

# **Portant - Protecting Sensitive Data from Multiple Sources**

## **Recommendations / Monitoring Plan**

**COMP3850 – Computing Industry Project**

**Group 37 – Socheat Chhun, Edward Morris, Arshita Jaryal, Md Mehedy Hasan, Jarrod Adair, Bradley Anderson**

# Contents

---

<b>Contents</b>	<b>2</b>
<b>1. Permissions Model</b>	<b>3</b>
<b>2. Contingency Plan</b>	<b>4</b>
2.1. Regular backup .....	5
2.2. Disaster recovery plan .....	5
2.3. Data loss prevention .....	6
2.4. Policies and education .....	6
<b>3. Deployment Security Measures</b>	<b>6</b>
3.1. Application Deployment platform .....	6
3.2. Encryption .....	6
3.2.1. Identity and Access Management .....	7
3.2.2. Token Security .....	7
3.2.3. Vulnerability and Patch Management .....	7
3.2.4. Continuous monitoring .....	8
3.2.5. Awareness training .....	8
<b>4. Future Features/Functionality</b>	<b>8</b>
<b>5. Installation/User Manual</b>	<b>9</b>
<b>6. Monitoring Benchmarks</b>	<b>11</b>
6.1. Application load balancer .....	11
6.2. Network load balancer .....	12
6.3. Health check .....	12
<b>7. Third-Party Packages/Plugins</b>	<b>12</b>
<b>8. Configuration of Cryptographic Measures</b>	<b>13</b>
8.1. Transport layer security .....	13
8.2. Upgrade Hashing for storing sensitive data. ....	14
<b>9. References</b>	<b>15</b>

# 1. Permissions Model

---

As an additional security measure, access control in the form of a permissions model can be implemented. A permission system would allow administrators and management level personnel a greater degree of control over who is able to create, view and respond to forms in the application.

This permissions model in mind will mimic permissions models that exist with products like Atlassian, AWS and Microsoft Teams. These products have very robust permissions allocation systems which allow for flexibility when managing what groups of people do and do not have access to.

For each organization either one or more head administrator/s would be delegated when a company first signs themselves up to use the application. The head administrator/s will then have full control to create groups, assign members to groups and allocate permissions to individuals or groups that head administrators create.

Designation of roles and permissions can be done in one of two ways, either individuals are given form creation, viewing and response permissions or individuals can be added to groups created by administrators which allow them certain permissions based on the rules in place for that group.

Groups and individuals can be granted the following kinds of permissions:

1. Administrator privileges (allowing for viewing of data pertaining to the use of the company's space as well as things such as deletion of the space, forms and users).
2. Enabling or disabling access to the company's space for the creation, viewing and response to forms.
3. Adding or removing permissions from individuals or groups.
4. Creation and deletion of groups.
5. Creation of forms (as well as which groups/individuals receive created forms).
6. Viewing of responded forms (as well as which groups forms can be looked at).
7. Deletion/Closing of forms for responses.

The ability to control which groups can distribute forms to other groups as well as the viewability of certain group's forms allows for scenarios where certain groups can only view relevant group's forms as well as only being able to send forms to those groups the require information from.

## Manage Permissions for 'Management'

Form Creation ☒ ▼

Which Groups/Individuals can 'Management' send forms to?

Groups:

Management	<input checked="" type="checkbox"/>
HR	<input type="checkbox"/>
Employees	<input checked="" type="checkbox"/>

Individuals:

John Citizen	<input type="checkbox"/>
Jane Citizen	<input checked="" type="checkbox"/>
. . .	

*Figure 1: A rough example screen showing how permissions would be allocated.*

Take the example above for managing permissions for a group named 'Management', by checking the form creation permission, Management can create and send forms. The groups that they're allowed to send it to are 'Management' and 'Employees'. An individual named 'Jane Citizen' has also been specified, it doesn't matter whether the individual in question belongs to either 'Management' or 'Employees', Management can create and send forms to this individual as Management has been explicitly granted the permission to do so.

Other screens can be used to create groups and add individuals to groups using text inputs or listing them out, thus allowing for groups to be made and managed to construct a permissions model that defines roles within an organization using the application.

