Hivint

Project Ref. JOB20295

By Harold Zang

LEXER ANNUAL PENETRATION TEST REPORT

This document details the security posture of the Lexer External Facing Systems based on the findings identified by Hivint during the External Penetration Test and AWS Infrastructure Security Review performed in August 2020.

1. Summary

Hivint conducted a penetration test of the Lexer's External Facing Systems. The purpose of the security test was to assess the security posture of Lexer External Facing Systems against attacks commonly used by external attackers from the public Internet.

Hivint performed a verification of the remediation effort in August 2020. This document reflects the findings that were validated as fixed during the subsequent re-testing.

The Lexer Client Dashboard is considered secure and robust against common web application and cloud infrastructuresecurity vulnerabilities.

Target Systems

- Lexer Client Dashboard – clients.lexer.io
- Lexer API – api.lexer.io
- Lexer AWS Configuration

Risk Level

Remediated

20 August 2020

Key Strengths

- The Lexer application has robust input validation and output encoding implemented preventing it against common web application injection attacks such as Cross-site Scripting (XSS) and SQL Injection
- The Lexer application has strong authentication and strict authorisation implemented preventing against anonymous and unauthorised access to sensitive application functionality and data
- The Lexer application does not unnecessarily disclose internal information
- The Lexer application server only supports the latest TLS protocols and ciphers that are considered cryptographically secure
- The Lexer cloud infrastructure is properly configured preventing against unauthorised access to AWS resources and the data stored within

Table of Contents

2. How to Read this Document

3. Priority of Weakness

This section provides the priority of the findings identified during the security assessment. The priority is based on the rated risk for each security issue.

Status	Risk	Ref.	Weakness
Remediated	Medium	APP-1	Insufficient Authentication: Exposed Slack Messaging Endpoints
Remediated	Low	APP-2	Insufficient Authentication: Slack Channels Enumeration
Remediated	Low	APP-3	Information Leakage: Internal Information Disclosure
Remediated	Low	INFRA-1	System Misconfiguration: Outdated SSL Version 3 Supported
Remediated	Best Practice	INFRA-2	Insufficient Authentication: Credentials Older than 90 days are Still in Use
Remediated	Best Practice	INFRA-3	Insufficient Authentication: Access Keys Older than 90 Days

4. Detailed Findings

This section provides detailed descriptions and analysis of the security issues identified during the security assessment of the Lexer's External Facing Systems.

4.1. Application Penetration Test

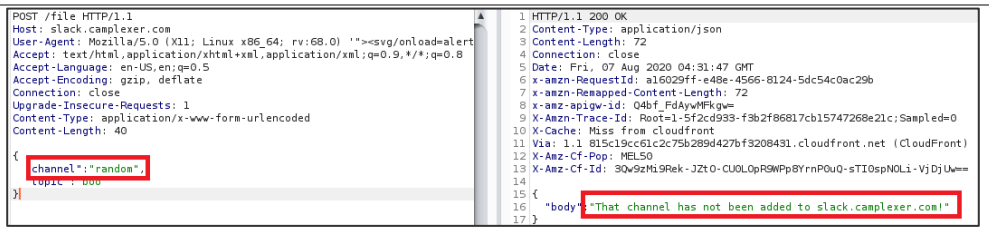
The following security issues were identified during the application review.

