTICS
HELP
VULNERABILITIES
Project: OpenEMR-Team04
Version: All
Severity: All
Status: Open
Tag: Seeker-Verified
Contains text
more filters
clear
Refresh
Export
Viewing 1-10 of 12
Vulnerability
Severity
#
Last Detected
Status
Missing Content-Type Header [Key: OpenEMR-Team04-8] Seeker-Verified
Low 47 20 minutes ago Detected
Endpoint: / Parameter: None Code location: None
Insufficient SSL Enforcement [Key: OpenEMR-Team04-2] Seeker-Verified
High 33 25 minutes ago Detected
Endpoint: / Parameter: None Code location: None
Missing XSS-Protection Header [Key: OpenEMR-Team04-3] Seeker-Verified
Low 33 25 minutes ago Detected
Endpoint: / Parameter: None Code location: None
Missing Content-Security-Policy header [Key: OpenEMR-Team04-4] Seeker-Verified
Info 33 25 minutes ago Detected
Endpoint: / Parameter: None Code location: None
Missing Referrer Policy Header [Key: OpenEMR-Team04-5] Seeker-Verified
Low 33 25 minutes ago Detected
Endpoint: / Parameter: None Code location: None
Missing Content-Type-Options Header [Key: OpenEMR-Team04-6] Seeker-Verified
Info 33 25 minutes ago Detected
Endpoint: / Parameter: None Code location: None
Clickjacking [Key: OpenEMR-Team04-7] Seeker-Verified
Low 33 25 minutes ago Detected
Endpoint: / Parameter: None Code location: None
SYNOPSYS
Seeker 2022.2.0

Screenshots of Seeker information on the five vulnerabilities:

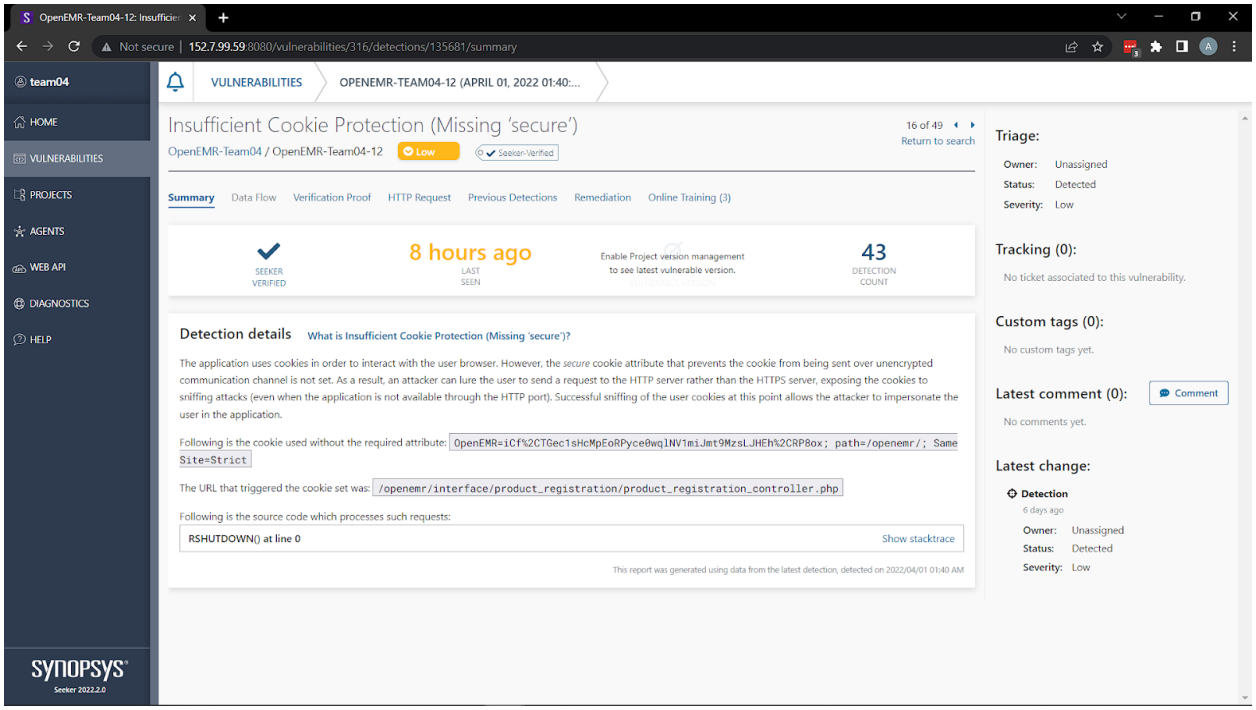
1. Sensitive Information Sent in URL



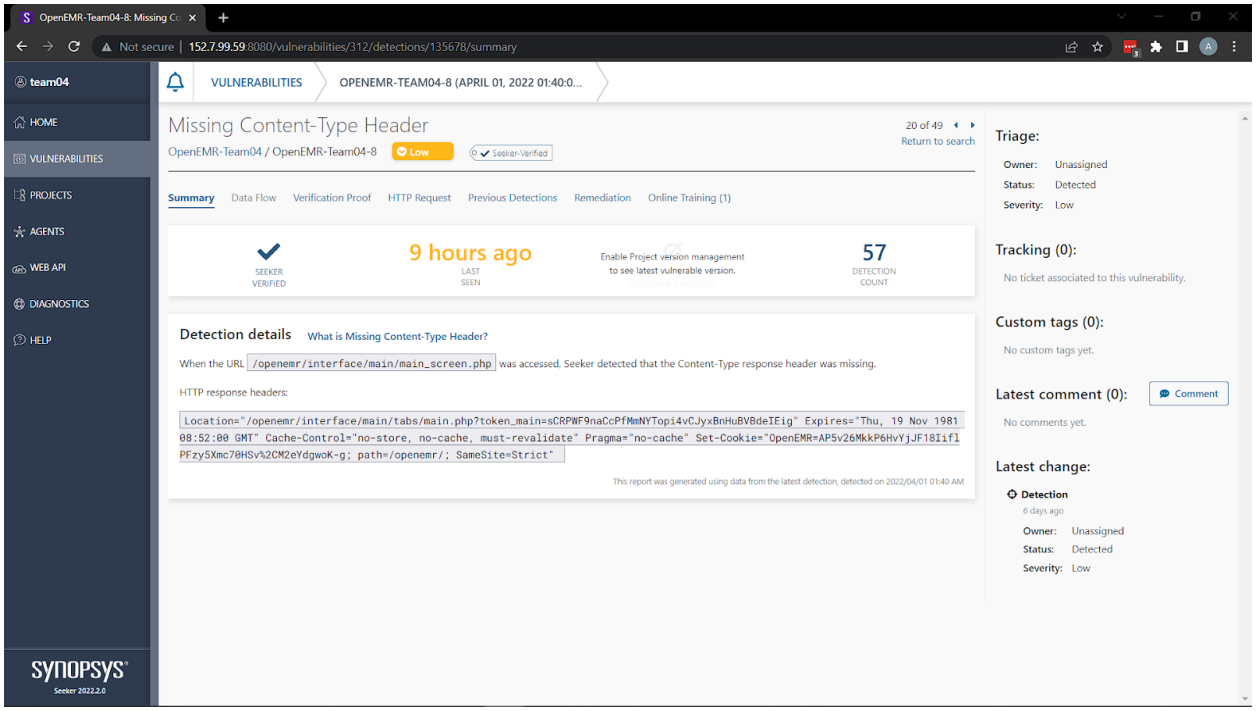
2. Insufficient Cookie Protection (Missing ‘HttpOnly’)



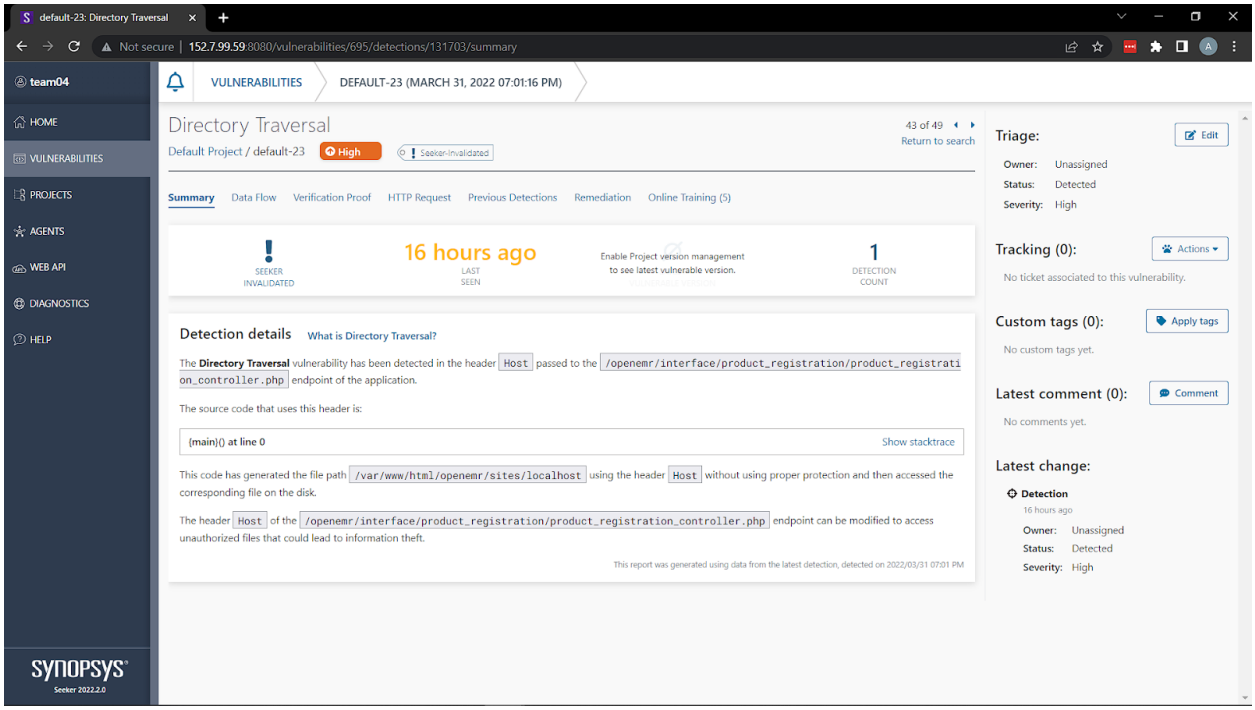
3. Insufficient Cookie Protection (Missing ‘secure’)



4. Missing Content-Type Header



5. Directory Traversal



Black-box test cases detected by Seeker:

1. Test Case ID: 1.5.1-1

Description: Verify that input and output requirements clearly define how to handle and process data based on type, content, and applicable laws, regulations, and other policy compliance.

Repeatable Steps:

1. Open openemr application in vcl, or navigate to http://localhost/openemr url in the firefox browser in vcl.
2. Open the developer tools. (Right click anywhere on the openemr webpage and select inspect).
3. Monitor the network tab in developer tools.
4. Now login as administrator. (username: admin_openemr, password: admin_openemr).
5. Take a look at the main_screen.php request in network tab,

Expected Results: The auth parameter which is part of the url should not be exposed. The auth parameter has been submitted in the URL /openemr/interface/main/main_screen.php, which causes it to be cached in the browser history and might also allow it to be stored in the application server log or other devices along the path such as load balancers, or proxies. This information will be compromised if an attacker gains access to the user's computer either by using malware or by physical access to the user's computer (for example if the user used a public computer), or by compromising information that has been stored in devices along the route.

CWE: 1029, Sensitive Data Exposure

2. Test Case ID: 3.4.2-1
Description: Verify that cookie-based session tokens have the 'HttpOnly' attribute set.
Repeatable Steps:
1. Open the openemr application in vcl, or navigate to http://localhost/openemr url in the firefox browser in vcl.
2. Open the developer tools. (Right click anywhere on the openemr webpage and select inspect).
3. Monitor the network tab in developer tools.
4. Now login as administrator. (username: admin_openemr, password: admin_openemr).
5. In the network tab navigate to the main_screen.php request, and select Cookies from the provided tabs.
6. Take a look at the HttpOnly column in Response cookies.

