SECURE PROGRAMMING
PROJECT - PART 2

WEB APPLICATION FOR COURSE GRADEBOOK

Version 2.0
Date: 12/09/2018

Submitted by:
1. Shloak Agarwal
2. Aman Arora
3. Shweta Baskaran
4. Vishesh Kumar Dwivedi
5. Tejasvini Ravi Patil
6. Anirudh Raghavendrara Deshpande
7. Himanshu Ajay Singh
8. Balasubramaniam Theetharappan
9. Sarita Maruti Tigadi

Indexes

Vignettes of the system

- Input/output validation
  - Regular Expression check:
    Checks are in place for validating the format of the inputs entered by the user - the login ID and password.

```
protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    HttpSession session = request.getSession();
    if (session.getAttribute(Constants.SP_USERNAME) != null
            && Constants.SP_USERTYPE_PROFESSOR.equals(session.getAttribute(Constants.SP_USERTYPE))) {
        ProfessorDao dao = new ProfessorDao();
        Integer profID = new Integer(session.getAttribute(Constants.SP_USERID).toString());
        String id = request.getParameter("ssid");
        String psid = request.getParameter("psid");
        String grade = request.getParameter("grade");
        if(id.matches("^[0-9]+$") &&
                grade.matches("^[A-F]{1}$")) {
            int ssid = Integer.parseInt(id);
            new UserDao().updateGrade(ssid, profID, grade);
            LOG.info("grade updated for studentid "+ssid+" by professor id "+profID);
            response.sendRedirect("/sp/professor/grades?psid="+psid);

        }

    }
```

  - Existence in database:
    Checks are made to ensure that the login details entered by the user are valid and exist in the database. If not, suitable error messages are displayed.

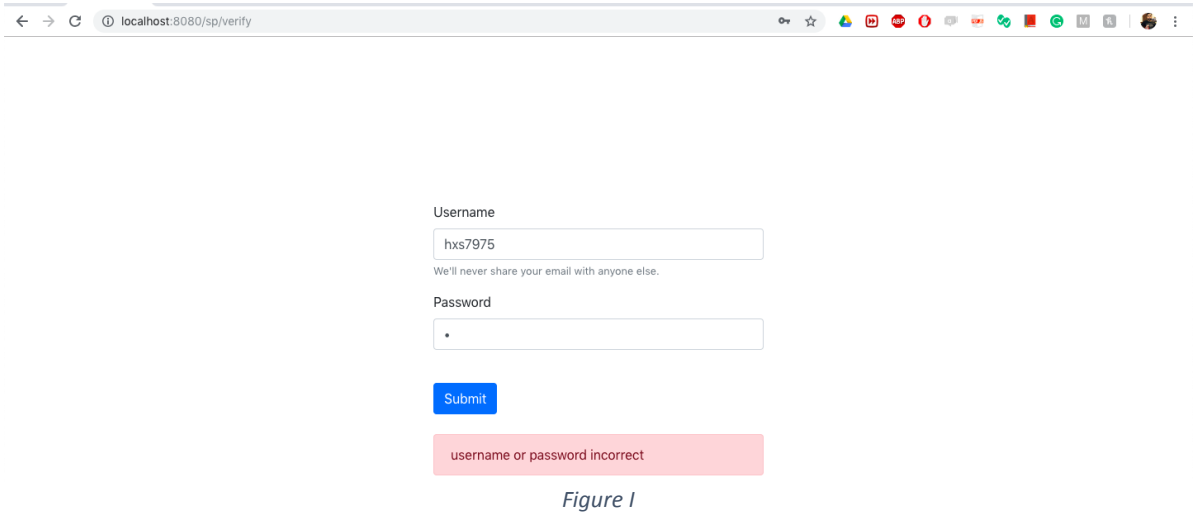


Figure I

- Use of centralized logging and logging of various application events
  A centralized login framework called log4j is used which is a logging package for Java. A centralized framework makes it easy to:
  - Provide one consistent view of the system reflected in the logs.
  - Facilitate changes, such as moving logging to another machine, switching from logging to a file to logging to a database, or updating validation or privacy measures.

- This is also preferred as it includes timestamps by default.

The events logged in the grade book are:

- User Login - authentication of user by Login ID and password
- Update of grades - by Professor
- User Logout - end of session
- This log is maintained on the server side and is not available to be accessed by students or professor. Only the database administrator and the website admin can access it.
- Both success as well as failure messages that are generated during the use of the grade book application is logged.

- List all events that your application specifically audits

  The following is the list of events which are audited by the application using log4j.properties
  - successful login
  - grade updated
  - successful logout

- Exception/error handling

  Exceptions are properly handled by using the try-catch statement block. Appropriate ways of handling exceptions are implemented for each exception thrown.

```
protected int update(String query, Object[] data) {
    createConnection();
    PreparedStatement preparedStatement;
    int i = 0;
    try {
        preparedStatement = connection.prepareStatement(query);
        for (int j = 0; j < data.length; j++) {
            Object object = data[j];
            if (object instanceof Integer) {
                preparedStatement.setInt(j + 1, ((Integer) object));
            } else if (object instanceof String) {
                preparedStatement.setString(j + 1, object.toString());
            }

        }
        i = preparedStatement.executeUpdate();
    } catch (Exception e) {
        LOGGER.error(e);
    }
    closeConnection();
    return i;
}
```

- Sample log file entries

  The centralized log maintained on the server side is updated whenever any important event takes place in the client end. The app.log file is generated by the application has the following log entries.

```
address => 0:0:0:0:0:0:0:1
2018-12-09 14:32:26 INFO  SPFilter:38 - valid request from ip-
address => 0:0:0:0:0:0:0:1
2018-12-09 14:32:27 INFO  SPFilter:70 - successfull login
request for ttj7975 from ip-address => 0:0:0:0:0:0:0:1
2018-12-09 14:32:27 INFO  SPFilter:48 - valid request from ip-
address => 0:0:0:0:0:0:0:1
2018-12-09 14:32:29 INFO  SPFilter:48 - valid request from ip-
address => 0:0:0:0:0:0:0:1
2018-12-09 14:32:34 INFO  SPFilter:48 - valid request from ip-
address => 0:0:0:0:0:0:0:1
2018-12-09 14:32:34 INFO  ProfessorContrller:87 - grade
updated for studentid5 by professor id 1
2018-12-09 14:32:34 INFO  SPFilter:48 - valid request from ip-
address => 0:0:0:0:0:0:0:1
2018-12-09 14:32:40 INFO  SPFilter:48 - valid request from ip-
address => 0:0:0:0:0:0:0:1
2018-12-09 14:32:40 INFO  ProfessorContrller:87 - grade
updated for studentid10 by professor id 1
2018-12-09 14:32:40 INFO  SPFilter:48 - valid request from ip-
address => 0:0:0:0:0:0:0:1
2018-12-09 16:56:28 ERROR SPFilter:43 - invalid request from
ip-address => 0:0:0:0:0:0:0:1
2018-12-09 16:56:28 INFO  SPFilter:38 - valid request from ip-
address => 0:0:0:0:0:0:0:1
2018-12-09 16:58:33 INFO  SPFilter:38 - valid request from ip-
address => 0:0:0:0:0:0:0:1
2018-12-09 16:58:33 INFO  SPFilter:70 - successfull login
request for ttj7975 from ip-address => 0:0:0:0:0:0:0:1
2018-12-09 16:58:33 INFO  SPFilter:48 - valid request from ip-
```

- Session management
  A new session begins every time a user is validated and the session identifier is generated on the server side. This session identifier is no longer valid and the session expires once the user logs out of the system.

  A timestamp is set for maximum session lifetime and is fixed to be for 30 minutes. If inactivity on the web application occurs for more than 30 minutes, the session expires.

```
<!-- ==================== Default Session Configuration ================ -->
<!-- You can set the default session timeout (in minutes) for all newly   -->
<!-- created sessions by modifying the value below.                       -->

  <session-config>
      <session-timeout>30</session-timeout>
  </session-config>
```

```
} else {
    LOG.info("successfull login request for "+clientUser.getUserName()+" from ip-address => "+request.getRemote
    dbUser.setLoginAttepmts(0);
    userDao.update(dbUser);
    request.setAttribute("error", "login successful");
    if(dbUser.getProfessorId()!=0) {
        Professor professor=new Professor();
        professor.setProfessorId(dbUser.getProfessorId());
        professor = new ProfessorDao().getOne(professor);
        HttpSession session = request.getSession();

        session.setAttribute(Constants.SP_USERNAME, professor.getName());
        session.setAttribute(Constants.SP_USERID, professor.getProfessorId());
        session.setAttribute(Constants.SP_USERTYPE, Constants.SP_USERTYPE_PROFESSOR);
        response.sendRedirect(request.getContextPath()+"/professor/dashboard");
    }else if(dbUser.getStudentId()!=0) {
        Student student=new Student();
        student.setStudentId(dbUser.getStudentId());
        student = new StudentDao().getOne(student);
        HttpSession session = request.getSession();

        session.setAttribute(Constants.SP_USERNAME, student.getName());
        session.setAttribute(Constants.SP_USERID, student.getStudentId());
        session.setAttribute(Constants.SP_USERTYPE, Constants.SP_USERTYPE_STUDENT);
        response.sendRedirect(request.getContextPath()+"/student/dashboard");
    }

}
```

- Authentication and authorization
  A user is validated using the login page where user enters login ID and password. Depending on the login details (as verified from the backend database), further views of the user differs and different privileges are granted. For example, if the user login details match with an entry in the database that is of a Student's, the grade edit feature is disabled and the student is authorized only to view the grades. Also, each student is restricted to viewing only his/her grade and not grade of any other student. If the login details are authenticated to be of a Professor, he/she can view grades of all students and also edit/update the grades. These changes will be logged.

- Cryptography
  SALT is used for encrypting the user password when it is stored and transmitted from client to server while user logs in. It is a random data that is used as an additional input to a one-way function that "hashes" data, a password or passphrase. Hence it is used to safeguard passwords in storage.

  A new salt is randomly generated for each password.

  The salt and the password (or its version after Key stretching) are concatenated and processed with a cryptographic hash function, and the resulting output (but not the original password) is stored with the salt in a user database. Hashing allows for later authentication without keeping and therefore risking the plaintext password in the event that the authentication data store is compromised.

  Here, SHA1 is used for key generation.

```
public boolean check(User clientUserObject) throws Exception {
    String[] saltAndHash = password.split("\\$");
    if (saltAndHash.length != 2) {
        throw new IllegalStateException("The stored password must have the form 'salt$hash'");
    }
    String hashOfInput = hash(clientUserObject.getPassword(), Base64.decodeBase64(saltAndHash[0]));
    return hashOfInput.equals(saltAndHash[1]);
}


private String hash(String password, byte[] salt) throws Exception {
    if (password == null || password.length() == 0)
        throw new IllegalArgumentException("Empty passwords are not supported.");
    SecretKeyFactory f = SecretKeyFactory.getInstance("PBKDF2WithHmacSHA1");
    SecretKey key = f.generateSecret(new PBEKeySpec(password.toCharArray(), salt, iterations, desiredKeyLen));
    return Base64.encodeBase64String(key.getEncoded());
}
```

- Privilege management
  Each user is associated with a Role ID which has a specific set of Privileges. User cannot perform operations for which he/she does not have privileges. Each time user logins into the application, the respective privileges are assigned to the user

  The log files and database information will be accessible only by the system administrator and not the Student or Professor.

- Mitigations for buffer overflows
  Java is designed to avoid the buffer overflow by checking the bounds of a buffer (like an array) and preventing any access beyond those bounds. Hence it is a safe language and takes care of buffer overflows by itself.

- Mitigations for Cross Site Request Forgeries (CSRF)
  Cross-site request forgery, is a type of malicious exploit of a website where unauthorized commands are transmitted from a user that the web application trusts. In our application we are using strong encryption and functions like SHA-1 and SALT that will generate hashes for password and paraphrase and SHA-1 used for key generation before transmitting to server.

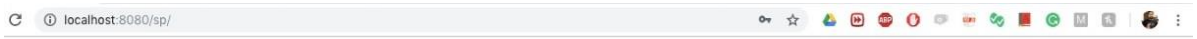
- Mitigations for injection attacks
  SQL injection is a code injection technique, used to attack data-driven applications, in which nefarious SQL statements are inserted into an entry field for execution. We prevent this type of an attack in our application by making use of SQL prepared statements and incorporating whitelisting data on server side.

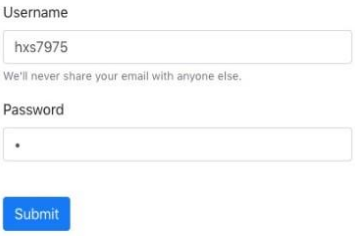
```
protected int update(String query, Object[] data) {
    createConnection();
    PreparedStatement preparedStatement;
    int i = 0;
    try {
        preparedStatement = connection.prepareStatement(query);
        for (int j = 0; j < data.length; j++) {
            Object object = data[j];
            if (object instanceof Integer) {
                preparedStatement.setInt(j + 1, ((Integer) object));
            } else if (object instanceof String) {
                preparedStatement.setString(j + 1, object.toString());
            }

        }
        i = preparedStatement.executeUpdate();
    } catch (Exception e) {
        LOGGER.error(e);
    }
    closeConnection();
    return i;
}
```

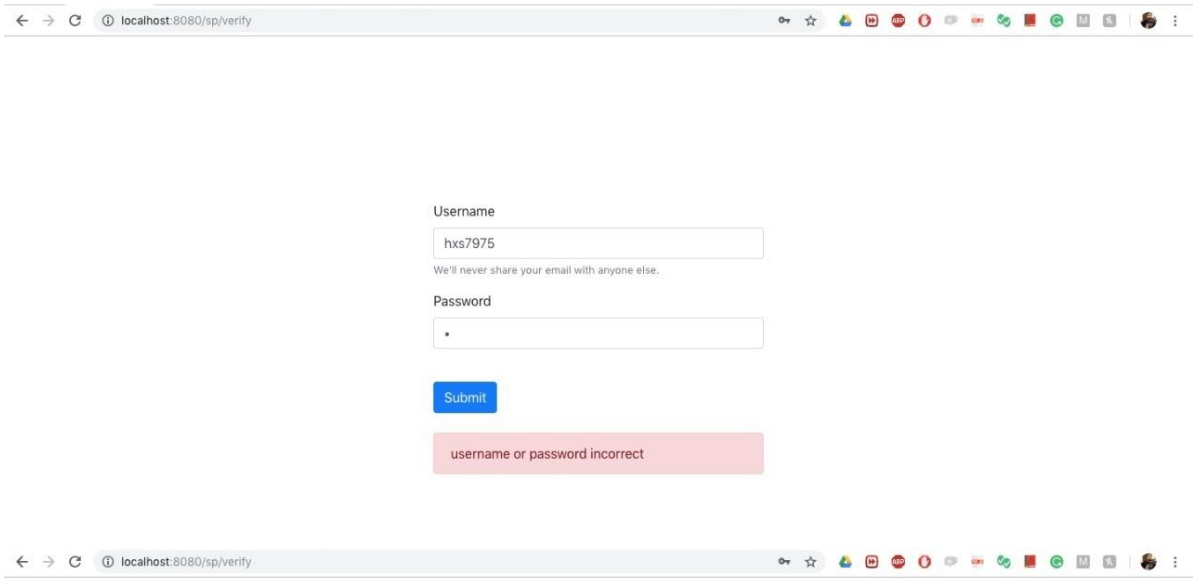
- Handling of sensitive information
  All the data with regards to the grade book application is stored securely on the database with only access to the system administrator. Passwords are encrypted using the SHA1 SALT encryption method so that passwords of all users are kept safe when stored and transmitted. Session ID is also a sensitive data and is randomly generated on successful authentication of the user.

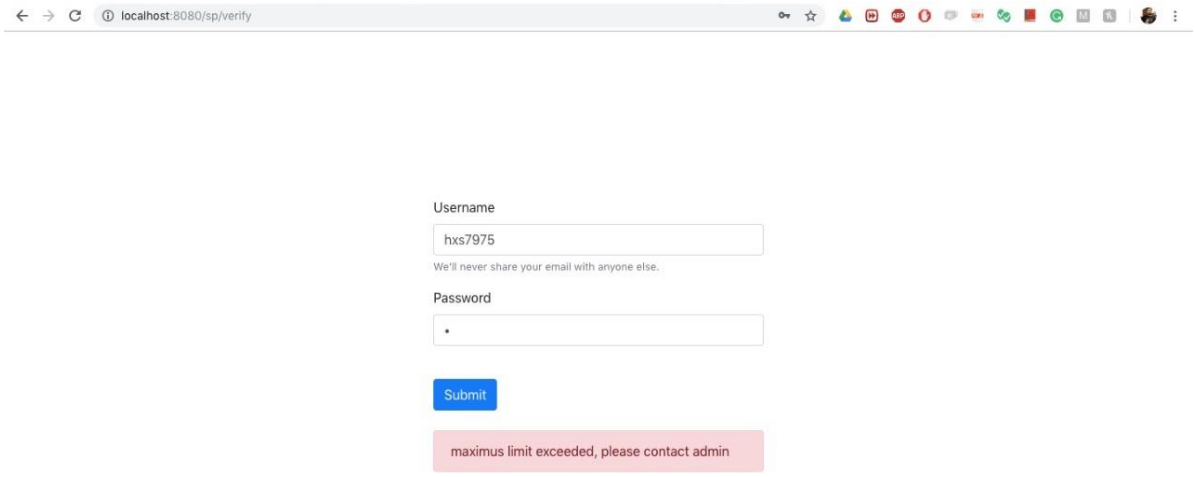
Execution of scenarios

- Scenarios:
  1. User (Student/Professor) will login to the system by entering the valid credentials.
  2. Student will login to the system to retrieve the course details (term, subjects, grades etc.).
  3. Professor will login to the system to retrieve and update course details.
- Screenshots of system execution:
  LOGIN:

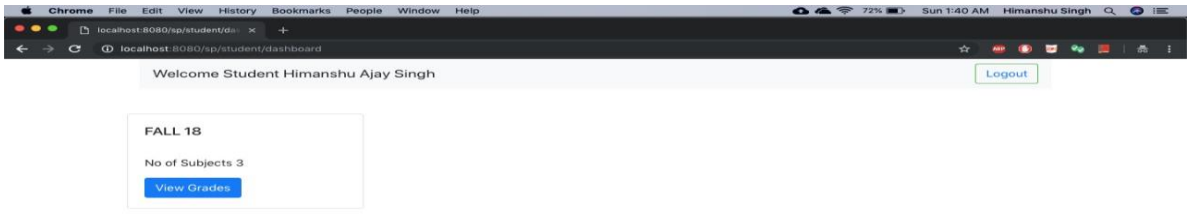




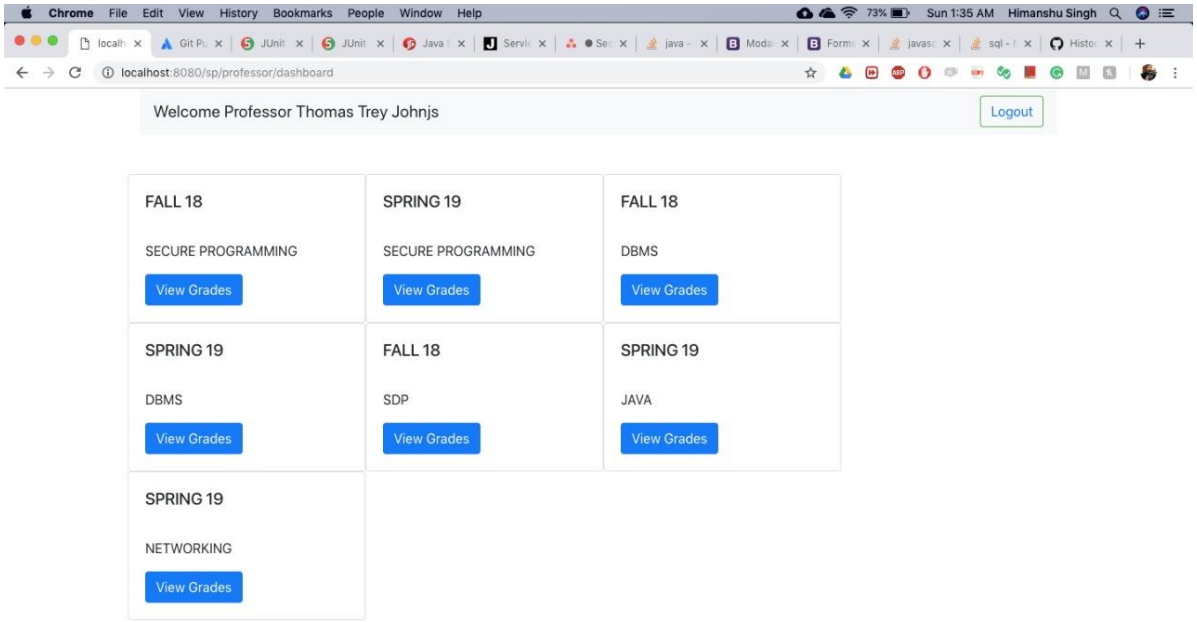


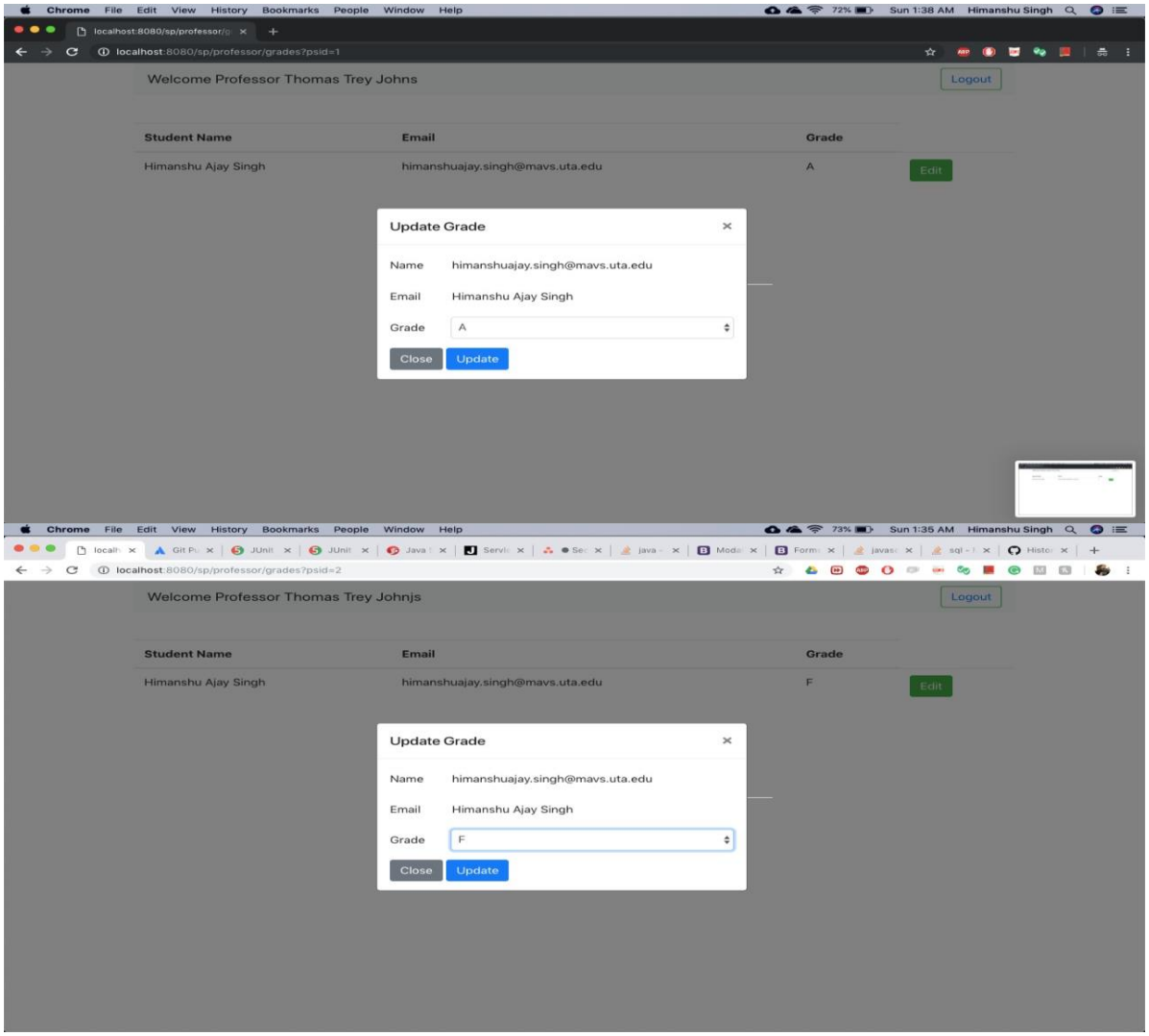


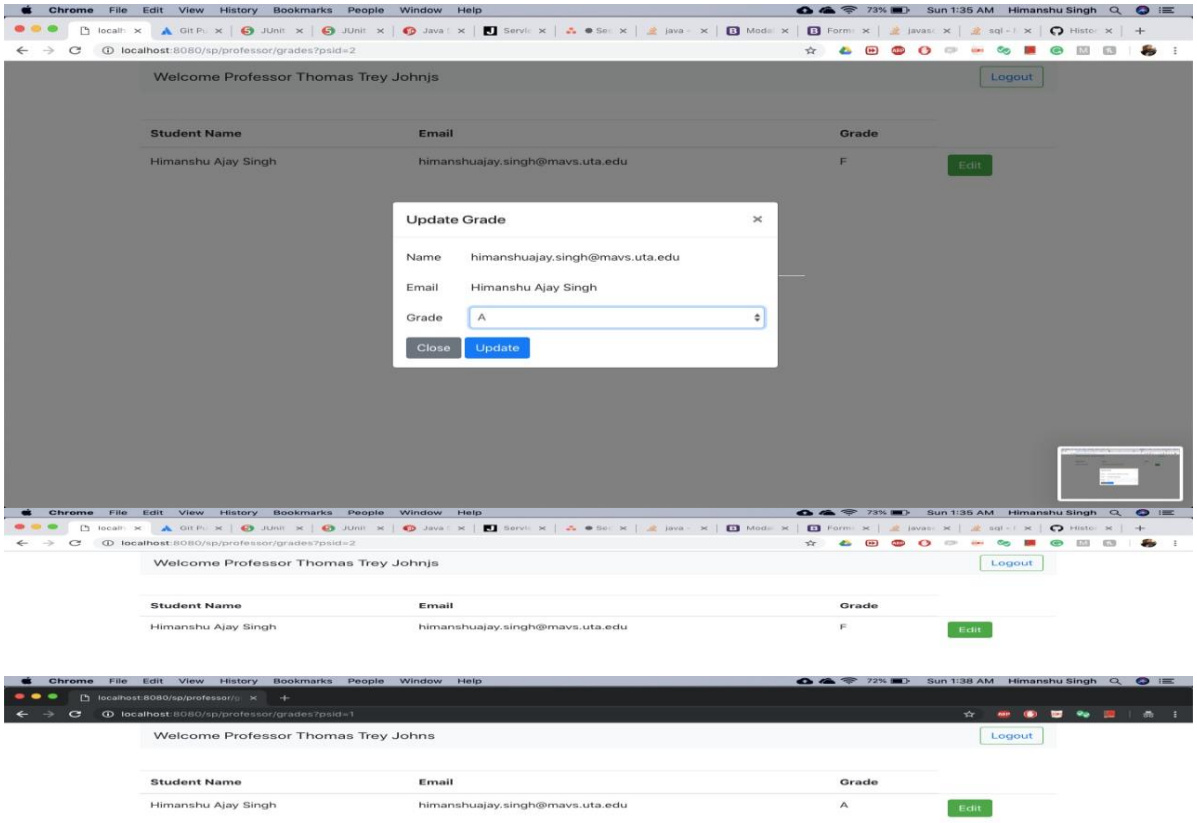
STUDENT DASHBOARD:



PROFESSOR DASHBOARD:

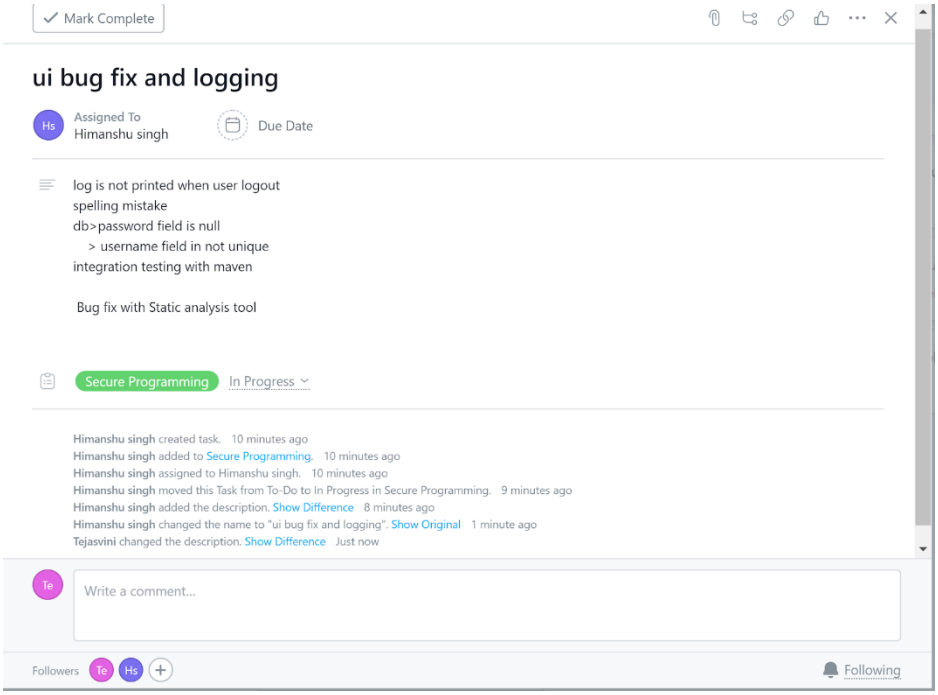




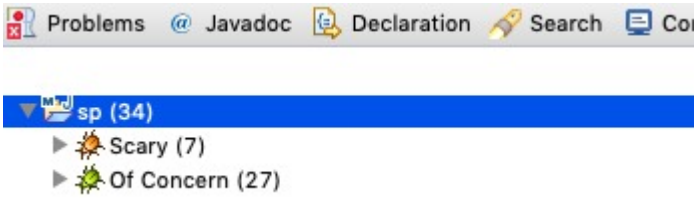


Software Version Description

- Final product release version: There is only one branch which is the master branch.
- List of issues resolved since prior release
  The links for the following tasks is as shown below:
  https://app.asana.com/0/939282246374097/939664138627054/f



- List of known issues that weren't resolved
  There is a total of 27 issues generated by FindBugs which are 'of-concern', which are not fixed



- Dependencies we have used for the application to operate and their version numbers
  - Version Control: GitHub
  - IDE: Eclipse ide as it supports Java EE and its open source
  - Database: MySQL (relational and open source)
  - Compilers: Java 1.8
  - Static analysis tool: find bug
  - Logging: log4j
  - Programming language: java 1.8, Html, JavaScript
  - Application server: Apache Tomcat 8
  - Technology: Java EE (servlets and JSP)
  - Frameworks: Bootstrap 4.1.3, Apache Standard Taglibs 1.2.3 (for avoiding script lets)

Deliverables from part 1

The report of project part 1 is in Zip folder and is updated accordingly.

Static code analysis evidence

- Manual code review
  Static Code Analysis Evidence including the following:

File name & line number	Defect Description	Disposition	Fix by
ProfessorController, Line 46	Type casting using new Integer object, used parseInt	Fixed	Himanshu Sing
ProfessorDao, Line 38	Prepared statement object was not closed	Fixed	Tejasvini Patil

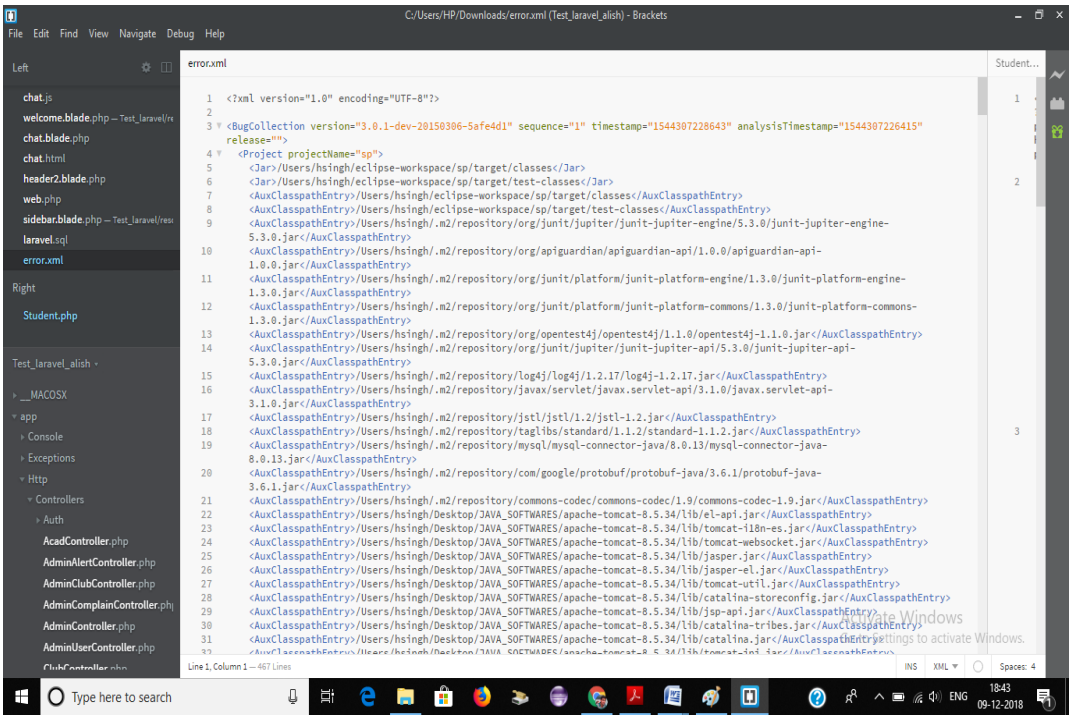
Snippet from ProfessorController.java:

- Automated

Bug: Null passed for non-null parameter of resultSetToArrayList(ResultSet) in com.uta.sp.dao.JdbcConnection.getMany(String) This method call passes a null value for a non-null method parameter. Either the parameter is annotated as a parameter that should always be non-null, or analysis has shown that it will always be dereferenced. Rank: Scary (8), confidence: Normal Pattern: NP_NULL_PARAM_DEREF Type: NP, Category: CORRECTNESS (Correctness)	fixed
Bug: HTTP parameter directly written to HTTP header output in com.uta.sp.controller.ProfessorContrller.doPost(HttpServletRequest, HttpServletResponse)	fixed
Bug: HTTP parameter directly written to HTTP header output in com.uta.sp.controller.ProfessorContrller.doPost(HttpServletRequest, HttpServletResponse)	fixed
Bug: Unresolvable reference to javax.crypto.SecretKey by com.uta.sp.dao.JdbcConnectionTest	In the test file so no need to change
Bug: Unresolvable reference to javax.crypto.SecretKey by com.uta.sp.dao.JdbcConnectionTest	In the test file so no

	need to change
Unresolvable reference to javax.crypto.SecretKey by com.uta.sp.dao.JdbcConnectionTest	In the test file so no need to change

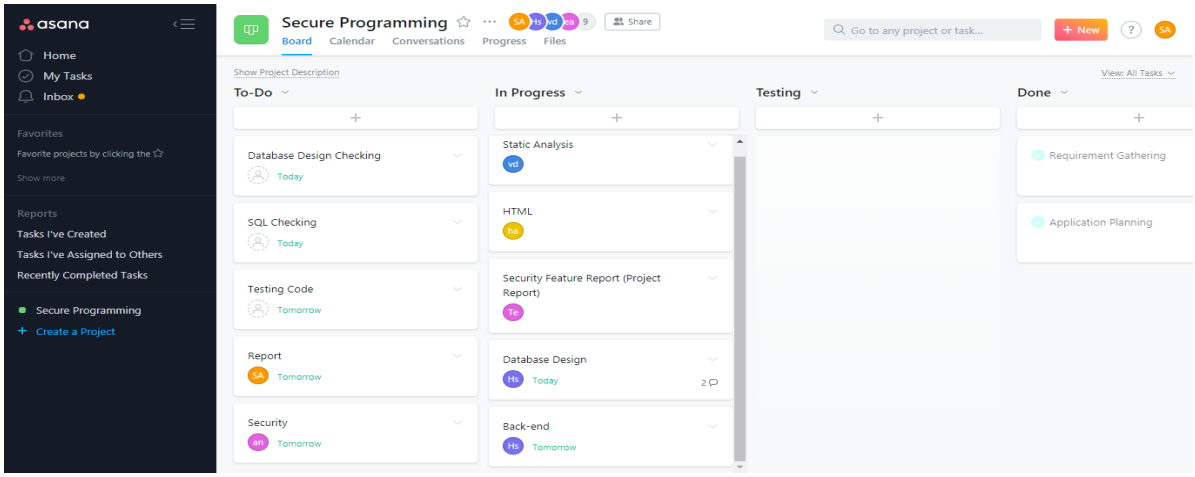
The following figure is the screenshot of **error.xml** file, it is a vulnerability report of the system. The report is generated using **FindBugs** vulnerability tool.



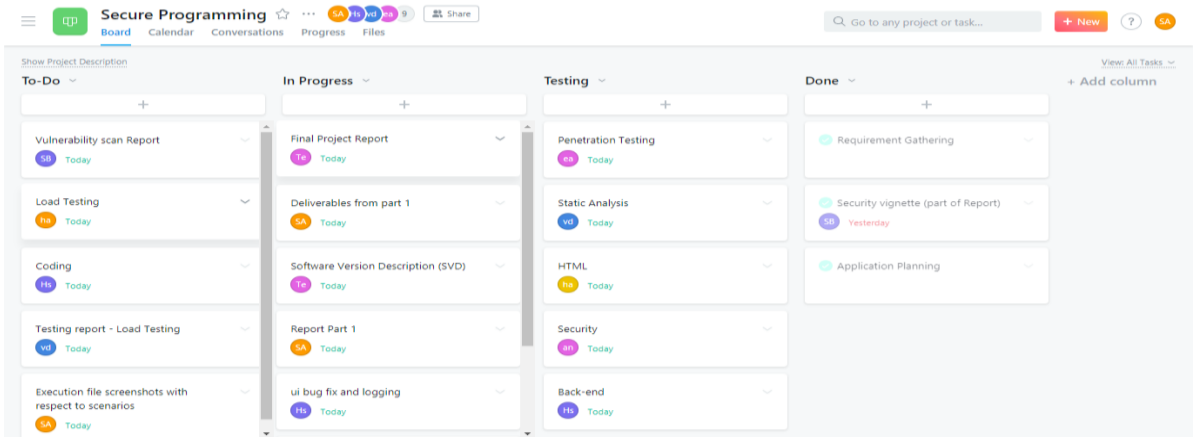
The error.xml file is included in the zip file.

Project management

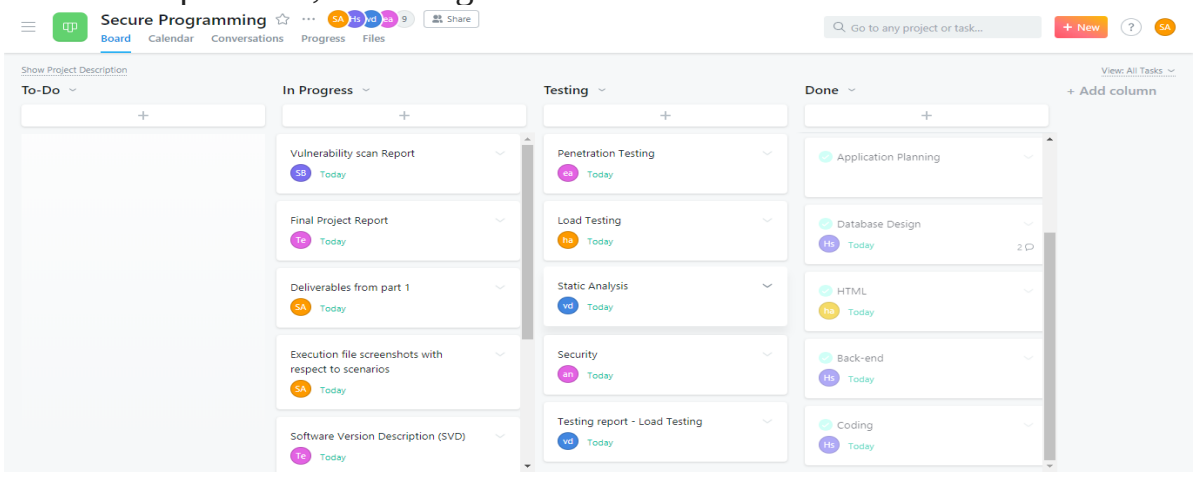
Asana is use in project management part. It is an application designed to help teams organize, track, and manage their work. Below are the screenshots of the dashboard. As the project was scheduled, task was assigned and was completed on accordingly to the time assigned.



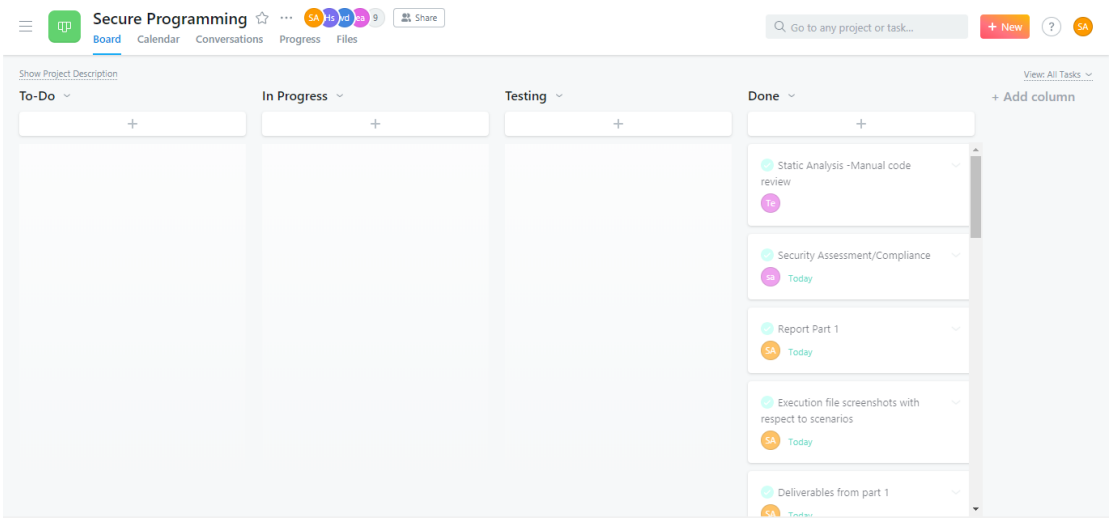
Different column were made like To-Do, In Progress, Testing and Done.



As the task proceeds, the task goes to different columns.



In last, task completes the owner mark it as complete.



