



## TABLE OF CONTENTS

<b>Executive Summary</b>	<b>3</b>
<b>Scope of Testing</b>	<b>4</b>
<b>Vulnerability Rating</b>	<b>5</b>
<b>Summary of Vulnerabilities &amp; Recommended Remediations</b>	<b>6</b>
• Cross Site Scripting (XSS)	6
• Outdated Library (jQuery)	7
• Unencrypted Loaded Resources (HTTP)	8
• Privilege Escalation (Google Bucket)	9
• Misconfigured Service Account	10
• Excess Permissions	11
• Instance Misconfiguration	12
• Low Password Complexity	13
• SSH Using Password	14
• Password Optimization	15
• Open Ports, filtering & logging off on Firewall	16
<b>Recommendations</b>	<b>17</b>
• Unattached Storage Bucket	17
• No Automatic Snapshots	18
<b>Security Matrix</b>	<b>19</b>
<b>Known Vulnerabilities / End of Life</b>	<b>20</b>
<b>Recommendations</b>	<b>21</b>
<b>Appendix</b>	<b>22</b>
• Criticality Rating	22
• Criticality Reference Table	23
• Tools Referenced	24
• Acronyms	25

## EXECUTIVE SUMMARY

A vulnerability assessment and cloud audit was conducted in accordance with the organization [REDACTED] regards to their Google Cloud Platform (GCP).

Efforts were based on analyzing nodes on the internal segment of the environment of organization [REDACTED] and to analyze for improved security posture and configuration.

After assessing the internal environment, we found several vulnerabilities that should be remediated in order to create better defensive posture and a more secure set of systems. Our findings, which will be outlined in this report, include the following areas of vulnerabilities:

- Cross Site Scripting (XSS)
- Outdated Library (jQuery)
- Unencrypted Loaded Resources (HTTP)
- Privilege Escalation (Google Bucket)
- Misconfigured Service Account
- Excess Permissions
- Instance Misconfiguration
- Low Password Complexity
- SSH Using Password
- Password Optimization
- Open Ports, filtering & logging off on Firewall

We have included remediation steps for the vulnerabilities in this report.

In conclusion, [REDACTED] should look to remediate these findings in order to increase the effectiveness of security within the organization.

## SCOPE OF TESTING

The scope of testing discussed and agreed upon with organization [AAAAA AAAAAA] included assessment of the AWS RDS and associated backend services including the S3, EC2, and the hosted WordPress web application. The test would cover vulnerabilities associated with the web application, associated services and the MySQL server hosting the application of organization [AAAAAA AAAAAA].

The testing was done utilizing a gray box approach. Some credentials were provided in order to speed up the process of testing and vulnerability finding.

The test included the EC2, S3 buckets, associated backend services and the server hosting the web application. There were three (3) external public facing Ips provided in scope for organization [AAAAAA AAAAAA] of the server hosting the web application, MySQL and its associated services. Furthermore, A minimalistic privileged user account for the SSH service on the Linux environment, and of the web server was provided so a deep internal assessment of the SSH environment could be completed, in which we did find significant warnings needing to be addressed.

## VULNERABILITY RATING (CRITICALITY)

A criticality score, between 0 to 49, is calculated by adding individual scores from “Time”, “Expertise”, “Knowledge required”, “Access to product by attacker”, “Type of equipment”. A following example is shown:

Factor	Value	Points
Time	< 1 week	1
Expertise	Expert	6
Knowledge required	Restricted information	3
Access from Attacker	Moderate	4
Type of equipment	Standard	0

**Criticality: Medium (14)**

Scores are also labeled based on three levels of criticality:

**Critical - Score between  $\geq 25$**

**Very High - Score Between 20-24**

**High - Score between 14-19**

**Medium - Score between 10-13**

**Low - Score between 0-9**

Please refer to the criticality matrix in the appendix for more information.