Expected Results: The HttpOnly should be set. If the HttpOnly is not set the an attacker can lure the user to run malicious code in his browser, exposing the session cookies. Successful access to the cookies at this point might allow the attacker to impersonate the user in the application.

CWE: 1004, Sensitive Cookie Without 'HttpOnly' Flag

3. Test Case ID: 3.4.1-1
Description: Verify that cookie-based session tokens have the 'Secure' attribute set.
Repeatable Steps:
1. Open the openemr application in vcl, or navigate to http://localhost/openemr url in the firefox browser in vcl.
2. Open the developer tools. (Right click anywhere on the openemr webpage and select inspect).
3. Monitor the network tab in developer tools.
4. Now login as administrator. (username: admin_openemr, password: admin_openemr).
5. In the network tab navigate to the main_screen.php request, and select Cookies from the provided tabs.
6. Take a look at the Secure column in Response cookies.

Expected Results: The Secure attribute should be set. If the Secure attribute is not set, an attacker can lure the user to send a request to the HTTP server rather than the HTTPS server, exposing the cookies to sniffing attacks (even when the application is not available through the HTTP port). Successful sniffing of the user cookies at this point allows the attacker to impersonate the user in the application.

CWE: 614, Sensitive Cookie in HTTPS Session Without 'Secure' Attribute

4. Test Case ID: 13.2.5-1
Description: Verify that REST services explicitly check the incoming Content-Type to be the expected one, such as application/xml or application/json.
Repeatable Steps:
1. Open the openemr application in vcl, or navigate to http://localhost/openemr url in the firefox browser in vcl.
2. Open the developer tools. (Right click anywhere on the openemr webpage and select inspect).
3. Monitor the network tab in developer tools.
4. Now login as administrator. (username: admin_openemr, password: admin_openemr).
5. In the network tab navigate to the main_screen.php request, and select Headers from the provided tabs.
6. Scroll down to the Request Headers section and check for a Content-Type header

Expected Results: The 'Content-Type' header should be sent with the request so it can be checked by the relevant REST service.

CWE: 436, Interpretation Conflict

5. Test Case ID: 4.3.2-2
Description: Verify that directory browsing is disabled unless deliberately desired. Additionally, applications should not allow discovery or disclosure of file or directory metadata, such as Thumbs.db, .DS_Store, .git or .svn folders.
Repeatable Steps:
1. Open the openemr application in vcl, or navigate to http://localhost/openemr url in the firefox browser in vcl.
2. The login page will look like http://localhost/openemr/interface/login/login.php?site=default .
3. Edit the URL to http://localhost/openemr/interface/product_registration/

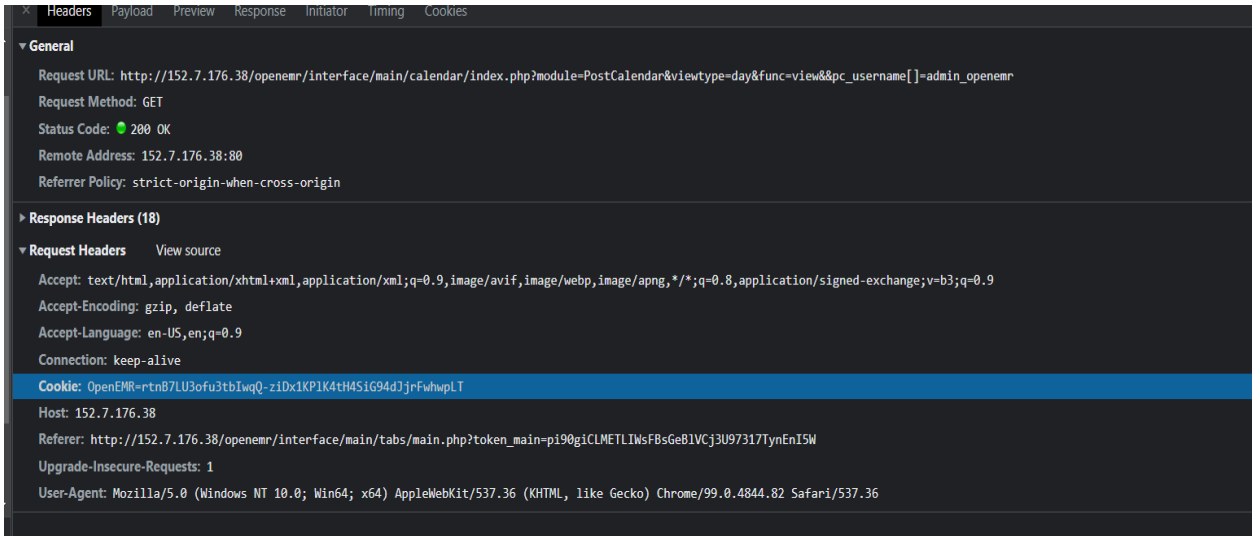
Expected Results: No one should have access to look at the files in the directory.

CWE: 548, Exposure of Information Through Directory Listing

Time Spent: 4 hours
True Positives: 5
True Positives per hour: 1.25

WebInspect

1. Test Case ID: 10.2.1-1
Description: Verify that the application source code and third-party libraries do not contain unauthorized phone home or data collection capabilities. Where such functionality exists, obtain the user's permission for it to operate before collecting any data.
Repeatable Step:
1. Login as any user (Eg: username: admin_openemr password: admin_openemr)
2. Open Developer tools to inspect the network communication within the openemr application.
3. Click on the calendar on the top left of the navigation bar.
4. The following GET API is called with username information as shown below,
http://152.7.176.38/openemr/interface/main/calendar/index.php?module=PostCalendar&viewtype=day&func=view&&pc_username[]=admin_openemr



Expected Results:
- The user name, phone number, or any other sensitive information should not be visible to the application code or any third-party libraries.

CWE Info: 359: Privacy Violation

Privacy Violation: HTTP GET (10965) View Description

CWE: 359

Kingdom: Security Features

Page: http://152.7.177.250:80/openemr/interface/main/calendar/index.php?module=PostCalendar&func=view&tplview=default&viewtype=day&Date=20220327&pc_username[0]=admin_openemr&pc_category=&pc_topic=

Request:

```
GET /openemr/interface/main/calendar/index.php?
module=PostCalendar&func=view&tplview=default&viewtype=day&Date=20220327&pc_
username[0]=admin_openemr&pc_category=&pc_topic= HTTP/1.1
Referer: http://...TRUNCATED...
```

Response:

```
HTTP/1.1 200 OK
Date: Tue, 29 Mar 2022 06:50:27 GMT
Server: Apache/2.4.29 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
```

2. Test Case ID: 4.3.2-1

Description: Verify that directory browsing is disabled unless deliberately desired. Additionally, applications should not allow discovery or disclosure of file or directory metadata, such as Thumbs.db, .DS_Store, .git or .svn folders.

Repeatable Step:

1. Login as any user (Eg: username: admin_openemr password: admin_openemr)
2. Open the following URL to open the directory listing,

   http://ip_adress/openemr/sites/default/

Index of /openemr/sites/default

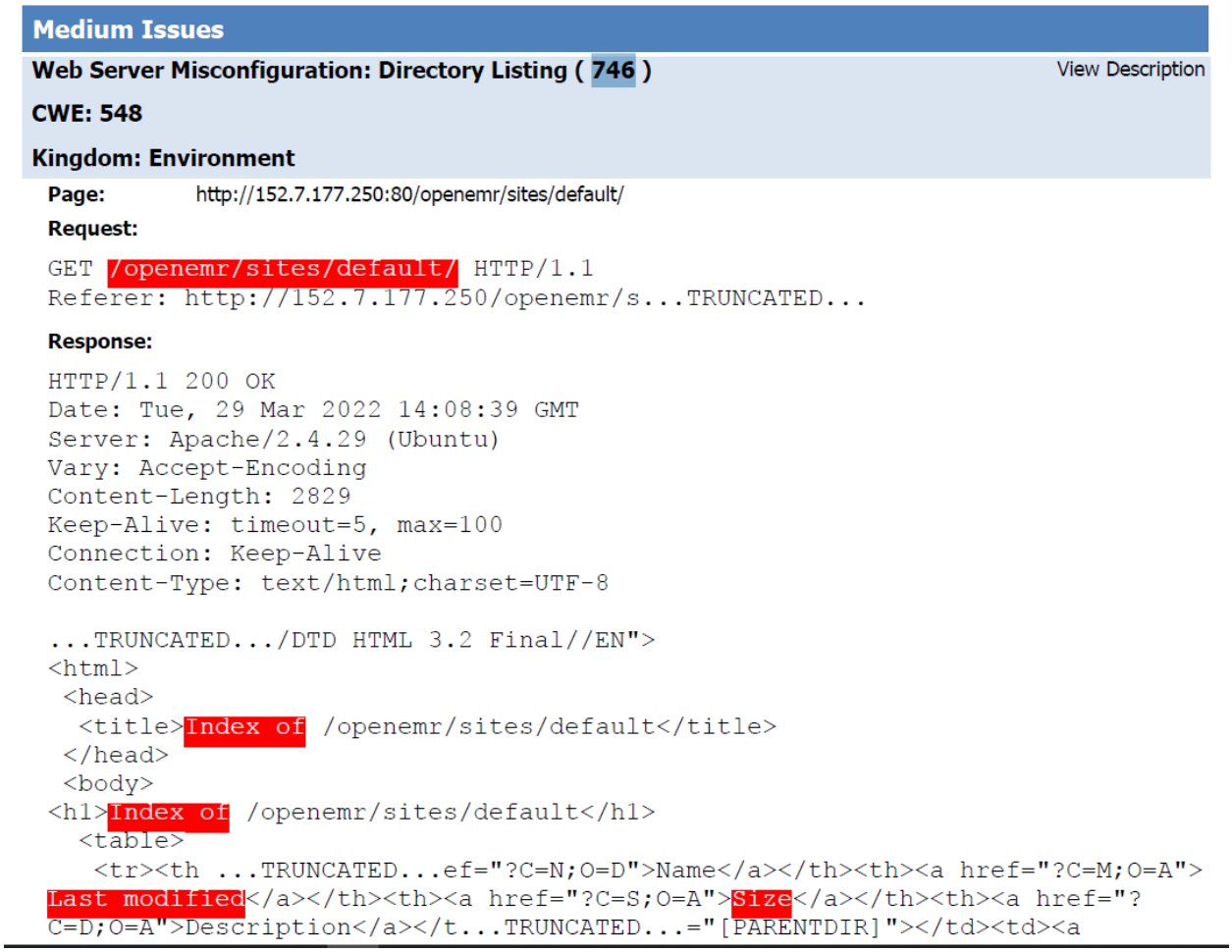
Name	Last modified	Size	Description
Parent Directory		-	
LBF/	2021-01-05 22:53	-	
clickoptions.txt	2021-01-05 22:53	2.2K	
config.php	2021-01-05 22:53	4.7K	
docker-version	2021-01-05 22:53	2	
faxcover.txt	2021-01-05 22:53	392	
faxtitle.eps	2021-01-05 22:53	11K	
images/	2021-01-05 22:53	-	
referral_template.html	2021-01-05 22:53	13K	
sqlconf.php	2022-01-31 15:24	651	
statement.inc.php	2021-01-05 22:53	49K	

Apache/2.4.29 (Ubuntu) Server at 152.7.177.250 Port 80

Expected Results:

- The directory listing should be disabled, the webserver should return an appropriate error response.