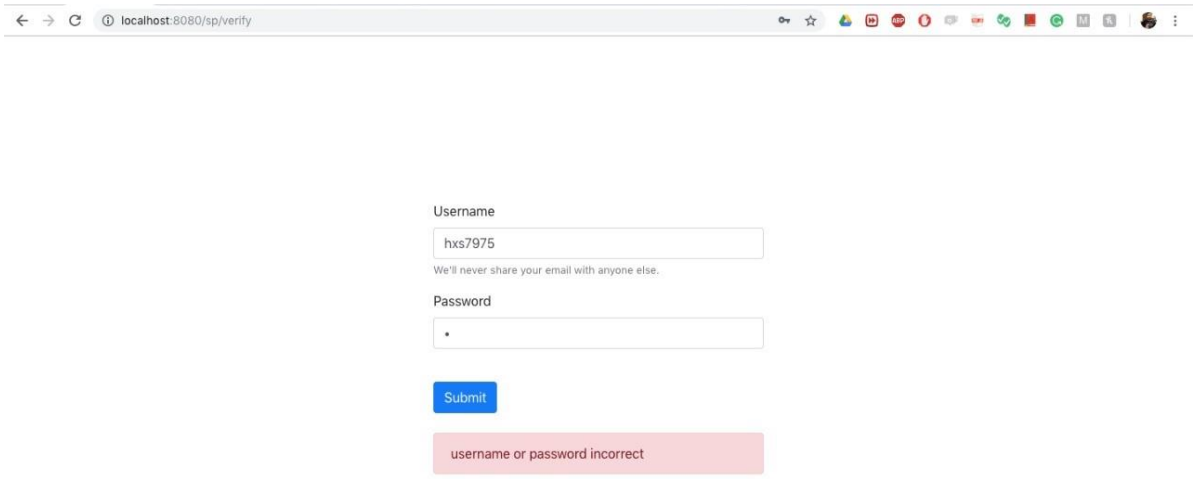
Testing report

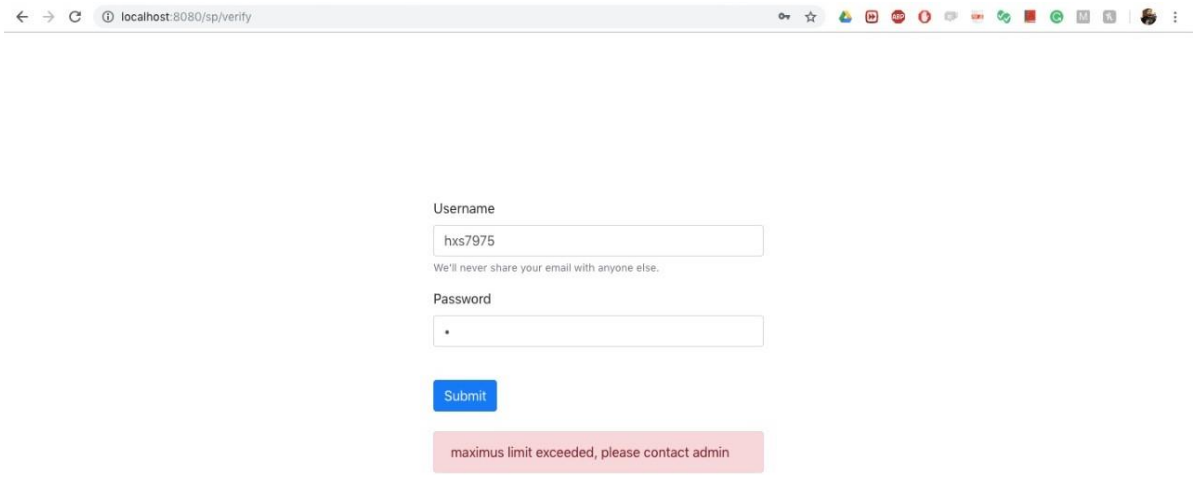
- Security testing result

  We have done all component testing of the system and here as a measure of security, we have included security snippet codes. For example, One of the most important part of the system as a user and as a security perspective is entering the system (signing in to the system).For that we have applied regular expression check on the input given by the user .

```
protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    HttpSession session = request.getSession();
    if (session.getAttribute(Constants.SP_USERNAME) != null
            && Constants.SP_USERTYPE_PROFESSOR.equals(session.getAttribute(Constants.SP_USERTYPE))) {
        ProfessorDao dao = new ProfessorDao();
        Integer profID = new Integer(session.getAttribute(Constants.SP_USERID).toString());
        String id = request.getParameter("ssid");
        String psid = request.getParameter("psid");
        String grade = request.getParameter("grade");
        if(id.matches("^[0-9]+$") &&
                grade.matches("^[A-F]{1}$")) {
            int ssid = Integer.parseInt(id);
            new UserDao().updateGrade(ssid, profID, grade);
            LOG.info("grade updated for studentid "+ssid+" by professor id "+profID);
            response.sendRedirect("/sp/professor/grades?psid="+psid);

        }

    }
```

  The regular expression check will help in proper input validation. We already did the penetration testing, preventing the brute-force attack and we have the result below in the penetration testing section below. Still, in this section, validation part of the system is shown by the following figures.

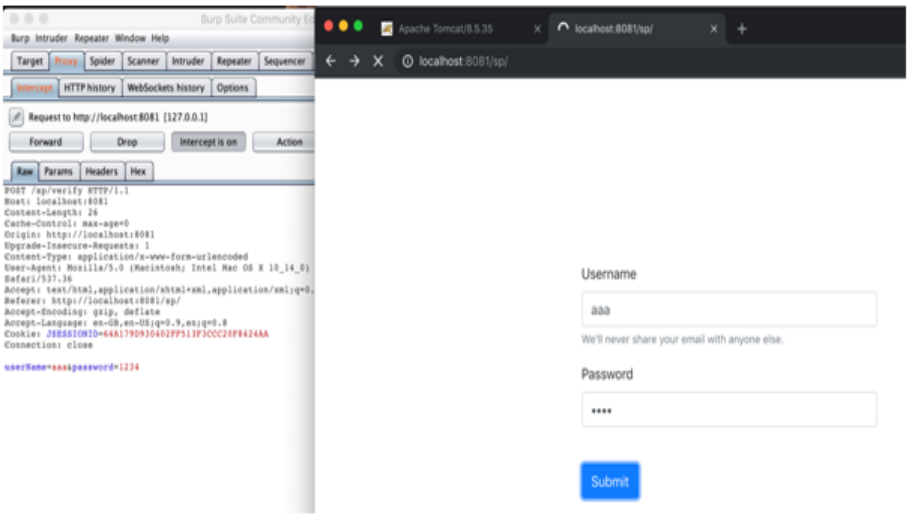




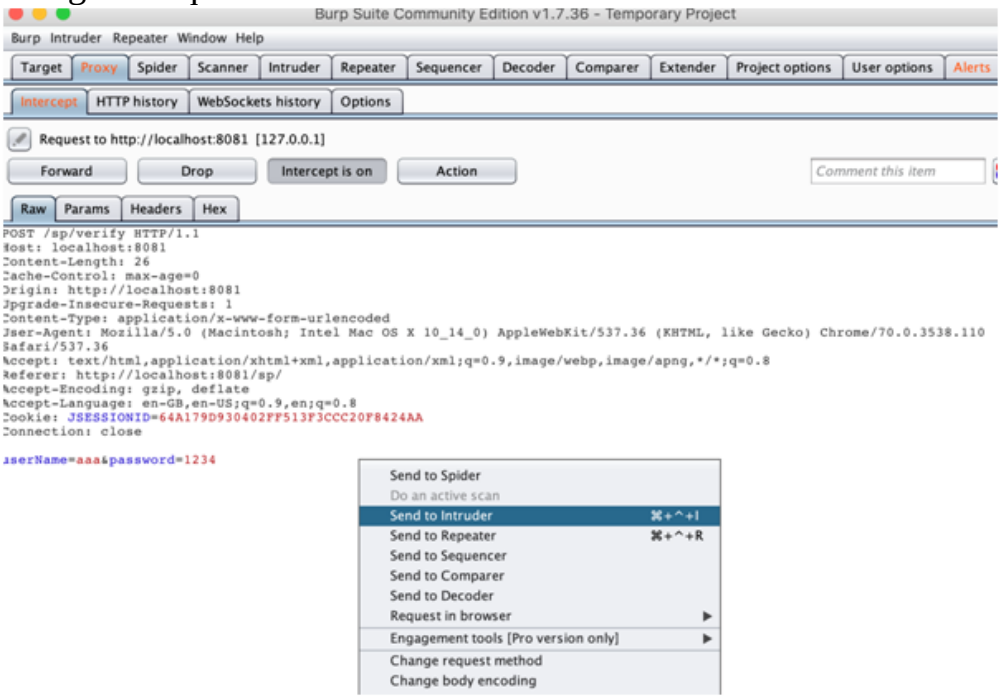
- Penetration testing result

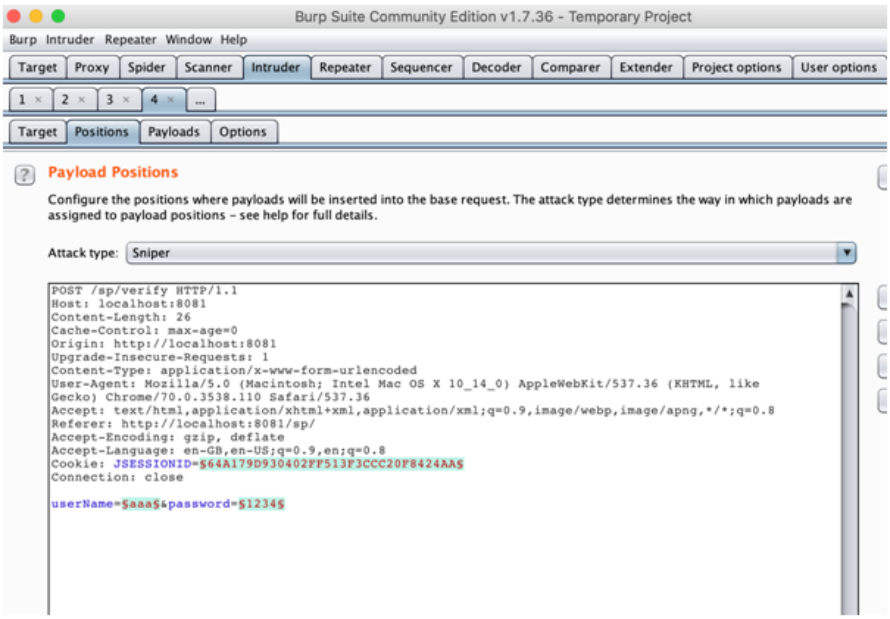
  Penetration Testing using Burp Suite

  With this we test the grade book application for the login protection by carrying out the brute force attack. Turning on the intercept after passing the credentials in our application we see the post request being intercepted. The two values listed are username and password, thus we brute force these values. In the burp suite we make use of the intruder which allows us to edit and manipulate the requests.



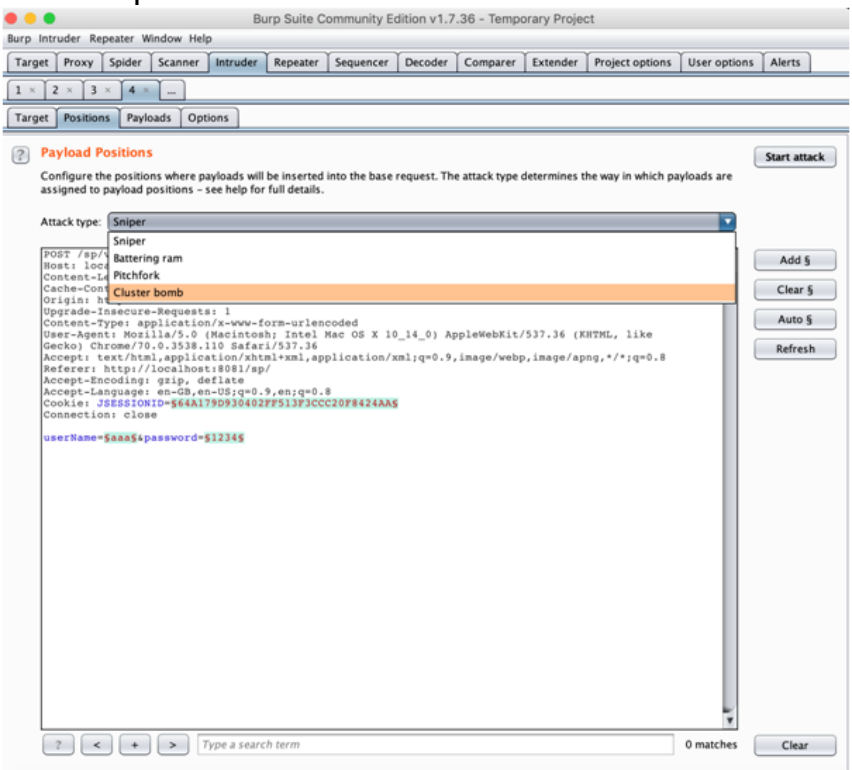
Sending the request to the intruder and hit forward.



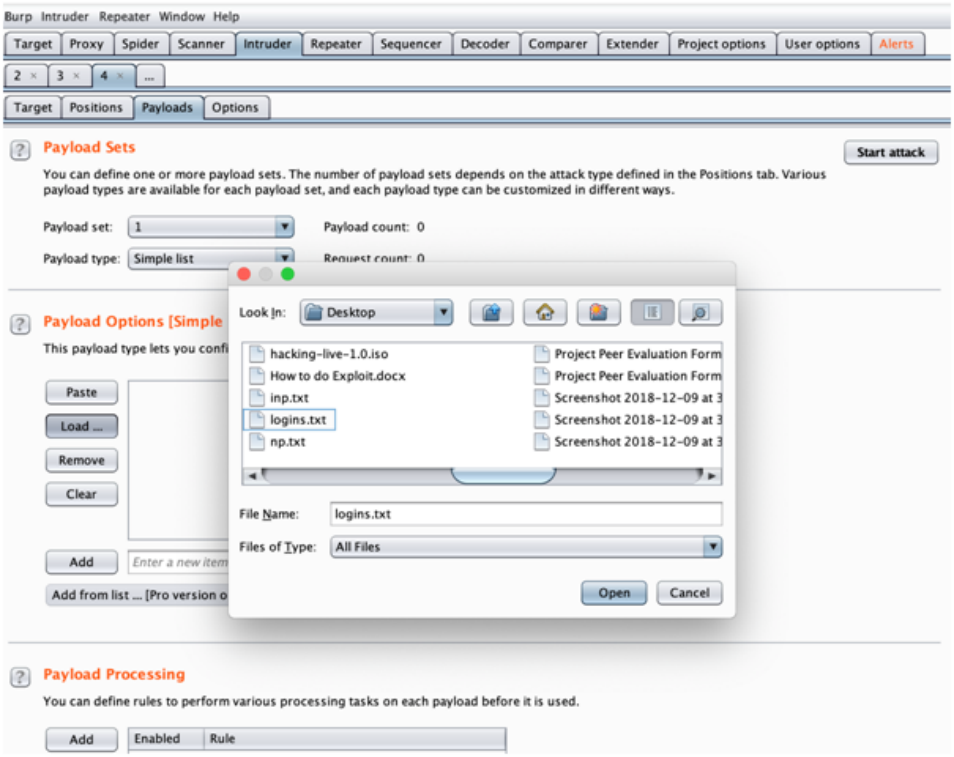


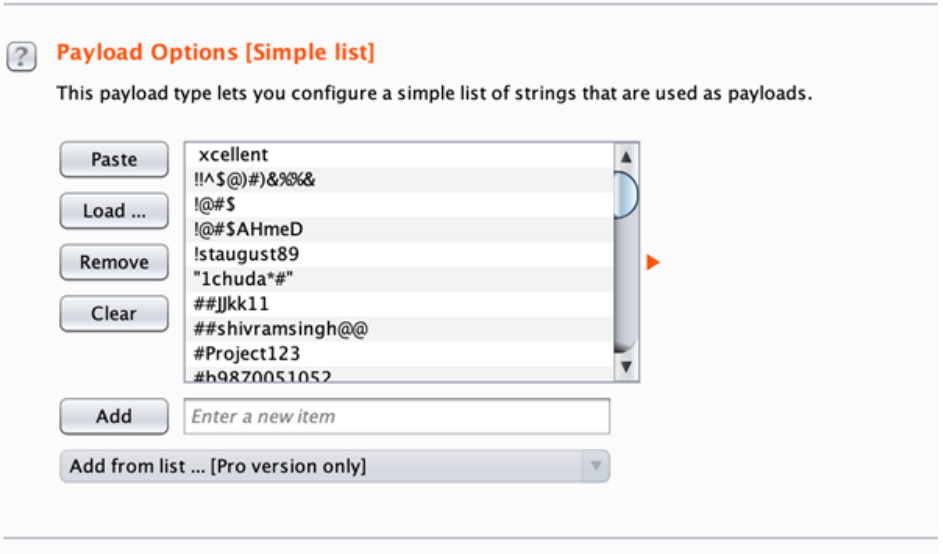
In the positions inside the intruder we see that we get the desired request that we intercepted and also, we see the payloads being highlighted for which we can carry out the brute force.

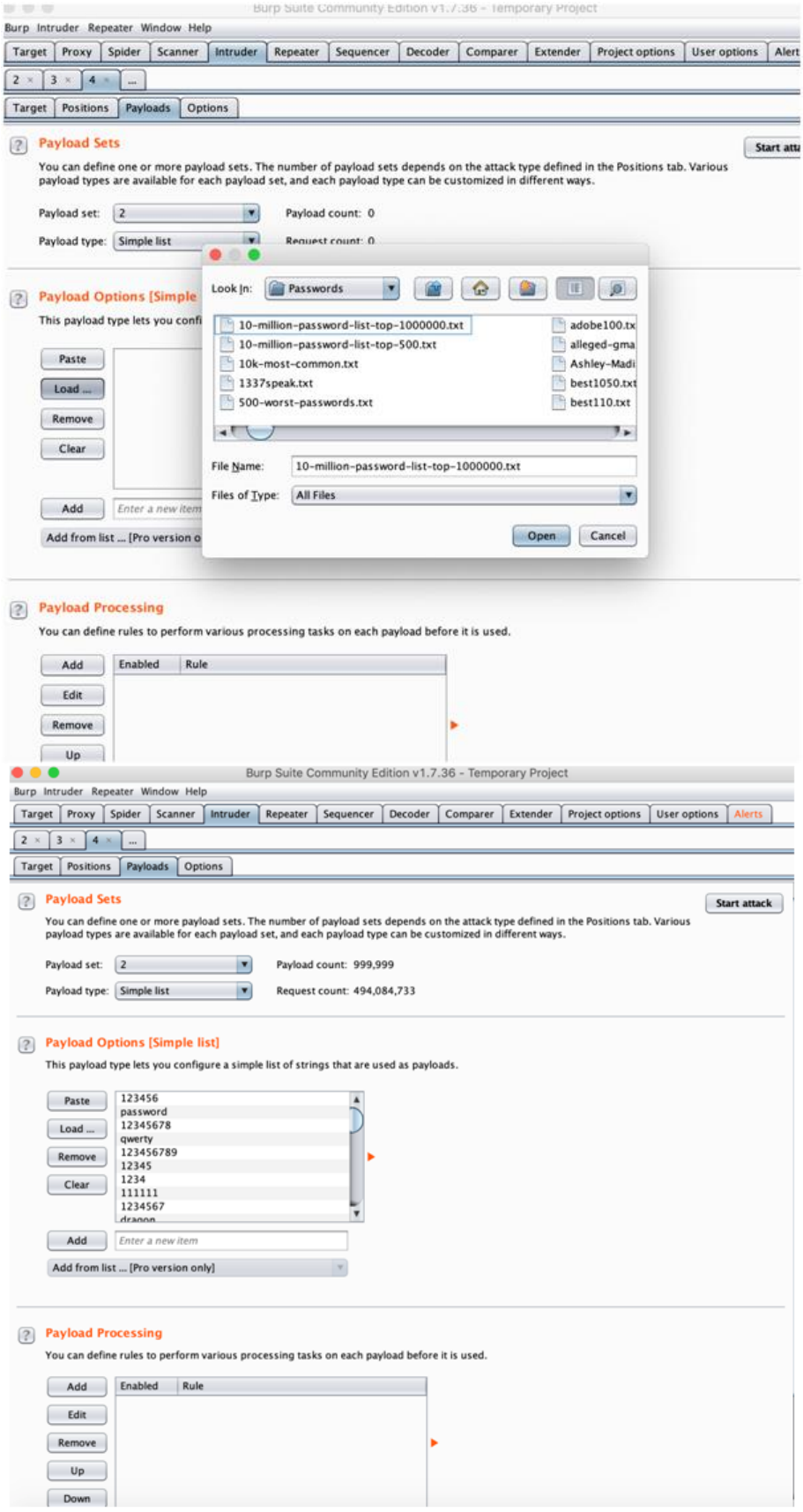
Here we just use the username and password to carry out the attack and we use the cluster bomb attack type as we are brute forcing against two values and these are required in combination.