APP-1	Insufficient Authentication: Exposed Slack Messaging Endpoints
Description	Application security is fundamentally reliant on the effective implementation of authentication controls. Insufficient authentication occurs when a web application permits an attacker to access content or functionality without having to properly authenticate. Hivint identified that the Lexer messaging end point can be accessed without authentication. A lack of user verification may allow an attacker to send unauthorised messages to Lexer Slack channels (slack.camplexer.com). In addition, Hivint was also able to enumerate available Slack channels. (See **APP-2**)
Proof of Concept	1. Open a Terminal client and send the following payload: `curl https://slack.camplexer.com -XPOST -d '{"channel": "sendingtest", "message": "hello world", "username": "Lily", "topic": "boo", "icon_emoji": ":lexer:", "attachments": []}'` 2. Observe a response indicating that the message was successfully sent as demonstrated in the screenshot below. 
Consequence	**Moderate:** Without authentication, an attacker is able to use the end point to send messages to the Lexer Slack platform. A malicious user might be able to use it to send phishing messages. A successful phishing attack may lead to the attacker gaining unauthorized access to other Lexer systems. Hivint was not able to verify if any of our messages were actually sent it to the Slack. However, Lexer confirmed the validity of the enumerated Slack channels.
Likelihood	**Possible:** The end point is exposed on the Internet without authentication. It is considered trivial for an attacker to craft valid requests to the Slack server.
Risk	Medium Remediated Authorisation has been implemented on the affected Slack endpoints.
Remediation	Do not unnecessarily exposed the Slack endpoints on the Internet. Implement network access restrictions if there's a requirement for it. In addition, implement authentication on the exposed Slack end points.

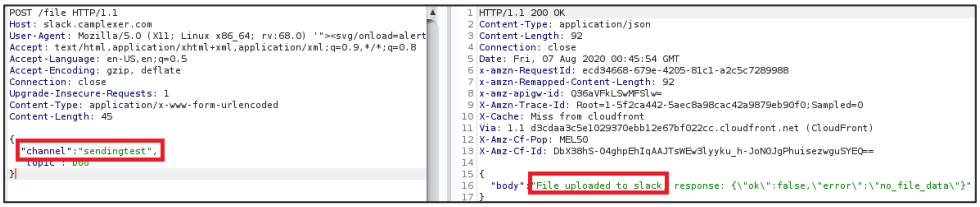
Reference(s)	https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

APP-2	Insufficient Authentication: Slack Channels Enumeration
Description	Application security is reliant on the effective implementation of authentication controls. Insufficient authentication processes that leak username information for example, allows an attacker to perform a targeted attack against the authentication mechanism. The Lexer Slack messaging end point responds differently when an existing channel is requested compared to when a non-existing channel is submitted. This allows an attacker to perform a targeted attacks, for example sending of malicious content against the enumerated Lexer Slack channels.

Proof of Concept	1. Send the following request to the automated tool, Intruder that is part of the Burp Suite web interception proxy.
	
	`POST /file HTTP/1.1`
	`Host: slack.camplexer.com`
	`User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0)`
	`Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8`
	`Accept-Language: en-US,en;q=0.5`
	`Accept-Encoding: gzip, deflate`
	`Connection: close`
	`Upgrade-Insecure-Requests: 1`
	`Content-Type: application/x-www-form-urlencoded`
	`Content-Length: 45`
	
	`{`
	`"channel": "random",`
	`"topic": "boo"`
	`}`
	
	2. Compare inputs to the channel field.
	
	3. Observe the application response regarding the available status of the channel.
	
	The following screenshot shows the response when a non-existing channel is submitted to the application:

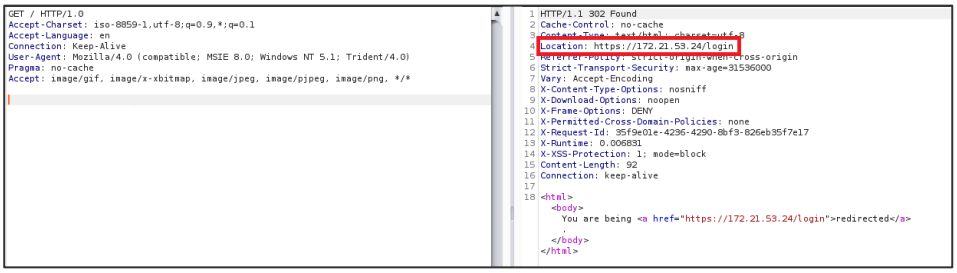


The following screenshot shows the response when an existing channel is submitted to the application:



Consequence	**Minor:** The disclosed Slack channels do not lead to direct compromise of the Slack channel but may assist an attacker in performing targeted attacks against Lexer employees.
Likelihood	**Possible:** Tools are publicly available to aid in enumerating the Slack channels in a short amount of time.
Risk	Low **Remediated** Authorisation has been implemented on the affected Slack endpoints.
Remediation	Configure the endpoint to return generic messages that do not disclose the validity of Slack channels.
Reference(s)	https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/03-Identity_Management_Testing/04-Testing_for_Account_Enumeration_and_Guessable_User_Account

APP-3	Information Leakage: Internal Information Disclosure
Description	Information leakage refers to a situation in which a system reveals internal and potentially sensitive data, often through incorrect system configurations. Several Lexer Internet facing end points disclose internal information such as: internal IP address, deployment bash scripts and a messaging end point in the application's header and response body. This may aid a malicious attacker in identifying information about the deployment bash scripts, internal IP address and a messaging end point in use to understand the internal structure of the Lexer web application and determine known vulnerabilities.

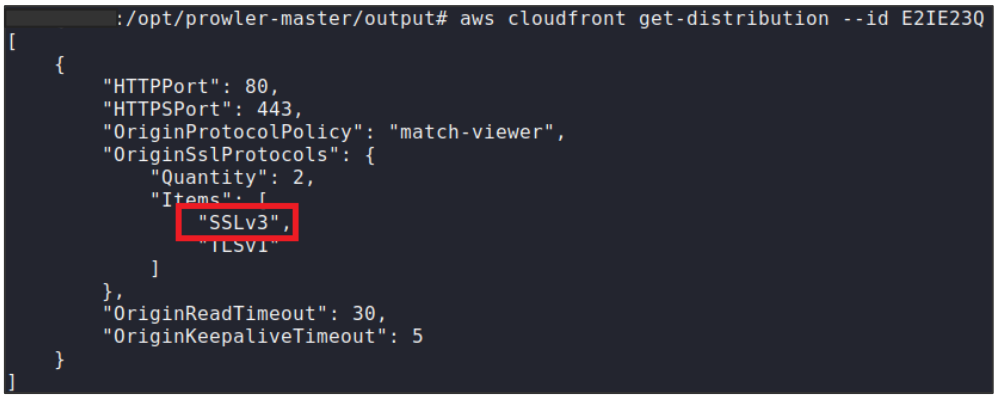
Proof of Concept	1. Navigate to the following URL using a web browser with a web interception proxy. https://clients.lexer.io
	
	2. Intercept the request and send the following payload.
	
	`GET / HTTP/1.0`
	`Accept-Charset: iso-8859-1,utf-8;q=0.9,*;q=0.1`
	`Accept-Language: en`
	`Connection: Keep-Alive`
	`User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)`
	`Pragma: no-cache`
	`Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, */*`
	
	3. Observe an internal IP address disclosed in the HTTP response header as shown in the screenshot below:
	 
	GET / HTTP/1.0
	Accept-Charset: iso-8859-1,utf-8;q=0.9,*;q=0.1
	Accept-Language: en
	Connection: Keep-Alive
	User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)
	Pragma: no-cache
	Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, */*
	
	HTTP/1.1 302 Found
	Cache-Control: no-cache
	Content-Type: text/html; charset=utf-8
	Location: https://172.21.53.24/login
	Referrer-Policy: strict-origin-when-cross-origin
	Strict-Transport-Security: max-age=31536000
	Vary: Accept-Encoding
	X-Content-Type-Options: nosniff
	X-Download-Options: noopen
	X-Frame-Options: DENY
	X-Permitted-Cross-Domain-Policies: none
	X-Request-Id: 35f9e01e-4236-4290-8bf3-826eb35f7e17
	X-Runtime: 0.006831
	X-XSS-Protection: 1; mode=block
	Content-Length: 92
	Connection: keep-alive
	