CWE Info: 548: Exposure of Information Through Directory Listing



Medium Issues

Web Server Misconfiguration: Directory Listing (746) View Description

CWE: 548

Kingdom: Environment

Page: http://152.7.177.250:80/openemr/sites/default/

Request:

```
GET /openemr/sites/default/ HTTP/1.1
Referer: http://152.7.177.250/openemr/s...TRUNCATED...
```

Response:

```
HTTP/1.1 200 OK
Date: Tue, 29 Mar 2022 14:08:39 GMT
Server: Apache/2.4.29 (Ubuntu)
Vary: Accept-Encoding
Content-Length: 2829
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html;charset=UTF-8

...TRUNCATED.../DTD HTML 3.2 Final//EN">
<html>
 <head>
  <title>Index of /openemr/sites/default</title>
 </head>
 <body>
<h1>Index of /openemr/sites/default</h1>
  <table>
   <tr><th ...TRUNCATED...ef="?C=N;O=D">Name</a></th><th><a href="?C=M;O=A">
Last modified</a></th><th><a href="?C=S;O=A">Size</a></th><th><a href="?
C=D;O=A">Description</a></t...TRUNCATED...="[PARENTDIR]"></td><td><a
```

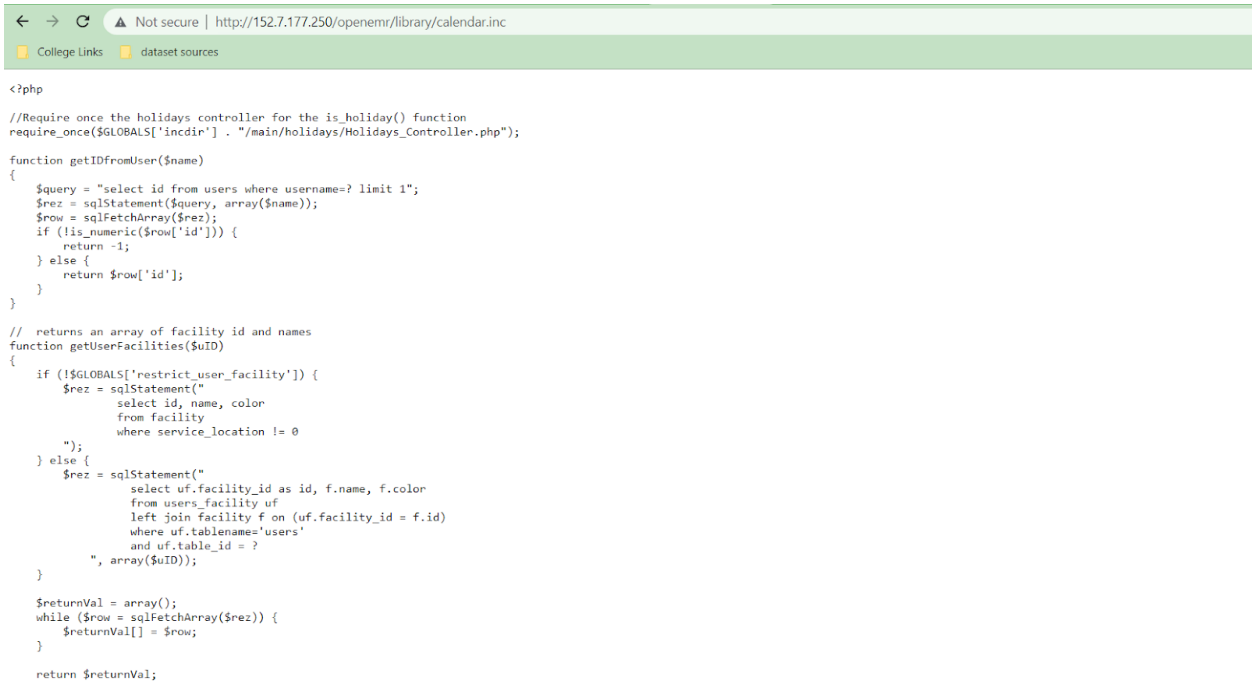
3. Test Case ID: 12.5.1-1

Description: Verify that the web tier is configured to serve only files with specific file extensions to prevent unintentional information and source code leakage. For example, backup files (e.g. .bak), temporary working files (e.g. .swp), compressed files (.zip, .tar.gz, etc) and other extensions commonly used by editors should be blocked unless required.

Repeatable Step:

1. Login as any user (Eg: username: admin_openemr password: admin_openemr)
2. Open the following URL to download the inc file,

   http://Ip_address/openemr/library/calendar.inc



Expected Results:

- The temporary files, backup files, and other config files should not be accessible. The webserver should return an appropriate error response.

CWE Info: 540: Inclusion of Sensitive Information in Source Code

High Issues

Web Server Misconfiguration: Unprotected File (1384) View Description

CWE: 540

Kingdom: Environment

Page: http://152.7.177.250:80/openemr/library/calendar.inc

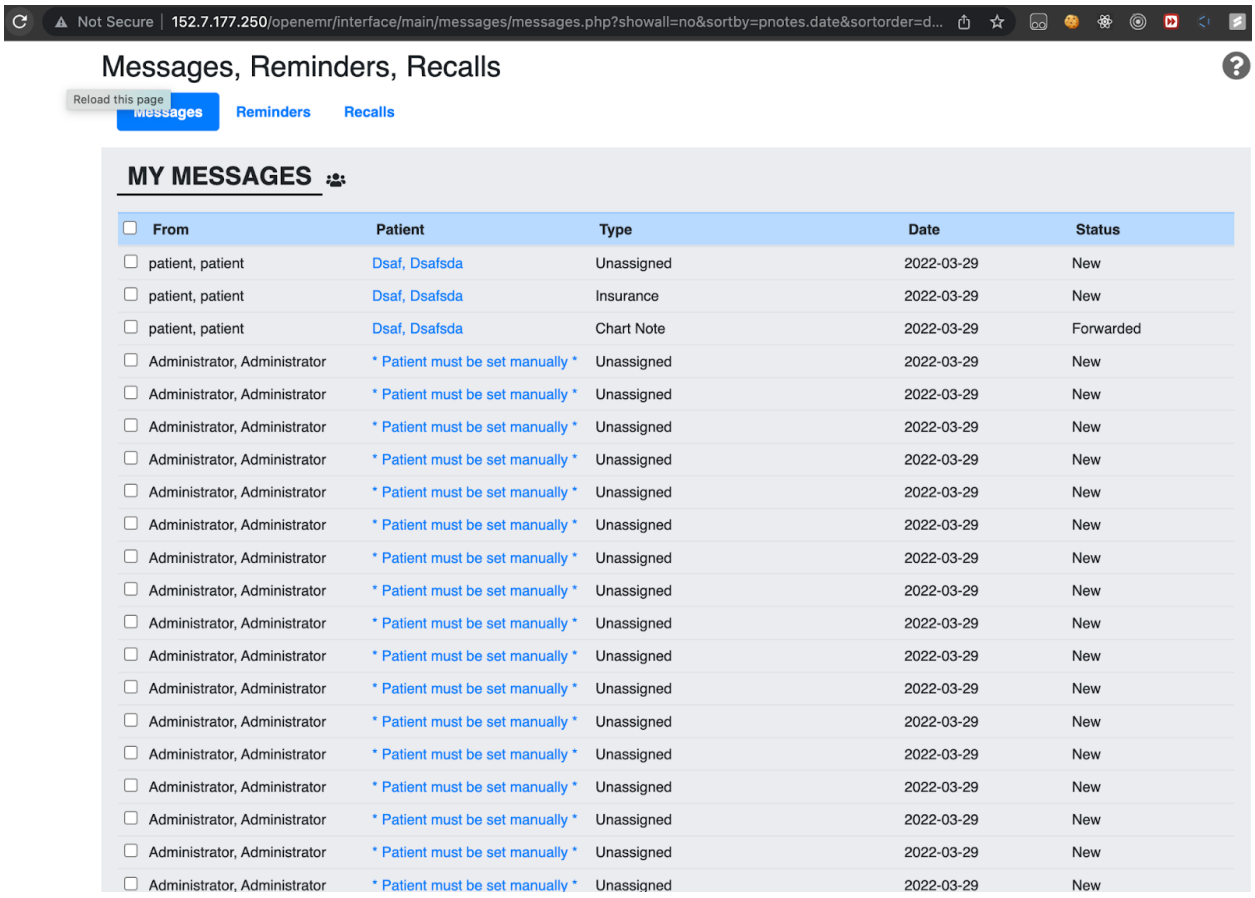
Request:

```
GET /openemr/library/calendar.inc HTTP/1.1
Referer: http://152.7.177.250/openemr/library/
Accept: */*
Pragma: no-cache
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:83.0)
Gecko/20100101 Firefox/83.0
Host: 152.7.177.250
Connection: Keep-Alive
X-WIPP: AscVersion=21.2.0.103
X-Scan-Memo:
Category="Crawl";SID="C15BAB9763C96058141F08C4B8ED8B98";PSID="2CC16286094BC8
52B6992B9FE7109237";SessionType="Crawl";CrawlType="HTML";AttackType="None";O
riginatingEngineID="00000000-0000-0000-0000-
000000000000";AttributeName="href";Format="NonRooted";LinkKind="HyperLink";L
ocations="HtmlNode";Source="ScriptExecution";tht="31";
```

4. Test Case ID: 5.3.5-1
Description: Verify that where parameterized or safer mechanisms are not present, context-specific output encoding is used to protect against injection attacks, such as the use of SQL escaping to protect against SQL injection.
Repeatable Step:
1. Login to the openemr application using credentials as (Eg: username: admin_openemr password: admin_openemr)
2. Now open a new tab and paste the following URL.

http://ip_address/openemr/interface/main/messages/messages.php?showall=no&sortby=pnotes.date&sortorder=desc&begin=0&task=1%27;%20return%20Boolean(%27false%27);%20%270&noteid=483&

3. Observe the URL it contains a section where the code is being injected and the output response is being rendered on the frontend.



Expected Results:

- The attacker should not be able to inject code in the URL or any of the GET and POST requests from the browser.

CWE Info: 89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')

NoSQL Injection: MongoDB (11687) View Description

CWE: 89

Kingdom: Input Validation and Representation

Page: http://152.7.177.250:80/openemr/interface/main/messages/messages.php?showall=no&sortby=pnotes.date&sortorder=desc&begin=0&task=1';%20return%20Boolean('false');%20'0&noteid=483&

Request:

```
GET /openemr/interface/main/messages/messages.php?
showall=no&sortby=pnotes.date&sortorder=desc&begin=0&task=
1';%20return%20Boolean('false');%20'0&noteid=483& HTTP/1.1
Referer: http://152.7.177.2...TRUNCATED...
```

Response:

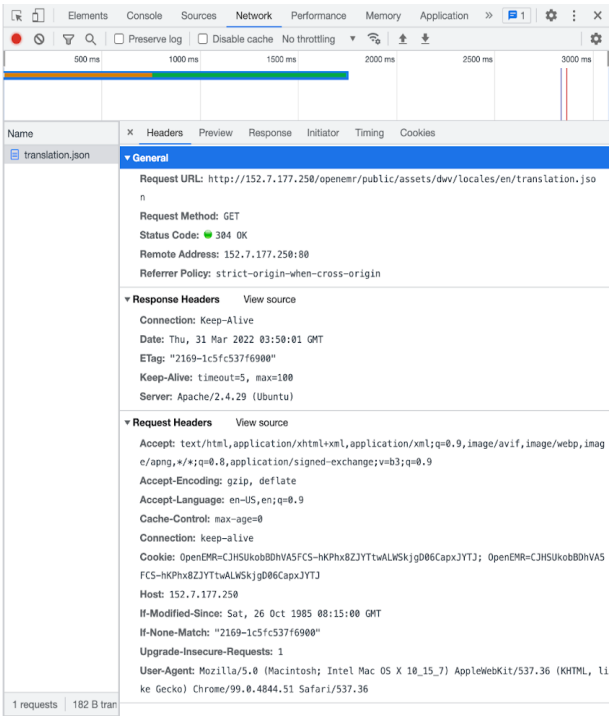
```
HTTP/1.1 200 OK
Date: Tue, 29 Mar 2022 14:06:51 GMT
Server: Apache/2.4.29 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache...TRUNCATED...
```