## SUMMARY OF VULNERABILITIES

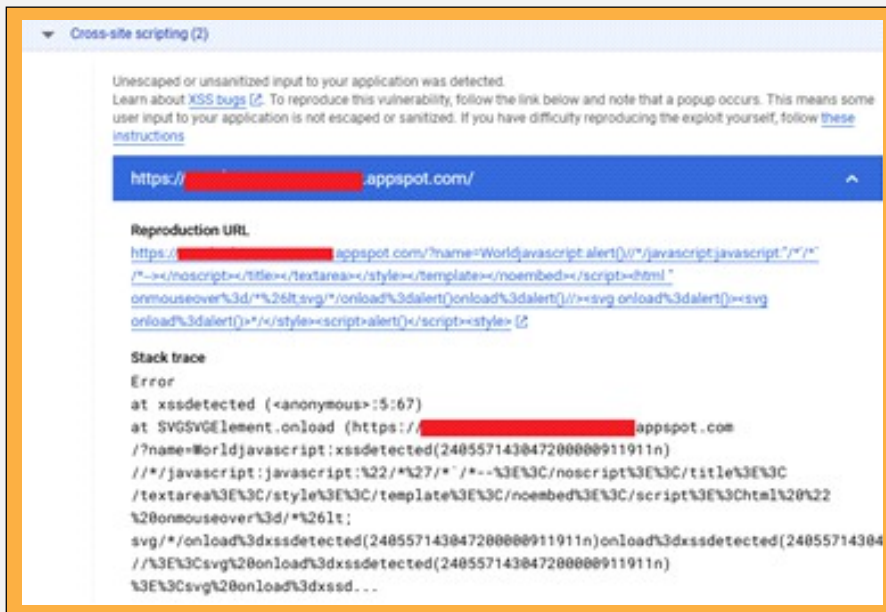
**VULNERABILITY #1**

**CRITICALITY: CRITICAL (33)**

**CROSS SITE SCRIPTING (XSS)**

**LOCATION: WEB SERVER  
GCP CLOUD**

With utilization of the Google Web Security scanner, Cross Site Scripting (XSS) was detected in the deployed NodeJS app. Cross Site Scripting (XSS) allows attackers to inject their own script onto the application, alter the behavior of the application, and ultimately create an attack that will execute the malicious script to any and all other legitimate visitors of the application.



**RECOMMENDED  
REMIEDIATION #1**

**CRITICALITY: CRITICAL (33)**

**CROSS SITE SCRIPTING (XSS)**

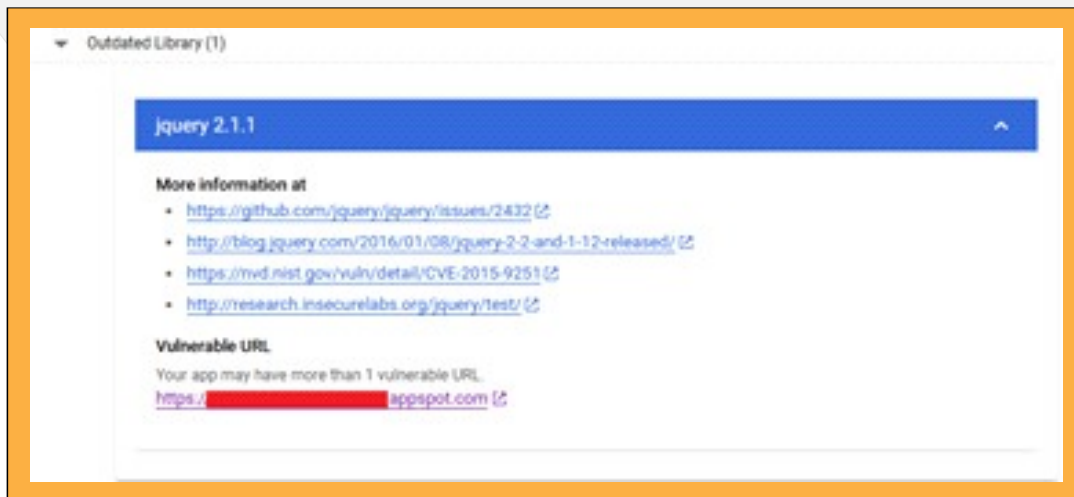
**LOCATION: WEB SERVER  
GCP CLOUD**

We recommend, if possible, to change the underlying code, to better sanitize inputs and not allow inserting of scripts. If changing the underlying code is not possible at this point in production, a robustly configured Web Application Firewall (WAF) would also provide the additional level of security.