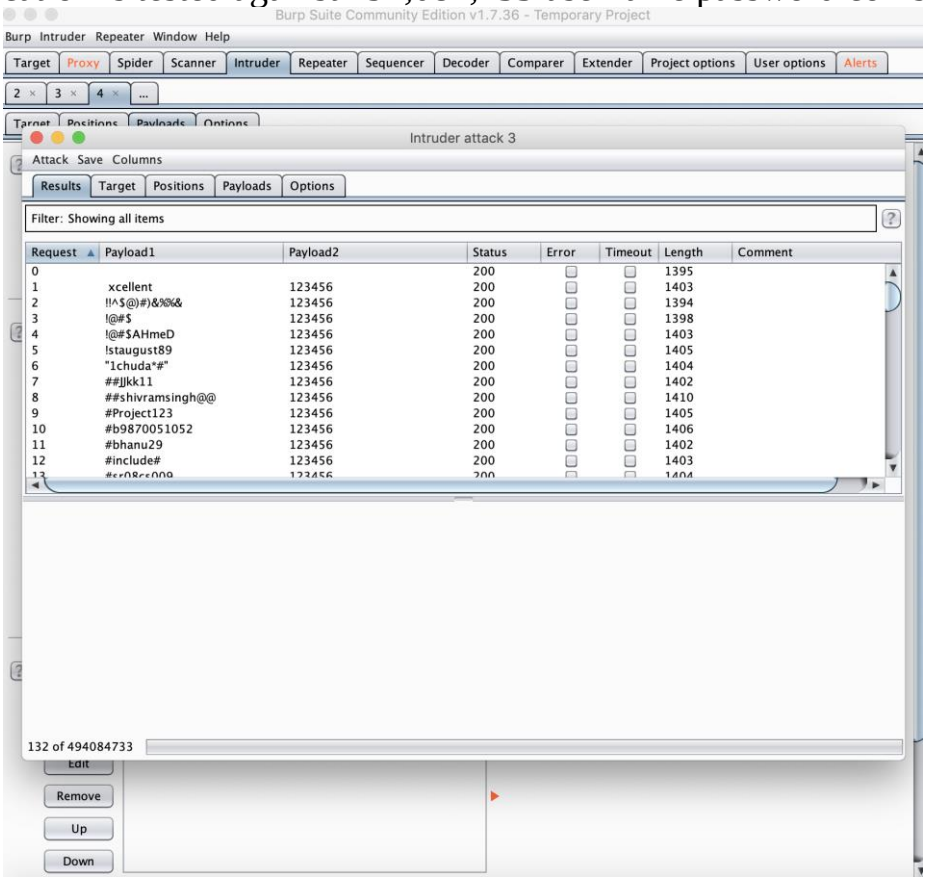
In the payloads we see that we two payloads set and, in the payload, set 1 we add the word list of user ids and the payload 2 for the passwords.


Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
Target Positions Payloads Options
Payload Sets
Start attack
You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.
Payload set: 1
Payload count: 0
Payload type: Simple list
Request count: 0
Payload Options [Simple
This payload type lets you confi
Paste
Load ...
Remove
Clear
Add
Enter a new item
Add from list ... [Pro version o
Look In: Desktop
hacking-live-1.0.iso
How to do Exploit.docx
inp.txt
logins.txt
np.txt
Project Peer Evaluation Form
Project Peer Evaluation Form
Screenshot 2018-12-09 at 3
Screenshot 2018-12-09 at 3
Screenshot 2018-12-09 at 3
File Name: logins.txt
Files of Type: All Files
Open
Cancel
Payload Processing
You can define rules to perform various processing tasks on each payload before it is used.
Add
Enabled
Rule

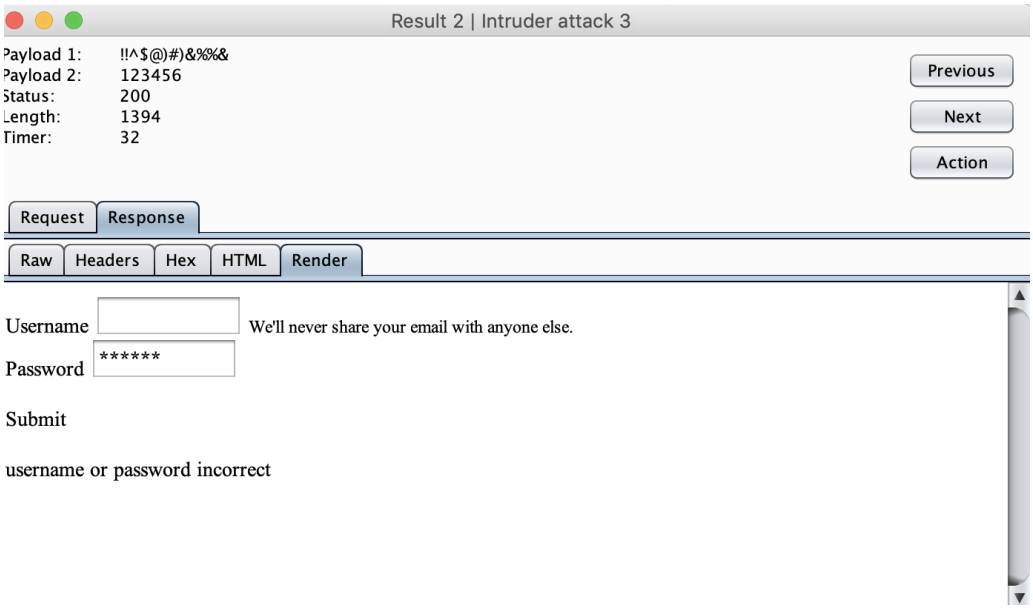

Payload Options [Simple list]
This payload type lets you configure a simple list of strings that are used as payloads.
Paste
Load ...
Remove
Clear
xcellent
!!^$@)#)&%%&
!@#$
!@#$AHmeD
!staugust89
"1chuda*#"
##Jjkk11
##shivramsingh@@
#Project123
#b9870051052
Add
Enter a new item
Add from list ... [Pro version only]

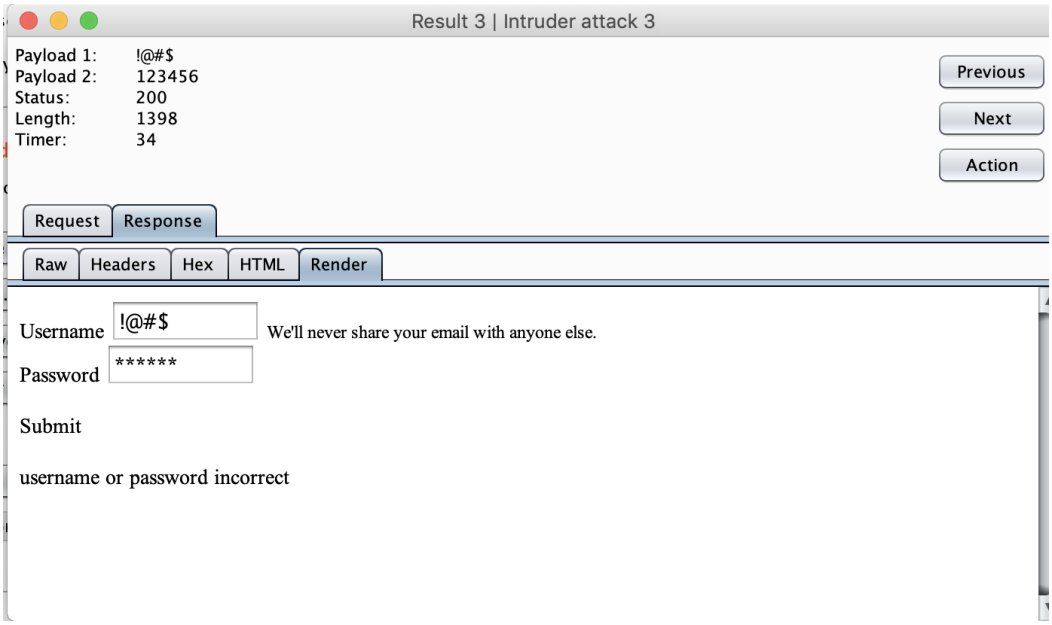

Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alert
Target Positions Payloads Options
Payload Sets
You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.
Payload set: 2
Payload count: 0
Payload type: Simple list
Payload Options [Simple
This payload type lets you confi
Paste
Load ...
Remove
Clear
Add
Enter a new item
Add from list ... [Pro version o
Look In: Passwords
10-million-password-list-top-1000000.txt
10-million-password-list-top-500.txt
10k-most-common.txt
1337speak.txt
500-worst-passwords.txt
adobe100.tx
alleged-gma
Ashley-Madi
best1050.txt
best110.txt
File Name: 10-million-password-list-top-1000000.txt
Files of Type: All Files
Open
Cancel
Payload Processing
You can define rules to perform various processing tasks on each payload before it is used.
Add
Edit
Remove
Up
Enabled
Rule
Burp Suite Community Edition v1.7.36 - Temporary Project
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
Target Positions Payloads Options
Payload Sets
Start attack
You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.
Payload set: 2
Payload count: 999,999
Payload type: Simple list
Request count: 494,084,733
Payload Options [Simple list]
This payload type lets you configure a simple list of strings that are used as payloads.
Paste
Load ...
Remove
Clear
123456
password
12345678
qwerty
123456789
12345
1234
111111
1234567
dragon
Add
Enter a new item
Add from list ... [Pro version only]
Payload Processing
You can define rules to perform various processing tasks on each payload before it is used.
Add
Edit
Remove
Up
Down
Enabled
Rule

Now we start the attack and it will go through all the combinations, the application is tested against 494,084,733 username password combinations.



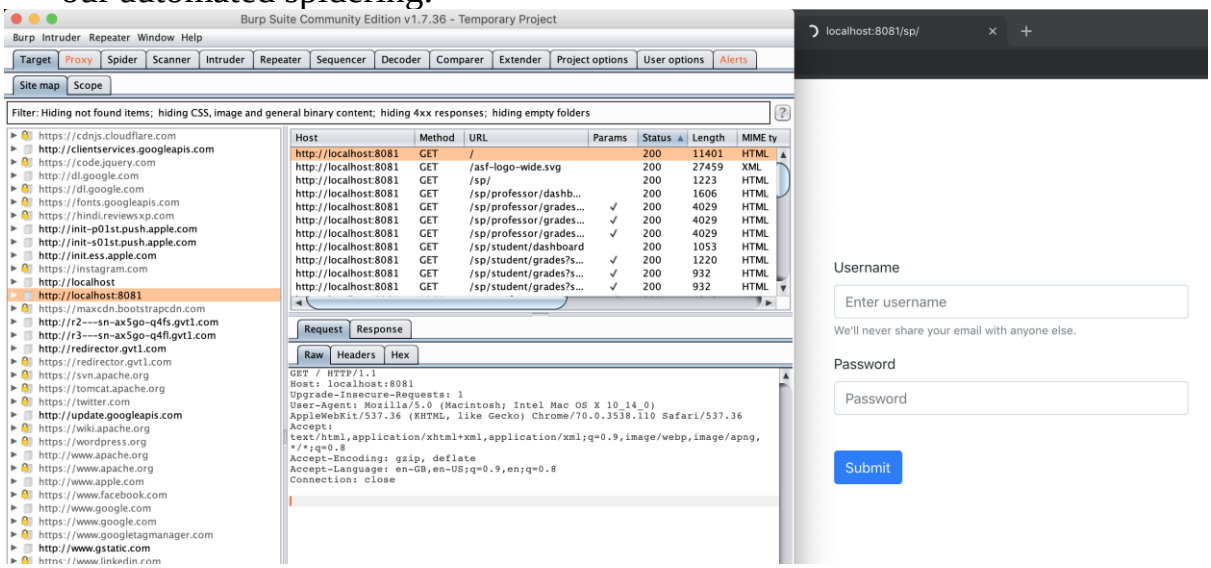
When we scan the combinations for different lengths and check the response and in render, we see that the potential combinations fail for our application. Thus, making our application stringent against the login attacks.

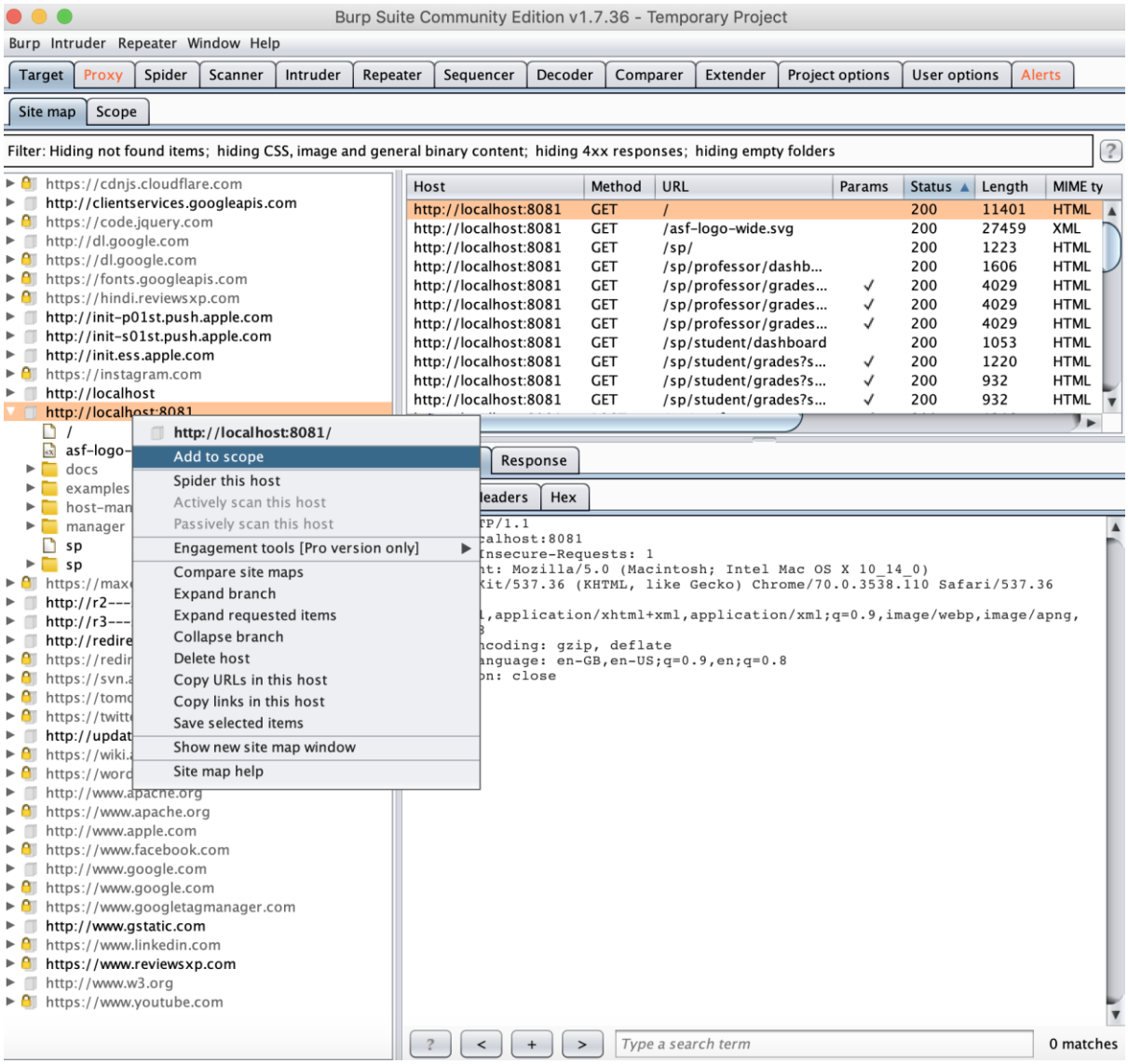




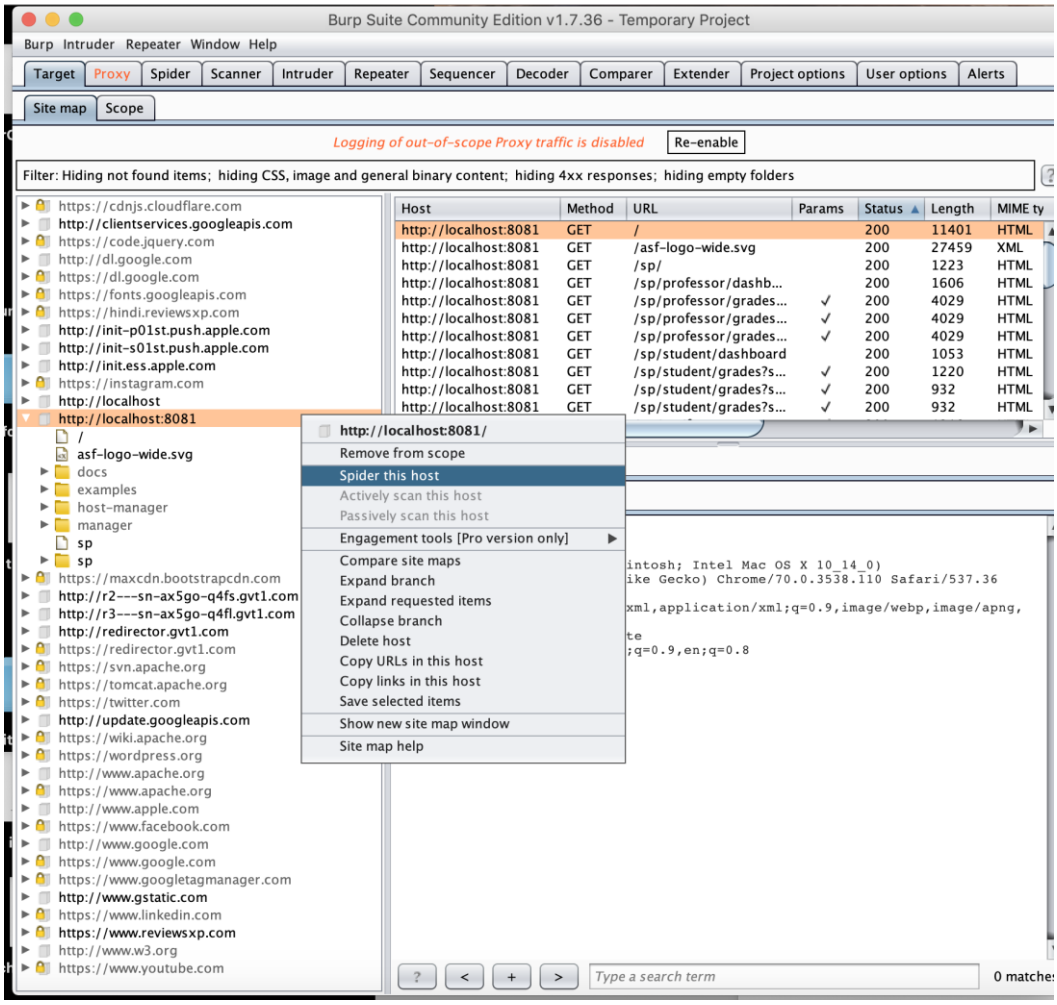
Target and Spidering

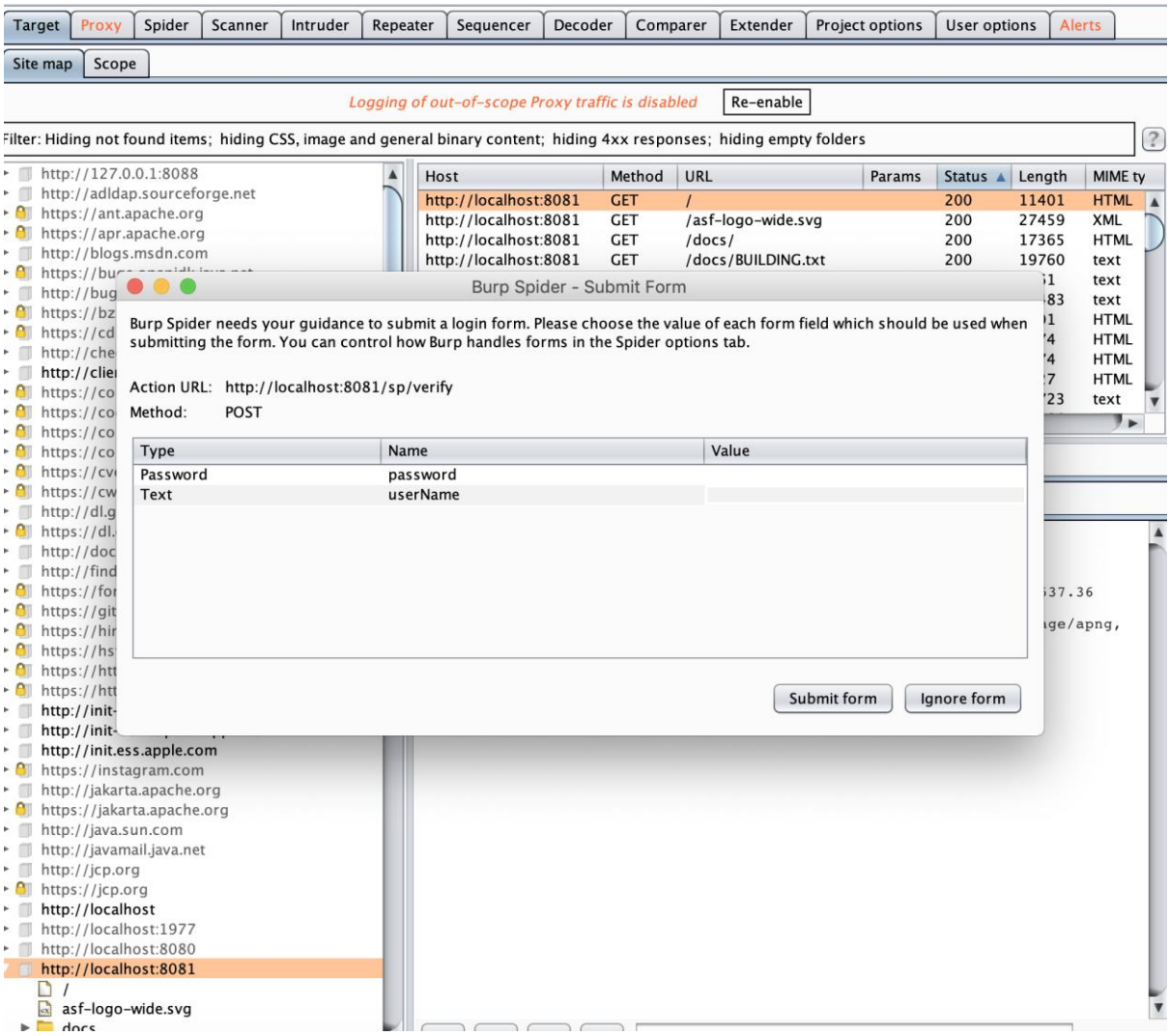
Here we have our target at http://localhost:8081 in our site map which is the structure of the web page and we add it to the scope which allows us to define our automated spidering.