5. Test Case ID: 14.5.3-1
Description: Verify that the Cross-Origin Resource Sharing (CORS) Access-Control-Allow Origin header uses a strict allow-list of trusted domains and subdomains to match against and does not support the "null" origin.
Repeatable Step:
1. Login to the openemr application using credentials as (Eg: username: admin_openemr password: admin_openemr)
2. Now open a new tab and paste the following URL with network tab open to see the request headers.

http://152.7.177.250:80/openemr/public/assets/dwv/locales/en/translation.json

3. The Request Headers does not contain an “Origin” key i.e the request is done with “null” origin. And as a result of this request, you should be able to view the content which is not intended to be visible in the frontend.

```
{
    "basics": {
        "open": "Open",
        "close": "Close",
        "help": "Help",
        "back": "Back",
        "reset": "Reset",
        "apply": "Apply",
        "name": "Name",
        "value": "Value",
        "dicomTags": "DICOM Tags",
        "columns": "Columns",
        "group": "Group",
        "element": "Element",
        "vr": "VR",
        "vl": "VL",
        "presets": "Presets",
        "toolbox": "Toolbox",
        "history": "History",
        "image": "Image",
        "info": "Info",
        "downloadState": "Download state",
        "drawList": "Annotations",
        "search": "Search",
        "id": "ID",
        "slice": "Slice",
        "frame": "Frame",
        "type": "Type",
        "color": "Color",
        "label": "Label",
        "description": "Description",
        "editMode": "Edit Mode",
        "deleteDraws": "Delete All",
        "visible": "Visible"
    },
    "colour": {
        "Yellow": {
            "name": "Yellow"
        },
        "Red": {
            "name": "Red"
        },
        "White": {
            "name": "White"
        },
        "Green": {
            "name": "Green"
        },
        "Blue": {
            "name": "Blue"
        },
        "Lime": {
            "name": "Lime"
        },
```

Expected Results:

- The attacker should not be able to view the translation json as the “Origin” attribute in the Request Headers is not specified or “null”.

CWE Info: 346: Origin Validation Error.

JavaScript Hijacking: JSONP (10731) View Description

CWE: 346

Kingdom: Encapsulation

Page: http://152.7.177.250:80/openemr/public/assets/dwv/locales/en/translation.json

Request:

```
GET /openemr/public/assets/dwv/locales/en/translation.json HTTP/1.1
Referer: http://152.7.177.250/openemr/library/dicom_frame.php
Host: 152.7.177.250
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-Requested-With: XMLHttpRequest
Pragma: no-cache
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:83.0)
Gecko/20100101 Firefox/83.0
Connection: Keep-Alive
X-WIPP: AscVersion=21.2.0.103
X-Scan-Memo:
ScriptEngine="Gecko";Category="Crawl";SID="E9473FFDA2A4917F4EF06C71862BD95B"
;PSID="727A138E6889B08326B6F8FB32F971C6";SessionType="Crawl";CrawlType="AJAX
Include";AttackType="None";OriginatingEngineID="00000000-0000-0000-0000-
000000000000";tht="21";
X-RequestManager-Memo: stid="54";stmi="0";sc="1";rid="2705852c";
X-Request-Memo: rid="ee135759";sc="2";thid="229";
Cookie:
CustomCookie=WebInspect175190ZX2CB257C069934F78807027414B0AAD7DYDB97;OpenEMR
=cxbYp2Ktr498IDoUldjOwsaOorHGSs-YA%2C%2CN9-z1Yidy2hw3
```

Response:

```
HTTP/1.1 200 OK
Date: Tue, 29 Mar 2022 08:05:35 GMT
Server: Apache/2.4.29 (Ubuntu)
Last-Modified: Sat, 26 Oct 1985 08:15:00 GMT
ETag: "2169-1c5fc537f6900"
Accept-Ranges: bytes
Content-Length: 8553
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: application/json
```

Time taken and true positives per hour:

Time Spent (Running the test cases and Going through webinspect): 5.5 hours

True Positives: 5

True Positives per hour: 1.1

Test Coverage

Test Coverage Report:

ASVS Section	Control	Test Covered	Coverage
V1	V1.5	1.5.1	**1/14 => 7%**
V2	V2.1	2.1.2, 2.1.3, 2.1.5, 2.1.6	**1/10 => 10%**
V3	V3.1	3.1.1	**4/7 => 57%**
	V3.2	3.2.1	
	V3.3	3.3.1	
	V3.4	3.4.1, 3.4.2	
V4	V4.1	4.1.1	**2/3 => 66%**
	V4.3	4.3.1, 4.3.2	
V5	V5.1	5.1.3	**4/5 => 80%**
	V5.2	5.2.1, 5.2.2, 5.2.3	
	V5.3	5.3.2, 5.3.4, 5.3.5	
	V5.4	5.4.2	
V6	-	-	**0/4 => 0%**
V7	V7.1	7.1.1, 7.1.3, 7.1.4	**3/4 => 75%**
	V7.2	7.2.1, 7.2.2	
	V7.3	7.3.4	
V8	-	-	**0/3 => 0%**
V9	-	-	**0/2 => 0%**
V10	V10.2	10.2.1, 10.2.2	**1/3 => 33%**
V11	-	-	**0/1 => %0**
V12	V12.1	12.1.1	**3/6 => 50%**
	V12.3	12.3.1	

	V12.5	12.5.1	
V13	V13.2	13.2.5	**1/4 => 25%**
V14	V14.5	14.5.3	**1/5 => 20%**
Total Coverage:			**21/71 => 29.58%**

Black box test cases:

1. Test Case ID: 1.2.2-1
Description: Verify that communications between application components, including APIs, middleware, and data layers, are authenticated. Components should have the least necessary privileges needed.
Repeatable Step:
1. Open the openemr application in vcl.
2. Login as any user (Eg: username: admin_openemr password: admin_openemr)
3. Open the following URL,
http://localhost/openemr/library/admin/
4. This in turn calls a /admin GET endpoint which is returning a 200 status response.
Expected Results:
- Access to privileged information such as remote site administration should be further authenticated before allowing access.

CWE Info: 306: Missing Authentication for Critical Function

2. Test Case ID: 2.3.2-1
Description: Verify that enrollment and use of user-provided authentication devices are supported, such as a U2F or FIDO tokens.
Repeatable Step:
1. Login as any user (Eg: username: admin_openemr password: admin_openemr)
2. Hover over name in top right corner (Eg: Administrator Administrator) and select MFA Management
3. Click the dropdown under Select/Add New Authentication Method for Administrator Administrator and see options

Expected Results:
- U2F is an option

CWE Info: 308: Use of Single-factor Authentication

3. Test Case ID: 2.2.1-1
Description: Verify that anti-automation controls are effective at mitigating breached credential testing, brute force, and account lockout attacks. Such controls include blocking the most common breached passwords, soft lockouts, rate limiting, CAPTCHA,

ever increasing delays between attempts, IP address restrictions, or risk-based restrictions such as location, first login on a device, recent attempts to unlock the account, or similar. Verify that no more than 100 failed attempts per hour is possible on a single account.

Repeatable Steps:

1. Open openemr application in vcl or navigate to firefox in vcl and open http://locahost/openemr.
2. Try login 102 times as admin with the wrong password. (username: admin_openemr, password: abcd).
3. After trying 102 times, try now with the correct password. (username: admin_openemr, password: admin_openemr).

Expected Results: More than 100 failed attempts per hour should not be possible on a single account.

CWE: 307 - Improper Restriction of Excessive Authentication Attempts

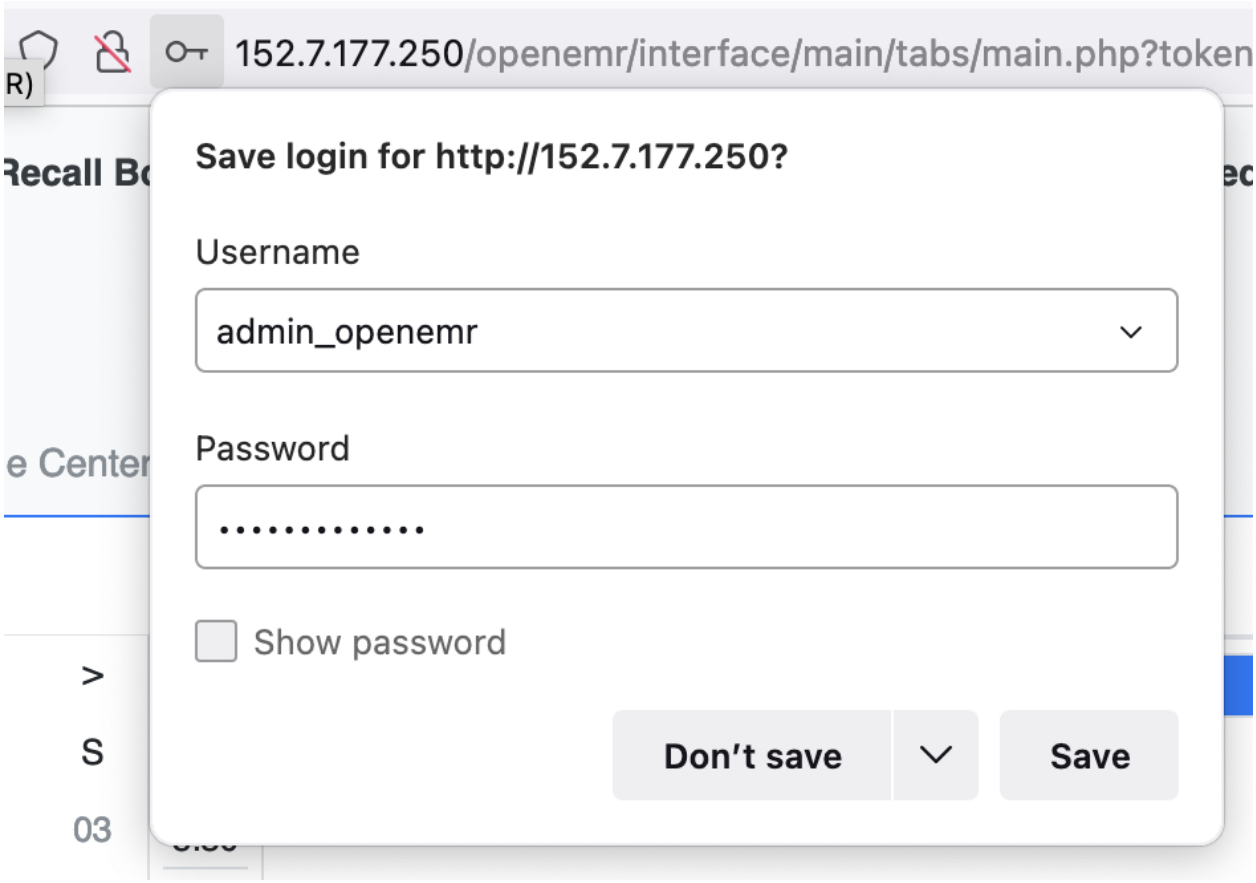
4. Test Case ID: 8.2.1-1

Description: Verify the application sets sufficient anti-caching headers so that sensitive data is not cached in modern browsers.