**VULNERABILITY #2**
**CRITICALITY: CRITICAL (34)**
**OUTDATED LIBRARY**
**LOCATION: WEB SERVER  
GCP CLOUD**

With utilization of the Google Web Security scanner, an outdated library of jquery 2.1.1 has been found. This library has known and detailed vulnerabilities associated with it and more information regarding the vulnerability can be found on, for example, <https://nvd.nist.gov/vuln/detail/CVE-2015-9251>

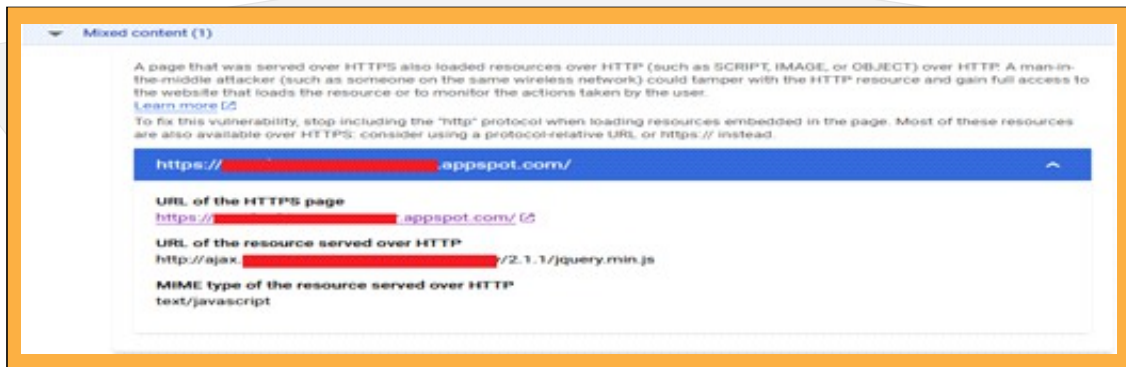
Based on the CVE above a quick description of the vulnerability is as follows: jQuery before 3.0.0 is vulnerable to Cross-site Scripting (XSS) attacks when a cross-domain Ajax request is performed without the dataType option, causing text/javascript responses to be executed.


**RECOMMENDED  
REMEDATION #2**
**CRITICALITY: CRITICAL (34)**
**OUTDATED LIBRARY**
**LOCATION: WEB SERVER  
GCP CLOUD**

It is recommended to upgrade to at least jquery version 3.0.0

**VULNERABILITY #3****CRITICALITY: MEDIUM (13)****UNENCRYPTED LOADED  
RESOURCES****LOCATION: WEB SERVER  
GCP CLOUD**

With utilization of the Google Web Security scanner, it was found that the application served as HTTPS has some loaded resources that are served over HTTP. HTTP is an unencrypted and unsecure protocol, hence we need to analyze the loaded resources to determine if an encrypted protocol would be the better option, which is most often the case.

**RECOMMENDED  
REMEDIAION #3****CRITICALITY: MEDIUM (33)****UNENCRYPTED LOADED  
RESOURCES****LOCATION: WEB SERVER  
GCP CLOUD**

Do not utilize HTTP, even for the loaded resources, utilize HTTPS instead.



VULNERABILITY #4

CRITICALITY: **CRITICAL (25)**

PRIVILEGE ESCALATION

LOCATION: GCP BUCKET

With utilization of the GCPBucketBrute script, we have found that the bucket utilized in the environment is vulnerable to privilege escalation.

```
$ python3 gcpbucketbrute.py -k my-bucket -f key.json
Generated 28 bucket permutations.

AUTHENTICATED ACCESS ALLOWED: my-bucket-mcpt
- VULNERABLE TO PRIVILEGE ESCALATION (storage.buckets.setIamPolicy)
- AUTHENTICATED LISTABLE (storage.objects.list)
- AUTHENTICATED READABLE (storage.objects.get)
- AUTHENTICATED WRITABLE (storage.objects.create, storage.objects.delete, and/or storage.objects.update)
- ALL PERMISSIONS:
[
  "storage.buckets.delete",
  "storage.buckets.get",
  "storage.buckets.getIamPolicy",
  "storage.buckets.setIamPolicy",
  "storage.buckets.update",
  "storage.objects.create",
  "storage.objects.delete",
  "storage.objects.get",
  "storage.objects.list",
  "storage.objects.update"
]
```