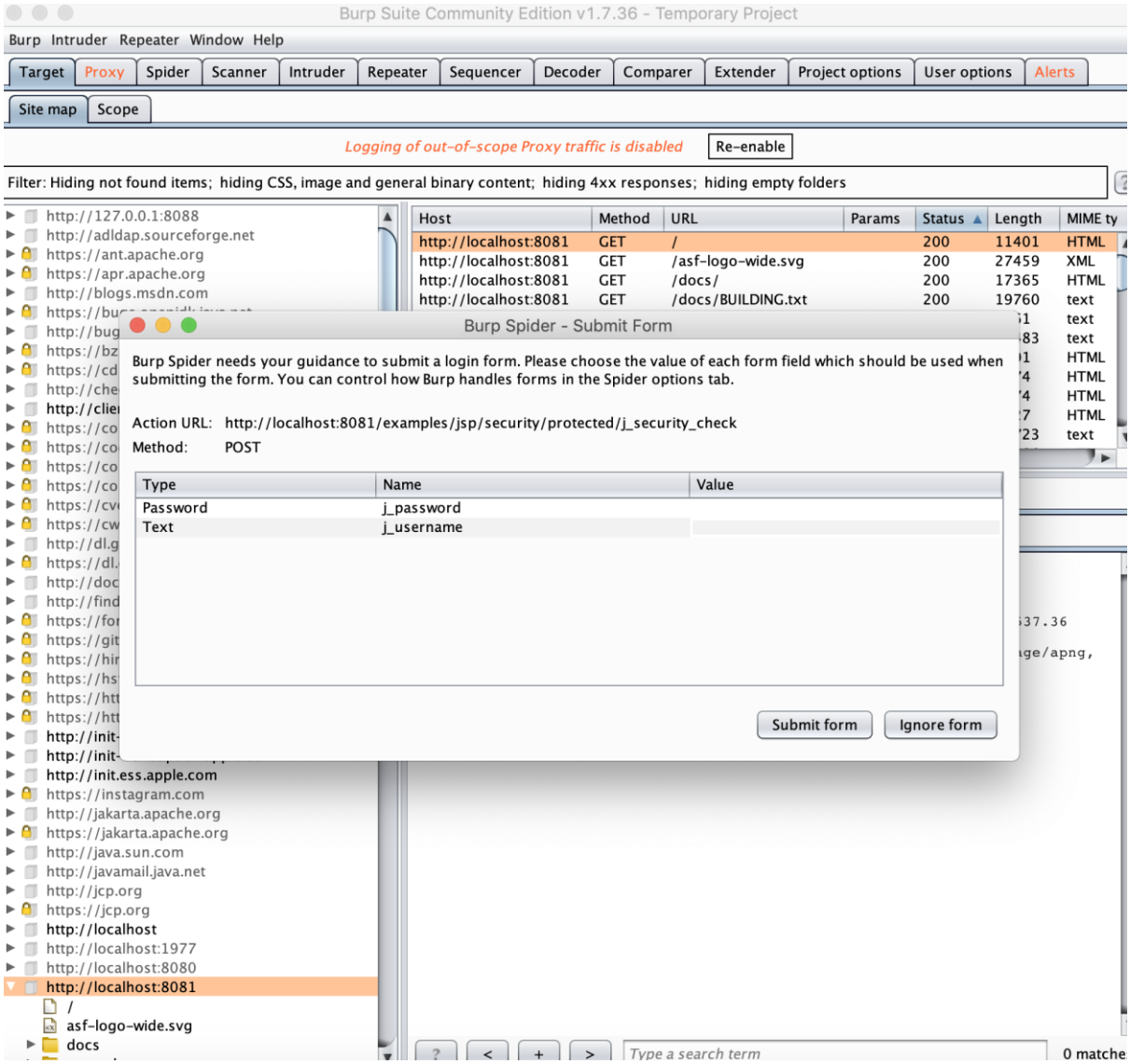




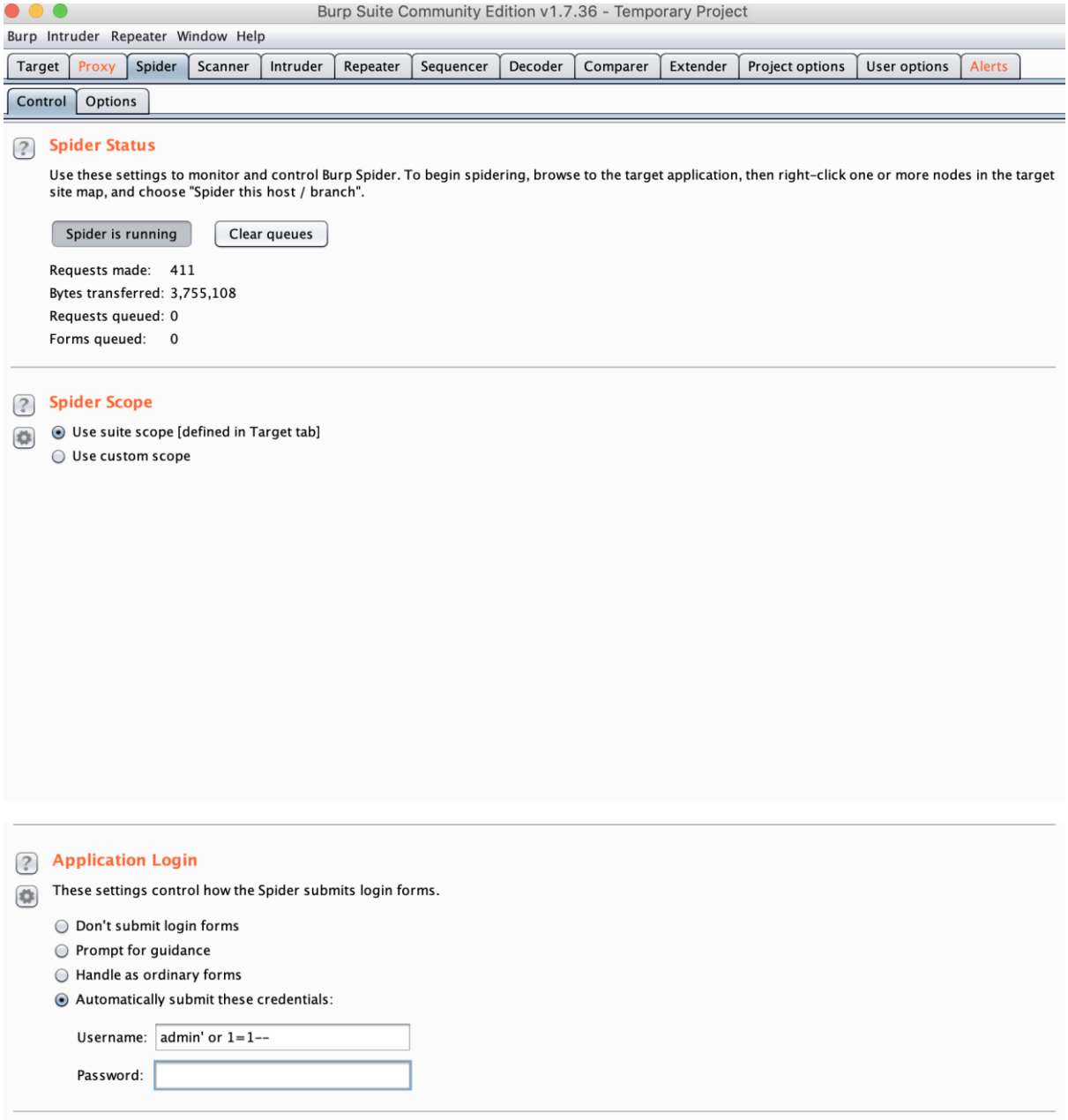
Spidering allows crawling through the website getting us idea how it is structured and how we can break through it. Thus, now we spider this host.


Burp Suite Community Edition v1.7.36 - Temporary Project
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
Site map Scope
Logging of out-of-scope Proxy traffic is disabled
Re-enable
Filter: Hiding not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders
Host Method URL Params Status Length MIME ty
http://localhost:8081 GET / 200 11401 HTML
http://localhost:8081 GET /asf-logo-wide.svg 200 27459 XML
http://localhost:8081 GET /sp/ 200 1223 HTML
http://localhost:8081 GET /sp/professor/dashb... 200 1606 HTML
http://localhost:8081 GET /sp/professor/grades... ✓ 200 4029 HTML
http://localhost:8081 GET /sp/professor/grades... ✓ 200 4029 HTML
http://localhost:8081 GET /sp/professor/grades... ✓ 200 4029 HTML
http://localhost:8081 GET /sp/student/dashboard 200 1053 HTML
http://localhost:8081 GET /sp/student/grades?s... ✓ 200 1220 HTML
http://localhost:8081 GET /sp/student/grades?s... ✓ 200 932 HTML
http://localhost:8081 GET /sp/student/grades?s... ✓ 200 932 HTML
https://cdnjs.cloudflare.com
http://clientservices.googleapis.com
https://code.jquery.com
http://dl.google.com
https://dl.google.com
https://fonts.googleapis.com
https://hindi.reviewsxp.com
http://init-p01st.push.apple.com
http://init-s01st.push.apple.com
http://init.ess.apple.com
https://instagram.com
http://localhost
http://localhost:8081
/
asf-logo-wide.svg
docs
examples
host-manager
manager
sp
sp
https://maxcdn.bootstrapcdn.com
http://r2---sn-ax5go-q4fs.gvt1.com
http://r3---sn-ax5go-q4fl.gvt1.com
http://redirector.gvt1.com
https://redirector.gvt1.com
https://svn.apache.org
https://tomcat.apache.org
https://twitter.com
http://update.googleapis.com
https://wiki.apache.org
https://wordpress.org
http://www.apache.org
https://www.apache.org
http://www.apple.com
https://www.facebook.com
http://www.google.com
https://www.google.com
https://www.googletagmanager.com
http://www.gstatic.com
https://www.linkedin.com
https://www.reviewsxp.com
http://www.w3.org
https://www.youtube.com
http://localhost:8081/
Remove from scope
Spider this host
Actively scan this host
Passively scan this host
Engagement tools [Pro version only]
Compare site maps
Expand branch
Expand requested items
Collapse branch
Delete host
Copy URLs in this host
Copy links in this host
Save selected items
Show new site map window
Site map help
intosh; Intel Mac OS X 10_14_0)
ike Gecko) Chrome/70.0.3538.110 Safari/537.36
xml,application/xml;q=0.9,image/webp,image/apng,
te
;q=0.9,en;q=0.8
Type a search term
0 matches


Target
Proxy
Spider
Scanner
Intruder
Repeater
Sequencer
Decoder
Comparer
Extender
Project options
User options
Alerts
Site map
Scope
Logging of out-of-scope Proxy traffic is disabled
Re-enable
Filter: Hiding not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders
Host
Method
URL
Params
Status
Length
MIME ty
http://localhost:8081 GET / 200 11401 HTML
http://localhost:8081 GET /asf-logo-wide.svg 200 27459 XML
http://localhost:8081 GET /docs/ 200 17365 HTML
http://localhost:8081 GET /docs/BUILDING.txt 200 19760 text
Burp Spider - Submit Form
Burp Spider needs your guidance to submit a login form. Please choose the value of each form field which should be used when submitting the form. You can control how Burp handles forms in the Spider options tab.
Action URL: http://localhost:8081/sp/verify
Method: POST
Type
Name
Value
Password password
Text userName
Submit form
Ignore form
http://init.ess.apple.com
https://instagram.com
http://jakarta.apache.org
https://jakarta.apache.org
http://java.sun.com
http://javamail.java.net
http://jcp.org
https://jcp.org
http://localhost
http://localhost:1977
http://localhost:8080
http://localhost:8081
/
asf-logo-wide.svg
docs

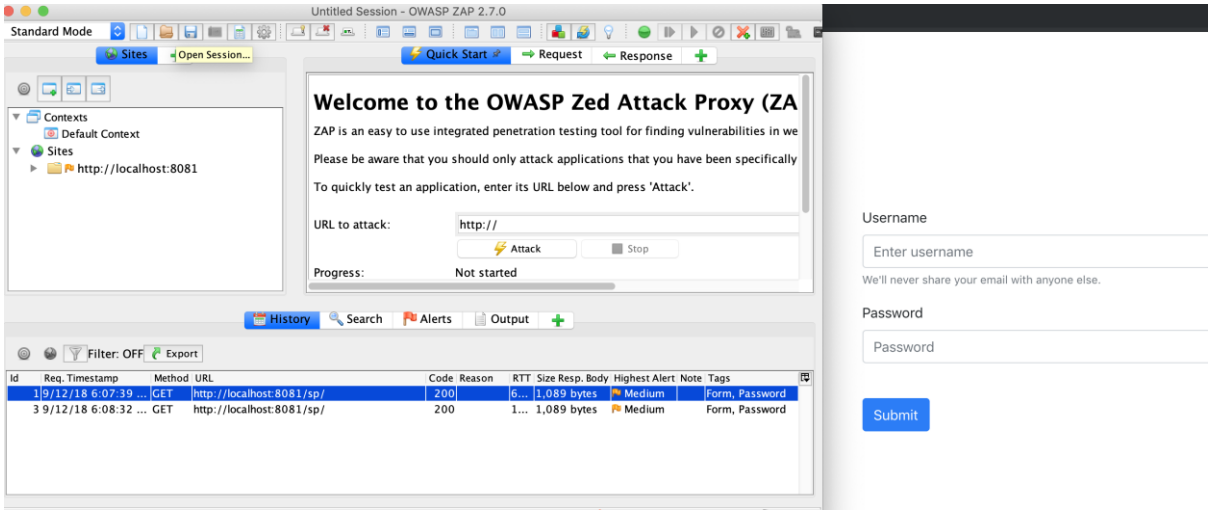


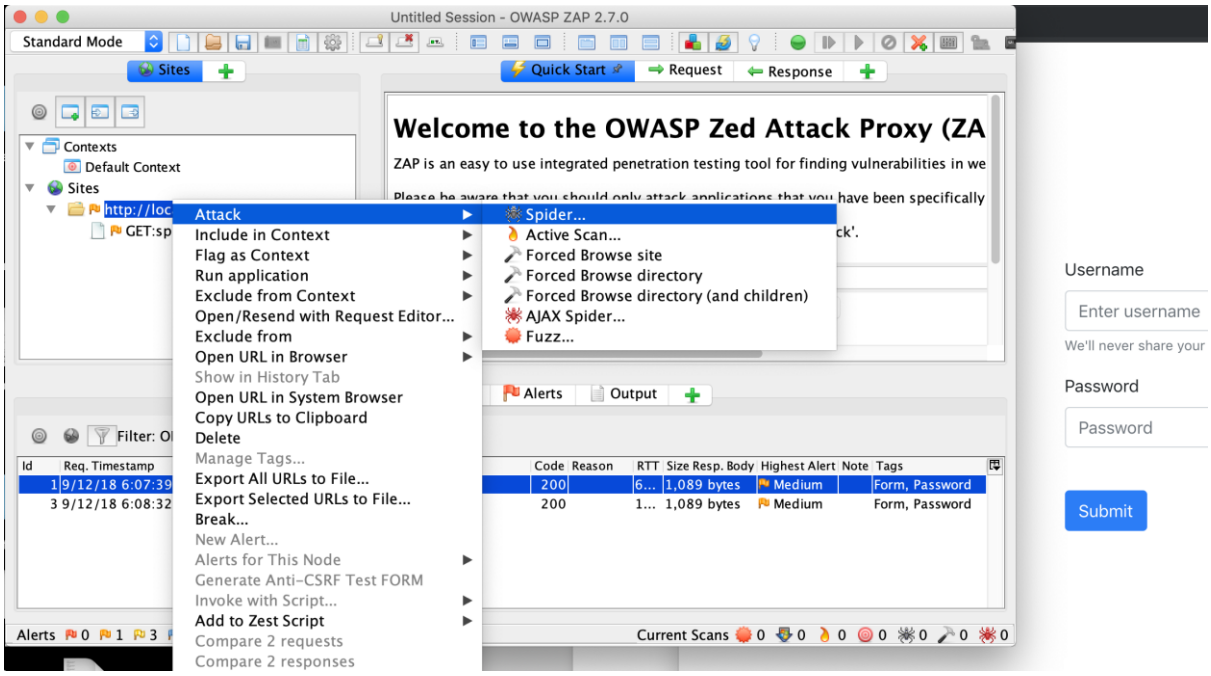
Now going into the spider, we see the requests it made and number of bytes it transferred. We further go into the application login and try to do sql injection and we again perform the spidering

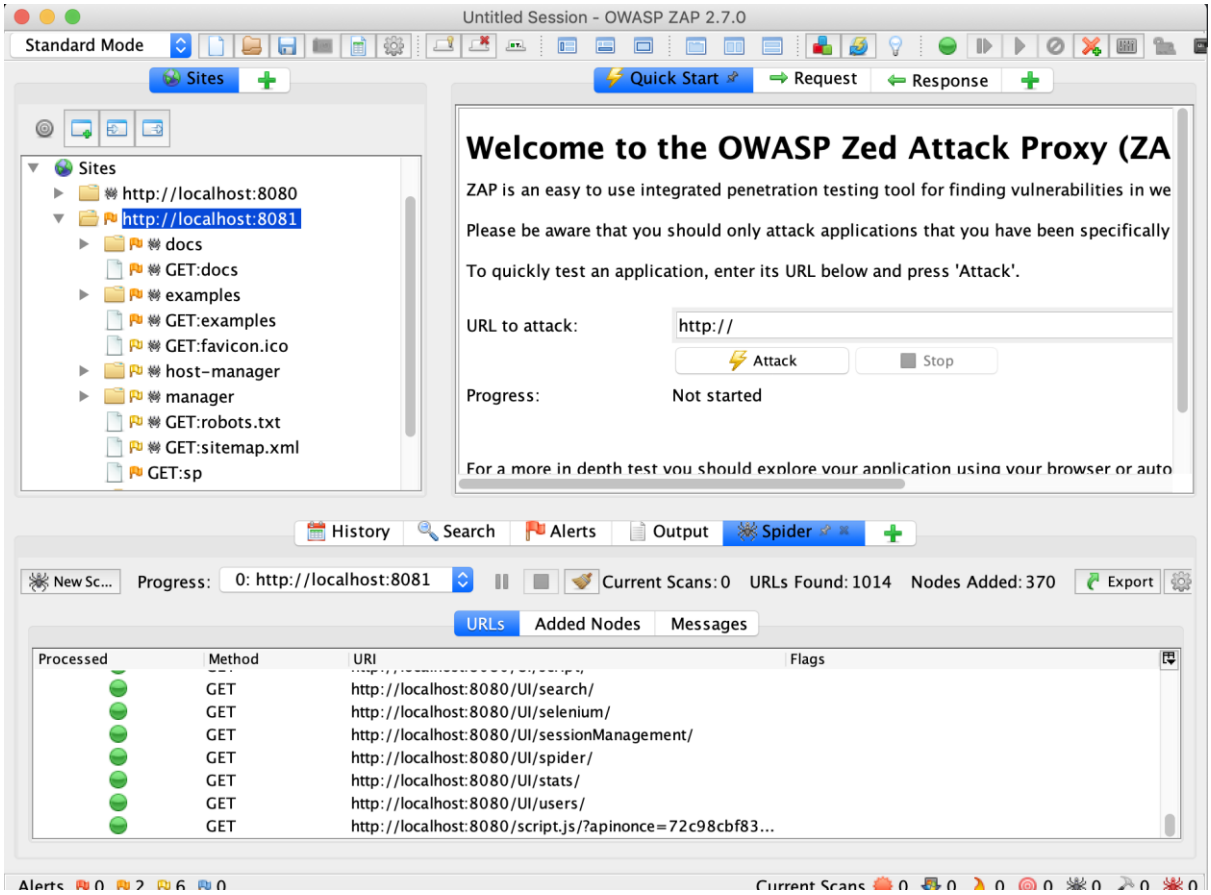


We have the structure of the application

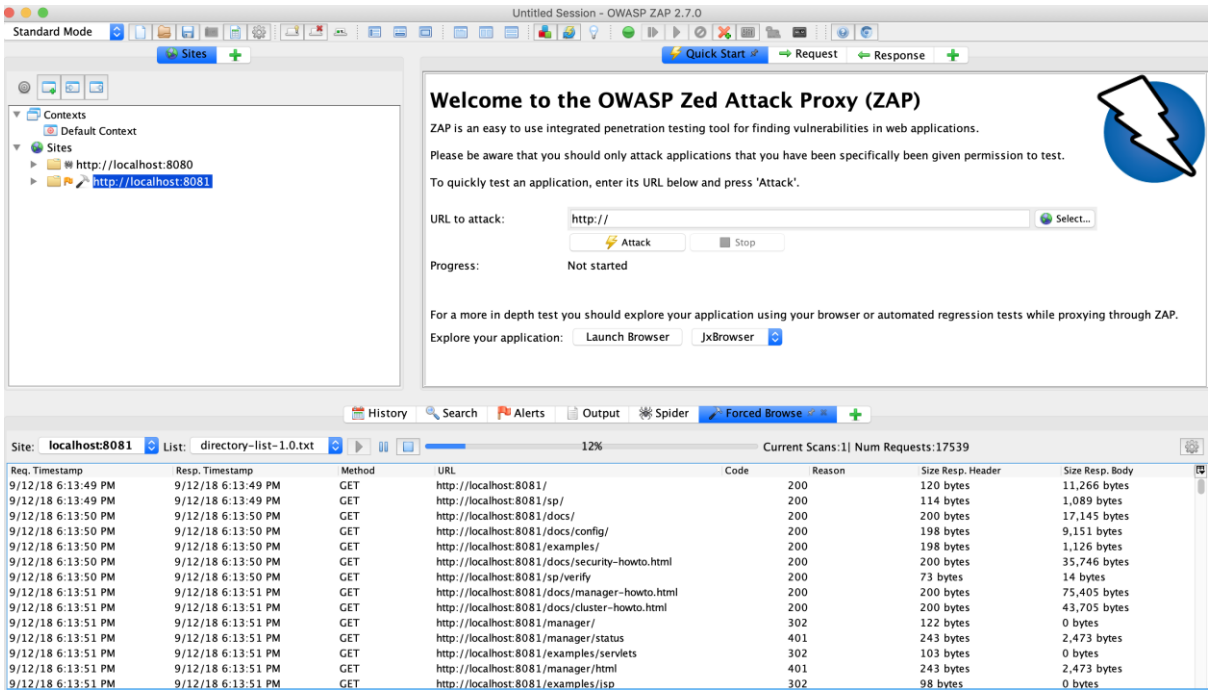
Finding the hidden files using ZAP



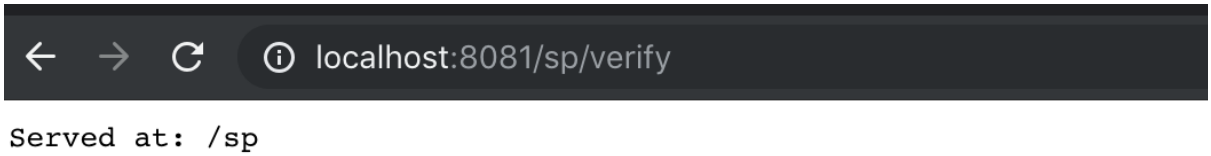




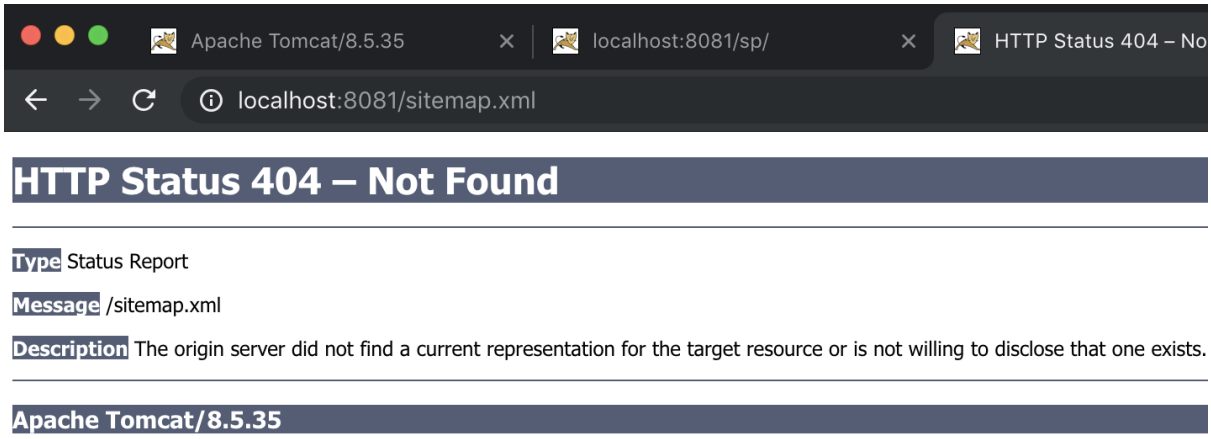
Carrying out the attack using Forced browser directory: -