## 2. Contingency Plan

---

The application has several security measures along with our dedicated team running constant tests as contingency. The current model consists of many measures taken by the Development team to ensure that the data is encrypted, and all private keys are secure.

To make sure breaches are contained, the system utilises the Private Key system. The data which flows from the user to the system has a personalised link with a pass key, which allows only the user to securely log into their work. We have communication channels going between the user to the application, the application to respondents and vice versa. As threat detection is a high priority for the application, A SIEM product will be implemented for the collection and analysis of logs for threat detection and incident response.

A SIEM monitors our web application in real time and collects a real time log. It also allows the user to set an alarm for any threat, providing a holistic view of the processes happening within the system.

Web Application Security and Data security are some of the main areas which require a high level of maintenance. To monitor the web application, external monitoring software's such as Nagios and SolarWinds can be utilised in contingency. The system uses SSL/TLS protocol for the detection of data tampering and falsifying. Data logs and anti-tampering measures help

ensure data integrity. As mentioned earlier Private keys will take the proper authorisation protocols which are layered with web availability safeguards ensure that the system is secure. Once encoded the user input is validated.

It is made sure that everyone who has access to the system has the right level of access as well. As stated in the Permissions Model (Section 1), Groups and Individuals can be granted a different level of access. So, in the case of an attempted attack, only those authorised can make changes, in turn decreasing the chance of internal threats.

End-To-End Testing is one of the main ways the team detects any vulnerabilities in the system. This helps ensure that it is secure from the beginning till the end. These are the following steps taken.

**Test Planning:** Consists of the team specifying the main tasks and scheduling the following steps which need to be taken. The Developers also specifies the resources available in retrospect to the time it will take.

**Test Design:** These are the main test specifications and where the developers generate the possible test cases. A risk and usage analysis is done to see which tests are a priority and how they are to be scheduled.

**Test Execution:** All selected tests are executed, and all results are documented.

**Analysis of results:** A test result analysis is generated by the team to evaluate the testing and check if there are any vulnerabilities or flaws in the code which need to be solved. This can also lead to requiring further testing in some cases.

## 2.1. Regular backup

---

Backup is a big part of information security. No business is immune from a cyber-attack and if data gets stolen or the system crashes, we can restore the system from backup and have the business or application up and running with less downtime.

If our system gets hacked or infected by malware, we can use a new system and restore the application or database data from the back up. A combination of backup such as incremental, differential, and full backup will be done periodically. It is also vital that the backup data is stored in a secure place and encrypted, and this will protect the backup from unauthorized access and snooping.

Back up files need be stored offline since some ransomware malware looks for backup files. It is also recommended to back up the data offsite. It is also necessary to test the backup's integrity and ability to successfully restore from the backup. It is also recommended to store cryptographic keys and certificates in a very secure place and offline and offsite.

## 2.2. Disaster recovery plan

---

We will need to develop a business recovery plan with detailed instructions on how to respond to any unplanned incidents and business disruptions.

It is recommended to establish a business continuity and disaster recovery plan and, in that plan, define critical system resources and develop a business impact analysis strategy. It should also include an annual risk assessment and testing, monitoring plan, data recovering and security and how to enable consistent, reliable backup processes. A successful business continuity and disaster recovery plan will ensure business operations and operational security.

## 2.3. Data loss prevention

---

Data loss prevention is a process used to ensure that sensitive is not lost, destroyed, miss used or unauthorized access to data. Data loss prevention system could be a set of tools or software that will monitor network data, endpoint data, data at rest, data in use etc. It could detect any leaking of data also. Data loss prevention could be implemented together with IDS, IPS, WAF and SIEM.

## 2.4. Policies and education

---

As discussed before, we need to develop a business continuity and disaster recovery plan policy. It is also important to update other policies. Employees should be trained properly and be put through an annual training and awareness program to ensure they have good knowledge about different attacks like phishing, ransomware etc.