RECOMMENDED  
REMIEDIATION #4CRITICALITY: **CRITICAL (25)**

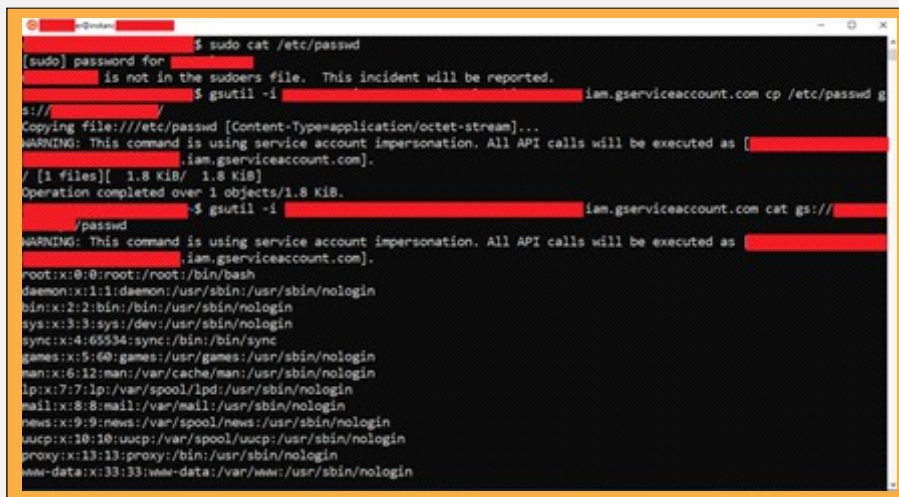
PRIVILEGE ESCALATION

LOCATION: GCP BUCKET

Configure the permissions of the bucket to not allow service accounts to escalate privileges. Having *storage.buckets.setIamPolicy* may allow a user to modify the IAM policy of the specified resource to grant himself/herself a role on it, thus granting him/her additional privileges at the resource level.

**VULNERABILITY #5**
**CRITICALITY: CRITICAL (27)**
**MISCONFIGURED SERVICE ACCOUNT**
**LOCATION: SESSION MANAGEMENT**

We were able to extract a credentials file (/etc/passwd) through a misconfigured service account. As can be shown in the following screenshot, we first try to read the /etc/passwd file using sudo, which then tells us that the user does not have the required permissions. Next, we utilize gsutil to copy the credentials file (/etc/passwd) to a bucket. The passwd file can then be read from the bucket as shown:



```

$ sudo cat /etc/passwd
[sudo] password for [REDACTED]:
[REDACTED] is not in the sudoers file. This incident will be reported.
$ gsutil -i [REDACTED] iam.gs-serviceaccount.com cp /etc/passwd gs://[REDACTED]
Copying file:///etc/passwd [Content-Type=application/octet-stream]...
WARNING: This command is using service account impersonation. All API calls will be executed as [REDACTED]
iam.gs-serviceaccount.com].
/ [1 files][ 1.8 KiB/ 1.8 KiB]
Operation completed over 1 objects/1.8 KiB.
$ gsutil -i [REDACTED] iam.gs-serviceaccount.com cat gs://[REDACTED]
/passwd
WARNING: This command is using service account impersonation. All API calls will be executed as [REDACTED]
iam.gs-serviceaccount.com].
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin

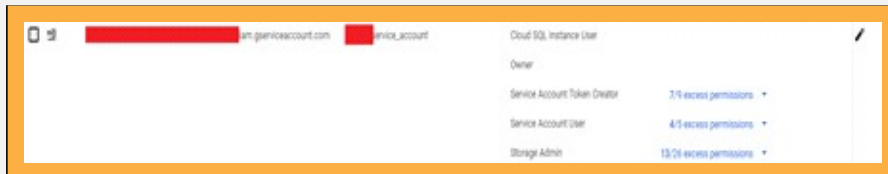
```

**RECOMMENDED REMEDIATION #5**
**CRITICALITY: CRITICAL (27)**
**MISCONFIGURED SERVICE ACCOUNT**
**LOCATION: SESSION MANAGEMENT**

Review and remove the bucket storage permission from the service account.

**VULNERABILITY #6****CRITICALITY: VERY HIGH (24)****MORE THAN REQUIRED  
PERMISSIONS****LOCATION: GCP IAM**

Authorization vulnerability and principles of least privilege are violated here. This service account is given more roles than it is required for its function, and these roles can lead to perform impersonation, letting users access files and folder without having permission to do so at the instance level.