	<html>
	<body>
	You are being redirected
	.
	</body>
	</html>

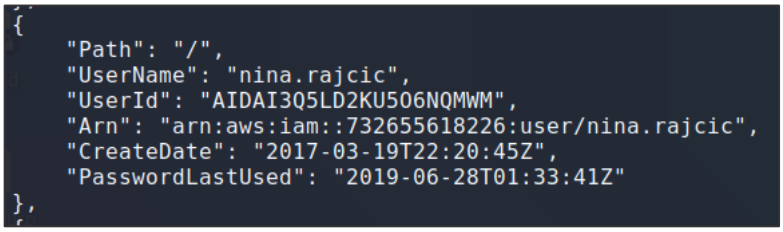
Consequence	**Minor:** While the information disclosed does not lead to a direct compromise of the application itself, the information disclosed will aid an attacker in identifying vulnerabilities within the utilised technology in order to craft targeted attack against the server.
Likelihood	**Unlikely:** It is considered trivial to browse to the pages unnecessarily disclosing internal IP address and deployment bash scripts. However, further exploitation of the affected system would require an attacker to first identify the vulnerability found in a particular server or software version.
Risk	Low Remediated

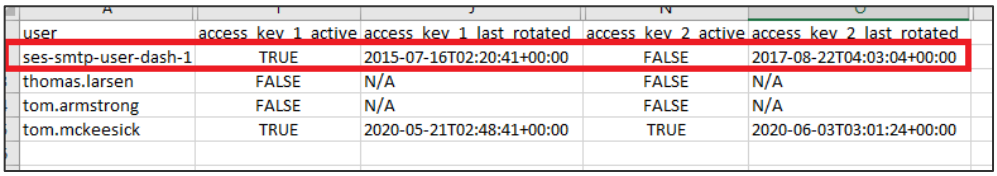
	Lexer was notified of this security misconfiguration testing, and a patch was applied immediately.
Remediation	Configure the application to not unnecessarily disclose internal information such as internal IP address and delete the deployment bash scripts. These information may assist in fingerprinting the target system.
Reference(s)	https://www.owasp.org/index.php/Testing_for_Web_Application_Fingerprint_(OWASP-IG-004)

4.2. Infrastructure Security Assessment

The following security issues were identified during the infrastructure review.

INFRA-1	System Misconfiguration: Outdated SSL Version 3 Supported
Description	Application misconfigurations are most often caused by unnecessary features enabled by default. These default configurations if left enabled, may provide an avenue for malicious attackers to bypass authentication methods, or gain access to system information to inform their attacks. Hivint identified a Lexer CloudFront instance (E2IE23QMWIGIPH) that supports the outdated SSL version 3.
Proof of Concept	1. Open the terminal and run the following command: `aws cloudfront get-distribution --id E2IE23QMWIGIPH --query 'Distribution.DistributionConfig.Origins.Items[*].CustomOriginConfig'` 2. Observe SSLv3 is enabled as shown in the screenshot below: 
Consequence	**Moderate:** A successful traffic interception attack against Lexer application would result in compromise of sensitive information or session information.
Likelihood	**Rare:** The attacker is required to be strategically located in the communication path between the user and Lexer application in order to intercept and decrypt messages.
Risk	Low **Remediated** Hivint verified that this has been fixed and SSL version 3.0 is no longer supported.
Remediation	Ensure that SSLv3 support is not supported on the CloudFront instance
Reference(s)	https://wiki.mozilla.org/Security/Server_Side_TLS#Recommended_configurations

INFRA-2	Insufficient Authentication: Credentials Older than 90 days are Still in Use
Description	Application security is fundamentally reliant on the effective implementation of authentication controls. Insufficient authentication occurs when a web application permits an attacker to access content or functionality without having to properly authenticate. Hivint identified that some AWS users has not logged in using their password in over 90 days and that these passwords are still enabled.
Proof of Concept	1. Open the terminal and run the following command: `aws iam list users` 2. Observe the following example account that has not logged in during the last 90 days or have not used their password at all.  Affected Users: • aya.cross • nina.rajcic • thomas.larsen
Consequence	While this doesn't lead to a direct compromise, removing these credentials if unused, will reduce the window of opportunity that a compromised or abandoned account will be used. This is a Best Practice recommendation and therefore does not have a consequence rating.
Likelihood	This is a Best Practice recommendation and therefore does not have a likelihood rating.
Risk	**Best Practice** **Remediated** Hivint verified that the password of the user "aya.cross" has been updated; The accounts "nina.rajcic" and "thomas.larsen" "password_enabled" attribute has been updated to `false`
Remediation	Ensure that all inactive accounts and credentials are disabled or removed
Reference(s)	CIS Amazon Web Services Foundations Benchmark section 1.3