# 3. Deployment Security Measures

---

## 3.1. Application Deployment platform

---

Our application will be deployed in the cloud environment. Deploying applications in the cloud is cost efficient, easier collaboration for development, resourceful development, better security, simplified application hosting, minimize resource cost etc. Upon doing research our team has decided on the cloud provider as amazon web services aka AWS.

AWS is a top cloud provider which has so many services that will help deploy and monitor our application better. Such services AWS Lambda, AWS CloudWatch, AWS CloudTrail, AWS GuardDuty, AWS WAF will be used for better and cost-effective deployment and intrusion detection and continuous monitoring of our application.

Since the application will be for wider use, AWS can ensure the application's hosting platform and its security.

## 3.2. Encryption

---

When application is used for wider use, implementing CIA triad will ensure application remains secure. Strong encryption protocol will ensure the security of data in transit and data at rest.

The security properties are confidentiality, integrity and availability which are also called CIA triad.

**Confidentiality** – protecting unauthorised access to data and subsequent misuse of it.

**Integrity** – Data has not been tampered with and is accurate and consistent.

**Availability** – Data should be available to authorised parties.

As mentioned in our scoping document, that TLS 1.3 will be used for transport layer security. Also, a strong hashing algorithm will be used for storing sensitive data.

Here are some additional security measures listed below.

- Disable any HTTP access for SSL enable services and resources.

- If any weak cipher is implemented, then disable them.
- Obtain SSL/TLS certificate from trusted and reputable certificate authority.
- TLS 1.3 will ensure that transport layer security header will not communicate with non-SSL per the Server Certificate Security policy.
- Allow only privilege user to connect to SSH server for management and must be authenticated using the private key.
- Store the certificate keys in a very secure place and only give access to an appropriate individual on a need-to-know basis. AWS certificate manager and AWS key manager are good options for storing keys.
- Rotate the keys regularly, renew them on time and revoke them if necessary.

### **3.2.1.Identity and Access Management**

Identity and access management (IAM) refers to managing access to network and web application services and resources. As discussed in section 1, different permission model will be used for users to access the application, using resources and perform administrative tasks. To ensure IAM, authentication, authorization and accounting will be implemented.

Since traditional web access management is becoming obsolete day by day, modern access management will be used for access management. Modern access management can control access to web applications and different APIs by supporting various deployment models.

Modern access management can integrate seamlessly with advanced multi factor authentication (MFA) and single sign on services. It is vital that we use MFA for user authentication which will ensure two factors such as user says what they are and what they have.

Once the user is authenticated then the user will be authorized to access various resources based on the permission they have. We will enforce different policies to access services and resources based on permission model.

A strong password policy will be enforced to ensure confidentiality of the information and integrity of the system.

All logs generated by IAM will be stored in a central log management system and will be analysed real-time and static. The log management system will also generate alerts based on policies and anomaly and response will be taken towards that alert will help us avoid a security breach.

### **3.2.2.Token Security**

By searching common developer websites for unsecured tokens, you can easily find them online. Instead of storing the token information somewhere safer, developers simply include them in their open-source repositories.

Securing the third-party tokens properly should be a straightforward device protection best practise. Please don't leave tokens that you've paid for lying around in your code, ready to be snatched.

### **3.2.3.Vulnerability and Patch Management**

Vulnerability management is a process to identify assets in the infrastructure or network and reports security vulnerability on target system. As discussed in the design and testing document, a vulnerability scanner will be used to scan regularly to find known vulnerabilities. A periodic penetration testing will also be conducted to find vulnerabilities.

Patch management is to apply missing patch to fix vulnerability and it also includes updating the software and application.

These two services can work together to update software and application to mitigate any ongoing or previous vulnerabilities.

### **3.2.4. Continuous monitoring**

Continuous monitoring is a technology process which will be implemented for our web application. Continuous monitoring helps quick detection of security risks, and it is one of the most important tools after the application is deployed. The goal of continuous monitoring is to find security risks in real-time and increase the visibility and transparency of our application and any activities within it. Below is a list of ways we can do continuous monitoring.