**RECOMMENDED  
REMEDiation #6****CRITICALITY: VERY HIGH (24)****MORE THAN REQUIRED  
PERMISSIONS****LOCATION: GCP IAM**

Follow the principles of least privilege and only allow required permissions on this service account. Do not permit additional permissions that are not required by the role.

**VULNERABILITY #7****CRITICALITY: CRITICAL (27)****INSTANCE MISCONFIGURATION****LOCATION: SQL INSTANCE**

Here the permissions given to the SQL instance are not limited and access to bucket storage can lead users who do not have permissions to access SQL to export the entire SQL database:

```
gcloud auth list
Credentialed Accounts
ACTIVE ACCOUNT
* [REDACTED] iam.gserviceaccount.com

To set the active account, run:
$ gcloud config set account `ACCOUNT`

[REDACTED] $ gcloud sql connect [REDACTED] --user=root
Allowlisting your IP for incoming connection for 5 minutes...done.
Connecting to database with SQL user [root].Enter password:
ERROR 1045 (28000): Access denied for user 'root'@[REDACTED]

[REDACTED] $ gcloud sql export csv [REDACTED] gs://[REDACTED] download --query="SELECT *
> FROM [REDACTED] " --database=[REDACTED]_db
Exporting Cloud SQL instance...done.
Exported [https://sqladmin.googleapis.com/sql/v1beta4/projects/[REDACTED]] to [gs://[REDACTED]
[REDACTED] download].
[REDACTED] $ gsutil cat gs://[REDACTED] download
[REDACTED]
",1
2
3
4
5
",6
3",7
```

**RECOMMENDED  
REMEDATION #7****CRITICALITY: CRITICAL (27)****INSTANCE MISCONFIGURATION****LOCATION: SQL INSTANCE**

Remove the permission of SQL instance to access the storage buckets.

VULNERABILITY #8

CRITICALITY: VERY HIGH (24)

PASSWORD COMPLEXITY

LOCATION: GCP SQL SERVER

Root access to the SQL server is enabled and password used is less than 6 characters containing only lower-case alphabets.

```
m [REDACTED] ~$ gcloud sql connect [REDACTED] --user=root
Allowlisting your IP for incoming connection for 5 minutes...done.
Connecting to database with SQL user [root].Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 52
Server version: 5.7.36-google (Google)

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql> _
```

RECOMMENDED  
REMEDATION #8

CRITICALITY: VERY HIGH (24)

PASSWORD COMPLEXITY

LOCATION: GCP SQL SERVER

Remove the root access via MySQL until absolutely necessary. Also, put in place a strong password policy.

**VULNERABILITY #10****CRITICALITY: HIGH (14)****PUBLIC KEY CRYPTOGRAPHY  
VS PASSWORD****LOCATION: APPLICATION  
INSTANCE**

SSH to the instance is possible via password. Since the instance is of critical sensitivity and importance to the environment, it should not utilize a password authentication mechanism, but instead public key cryptography.

```
DESKTOP [REDACTED]:~$ ssh -i [REDACTED] m [REDACTED] 140
Warning: Identity file [REDACTED] not accessible: No such file or directory.
m [REDACTED].140's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1072-gcp x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of [REDACTED]

System load:  0.0      Processes:    114
Usage of /:   [REDACTED] GB   Users logged in:  0
Memory usage: 20%     IP address for ens4: [REDACTED]
Swap usage:   0%

 * Super-optimized for small spaces - read how we shrank the memory
   footprint of MicroK8s to make it the smallest full K8s around.

https://ubuntu.com/blog/microk8s-memory-optimisation

2 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

New release '20.04.4 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: [REDACTED]
```

**RECOMMENDED  
REMIEDIATION #10****CRITICALITY: HIGH (14)****PUBLIC KEY CRYPTOGRAPHY  
VS PASSWORD****LOCATION: APPLICATION  
INSTANCE**

Public/Private keys and public key cryptography infrastructure should replace password authentication.