Accessing the hidden files

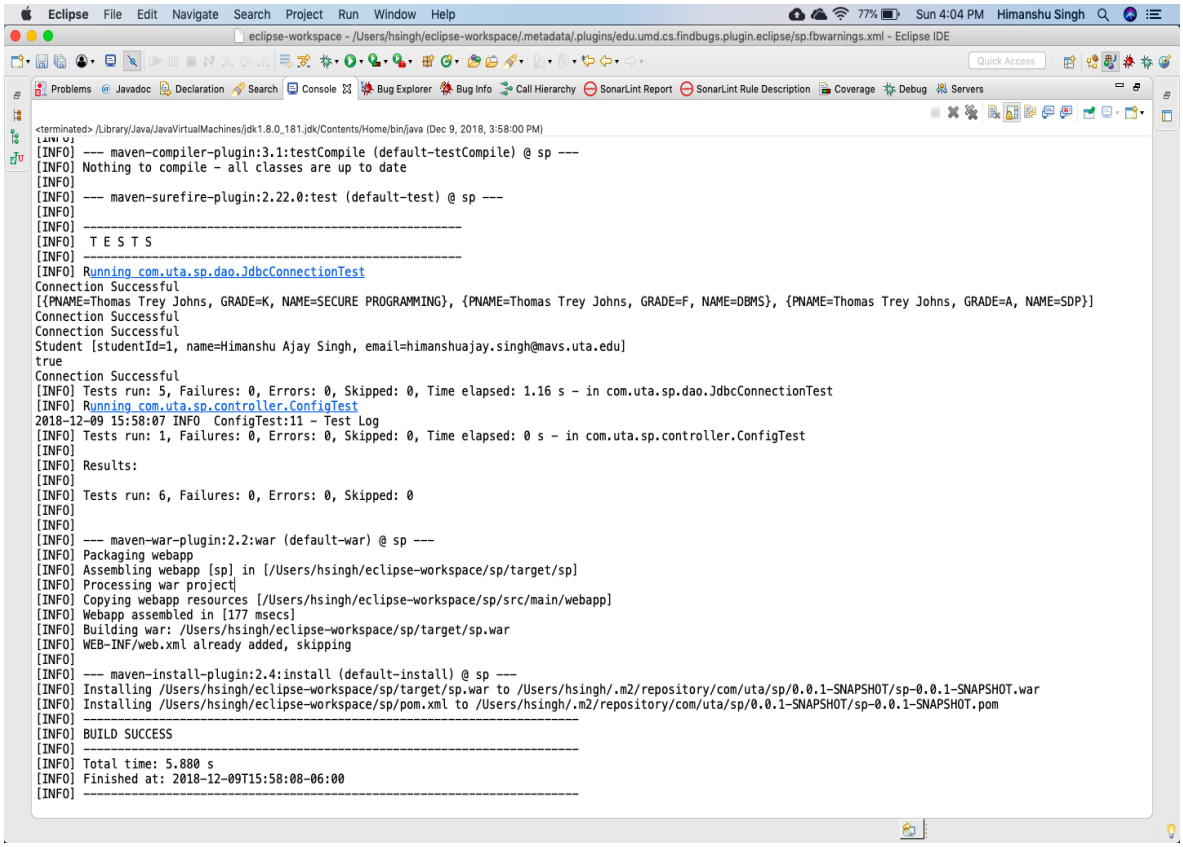


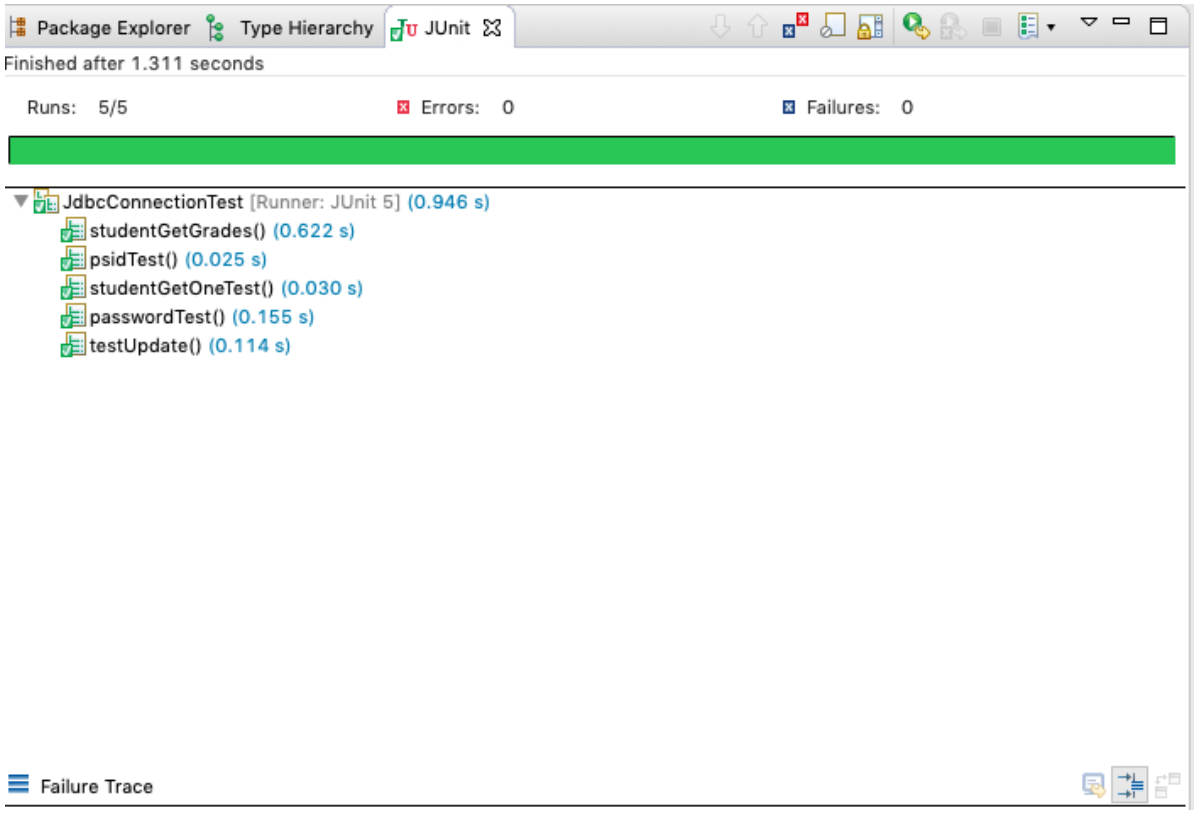
Served at: /sp



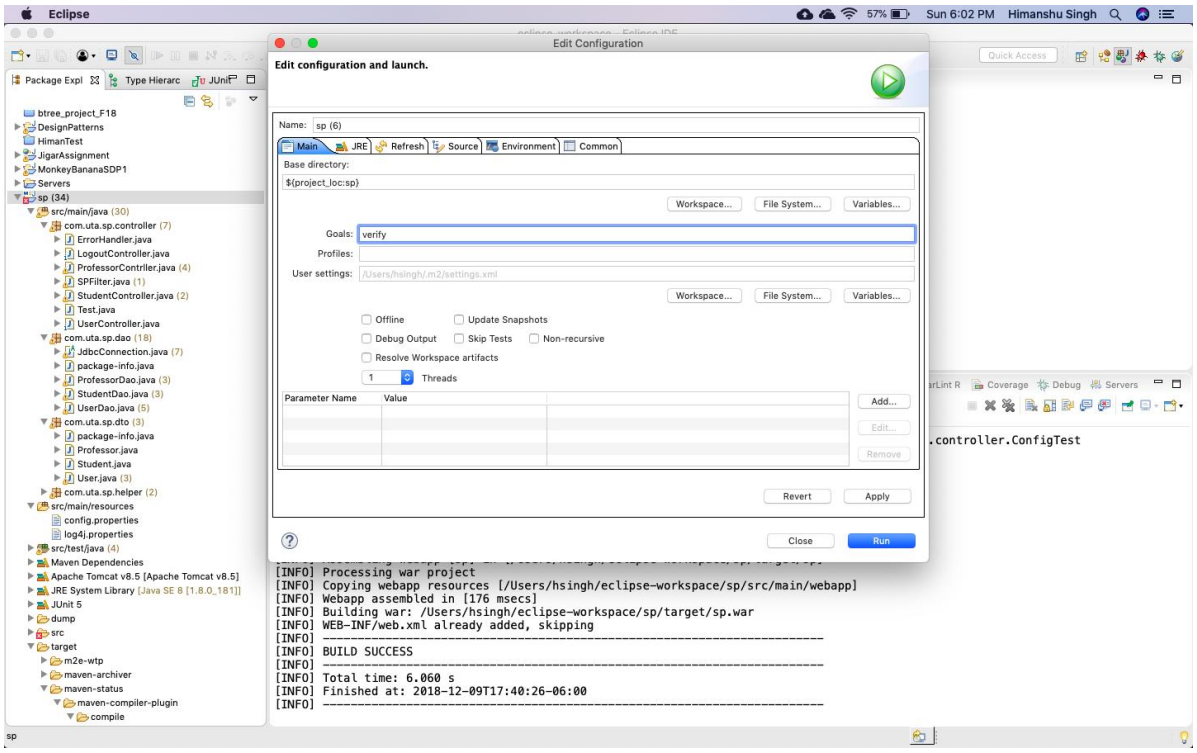
Hidden files don't give the direct access to the pages and the sensitive parts of the application in complete structure of the application are not obtained both in burp suite and Zap thus making the application secure.

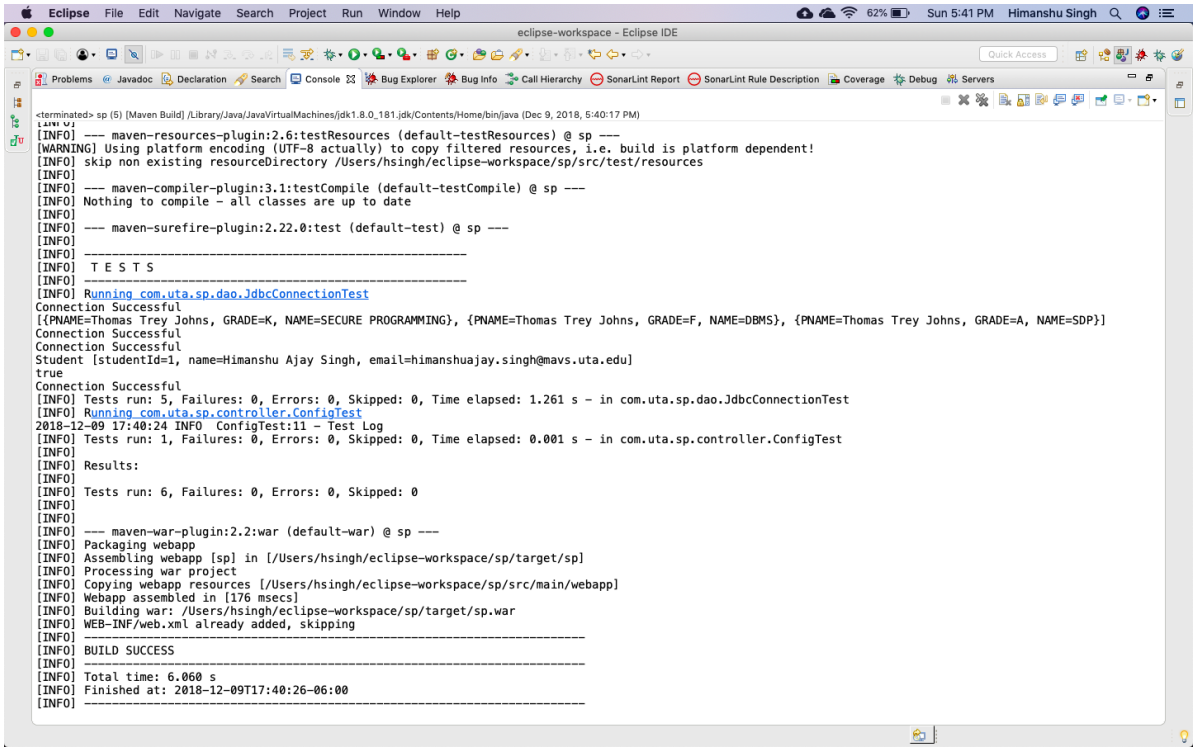
- Unit testing result
  Here is the unit testing report (actual screenshot) using **Junit unit testing framework**.


Eclipse File Edit Navigate Search Project Run Window Help
Sun 4:04 PM Himanshu Singh
eclipse-workspace - /Users/hsingh/eclipse-workspace/.metadata/.plugins/edu.umd.cs.findbugs.plugin.eclipse/sp.fbwarnings.xml - Eclipse IDE
Problems Javadoc Declaration Search Console Bug Explorer Bug Info Call Hierarchy SonarLint Report SonarLint Rule Description Coverage Debug Servers
<terminated> /Library/Java/JavaVirtualMachines/jdk1.8.0_181.jdk/Contents/Home/bin/java (Dec 9, 2018, 3:58:00 PM)
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ sp ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:2.22.0:test (default-test) @ sp ---
[INFO]
[INFO] ---
[INFO] T E S T S
[INFO] ---
[INFO] Running com.uta.sp.dao.JdbcConnectionTest
Connection Successful
[{PNAME=Thomas Trey Johns, GRADE=K, NAME=SECURE PROGRAMMING}, {PNAME=Thomas Trey Johns, GRADE=F, NAME=DBMS}, {PNAME=Thomas Trey Johns, GRADE=A, NAME=SDP}]
Connection Successful
Connection Successful
Student [studentId=1, name=Himanshu Ajay Singh, email=himanshuajay.singh@mavs.uta.edu]
true
Connection Successful
[INFO] Tests run: 5, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.16 s - in com.uta.sp.dao.JdbcConnectionTest
[INFO] Running com.uta.sp.controller.ConfigTest
2018-12-09 15:58:07 INFO ConfigTest:11 - Test Log
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0 s - in com.uta.sp.controller.ConfigTest
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 6, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- maven-war-plugin:2.2:war (default-war) @ sp ---
[INFO] Packaging webapp
[INFO] Assembling webapp [sp] in [/Users/hsingh/eclipse-workspace/sp/target/sp]
[INFO] Processing war project
[INFO] Copying webapp resources [/Users/hsingh/eclipse-workspace/sp/src/main/webapp]
[INFO] Webapp assembled in [177 msecs]
[INFO] Building war: /Users/hsingh/eclipse-workspace/sp/target/sp.war
[INFO] WEB-INF/web.xml already added, skipping
[INFO]
[INFO] --- maven-install-plugin:2.4:install (default-install) @ sp ---
[INFO] Installing /Users/hsingh/eclipse-workspace/sp/target/sp.war to /Users/hsingh/.m2/repository/com/uta/sp/0.0.1-SNAPSHOT/sp-0.0.1-SNAPSHOT.war
[INFO] Installing /Users/hsingh/eclipse-workspace/sp/pom.xml to /Users/hsingh/.m2/repository/com/uta/sp/0.0.1-SNAPSHOT/sp-0.0.1-SNAPSHOT.pom
[INFO] --
[INFO] BUILD SUCCESS
[INFO] --
[INFO] Total time: 5.880 s
[INFO] Finished at: 2018-12-09T15:58:08-06:00
[INFO] --


Package Explorer
Type Hierarchy
JUnit
Finished after 1.311 seconds
Runs: 5/5
Errors: 0
Failures: 0
JdbcConnectionTest [Runner: JUnit 5] (0.946 s)
studentGetGrades() (0.622 s)
psidTest() (0.025 s)
studentGetOneTest() (0.030 s)
passwordTest() (0.155 s)
testUpdate() (0.114 s)
Failure Trace

- Integration testing results
  The integration testing is done with maven configuration file using **maven verifies command**.





Security assessment/Compliance

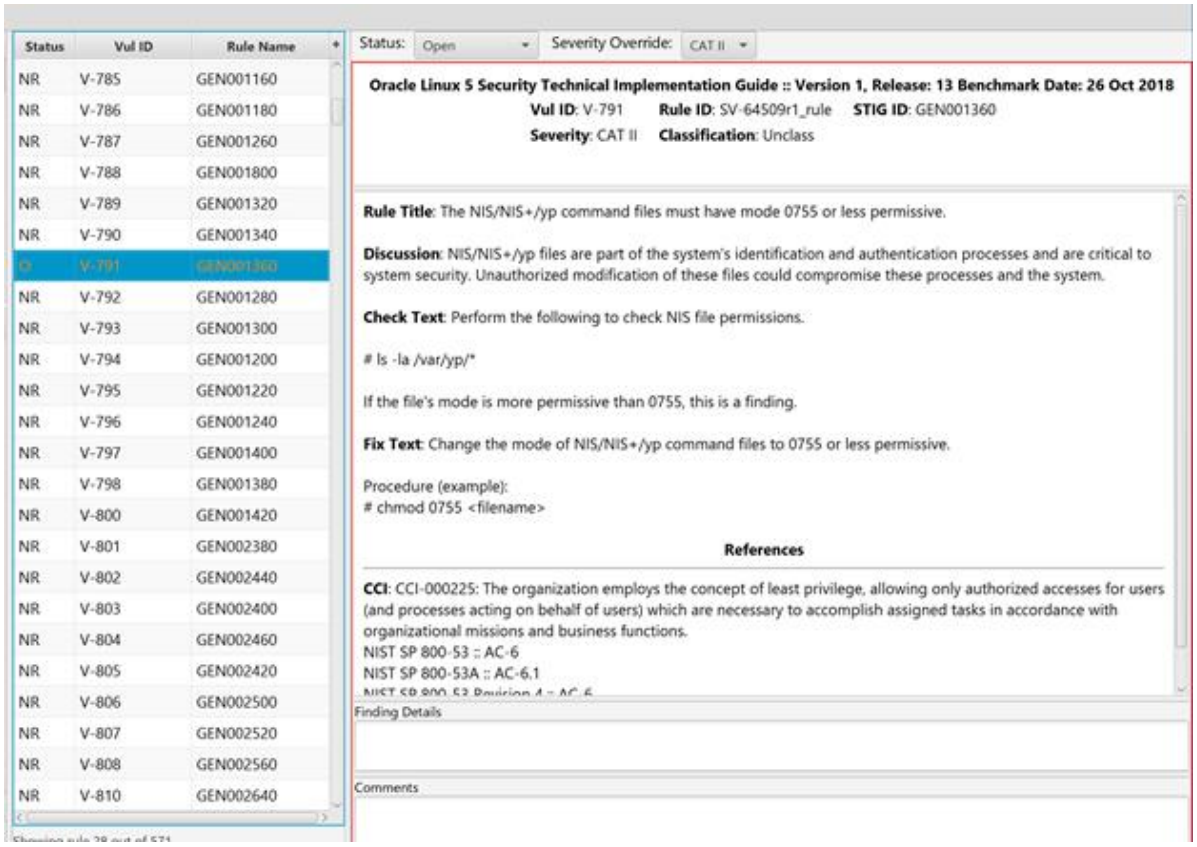
- STIG checklist

1.Oracle Linux 5 Security Technical Implementation Guide: Version 1, Release: 13 Benchmark Date: 26 Oct 2018