1. Risk and threat assessment
2. Regular audit
3. Intrusion detection system
4. Web application firewall
5. Keep and review audit trail.
6. Log analysis
7. Incident response plan

### **3.2.5. Awareness training**

Security is a shared responsibility. Everyone is responsible for protecting the application and protecting themselves. As for our organisation, security awareness training will be provided to our employees as well as guidelines for our users. These training and guidelines will be provided half yearly at a minimum.

## **4. Future Features/Functionality**

---

Functionality planned in the future involves new additions as well as upgrades to functionality. These are as such:

An upgrade to Google Authenticator would provide facial recognition, improving the current method of SMS as authentication can be bypassed through the possession of the user's mobile device.

The ability to designate form input fields (i.e., integers, text boxes, text fields, sliders, etc.) as well as allowing for images and files would allow for a greater degree of control when making forms as to what specific kinds of responses are required for each question. It should also be noted that incorporating functionality that allows for image and file upload and transfer will require added security measures to go along with them. Mainly the sanitisation of files to ensure that no malware is being sent as well as proper encoding of images and files for encryption and transfer.

Another planned inclusion is an upgrade to the UX design of the web application. This involves a redesign of the form layout on the user's perspective such as a review of the button layout and a consideration of adding more pages or utilising white space on the screen in a more thoughtful manner.



If the permissions model in Section 1 is implemented, the form creation screen can also be updated to include a 'contacts book' style of being able to designate forms being sent to either groups or individuals without needing to manually type their emails in. Plus with the permissions model in place, it's possible to make it so that only the people and groups you're allowed to send forms to show up in the contacts list.

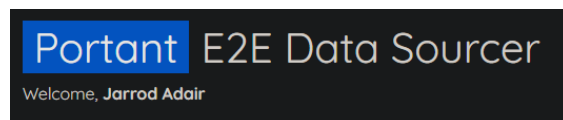
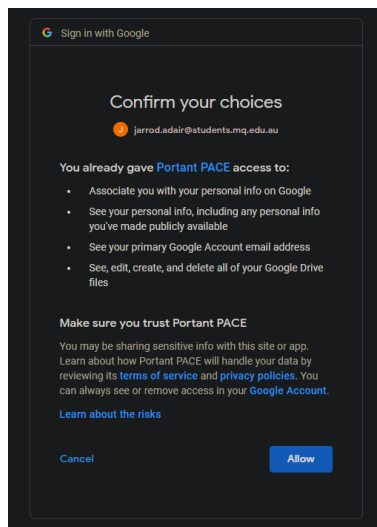
## 5. Installation/User Manual

---

Installation of our app is quite simple and has steps listed in the readme file but a typical install would go as follows:

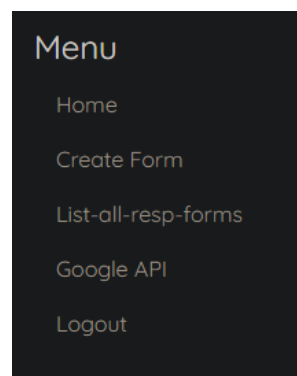
Firstly, python 3.7 at minimum is required for the app to work; additionally, the latest virtual environment is required to isolate the python dependencies and working python environment to the project. Then we use both virtual environments to initialise the project with python and use the inbuilt activation script generated to start the isolated virtual environment. Once inside the virtual environment, all the dependencies can be installed using python's pip module and after they are successfully installed, the app can be run using python on the app.py file, opening a web browser with the localhost address.

To use the application, the user will firstly have to authorise themselves using google OAuth2. This will direct the user to Google's OAuth services and then redirect the user back to the app as a 'logged in' user.