Repeatable Steps:

1. Open openemr application in one of the following autocomplete enabled browsers.
    - Internet Explorer version 11 or above
    - Firefox version 30 or above
    - Chrome version 34 or above
2. Try login using the credentials. (username: admin_openemr, password: admin_openemr). For example in firefox, during login you will notice some pop ups to save credentials.


openEMR
The most popular open-source Electronic Health Record and Medical Practice Management solution.
Acknowledgments, Licensing and Certification
Username:
Username:
This connection is not secure. Logins entered here could be compromised. Learn More
View Saved Logins
Password:
Language:
Default - English (Standard)
Login



3. Now try to re-login using saved password credentials in the browser.

Expected Results: When autocomplete is enabled, hackers can directly steal your password from local storage. So autocomplete should not be enabled for the application.
CWE-525: Use of Web Browser Cache Containing Sensitive Information.

Time is taken and true positives per hour: About 1 hr 30 mins to plan the black-box test cases and about 1 hr to execute them.
True Positives: 3
True Positives Per hour: 2

New Test Coverage Report:

Before the above 4 black-box test cases, the coverage is **21/71 = 29.58%.**
Now it's **25/71 = 35.21%**

True Positives in Project 1: 7
Time taken for black-box test cases in Project 1: 19 hours 35 minutes

True Positives in Project 2: 5
Time taken for black-box test cases in Project 2: 21 hours 30 minutes

True Positives in Project 3: 14
Time taken for black-box test cases in Project 3: 17 hours

Total True Positives: 26
Total Time takes: 58 hours 5 minutes
True Positives per hour (Project 1,2 & 3): 0.45

Reflect on the controls where we have lower coverage for:

1. V1 is all about Architecture, Design and Threat Modeling. Since we are not part of architecture and design in openemr we got very low coverage there. In threat modeling we didn't write any black-box test cases.
2. Before project 6 part5, we found black-box test cases for just 1 control in V2. But in part 6 we explored 3 more controls. This is probably because we haven't explored much in V2 though they are easy to explore.
3. V3, V4, V5, V7, V12 has coverage greater than 50%. We feel like we explored enough for these controls for the project we have done.
4. For V6, we find stored secrets as part of secret detection tools. But we haven't written any black-box test cases for it. That's the reason why there is no coverage for V6.
5. We didn't explore much on V8 - data protection.
6. We had no clue on how to test the controls in V9.
7. V10 is all about malicious code used as part of 3rd party libraries. This is done as part dependency checker. Since we didn't write black-box test cases the coverage is low. Though we were able to achieve 33%.
8. V11 is about business logic security. We felt it's hard to explore such vulnerabilities.
9. It's easy to explore V13, but we didn't get through it as part of this project.
10. V14 is about configuration management which we haven't explored much.

Attack Tree Exercise

Table 1

Attack Groups	APT41	Deep Panda	Leviathan	menuPass	Tonto Team
T1071 (.001, .002,.003, .004)	X				
T1560 (.001)	X				
T1560			X		
T1197	X		X		
T1547 (.001)	X				
T1547 (.001,.009)			X		
T1110 (.002)	X				
T1059 (.001,.003,. 004)	X				
T1136 (.001)	X				
T1543 (.003)	X				
T1486	X				
T1005	X			X	
T1059 (.001)		X			
T1059 (.001, .005)			X		
T1059 (.001,.003)				X	

T1059 (.001,.006)					X
T1546 (.008)		X			
T1564 (.003)		X			
T1027 (.005)		X			
T1057		X			
T1021 (.002)		X			
T1018		X			
T1505 (.003)		X			X
T1218 (.010)		X			
T1047		X			
T1056 (.001)					X
T1566 (.001)					X
T1090 (.002)					X
T1203					X
T1068					X
T1210					X
T1574 (.001)					X
T1105					X
T1135					X
T1003					X
T1069					X

(.001)					
T1087 (.002)				X	
T1583 (.001)			X	X	
T1560 (.001)				X	
T1119				X	
T1039				X	
T1074 (.001, .002)			X	X	
T1140			X	X	
T1568 (.001)				X	
T1586 (.001, .002)			X		
T1189			X		
T1585			X		

Table 2

Techniques	**Tactics**
T1071	Command and Control
T1560	Collection
T1197	Defense Evasion, Persistence
T1547	Persistence, Privilege Escalation
T1110	Credential Access
T1059	Execution
T1136	Persistence

T1543	Persistence, Privilege Escalation
T1486	Impact
T1005	Collection
T1546	Privilege Escalation, Persistence
T1564	Defense Evasion
T1027	Defense Evasion
T1057	Discovery
T1021	Lateral Movement
T1018	Discovery
T1505	Persistence
T1218	Defense Evasion
T1047	Execution
T1056	Collection, Credential Access
T1566	Initial Access
T1090	Command and Control
T1203	Execution
T1068	Privilege Escalation, Persistence
T1210	Lateral Movement
T1574	Persistence, Privilege Escalation, Defense Evasion
T1105	Command and Control
T1135	Discovery
T1003	Credential Access
T1069	Discovery
T1087	Discovery
T1583	Resource Development
T1119	Collection

T1039	Collection
T1074	Collection
T1140	Defense Evasion
T1568	Command and Control
T1586	Resource Development
T1189	Initial Access
T1585	Resource Development

Table 3

Techniques	Mitigations
T1071	M101, Network Intrusion Prevention
T1560	M1047, Audit
T1197	M1037 Filter Network Traffic M1028 Operating System Configuration M1018 User Account Management
T1110	M1036 ,Account Use Policies M1032 ,Multi-factor Authentication M1027 ,Password Policies M1018 ,User Account Management
T1059	M1049 ,Antivirus/Antimalware M1040 ,Behavior Prevention on Endpoint M1045 ,Code Signing M1042 ,Disable or Remove Feature or Program M1038 ,Execution Prevention M1026 ,Privileged Account Management M1021 ,Restrict Web-Based Content
T1136	M1032 ,Multi-factor Authentication M1030 ,Network Segmentation M1028 ,Operating System Configuration M1026 ,Privileged Account Management
T1543	M1047 Audit M1033 Limit Software Installation M1022 Restrict File and Directory

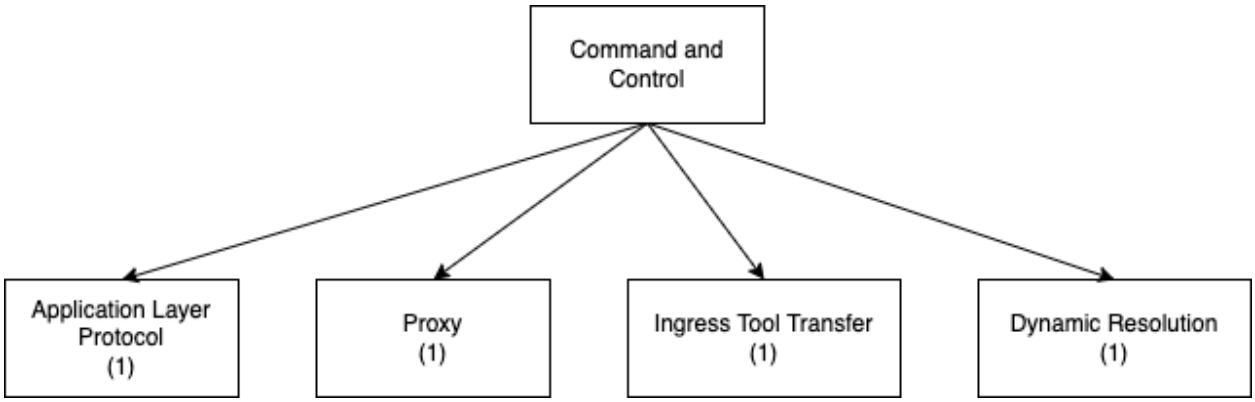
	Permissions M1018 User Account Management
T1486	M1040 Behavior Prevention on Endpoint M1053 Data Backup
T1005	M1057,Data Loss Prevention
T1027	M1049 Antivirus/Antimalware M1040 Behavior Prevention on Endpoint
T1021	M1032 Multi-factor Authentication M1018 User Account Management
T1505	M1047 Audit M1045 Code Signing M1042 Disable or Remove Feature or Program M1026 Privileged Account Management M1018 User Account Management
T1218	M1042 Disable or Remove Feature or Program M1038 Execution Prevention M1050 Exploit Protection M1026 Privileged Account Management
T1047	M1040 Behavior Prevention on Endpoint M1038 Execution Prevention M1026 Privileged Account Management M1018 User Account Management
T1566	M1049, Antivirus/Antimalware M1031 Network Intrusion Prevention M1021 Restrict Web-Based Content M1054 Software Configuration M1017 User Training
T1090	M1037, Filter Network Traffic M1031, Network Intrusion Prevention M1020, SSL/TLS Inspection
T1203	M1048 Application Isolation and Sandboxing M1050 Exploit Protection
T1210	M1048 Application Isolation and Sandboxing M1042 Disable or Remove Feature or Program M1050 Exploit Protection

	M1030 Network Segmentation M1026 Privileged Account Management M1019 Threat Intelligence Program M1051 Update Software M1016 Vulnerability Scanning
T1574	M1013 Application Developer Guidance M1047 Audit M1038 Execution Prevention M1022 Restrict File and Directory Permissions M1044 Restrict Library Loading M1024 Restrict Registry Permissions M1051 Update Software M1052 User Account Control M1018 User Account Management
T1105	M1031 Network Intrusion Prevention
T1135	M1028 Operating System Configuration
T1003	M1015 Active Directory Configuration M1040 Behavior Prevention on Endpoint M1043 Credential Access Protection M1041 Encrypt Sensitive Information M1028 Operating System Configuration M1027 Password Policies M1026 Privileged Account Management M1025 Privileged Process Integrity M1017 User Training
T1087	M1028 Operating System Configuration
T1583	M1056 Pre-compromise
T1119	M1041 Encrypt Sensitive Information M1029 Remote Data Storage
T1568	M1031 Network Intrusion Prevention M1021 Restrict Web-Based Content
T1586	M1056 Pre-compromise
T1189	M1048 Application Isolation and Sandboxing M1050 Exploit Protection M1021 Restrict Web-Based Content M1051 Update Software
T1585	M1056 Pre-compromise