Vul ID: V-791 **Rule ID**: SV-64509r1_rule **STIG ID**: GEN001360

Severity: CAT II **Classification**: Unclass

We have two files configuration. Properties and log4j.properties change the mode using # chmod 0755 <filename> for the above two files to prevent unauthorized modification of these files



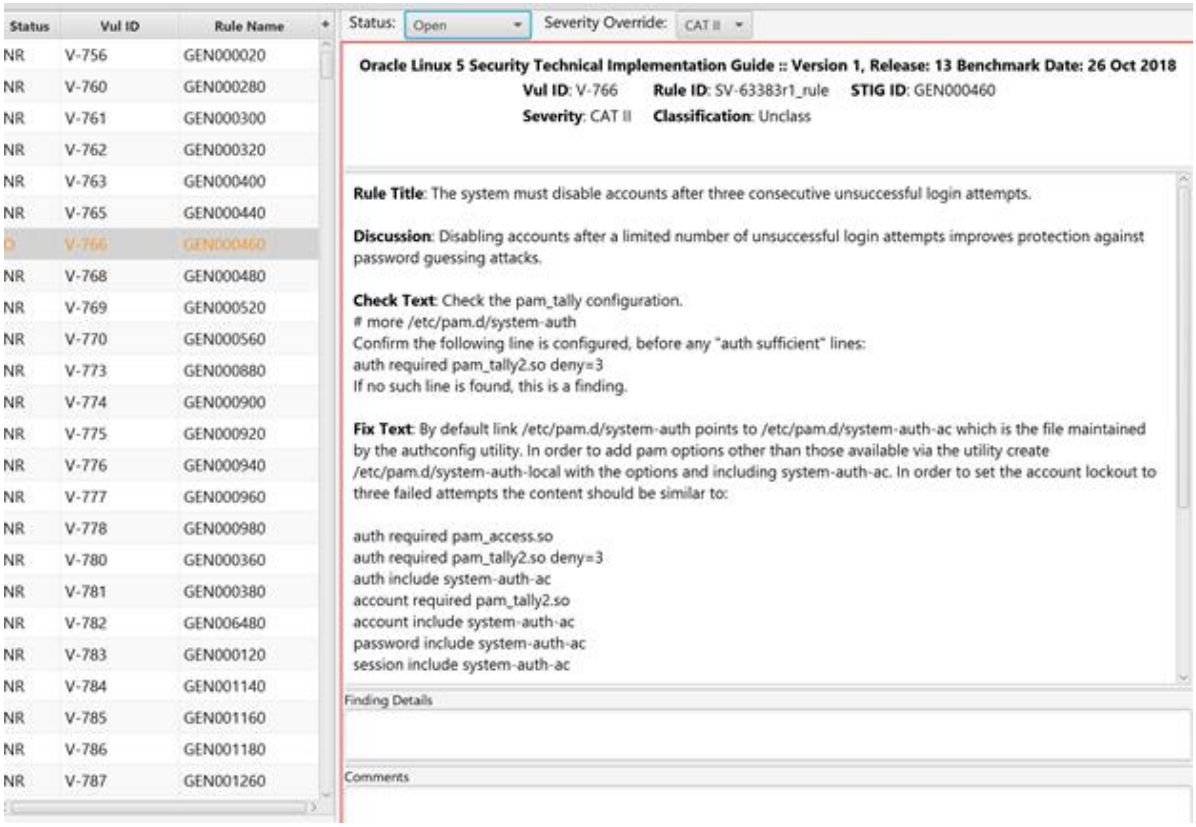

```
[Himanshus-Air-a70a:classes hsingh$ ls -l
total 16
drwxr-x---  3 hsingh  staff   96 Dec  9 14:32 com
-rw-r-----  1 hsingh  staff   74 Dec  9 02:25 config.properties
-rw-r-----  1 hsingh  staff  714 Dec  9 01:16 log4j.properties
[Himanshus-Air-a70a:classes hsingh$ chmod  0755 co
com/                config.properties
[Himanshus-Air-a70a:classes hsingh$ chmod  0755 config.properties
[Himanshus-Air-a70a:classes hsingh$ ls -l
total 16
drwxr-x---  3 hsingh  staff   96 Dec  9 14:32 com
-rwxr-xr-x  1 hsingh  staff   74 Dec  9 02:25 config.properties
-rw-r-----  1 hsingh  staff  714 Dec  9 01:16 log4j.properties
Himanshus-Air-a70a:classes hsingh$
```

2. Oracle Linux 5 Security Technical Implementation Guide: Version 1, Release: 13 Benchmark Date: 26 Oct 2018

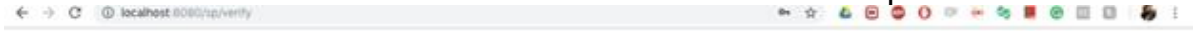
Vul ID: V-766 **Rule ID**: SV-63383r1_rule **STIG ID**: GEN000460

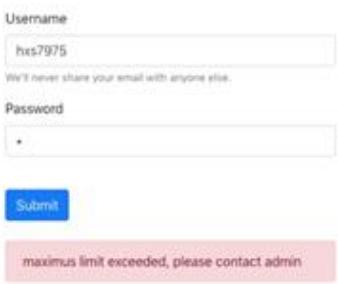
Severity: CAT II **Classification**: Unclass

Disabling accounts after a limited number of unsuccessful login attempts improves protection against passwords guessing attacks



Fix: The below screenshot shows the fix which is implemented



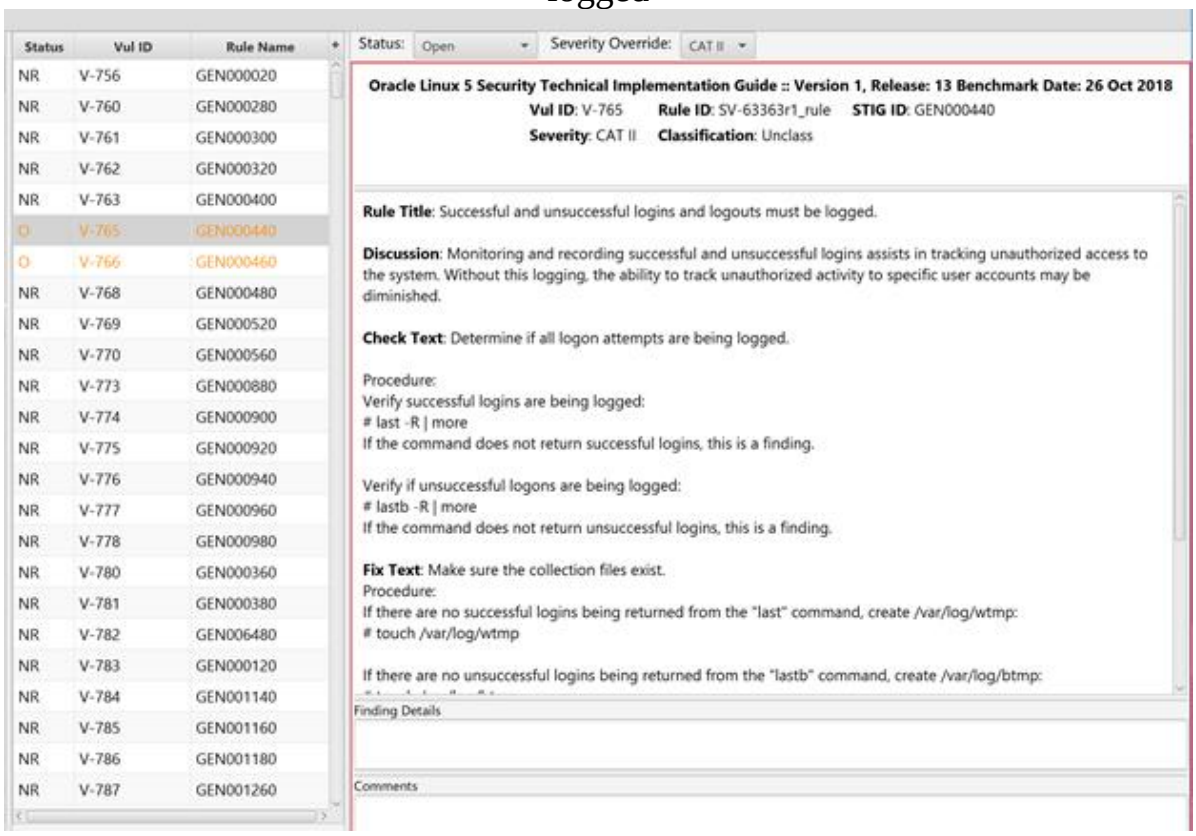


3. Oracle Linux 5 Security Technical Implementation Guide: Version 1, Release: 13 Benchmark Date: 26 Oct 2018

Vul ID: V-765 **Rule ID**: SV-63363r1_rule **STIG ID**: GEN000440

Severity: CAT II **Classification**: Unclass

The rule says that all successful and unsuccessful logins and logouts must be logged



We have fixed the above rule by logging in all these details:

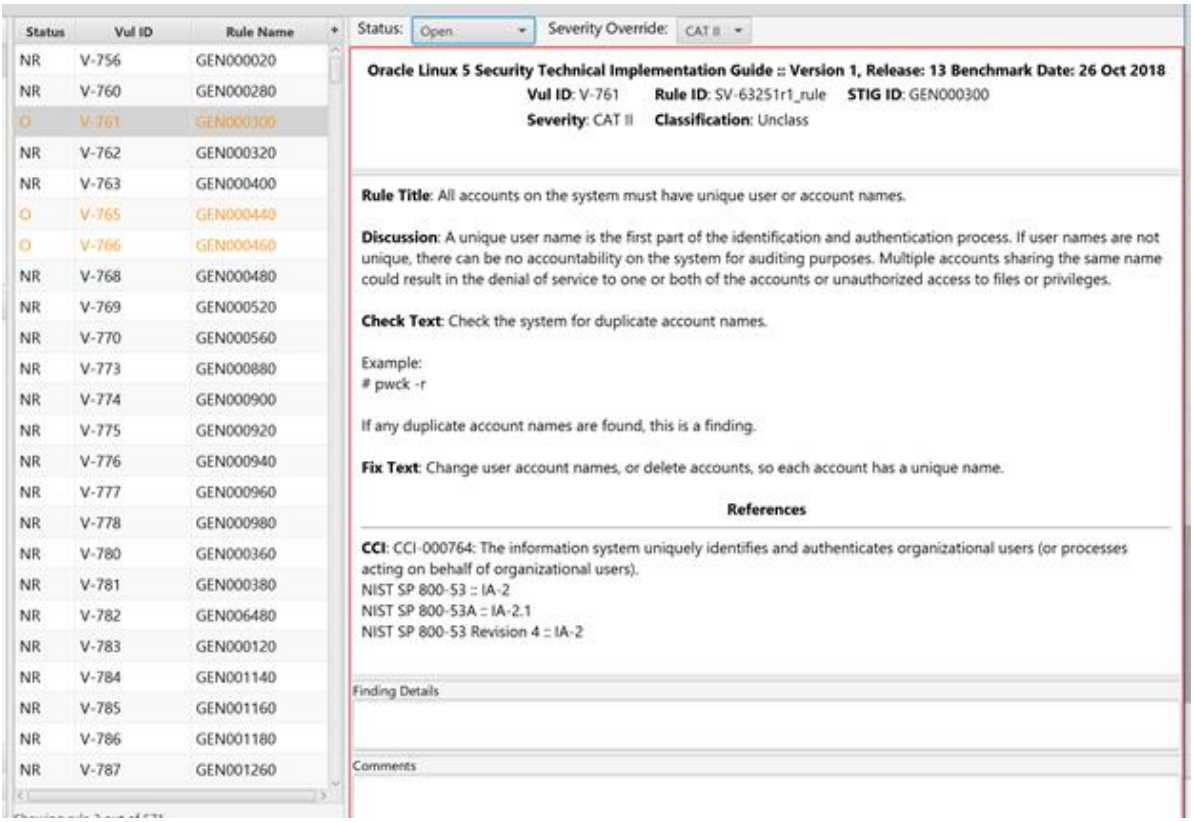
2018-12-09 14:32:20 INFO SPFilter:38 - valid request from ip-address => 0:0:0:0:0:0:0:1
2018-12-09 14:32:27 INFO SPFilter:70 - successfull login request for ttj7975 from ip-address => 0:0:0:0:0:0:0:1
2018-12-09 14:32:27 INFO SPFilter:48 - valid request from ip-address => 0:0:0:0:0:0:0:1
2018-12-09 14:32:29 INFO SPFilter:48 - valid request from ip-address => 0:0:0:0:0:0:0:1
2018-12-09 14:32:34 INFO SPFilter:48 - valid request from ip-address => 0:0:0:0:0:0:0:1
2018-12-09 14:32:34 INFO ProfessorContrller:87 - grage updated for studentid5 by professor id 1
2018-12-09 14:32:34 INFO SPFilter:48 - valid request from ip-address => 0:0:0:0:0:0:0:1
2018-12-09 14:32:40 INFO SPFilter:48 - valid request from ip-address => 0:0:0:0:0:0:0:1
2018-12-09 14:32:40 INFO ProfessorContrller:87 - grage updated for studentid10 by professor id 1
2018-12-09 14:32:40 INFO SPFilter:48 - valid request from ip-address => 0:0:0:0:0:0:0:1
2018-12-09 16:56:28 ERROR SPFilter:43 - invalid request from ip-address => 0:0:0:0:0:0:0:1
2018-12-09 16:56:28 INFO SPFilter:38 - valid request from ip-address => 0:0:0:0:0:0:0:1
2018-12-09 16:58:33 INFO SPFilter:38 - valid request from ip-address => 0:0:0:0:0:0:0:1
2018-12-09 16:58:33 INFO SPFilter:70 - successfull login request for ttj7975 from ip-address => 0:0:0:0:0:0:0:1
2018-12-09 16:58:33 INFO SPFilter:48 - valid request from ip-address => 0:0:0:0:0:0:0:1
2018-12-09 16:59:22 INFO SPFilter:48 - valid request from ip-address => 0:0:0:0:0:0:0:1
2018-12-09 16:59:22 INFO LogoutController:38 - Thomas Trey Johns =>User successfull logged out
2018-12-09 16:59:23 INFO SPFilter:38 - valid request from ip-address => 0:0:0:0:0:0:0:1

4. Oracle Linux 5 Security Technical Implementation Guide: Version 1, Release: 13 Benchmark Date: 26 Oct 2018

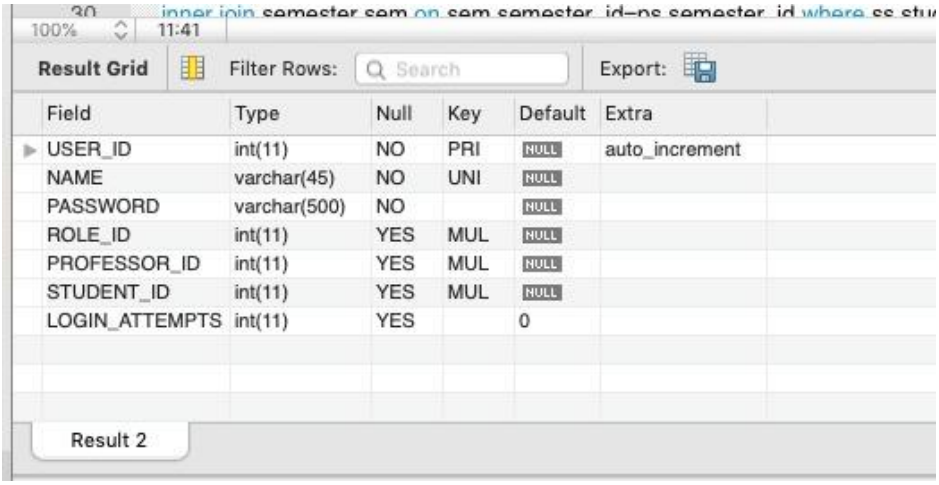
Vul ID: V-761 **Rule ID**: SV-63251r1_rule **STIG ID**: GEN000300

Severity: CAT II **Classification**: Unclass

All accounts on the system must have unique user or account names.



Fix: In the database we have implemented this by using unique keyword for all user names:

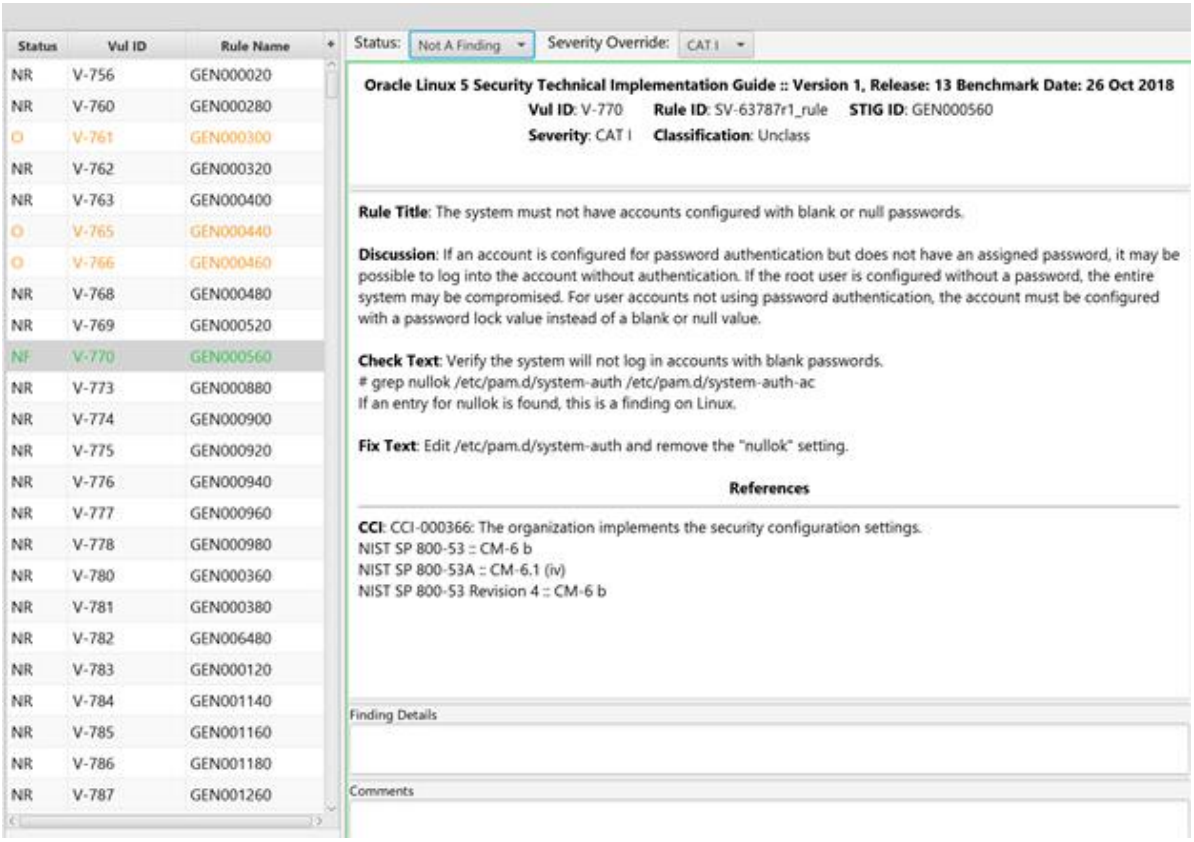


5. Oracle Linux 5 Security Technical Implementation Guide: Version 1, Release: 13 Benchmark Date: 26 Oct 2018

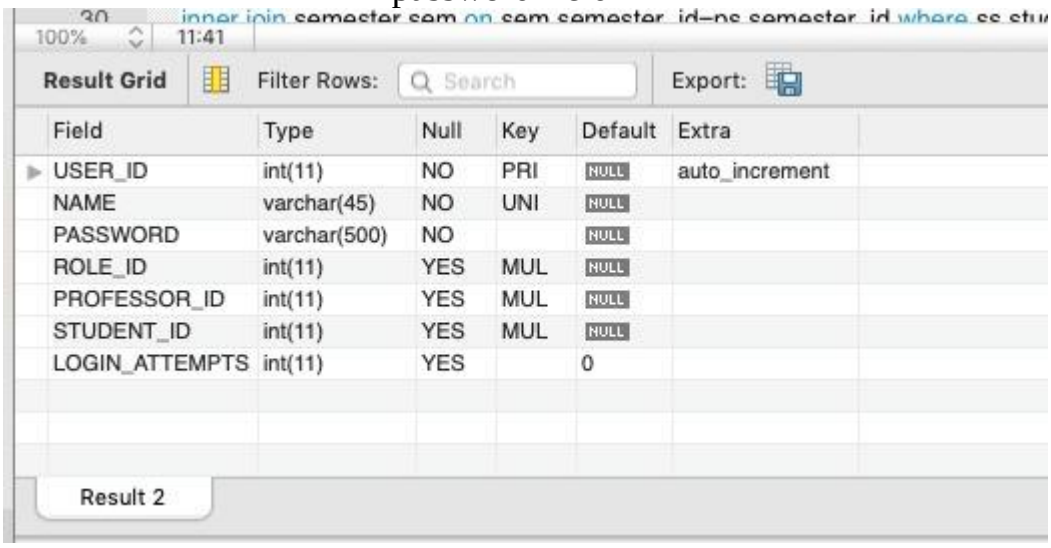
Vul ID: V-770 **Rule ID**: SV-63787r1_rule **STIG ID**: GEN000560

Severity: CAT I **Classification**: Unclass

The system must not have accounts configured with blank or null passwords.