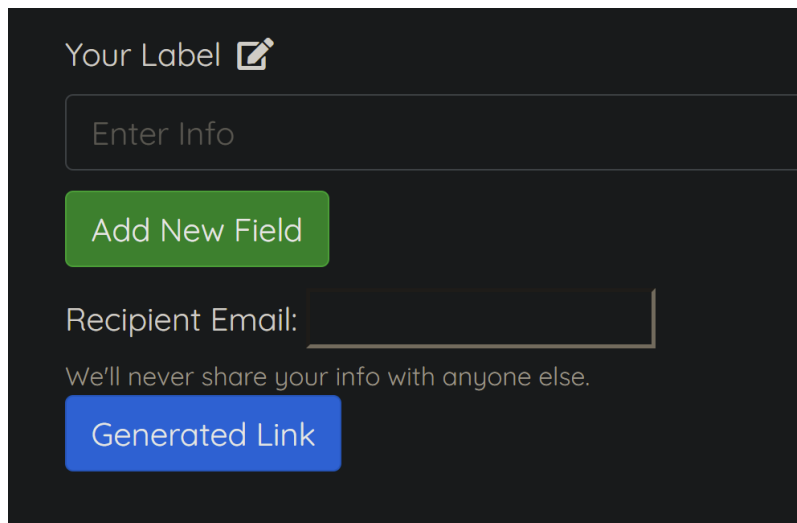


Google Auth page and welcome screen within app after logging in.

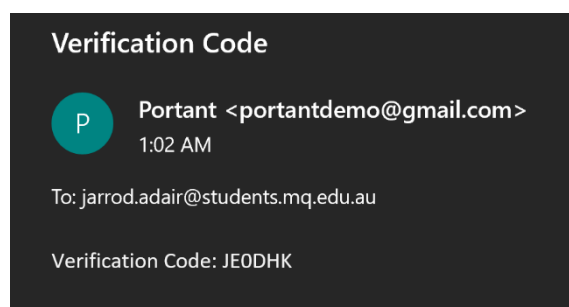
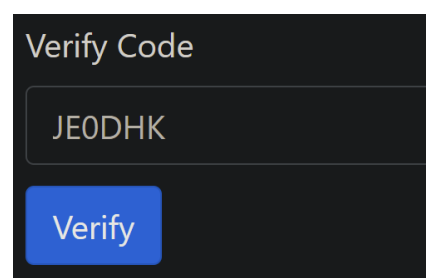
At the time of this deliverable, the following pages are available:



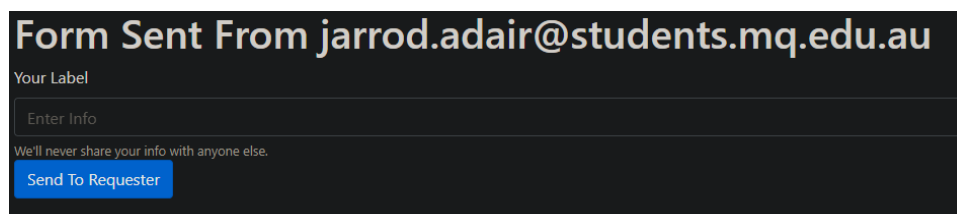
The Create Form page is one of the most important screens within the app, being the page where forms are created to be sent to recipients of the user for data to be retrieved from.

A dark-themed mobile app screen titled 'Your Label' with a pencil icon. It features a text input field labeled 'Enter Info', a green 'Add New Field' button, a 'Recipient Email:' label followed by a text input field, a line of text 'We'll never share your info with anyone else.', and a blue 'Generated Link' button.

Here you will find the ability to customise the form as well as add new fields and the generate link button which is used to create the link to share the form. The form is automatically shared with all emails listed inside the "Recipient Email" input box via email. These emails contain our MFA two step verification code which is then entered on the secure respondent URL generated.

An email interface on a dark background. The header is 'Verification Code'. Below it is a teal circle with a white 'P' icon, followed by the text 'Portant <portantdemo@gmail.com>' and '1:02 AM'. The body of the email says 'To: jarrod.adair@students.mq.edu.au' and 'Verification Code: JE0DHK'.A dark-themed mobile app screen titled 'Verify Code'. It has a text input field containing the code 'JE0DHK' and a blue 'Verify' button below it.

Following the natural flow of the app will take us to the respondent's form page. This page is used to simply send data that is entered from the recipient back to the app using end to end encryption.

A dark-themed mobile app screen titled 'Form Sent From jarrod.adair@students.mq.edu.au'. It features a 'Your Label' section with a text input field labeled 'Enter Info', a line of text 'We'll never share your info with anyone else.', and a blue 'Send To Requester' button.