INFRA-3	Insufficient Authentication: Access Keys Older than 90 Days
Description	Application security is fundamentally reliant on the effective implementation of authentication controls. Insufficient authentication occurs when a web application permits an attacker to access content or functionality without having to properly authenticate. Hivint identified that two AWS access keys have not been rotated in the last 90 days.
Proof of Concept	1. Navigate to Lexer AWS Management Console with valid credential 2. Click “Services” 3. Click “IAM” 4. Click “Credential Report” 5. Download the report. 6. Observe that “ses-smtp-user-dash-1” has not rotated access keys 1 or 2 in over 90 days. 
Consequence	While this doesn’t lead to a direct compromise, rotating these access keys will reduce the window of opportunity for old, leaked or compromised access keys to be used. This is a Best Practice recommendation and therefore does not have a consequence rating.
Likelihood	This is a Best Practice recommendation and therefore does not have a likelihood rating.
Risk	**Best Practice** **Remediated** Hivint verified that the user “ses-smtp-user-dash-1” has been deleted.
Remediation	Ensure that access key rotation occurs every 90 days and if the keys are no longer in use, they are revoked.
Reference(s)	CIS Amazon Web Services Foundations Benchmark section 1.4

user	access_key_1_active	access_key_1_last_rotated	access_key_2_active	access_key_2_last_rotated
ses-smtp-user-dash-1	TRUE	2015-07-16T02:20:41+00:00	FALSE	2017-08-22T04:03:04+00:00
thomas.larsen	FALSE	N/A	FALSE	N/A
tom.armstrong	FALSE	N/A	FALSE	N/A
tom.mckeesick	TRUE	2020-05-21T02:48:41+00:00	TRUE	2020-06-03T03:01:24+00:00

Appendix I – Assessed Targets

As part of Lexer security assurance process, the following systems were assessed to determine the security posture of the External Facing Systems.

- Web Application Security Assessment
    - clients.lexer.io
    - api.lexer.io
- Cloud Security Assessment
    - AWS Infrastructure

Appendix II – Project Schedule

The following is the Hivint security assessment schedule and roles and responsibilities for this engagement:

Date	Name	Role and Responsibility
3 Aug 2020 – 14 Aug 2020	Jamie Ooi	Project Management
3 Aug 2020 – 14 Aug 2020	Harold Zang	Technical Security Testing
14 Aug 2020	Sarath Nair	Quality Assurance
20 Aug 2020	Harold Zang	Remediation Test

Appendix III – Test Methodology

Application Testing – Test Cases

Hivint has developed an application testing methodology that can be adapted to a range of security testing targets and with consideration of a range of industry leading benchmarks and approaches:

- Open Source Security Testing Methodology Manual (OSSTMM) v3
- SANS/MITRE Common Weakness Enumeration (CWE) Top 25
- Open Web Application Security Project (OWASP) Top 10 Vulnerabilities
- Open Web Application Security Project (OWASP) API Security Top 10
- Web Application Security Consortium (WASC)

Through building our methodology around Weaknesses rather than Attacks, we can ensure that the methodology remains relevant for a broad spectrum of system types.

We conduct our testing using a structured approach. Our testing process involves initial application familiarisation – that is, getting a thorough understanding of how the system works, how the security elements are intended to operate, and the key business logic underpinning any core transactional functionality – followed by in-depth and comprehensive assessment of the technology itself.

The test cases described below are used as a starting point for response and behaviour analysis, with the responses then used to guide subsequent phases of analysis and attack.