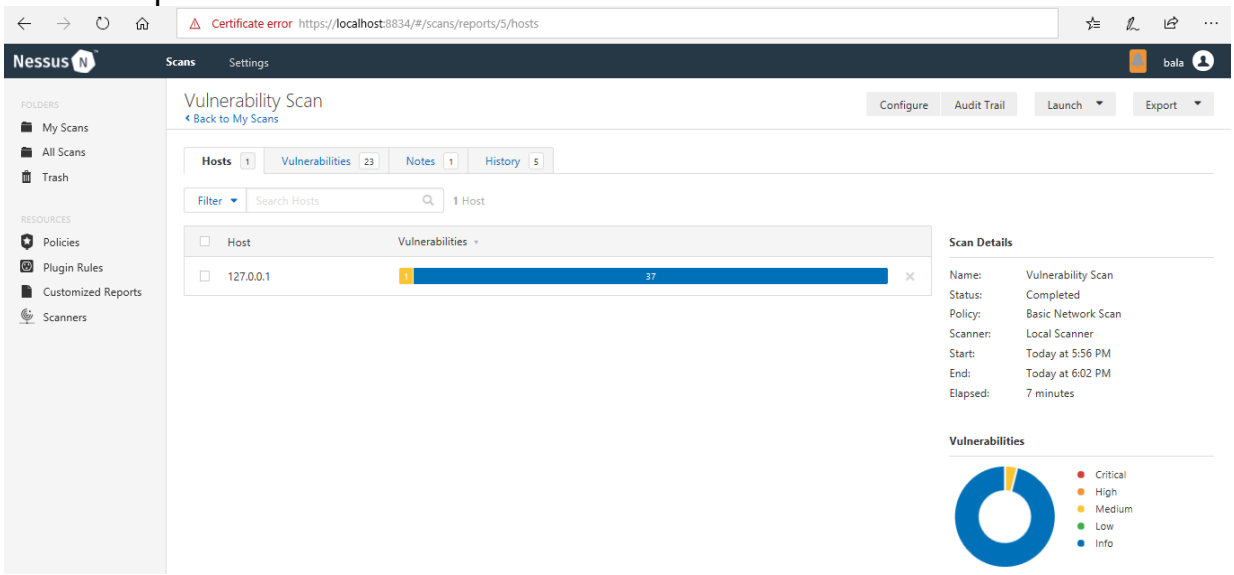
Fix: In the database we have implemented this by specifying No null in password field

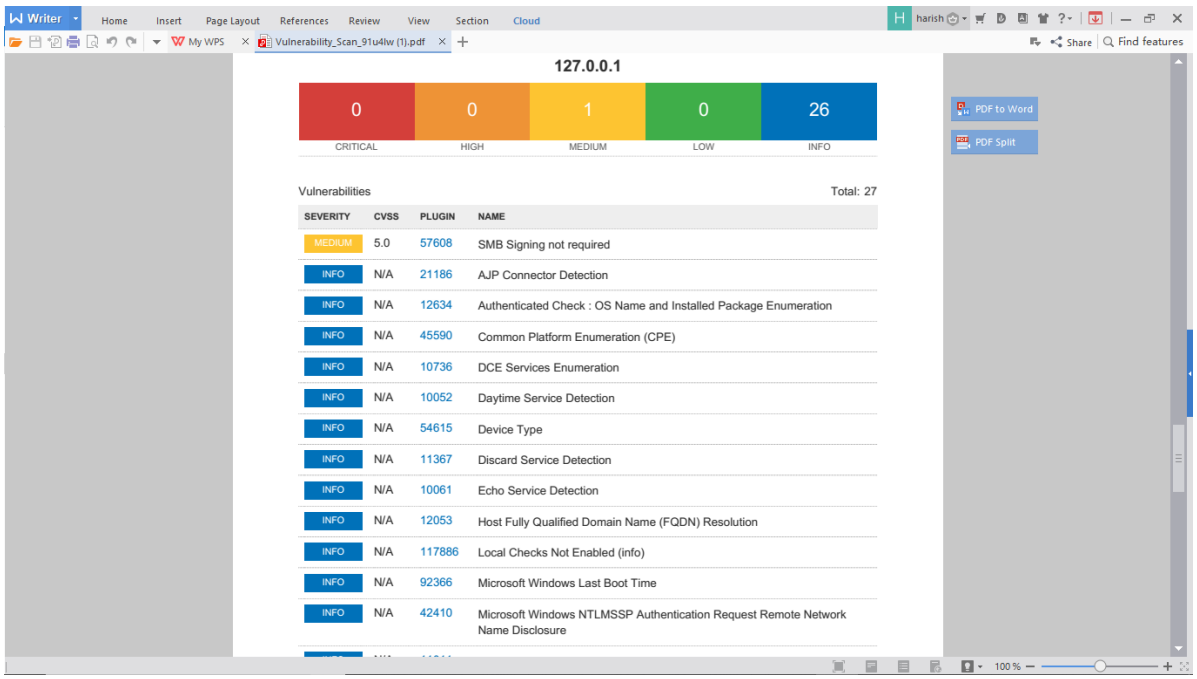


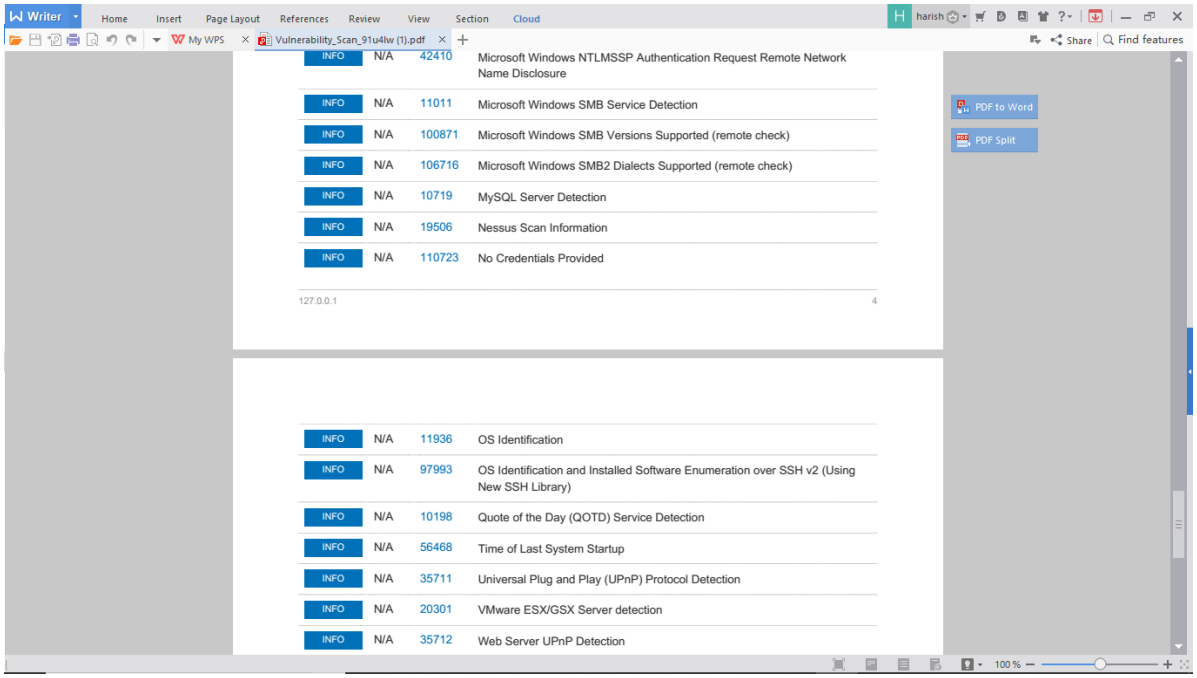
- Changes made in web server, application server for security hardening.
  The session-timeout is updated from the default value of 30 to 15 in tomcat/config/web.xml

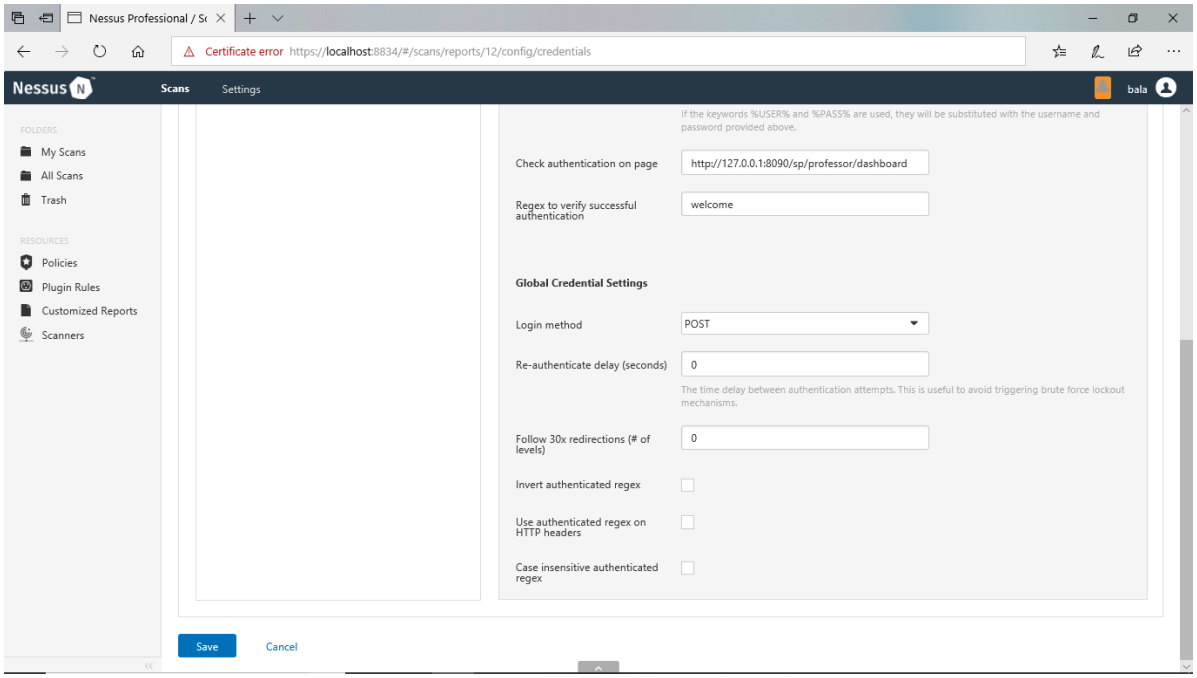
```
<session-config>
        <session-timeout>15</session-timeout>
</session-config>
```

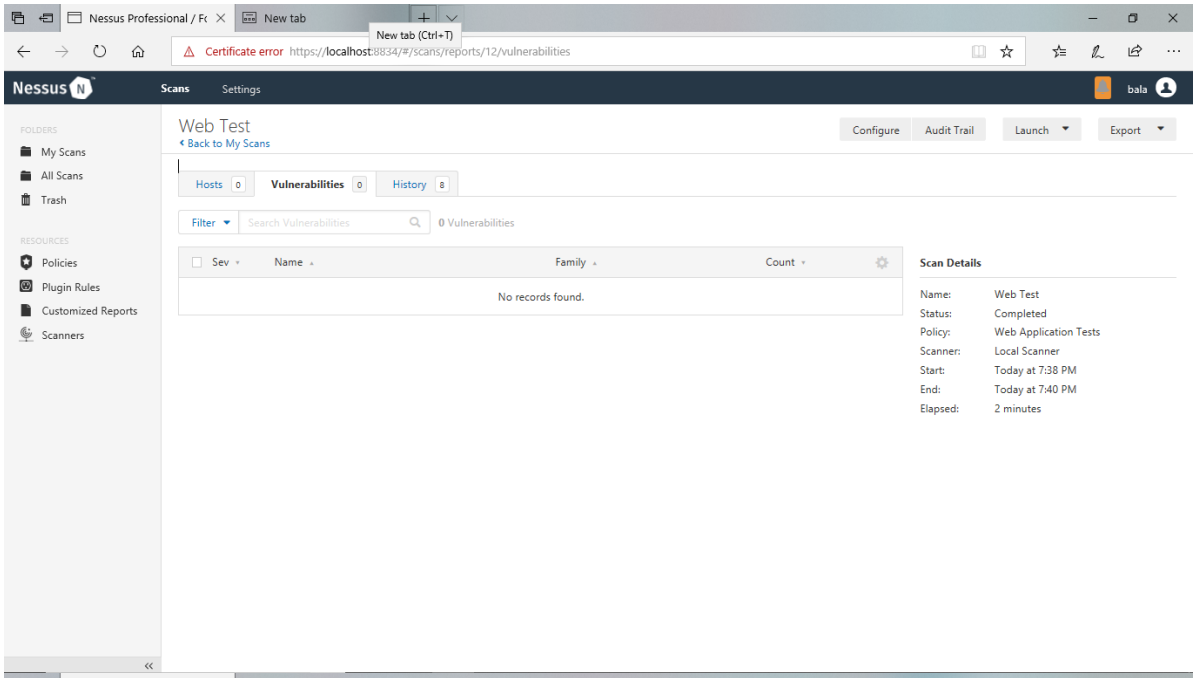
- Vulnerability scan reports
  Reports were generated for listing vulnerabilities in network and web application using Nessus. Screenshots of overall vulnerabilities are as below and reports are included in the file.

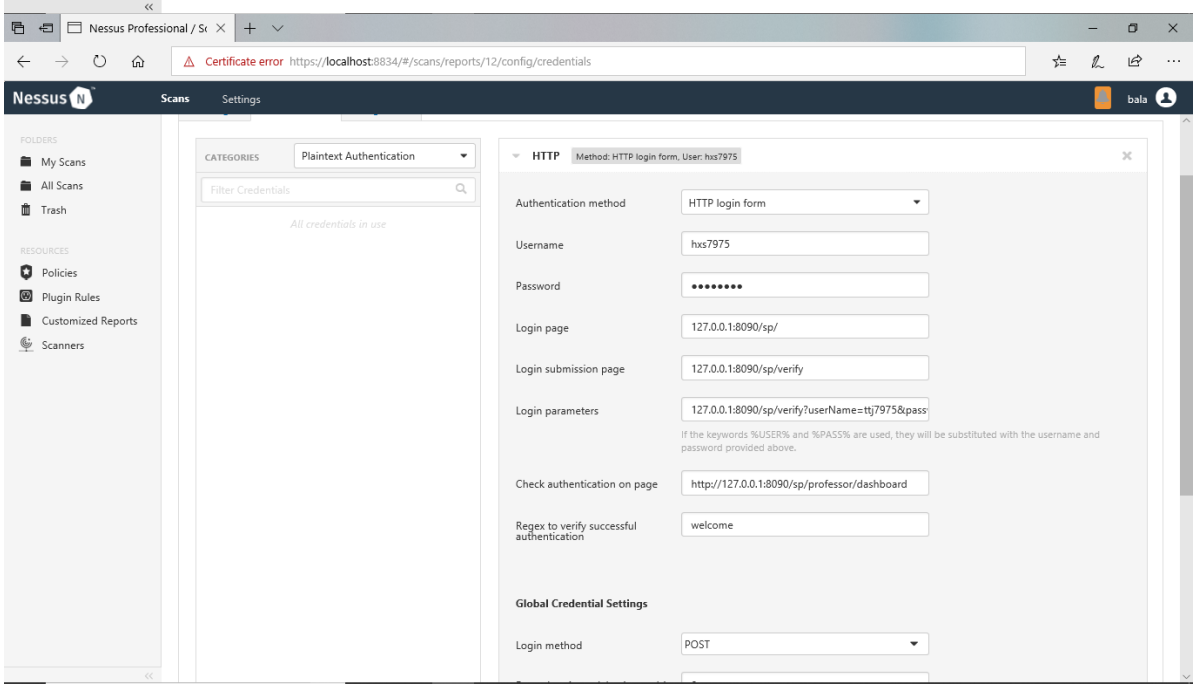


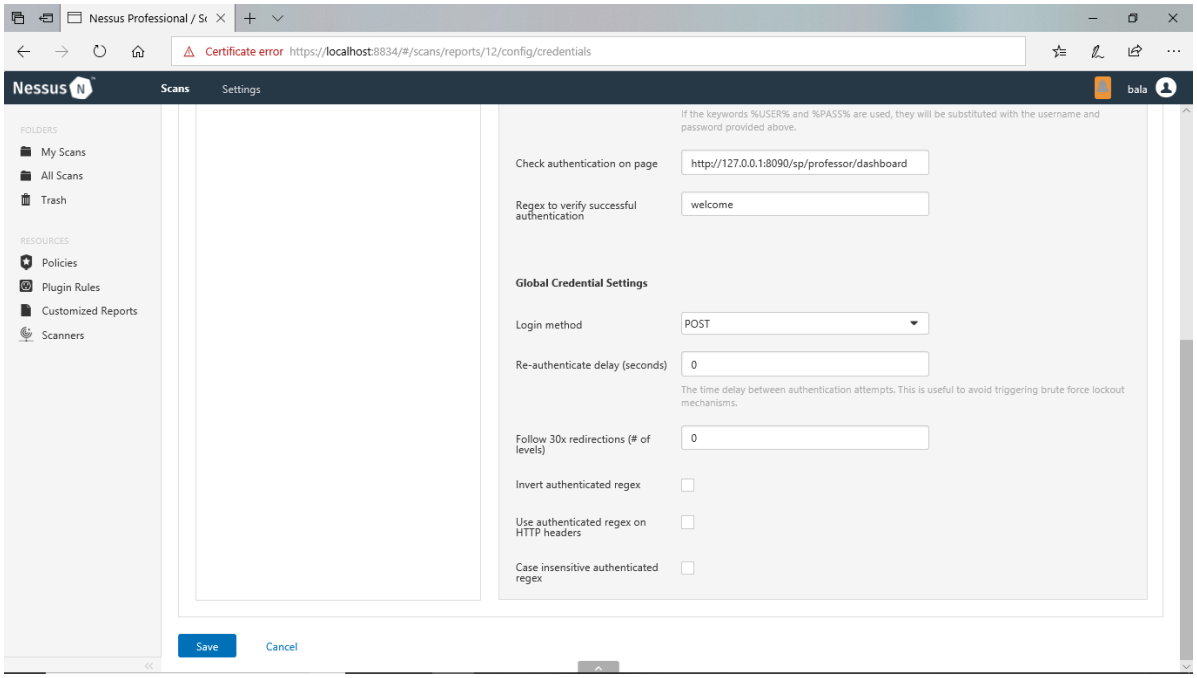



Writer
Home
Insert
Page Layout
References
Review
View
Section
Cloud
Vulnerability_Scan_91u4lw (1).pdf
Share
Find features
INFO N/A 42410 Microsoft Windows NTLMSSP Authentication Request Remote Network Name Disclosure
INFO N/A 11011 Microsoft Windows SMB Service Detection
INFO N/A 100871 Microsoft Windows SMB Versions Supported (remote check)
INFO N/A 106716 Microsoft Windows SMB2 Dialects Supported (remote check)
INFO N/A 10719 MySQL Server Detection
INFO N/A 19506 Nessus Scan Information
INFO N/A 110723 No Credentials Provided
127.0.0.1
4
PDF to Word
PDF Split
INFO N/A 11936 OS Identification
INFO N/A 97993 OS Identification and Installed Software Enumeration over SSH v2 (Using New SSH Library)
INFO N/A 10198 Quote of the Day (QOTD) Service Detection
INFO N/A 56468 Time of Last System Startup
INFO N/A 35711 Universal Plug and Play (UPnP) Protocol Detection
INFO N/A 20301 VMware ESX/GSX Server detection
INFO N/A 35712 Web Server UPnP Detection
100 %


Nessus Professional / Sc
Certificate error https://localhost:8834/#/scans/reports/12/config/credentials
Nessus
Scans
Settings
bala
FOLDERS
My Scans
All Scans
Trash
RESOURCES
Policies
Plugin Rules
Customized Reports
Scanners
If the keywords %USER% and %PASS% are used, they will be substituted with the username and password provided above.
Check authentication on page
http://127.0.0.1:8090/sp/professor/dashboard
Regex to verify successful authentication
welcome
Global Credential Settings
Login method
POST
Re-authenticate delay (seconds)
0
The time delay between authentication attempts. This is useful to avoid triggering brute force lockout mechanisms.
Follow 30x redirections (# of levels)
0
Invert authenticated regex
Use authenticated regex on HTTP headers
Case insensitive authenticated regex
Save
Cancel


Nessus Professional / Fc
New tab
New tab (Ctrl+T)
Certificate error https://localhost:8834/#/scans/reports/12/vulnerabilities
Nessus
Scans
Settings
bala
FOLDERS
My Scans
All Scans
Trash
RESOURCES
Policies
Plugin Rules
Customized Reports
Scanners
Web Test
Back to My Scans
Configure
Audit Trail
Launch
Export
Hosts 0
Vulnerabilities 0
History 8
Filter
Search Vulnerabilities
0 Vulnerabilities
Sev
Name
Family
Count
No records found.
Scan Details
Name: Web Test
Status: Completed
Policy: Web Application Tests
Scanner: Local Scanner
Start: Today at 7:38 PM
End: Today at 7:40 PM
Elapsed: 2 minutes


Nessus Professional / Sc
Certificate error https://localhost:8834/#/scans/reports/12/config/credentials
Nessus
Scans
Settings
bala
FOLDERS
My Scans
All Scans
Trash
RESOURCES
Policies
Plugin Rules
Customized Reports
Scanners
CATEGORIES
Plaintext Authentication
Filter Credentials
All credentials in use
HTTP
Method: HTTP login form, User: hxs7975
Authentication method
HTTP login form
Username
hxs7975
Password
Login page
127.0.0.1:8090/sp/
Login submission page
127.0.0.1:8090/sp/verify
Login parameters
127.0.0.1:8090/sp/verify?userName=ttj7975&pass
If the keywords %USER% and %PASS% are used, they will be substituted with the username and password provided above.
Check authentication on page
http://127.0.0.1:8090/sp/professor/dashboard
Regex to verify successful authentication
welcome
Global Credential Settings
Login method
POST



References

1. *https://www.owasp.org/images/5/57/OWASP_Proactive_Controls_2.pdf*
2. *http://web.cerritos.edu/dwhitney/SitePages/CIS201/LectureNotesOnTalonNet/Chapter12Lecture.pdf*
3. https://www.symantec.com/avcenter/reference/attack.surface.analysis.of.blackberry.devices.pdf
4. http://ethesis.nitrkl.ac.in/5793/1/E-9.pdf
5. *https://google.github.io/styleguide/javaguide.htm*