The Google API page is the page where all the received forms are located for the user. This page will allow the user to select multiple forms to construct a Google Docs document using the selected forms data with a custom document title.

The List-all-resp-forms page is currently a temporary page to list all forms that have been responded to too for the developers of the app. The final product will not include this page.

## 6. Monitoring Benchmarks

### 6.1. Application load balancer

The application load balancer provides a single point of contact for users. The load balancer receives requests from clients and distributes them to targets. It operates at the application layers of OSI or TCP/IP model. An application load balancer is used for simplifying and improving security by ensuring the current SSL/TLS ciphers and protocols are always used.

A set of rules and policies will be implemented to our application load balancer and based on that the load balancer will forward the traffic to the destination. For example, if we have a redundant server for our application, we want to forward traffic to our main and redundant server proportionally. This will ensure low latency for our client and better user experiences.

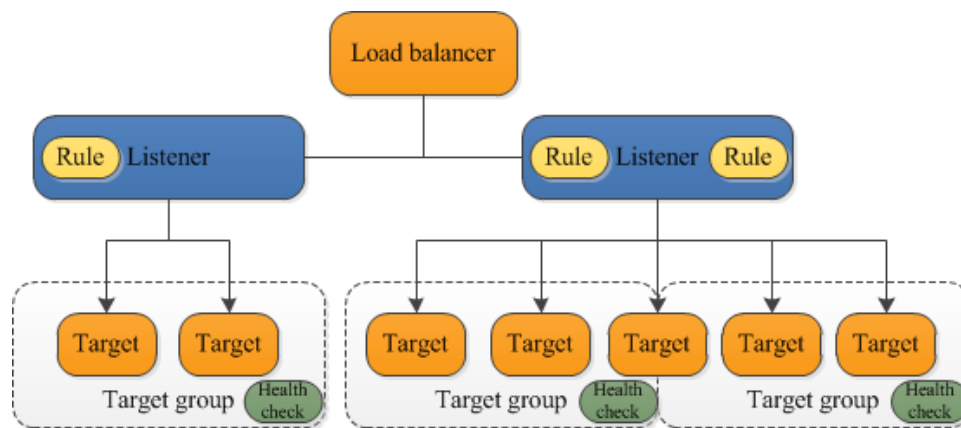


Figure 2: Application load balancer, image source: AWS

We can set matrices to our load balancer and based on that the load balancer will decide where to send the traffic. Some example metrics listed below.

Metric	Description
Active connection count	The number of active connections.
HTTP redirect count	The number of redirect actions that were successful.
HTTP request count	The number request to application was successful.

HTTP server-side error status count	The number of status 500 count
-------------------------------------	--------------------------------

An application load balancer usually can send a notification if any of the matrices.

## 6.2. Network load balancer

A network load balancer is features that distribute traffic among multiple servers. It works on layer 3 of OSI model and distributes traffic based on TCP and UDP protocol. It can handle lots of requests per second and can maintain low latencies. If there is a sudden change in the network traffic, network load balancer can handle these issues. It can also handle traffic based on routing and DNS requests.

## 6.3. Health check

Health checks are a way to check if a service is performing successfully. It will check periodically to determine which application server is safe to redirect traffic to. It will also be configured in a way it will send notification if any failover or unhealthy server or services. There are various ways to measure health such as liveness health check, local health check, dependency health check, anomaly detection etc.

# 7. Third-Party Packages/Plugins

Since we are using Python Flask as our development library, the library is a server-side rendering library. This means that we need to use the library for JavaScript and for Python.

For the JavaScript library, we focused on encryption library because the purpose of the application is to encrypt the data of the form and send this form to the python server to decrypt it.

For JavaScript we use:

List of JavaScript Libraries		
node-forge: 0.10.0	Crypto-js: 4.0.0	JQuery: 3.6.0
Bootstrap: 5.0.0	Font-awesome: 5.15.3	

For the Python library, we focused both on the login of the user from Google OAuth and decrypting of the form data that we received from the client.