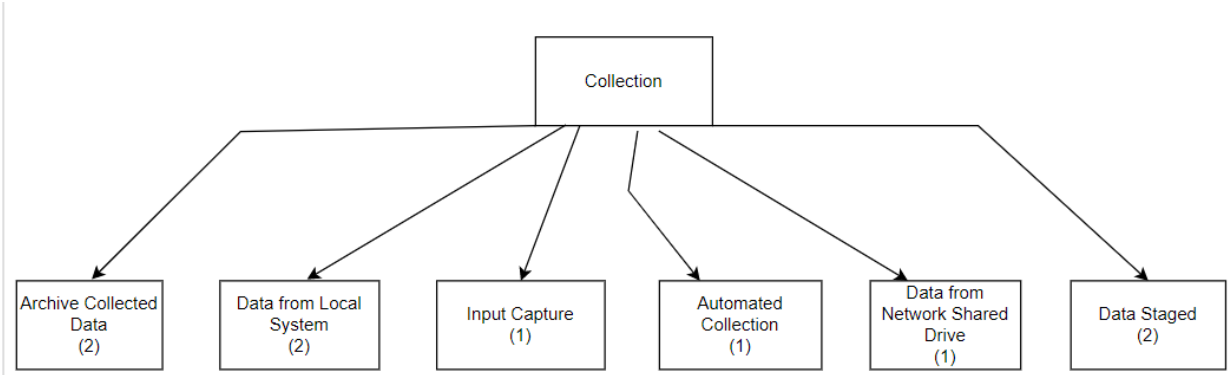
Attack Trees

---

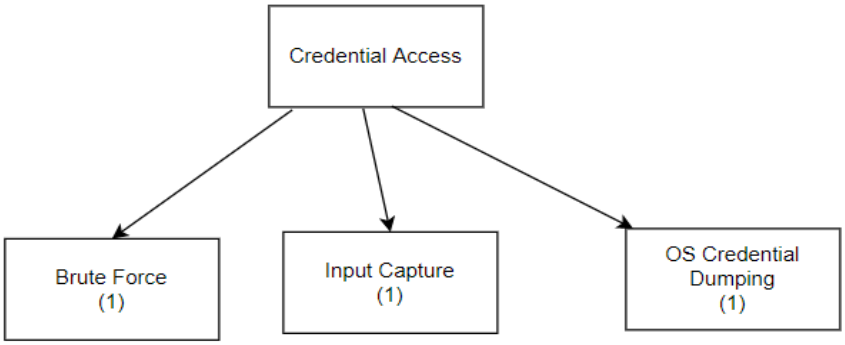
1. Tactic - Command and Control



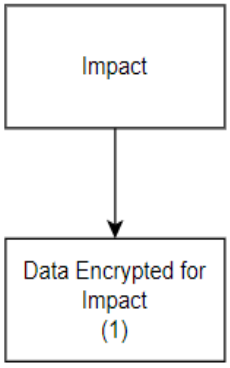
2. Tactic - Collection



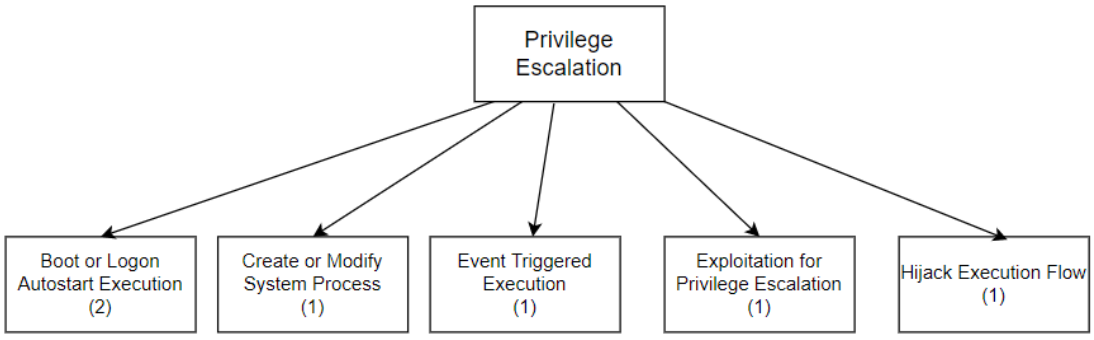
3. Tactic - Credential Access



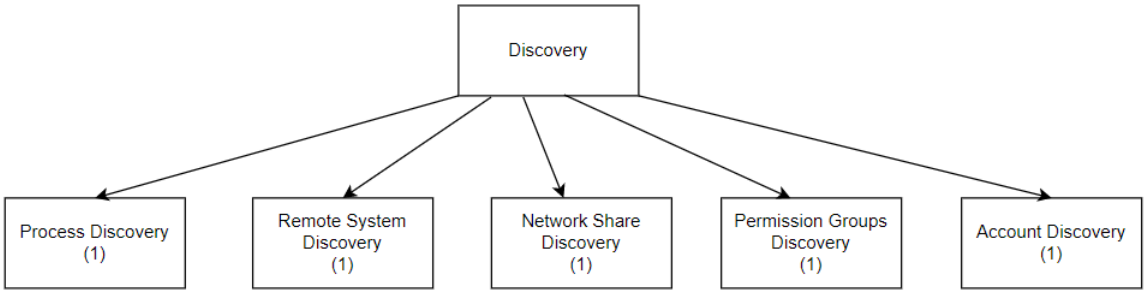
4. Tactic - Impact



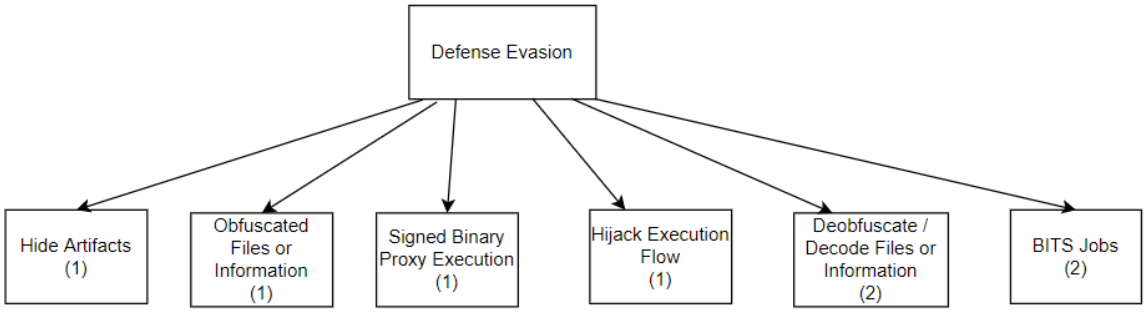
5. Tactic - Privilege Escalation



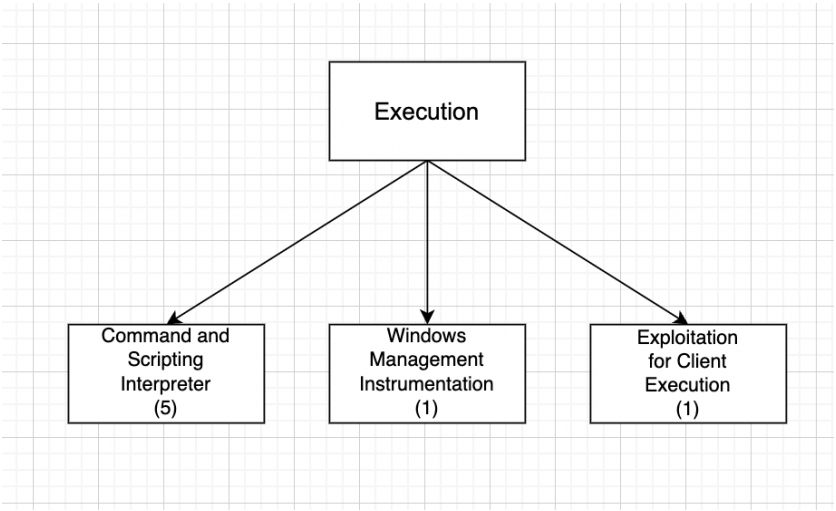
6. Tactic - Discovery



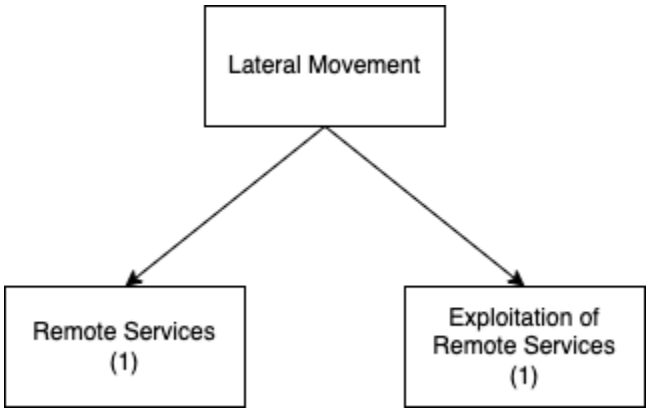
7. Tactic - Defense Evasion



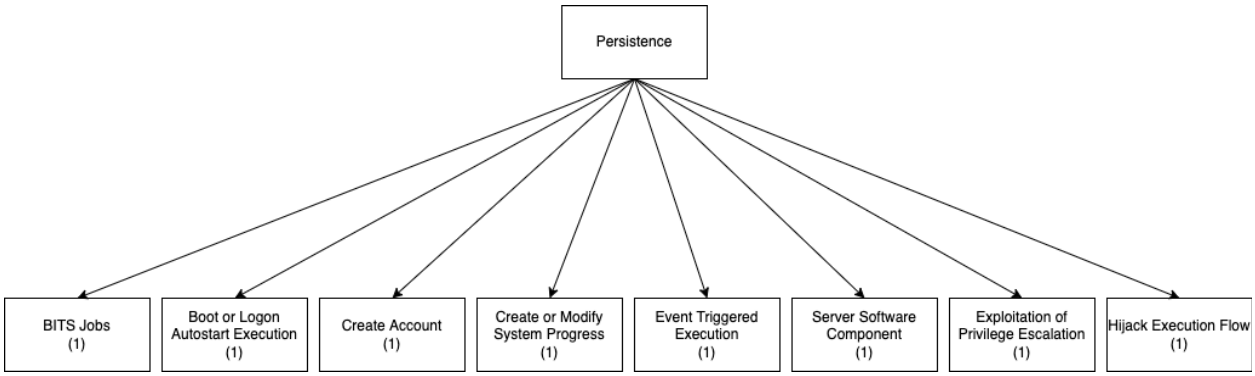
8. Tactic - Execution



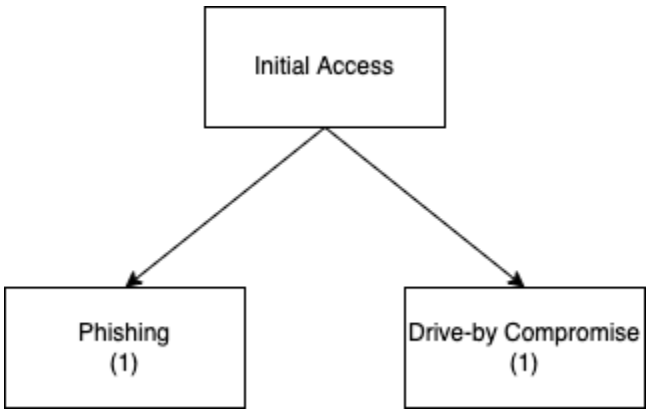
9. Tactic - Lateral Movement



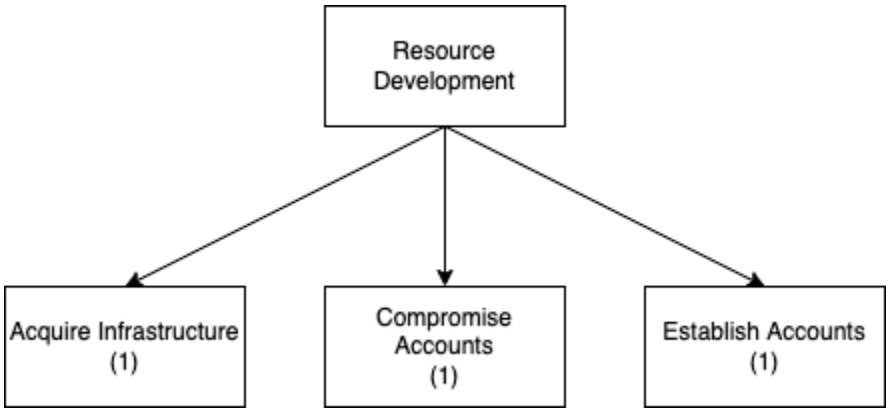
10. Tactic - Persistence



11. Tactic - Initial Access



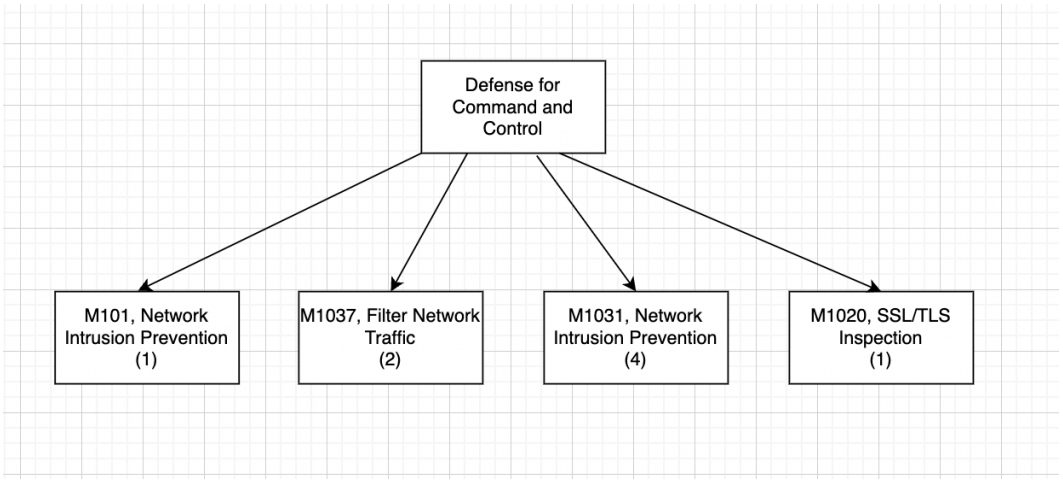
12. Tactic - Resource Development



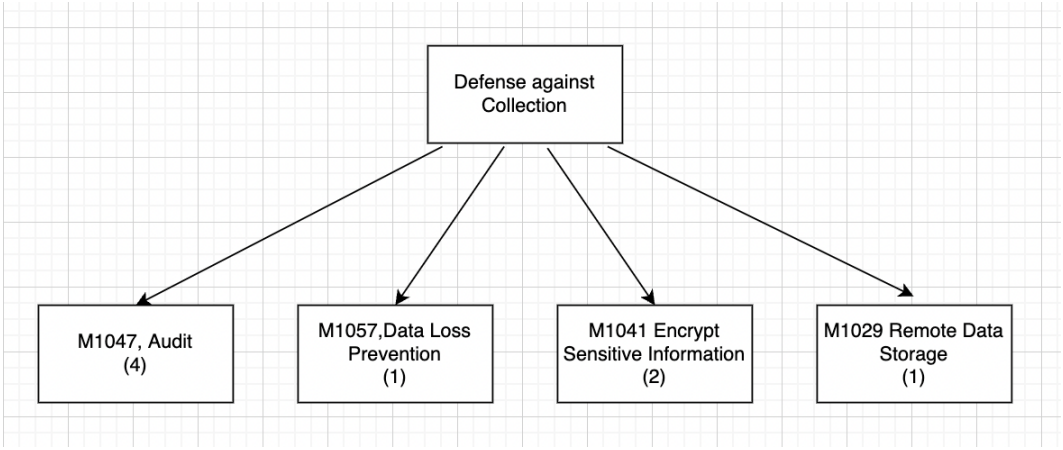
Defense Trees

---

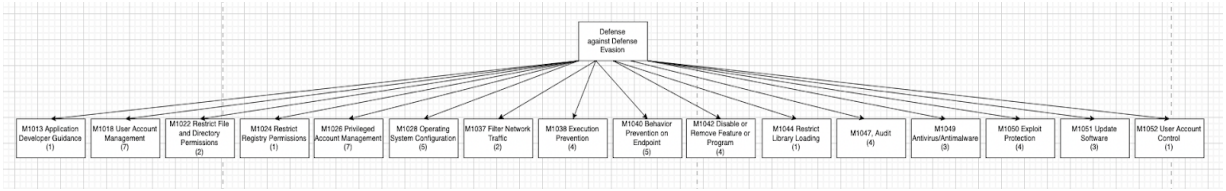
1. Command and Control



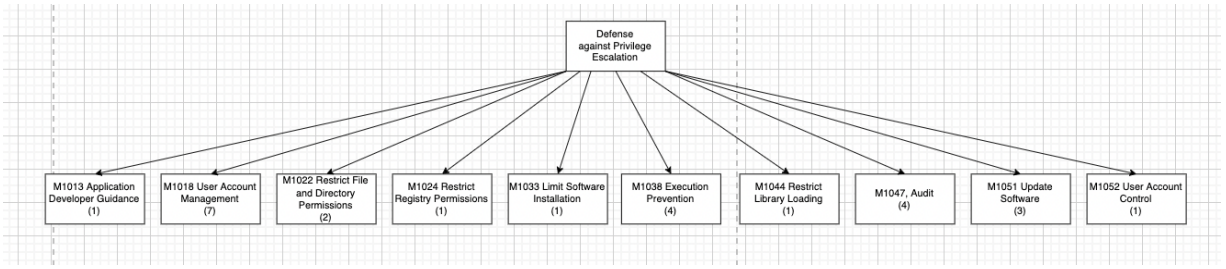
2. Collection



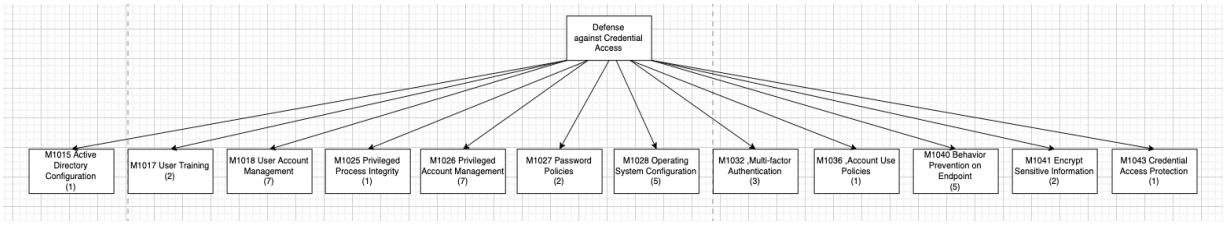
3. Defense Evasion



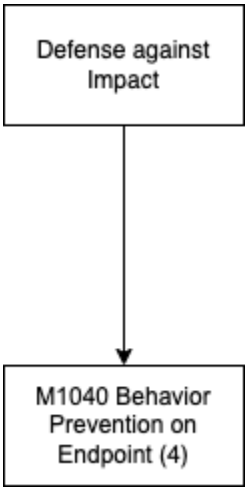
4. Privilege Escalation



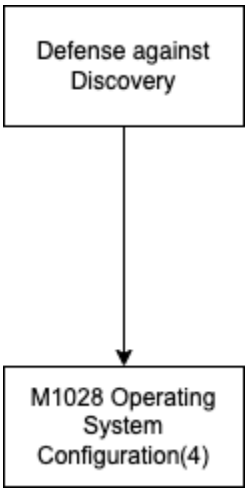
5. Credential Access



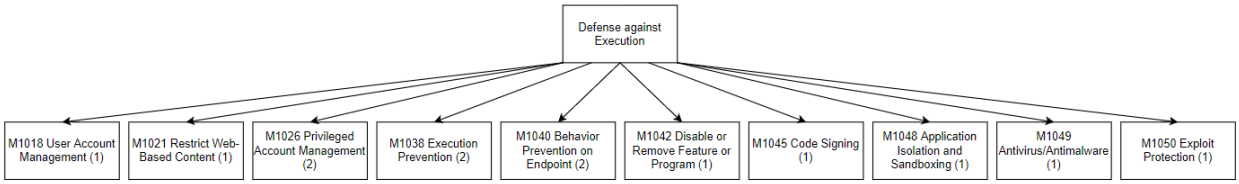
6. Impact



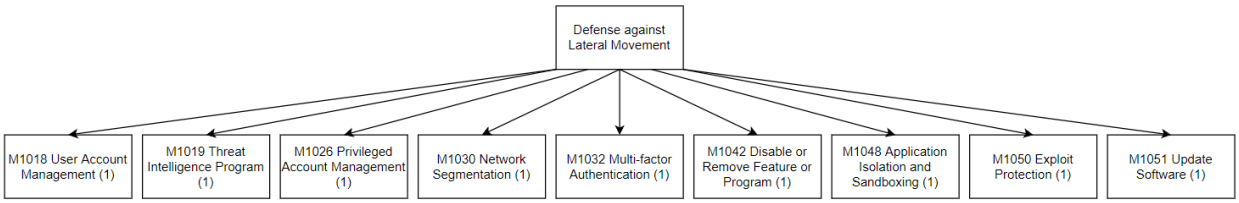
7. Discovery



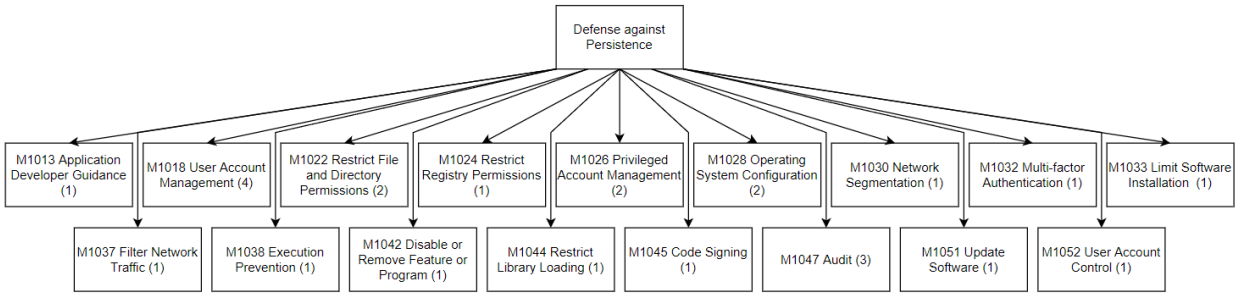
8. Execution



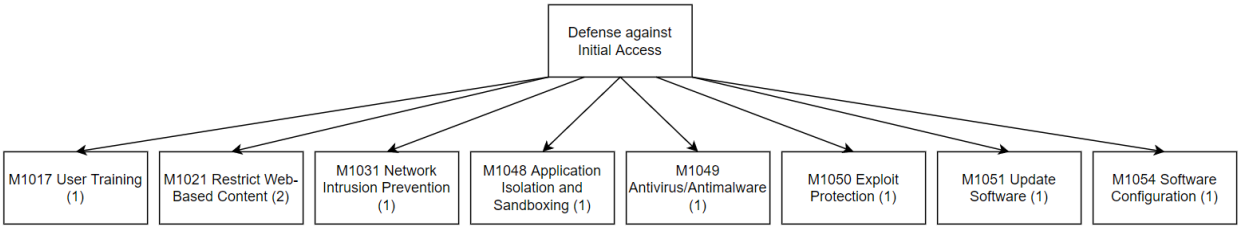
9. Lateral Movement



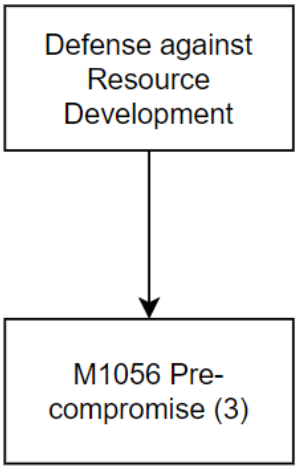
10. Persistence



11. Initial Access



12. Resource Development



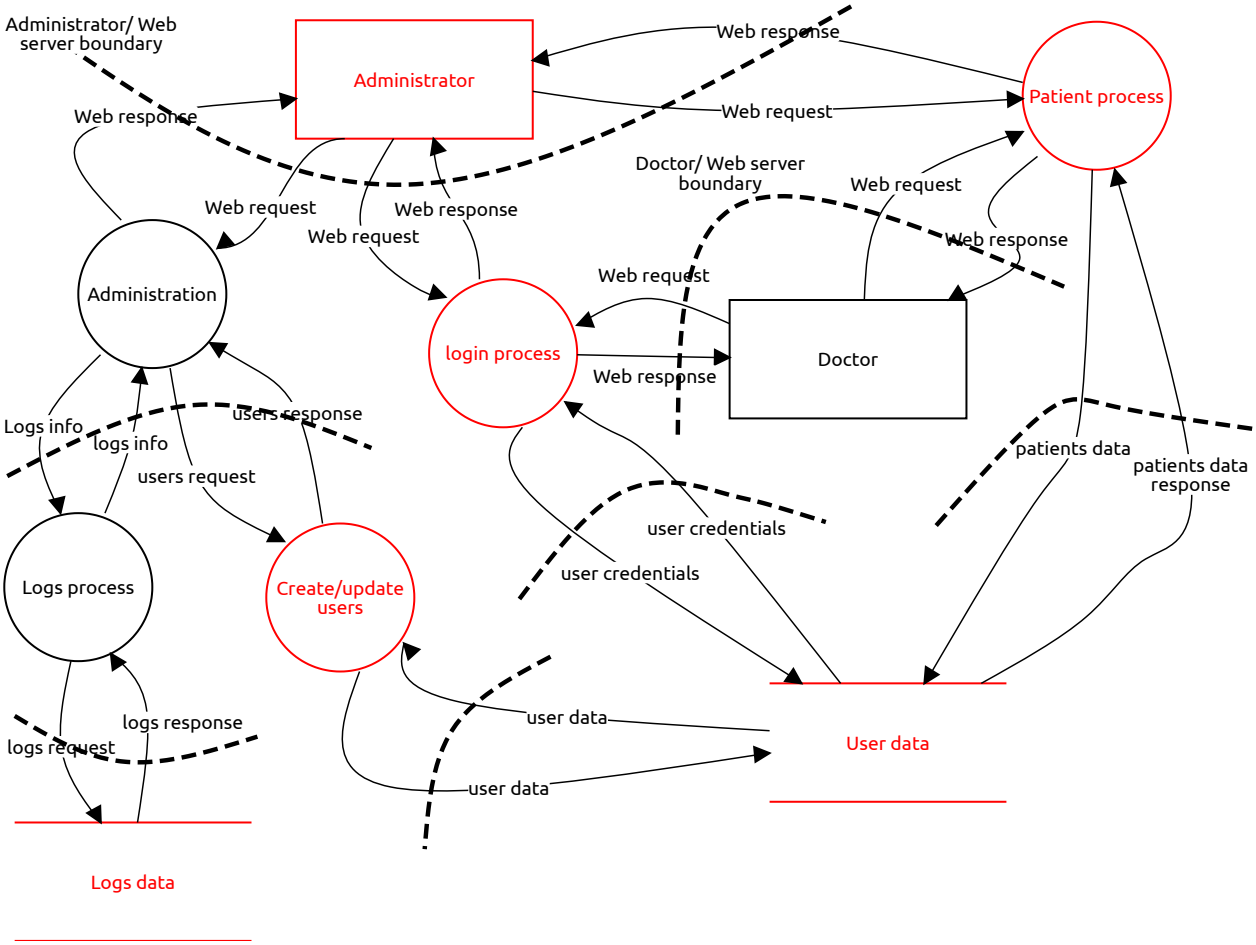
Threat model report for Demo Threat Model

Owner:
Mike Goodwin
Reviewer:
Jane Smith
Contributors:
Tom Brown; Albert Moneypenny

High level system description

A sample model of a web application, with a queue-decoupled background process.

Main Request Data Flow



Administrator (External Actor)

Description:

Information Disclosure threat

Information disclosure, Open, Medium Priority

Description:
An attacker can read data because it's hidden or occluded (for undo or change tracking) and the user might forget that it's there.

The webserver's in general create hidden / backup of the files it is currently serving in different extensions like .old, .bac, .bak files. This allows the attacker to view the logic of the scripts and extract useful information such as code bugs or logins and passwords.

Mitigation:
Removing the scripts from the web server to keep the web root clean. Write automated scripts to constantly remove all the backup files created by the web server.

Elevation of Privilege

Elevation of privilege, Open, Medium Priority

Description:
An attacker can force data through different validation paths which give different results.

In Openemr, the user logged in as admin can view or edit information about all the users in the system. But this should not be allowed for the user who is a front end office worked. We are able to elevate the privileges of the front end office worker as he is able to view/edit or create new facilities by accessing an endpoint which is should only be accessible for the facilities admin.
http://ip_address/openemr/interface/usergroup/facility_admin.php?token_main=token

Mitigation:
Per role authorization rules should be updated to restrict such access. So, all such requests would be denied by default.

login process (Process)

Description:

Generic spoofing threat

Spoofing, Open, Medium Priority

Description:
S3 - A user can try any number of login attempts in the openemr login page.

Mitigation:
After 3-4 login attempt failures the particular username should be blocked by 24-48 hours.

Generic spoofing threat

Spoofing, Open, Medium Priority

Description:
S5 - Typo squatting is not implemented in openemr. An attacker can create a website with small typo. If a user misspelled the website name and if there is a website created with that name by attacker, the user info will be at risk.

Mitigation:
The company should buy all the domains which could be easily used in a common typo squatting attack.

Cornucopia threat

Elevation of privilege, Open, Medium Priority

Description:
Data Validation & Encoding 2 - Brian can gather information about the underlying configurations, schemas, logic, code, software, services and infrastructure due to the content of error messages, or poor configuration, or the presence of default installation files or old, test, backup or copies of resources, or exposure of source code