**VULNERABILITY #11**
**CRITICALITY: HIGH (14)**
**PASSWORD POLICY OPTIMIZATION**
**LOCATION: APPLICATION INSTANCE**

The password policy for users of the instance is not optimized. Passwords with less than 6 characters and only alphabets are allowed. Furthermore, password expiration days are set to default, 99999, and should be shortened to allow for password rotation.

```
m [REDACTED] $ sudo cat /etc/pam.d/common-password | grep password
# /etc/pam.d/common-password - password-related modules common to all services
# used to change user passwords. The default is pam_unix.
# The "sha512" option enables salted SHA512 passwords. Without this option,
password [success=1 default=ignore] pam_unix.so obscure sha512
password requisite pam_deny.so
password required pam_permit.so
```

```
m [REDACTED] $ sudo cat /etc/login.defs | grep PASS_
# PASS_MAX_DAYS Maximum number of days a password may be used.
# PASS_MIN_DAYS Minimum number of days allowed between password changes.
# PASS_WARN_AGE Number of days warning given before a password expires.
PASS_MAX_DAYS 99999
PASS_MIN_DAYS 0
PASS_WARN_AGE 7
#PASS_CHANGE_TRIES
#PASS_ALWAYS_WARN
#PASS_MIN_LEN
#PASS_MAX_LEN
```

**RECOMMENDED REMEDIATION #11**
**CRITICALITY: HIGH (14)**
**PASSWORD POLICY OPTIMIZATION**
**LOCATION: APPLICATION INSTANCE**

Make password policy strong for example set minimum length to 12 while using upper case, lower case, numbers and special characters within.

**VULNERABILITY #11**
**CRITICALITY: HIGH (14)**
**OPEN PORTS AND LOGGING OFF**
**LOCATION: GCP FIREWALL**

There present several default ports and other ports maybe opened during pre-production environment which are still present for example testservices which lets TCP ports 1111 and 1234 publicly accessible. On top of this only secure protocols are marked with logs "On". It is not a good practice to turn logging off for ingress ports especially when they are publicly accessible.

**Filter** Enter property name or value

<input type="checkbox"/>	Name	Type	Targets	Filters	Protocols / ports	Action	Priority	Network ↑	Logs
<input type="checkbox"/>	default-allow-http	Ingress	http-server	IP ranges: 0.0.0.0/0	tcp:80	Allow	1000	default	Off
<input type="checkbox"/>	default-allow-https	Ingress	https-server	IP ranges: 0.0.0.0/0	tcp:443	Allow	1000	default	On
<input type="checkbox"/>	httppython	Ingress	Apply to all	IP ranges: 0.0.0.0/0	tcp:1234	Allow	1000	default	Off
<input type="checkbox"/>	testservices	Ingress	testservice	IP ranges: 0.0.0.0	tcp:1111, 1234	Allow	1000	default	Off
<input type="checkbox"/>	default-allow-icmp	Ingress	Apply to all	IP ranges: 0.0.0.0/0	icmp	Allow	65534	default	Off
<input type="checkbox"/>	default-allow-internal	Ingress	Apply to all	IP ranges: 10.128.0.0/9	tcp:0-65535 udp:0-65535 icmp	Allow	65534	default	Off
<input type="checkbox"/>	default-allow-rdp	Ingress	Apply to all	IP ranges: 0.0.0.0/0	tcp:3389	Allow	65534	default	Off
<input type="checkbox"/>	default-allow-ssh	Ingress	Apply to all	IP ranges: 0.0.0.0/0	tcp:22	Allow	65534	default	On

**RECOMMENDED REMEDIATION #11**
**CRITICALITY: HIGH (14)**
**OPEN PORTS AND LOGGING OFF**
**LOCATION: GCP FIREWALL**

Turn off all nonsecure ports such as HTTP (80) and ports meant for pre-production. Turn on filtering logging on those ports which are open.



## RECOMMENDATIONS FOR BEST PRACTICES

### RECOMMENDATION [1] UNATTACHED STORAGE BUCKET

**CRITICALITY: (N.A.)**

There are unattached disks with prefix “test” found. These unattached disks are currently participating in a significant amount of monthly billing, consider backing them up and removing them if they are no longer in utilization.

<input type="checkbox"/>	Status	Name ↑	Type	Size	Zone(s)	In use by	Snapshot schedule	Actions
<input type="checkbox"/>	✓	test-1	Balanced persistent disk	40 GB	us-central1-a		None	⋮
<input type="checkbox"/>	✓	test-2	Balanced persistent disk	45 GB	us-central1-a		None	⋮

## RECOMMENDATIONS FOR BEST PRACTICES

### RECOMMENDATION [2] NO AUTOMATIC SNAPSHOTS

CRITICALITY: (N.A.)