For Python server, we use:

List of Python Libraries			
cachelib==0.1.1	certifi==2020.12.5	httplib2==0.19.0	requests-oauthlib==1.3.0
chardet==4.0.0	Click==7.0	idna==2.10	rsa==4.7.2
Flask==1.0.2	Flask-Session==0.3.2	itsdangerous==1.1.0	six==1.15.0
google-api-core==1.26.0	google-api-python-client==1.12.8	Jinja2==2.10	uritemplate==3.0.1

google-auth==1.27.0	google-auth-http httplib2==0.0.4	MarkupSafe==1.1.1	urllib3==1.26.3
google-auth-oauthlib==0.4.2	googleapis-common-protos==1.53.0	numpy==1.19.4	Werkzeug==0.14.1
oauthlib==3.1.0	packaging==20.9	protobuf==3.15.3	pymongo==3.11.3
pyasn1==0.4.8	pyasn1-modules==0.2.8	pyparsing==2.4.7	uuid==1.30
python-dotenv==0.15.0	pytz==2021.1	requests==2.25.1	cryptography==3.4.7
pycryptodome==3.10.1			

## 8. Configuration of Cryptographic Measures

### 8.1. Transport layer security

As discussed in our previous deliverable that we should implement SSL/TLS 1.3 for data in transit because SSL/TLS 1.2 have some exploitable vulnerability. A new version of this protocol SSL/TLS 1.3 will be implemented in our web application for data in transit security.

Initially we recommend supporting both TLS 1.2 and 1.3 since a lot of browsers are still not compatible with the latest version. This could be done in Apache server's

“/etc/apache2/mods-available/ssl.conf” file. See example below.

#### Prerequisite

**Apache version** - Apache 2.4.37

**OpenSSH** - OpenSSL 1.1.1

**Apache module** – mod\_security

#### Step 1

Check Apache and OpenSSL version

```

hasan@ds-mq:~$ apache2 -version
Server version: Apache/2.4.41 (Ubuntu)
Server built:   2020-08-12T19:46:17
hasan@ds-mq:~$ 
hasan@ds-mq:~$ openssl version
OpenSSL 1.1.1f  31 Mar 2020
hasan@ds-mq:~$

```

Install additional security module mod-security.

```
hasan@ds-mq:~$ sudo apt-get install libapache2-mod-security2
```

Enable security module.

```
hasan@ds-mq:~$ sudo a2enmod rewrite ssl security2
```

Finally restart the Apache server

```
hasan@ds-mq:~$ sudo systemctl restart apache2
```

## Step 2

Open the configuration file.

```
hasan@ds-mq:~$ sudo nano /etc/apache2/mods-available/ssl.conf
```

## Step 3

Update the SSL protocol to both version 1.2 and 1.3.

```
GNU nano 4.8 /etc/apache2/mods-available/ssl.conf Modified
# options.
# Enable only secure ciphers:
#SSLCipherSuite HIGH:!aNULL
# Updates

SSLCipherSuite HIGH:!aNULL

SSLHonorCipherOrder on

#SSLProtocol -all +TLSv1.2
SSLProtocol -all +TLSv1.2 +TLSv1.3

# Updates

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Paste Text ^T To Spell ^_ Go To Line
```

Once done save the file and restart the Apache again.

## 8.2. Upgrade Hashing for storing sensitive data.

The hashing algorithm that we are planning to use is the Blowfish hashing algorithm. However, this hashing algorithm was built for security and uniqueness. This means that this hashing algorithm will be slow hence at some point in the future, we will have to move on to a faster hashing algorithm. We can use FNV-1 if we want to avoid less collision but faster hash. We can also use SuperFastHas for fast hashing but more collision. In short, the algorithm we are using now is slow but secure. However, we can change this hashing algorithm to a faster one.

## 9. References

---

*CloudWatch metrics for your Application Load Balancer - Elastic Load Balancing.*

Docs.aws.amazon.com. (2021). Retrieved 18 May 2021, from  
<https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-cloudwatch-metrics.html>.

*Implementing health checks.* Amazon Web Services, Inc. (2021). Retrieved 18 May 2021, from  
<https://aws.amazon.com/builders-library/implementing-health-checks/>.