Our core application security testing model is based around the WASC Threat Classification view of Weaknesses. This approach allows for the key issues with web applications to be analysed, while ensuring that an 'all threats' approach is taken as to *how* that weakness could arise.

Ref.	Weakness	OWASP Top 10 X-Ref[1]
AW1	Application/Server Misconfiguration	2017-A6 – Security Misconfigurations 2017-A9 – Using Components with Known Vulnerabilities 2019-API7 – Security Misconfiguration
AW2	Directory Indexing	2017-A6 – Security Misconfigurations 2019-API7 – Security Misconfiguration
AW3	Improper Filesystem Permission	2017-A5 – Broken Access Control 2019-API1 – Broken Object Level Authorization 2019-API5 – Broken Function Level Authorization
AW4	Improper Input Handling	2017-A1 – Injection 2017-A4 – XML External Entities (XXE) 2017-A7 – Cross-Site Scripting (XSS) 2017-A8 – Insecure Deserialization 2019-API8 - Injection

[1] The Open Web Application Security project (OWASP): https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

Ref.	Weakness	OWASP Top 10 X-Ref[1]
AW5	Improper Output Handling	2017-A1 – Injection 2017-A7 – Cross-Site Scripting (XSS) 2013-A10 – Unvalidated Redirects and Forwards 2019-API8 – Injection
AW6	Information Leakage	2017-A3 – Sensitive Data Exposure 2019-API3 – Excessive Data Exposure
AW7	Insecure Indexing	2017-A6 – Security Misconfigurations 2019-API7 – Security Misconfiguration
AW8	Insufficient Anti-automation	2017-A2 – Broken Authentication 2017-A6 – Security Misconfigurations 2019-API2 – Broken User Authentication 2019-API7 – Security Misconfiguration 2019-API4 – Lack of Resource & Rate limiting
AW9	Insufficient Authentication	2017-A2 – Broken Authentication 2019-API2 – Broken User Authentication
AW10	Insufficient Authorisation	2017-A5 – Broken Access Control 2019-API1 – Broken Object Level Authorization 2019-API5 – Broken Function Level Authorization 2019-API6 – Mass Assignment
AW11	Password Circumvention	2017-A2 – Broken Authentication 2019-API2 – Broken User Authentication
AW12	Insufficient Process Validation	-
AW13	Insufficient Session Expiration	2017-A2 – Broken Authentication 2019-API2 – Broken User Authentication
AW14	Insufficient Transport Layer Protection	2017-A6 – Security misconfigurations 2017-A9 – Using Components with Known Vulnerabilities 2019-API7 – Security Misconfiguration
AW15	Insufficient Auditing and Logging	2017-A10 – Insufficient Logging & Monitoring 2019-API10 – Insufficient Logging & Monitoring

Infrastructure Testing – Test Cases

Infrastructure security testing involves specialist consultants attempting to compromise a target system using the same techniques commonly used by malicious attackers, focused on infrastructure components such as servers, operating systems, network and security devices.

Infrastructure penetration tests are generally combined with application tests due to the significant prevalence of application level vulnerabilities and compromises originating from this source. However, infrastructure level penetration tests and vulnerability scans continue to be of value to identify misconfiguration of devices, out of date components and missing patches.

Our infrastructure security assessment process uses a 'drop in' scanning system, and runs a series of scans to identify key infrastructure security issues as detailed in the test cases below. Based on the data identified from these scans, additional testing activities may be discussed with the client to provide concrete demonstration of vulnerability and removal of false positives.