In openemr, when we navigate to http://localhost/openemr/interface , we will get to see all folders and files which are part of project.

This threat is found as part of Cornucopia game.

Mitigation:
Set open_basedir in php.ini configuration file.

Cornucopia threat

Elevation of privilege, Open, Medium Priority

Description:
AuthN3 - Muhammad can obtain a user's password or other secrets such as security questions, by observation during entry, or from a local cache, or from memory, or in transit, or by reading it from some unprotected location, or because it is widely known, or because it never expires, or because the user cannot change her own password. Example: This threat exists in openemr. Keep the network tab open and try logging in using username:"admin_openemr" and password: "admin_openemr". In the network tab we must be able to see a POST call being made which shows password in plaintext in the input payload parameters in transit.

Mitigation:
The password should not be shown in plain text format even in the frontend of a browser. HTTPS with SSL protocol must be used to encrypt sensitive data like passwords in transit.

Generic spoofing threat

Spoofing, Open, Medium Priority

Description:
S9 - An attacker who gets a password can reuse it (Use stronger authenticators). If an attacker obtains a doctor's password from another source, they can log into OpenEMR if the doctor has reused that password for OpenEMR.

Mitigation:
Require OpenEMR users to set up multi factor authenticaton

Cornucopia threat

Elevation of privilege, Open, Medium Priority

Description:
Authorization 3 - Christian can access information, which they should not have permission to, through another mechanism that does have permission (e.g. search indexer, logger, reporting), or because it is cached, or kept for longer than necessary, or other information leakage.

The login process can be skipped completely by visiting localhost/openemr/interface. The user will then have access to the file directory

Mitigation:
Do not expose any endpoints that allow access to the directory structure

Web request (Data Flow)

Description:

No threats listed.

Web response (Data Flow)

Description:

No threats listed.

User data (Data Store)

Description:

LINDDUN Identifiability

Identifiability, Open, Medium Priority

Description:
LINDDUN - Identifying Stored Data - Personal data being stored can be identified (because they are insufficiently minimized/de-identified before storage).

User data is stored with identifiable attributes attributes in the database (i.e. full name, phone number, email address, home address).

Mitigation:
The data can be de-identified by replacing identifying attributes (e.g. name, address) by an internal identifier. A link to the actual identity can be kept however , which still allows identifiability.

Data can be stored with username, email address or SSN as (internal) identifier

user credentials (Data Flow)

Description:

No threats listed.

user credentials (Data Flow)

Description:

No threats listed.

Administration (Process)

Description:

No threats listed.

Create/update users (Process)

Description:

Elevation of Privilege threat

Elevation of privilege, Open, Low Priority

Description:
EK - An attacker can inject a command that the system will run at a higher privilege level.

When creating a user, we can use Cross-site scripting text to inject malicious code. Though the system accepts the malicious code, it doesn't execute the malicious code.

Mitigation:
Deny list, allow lists should be used so that no one can inject any html text into the web application.

LINDDUN Non-Repudiation Threat

Non-repudiation, Open, Medium Priority

Description:
Is the origin of incoming communication known and traceable to the sender? Is it a problem if a trace of this information is kept?
Example: An employee shares gossip among his co-workers via a digitally signed email. When his boss received the forwarded message, it is difficult for the employee to deny having spread the gossip.

Mitigation:
There are two types of security mechanisms for generating non-repudiation evidence: secure envelopes and digital signatures. A secure envelope provides protection of the origin and the integrity of a message based on a shared secret key between communication parties.

Elevation of Privilege threat

Elevation of privilege, Open, Low Priority

Description:
EQ - You include user-generated content within your page, possibly including the content of random URLs.