Schedule periodic snapshots of the instances which would assist in data recovery or rolling back to previous versions.

<input type="checkbox"/>	Status	Name ↑	Type	Size	Zone(s)	In use by	Snapshot schedule	Actions
<input type="checkbox"/>	✓	[REDACTED]	Balanced persistent disk	3GB	europa-west3-c	[REDACTED]	None	⋮
<input type="checkbox"/>	✓	[REDACTED]	Balanced persistent disk	3GB	europa-west3-c	[REDACTED]	None	⋮

## SECURITY MATRIX

No.	Interface	Attack Path	Result
1.	GCP Bucket	Misconfigured permissions on Google bucket can lead to privilege escalation.	Exploitable
2.	Google Instance	Miconfigured service let a user export sensitive data and read which he or she can't normally.	Exploitable
3.	GCP IAM	Service account is giving some storage and admin permissions which is not a good practice.	N.A.
4.	SQL Server	Misconfigured permission to storage handling liked to SQL can let a user to export data and read it wehich he or she can't normally.	Exploitable
5.	IAM	Password complexity is very low, have demonstrated accessing services like SQL server by burteforcing the password easily.	Exploitable
6.	SSH	SSH allows connection via password when key is not provided. Since password complexity is low it is not advicable.	N.A.
7.	GCP Firewall	There are unsued open ports and services like filtering and logging is off for a number of ports.	N.A.

## KNOWN VULNERABILITIES / END OF LIFE

No.	CVE	Affected	CVE - Details	CVSS	Result
1.	CVE-2015-9251	Jquery 2.1.1	jQuery before 3.0.0 is vulnerable to Cross-site Scripting (XSS) attacks when a cross-domain Ajax request is performed without the dataType option, causing text/javascript responses to be executed.	6.1	Exploitable

## RECOMMENDATIONS

<b>Recommendations</b>	<b>Description</b>	<b>Required immediate remediation</b>
REC [1]: Unattached google storage buckets.	There are unattached disks with prefix "test" found. These unattached disks are currently participating in a significant amount of monthly billing, consider backing them up and removing them if they are no longer in utilization.	NO
REC [2]: No automatic snapshots.	Schedule periodic snapshots of the instances which would assist in data recovery or rolling back to previous versions.	NO

## APPENDIX

### 1. CRITICALITY RATING:

Listed below are the vulnerability ratings for the first two vulnerabilities. This section has been redacted, please refer to the full report for criticality ratings for all the vulnerabilities found.

#### 1.A VULN [1]:

Factor	Value	Points
Time	<= 1 week	15
Expertise	Layman	8
Knowledge required	Restricted information	7
Access to product by	Moderate	1
Type of equipment	Standard	2
<b>Total</b>	<b>33</b>	

#### 1.B VULN [2]:

Factor	Value	Points
Time	<= 1 week	15
Expertise	Competent	6
Knowledge required	Restricted information	7
Access to product by	Easy	4
Type of equipment	Standard	2
<b>Total</b>	<b>34</b>	

## 2. CRITICALITY REFERENCE TABLE:

Factor	Value	
Time taken for the exploitation	<= 1 day	18
	<= 1 week	15
	<= 2 weeks	13
	<= 1 month	10
	<= 2 months	7
	<= 3 months	4
	<= 4 months	2
	<= 5 months	1
	>5 months	0
Attacker skills	Layman	8
	Competent	6
	Expert	3
	Multiple experts	0
Knowledge required by the attacker	None	11
	Restricted information	7
	Sensitive information	3
	Critical information	0

Factor	Value	
Access to the product by the attacker	Not necessary/unlimited	10
	Easy	4
	Moderate	1
	Difficult	0
	None	N.A.
Type of equipment needed	None/ standard	2
	Specialised software	0

### 3. TOOLS REFERENCED:

Tool	Version
Burp Suite professional	2021.4
Nmap	7.4p
Firefox browser	21
GCPBucketBrute	N.A.
gcloud	N.A.



#### 4. ACRONYMS:

Acronyms	Full Form
SSH	Secure Shell
HTTP	HyperText Transfer Protocol
HTTPS	Secure HyperText Transfer Protocol
GCP	Google Coud Platform