When creating a new user, cross site scripting can be attempted in the first or last name fields. This input is not sanitized, and is stored in the database. When the newly created user is returned, the script is not executed; however, the user generated code is displayed on the page.

Mitigation:
Deny list, allow lists should be used so that no one can inject any html text into the web application.

Logs process (Process)

Description:

No threats listed.

Logs info (Data Flow)

Description:

No threats listed.

logs info (Data Flow)

Description:

No threats listed.

users request (Data Flow)

Description:

No threats listed.

users response (Data Flow)

Description:

No threats listed.

user data (Data Flow)

Description:

No threats listed.

user data (Data Flow)

Description:

No threats listed.

Logs data (Data Store)

Description:

Generic threat to Non-repudiation

Non-repudiation, Mitigated, High Priority

Description:
Non-repudiation: There is evidence that can link the data subject to a certain action.

When a user logs in, the username is stored in logs. So the actor responsible for the action cannot deny the login.

Mitigation:
This threat is mitigated. OpenEMR logs when there is a failed login and for a successful login.

Generic threat to Non Compliance

Non-compliance, Open, Medium Priority

Description:
The system does not comply with data protection principles.

In Openemr system when an operation like creation of a new patient record by any user like Admin or Doctor logs a set of records like creation of patient record within the table and creation of insurance records for the patient.

Mitigation:
The sensitive information like the SSN within the SQL query that is being logged in log table and other details should either be encrypted or masked before logging.

logs request (Data Flow)

Description:

No threats listed.

logs response (Data Flow)

Description:

No threats listed.

Web response (Data Flow)

Description:

No threats listed.

Web request (Data Flow)

Description:

No threats listed.

Patient process (Process)

Description:

Cornucopia threat

Tampering, Open, Medium Priority

Description:
Data Validation & Encoding 4:
Dave can input malicious field names or data because it is not being checked within the context of the current user and process

In openemr system, during the new patient creation/edit process, the first name, last name or any other form elements like SSN can have malicious input like <script src=":alert(document.cookie)"></script>. The current implemented system accepts these kind of input.

Mitigation:
There should be strict validations in place not only in the front end but also API validation in order reject this kind of malicious input. Instead of encoding them and storing them in the database

Spoofing threat

Spoofing, Open, Medium Priority

Description:
An attacker could steal credentials stored on the server and reuse them.
(for example, a key is stored in a world readable file).
Examples:
1. https://tinyurl.com/nz7hcpnr contains a PGP private key. It can be used to decrypt some confidential patient data in our openemr application.
2. https://tinyurl.com/2p8vp8xw contains the password for mysql database. It can be used to login to the backend database and perform CRUD operations on the data in our openemr application.

Mitigation:
The passwords or keys should not be stored in the source code files. They should be stores in secret-managers (For example: AWS Secret Manager) and then should be fetched using API calls.

Elevation of privilege threat

Elevation of privilege, Open, Medium Priority

Description:
An attacker can reflect input back to a user, like cross site scripting. Example in openemr: In the patients process if there is a message to be sent between patient and administrator, the network call can be intercepted and we can modify

form_note_type=<script>alert('xss')</script>. This will reflect the given script input to the frontend.

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

Web request (Data Flow)

Description:

No threats listed.

Web response (Data Flow)

Description:

No threats listed.

patients data (Data Flow)

Description:

No threats listed.

patients data response (Data Flow)

Description:

No threats listed.

Doctor (External Actor)

Description:

No threats listed.

Web request (Data Flow)

Description:

No threats listed.

Web response (Data Flow)

Description:

No threats listed.

Web request (Data Flow)

Description:

No threats listed.

Web response (Data Flow)

Description:

No threats listed.