Ref.	Weakness
IW1	Software Flaws
IW2	System Misconfiguration (Servers)
IW3	System Misconfiguration (Security Devices)
IW4	Information Leakage

This usually follows the following process:

- **Network Discovery**: The purpose of this step is to discover and map out the local infrastructure of the target network. At the end of the network discovery, the penetration tester should have a basic layout of the local network infrastructure.
- **Target Identification**: This step aims to identify a host of interest. This is usually a specific IP range, or a single host/server with many available open ports and corresponding services. At the completion of the target identification step, the penetration tester would have identified a specific target that is most likely to allow penetration of the target network. This may sometimes include additional infrastructure, such additional subnets, that were discovered during the detailed assessment and analysis.
- **Vulnerability Assessment**: This step includes detailed assessment and analysis of the security posture of the identified target. This includes assessing and analysing the services and software packages running on the identified network, and vulnerabilities that are commonly found on them.
- **Vulnerability Exploitation**: The step requires that the penetration tester perform manual verifications of the vulnerabilities that are commonly found on the available services on the target system. This usually includes attempts to bypass security controls, and the lack of, to perform unauthorised and most often unauthenticated transactions with the vulnerable services identified in the previous step.

- **Network Penetration**: Successful exploitation of the identified vulnerabilities will allow unauthorised penetration of the local network infrastructure and subsequent privilege escalation activities to access sensitive data and functionality.

Security Assessment Toolset

Security assessment tools are software applications that are designed to assist in identification of security vulnerabilities, reducing the time and effort to execute repeat processes. The following tools were used during the security assessment:

- Burp Suite Pro web interception proxy
- Nessus Professional vulnerability scanner
- Nmap network security scanner
- Metasploit exploitation toolkit
- Wireshark network analysis tool
- Sqlmap automated SQL injection auditing tool
- SSLScan SSL configuration scanner
- Recursebuster directory brute forcing tool
- Prowler (AWS CIS Benchmark Tool)

Time Boxing

Many applications would require an unfeasibly large amount of testing to provide coverage of all functions within the application with respect to all user types and the permutations of such users and access. This is particularly the case for systems with a high number of user types and/or privilege levels (as testing every permutation of one account's ability to interact with every other account can create hundreds, or thousands, of such permutations).

As a result, most tests are effectively "time boxed", which means that a set amount of time is allocated for testing based on the assessed risk presented by the application and the budget available, and within that time, test tasks are prioritised based on the areas of highest risk – both the most likely vulnerabilities to exist; and those that would cause the greatest harm.

Constraints

The environment provisioned for the security assessment will influence the results of the test. Where a fragile and sensitive environment is used and where network access controls are present, it may be necessary to take a 'gentler' approach to the test with a corresponding reduction in the level of coverage able to be achieved in a certain time period.

Appendix IV – Risk Assessment

The ISO (International Organisation of Standardisation) 31000 series is a family of risk management standards used widely within various industries as a guideline to internal or external audit programmes. The security assessment adopts the ISO 31000 risk assessment approach, incorporating risk assessment concepts from the MITRE organisations. These form the risk ratings assessed in this report. The following tables provide description of the likelihood, consequence and resulting risk rating used in this security assessment.

The interpretation of the likelihood of an event occurring is described as per below:

Likelihood Rating	Interpretation
Almost certain	The event is expected to occur. (e.g. 1 incident every month)
Likely	The event will probably occur. (e.g. 1 incident every 6 months)
Possible	The event should occur at some time. (e.g. 1 incident every year)
Unlikely	The event could occur at some time. (e.g. 1 incident every 2 years)
Rare	The event may occur only in exceptional circumstances. (e.g. 1 incident every 5 or more years)

Hivint considers the following as contributing factors to the likelihood of an event occurring.

- The **value** of assets contained within the vulnerable system
  E.g. Credit card details or dummy test data
- The **skills** required to successfully exploit the vulnerable system using the vulnerability identified
- The availability of **exploits** on the public domain
- The **complexity** of the exploit
- The **level of access** on the vulnerable system required to exploit the security issue
  E.g. Privileged administrative user or anonymous user

The interpretation of the consequence of an event occurring is described as per below:

Consequence Rating	Sample Interpretation
Insignificant	Little disruption to the user community. Technologies in use will require little/no effort to change. Isolated complaint from individual stakeholder able to be managed via business as usual operations.
Minor	Minor disruption to user community. The ability to provide the required service is impaired. Complaints from key stakeholder requiring management attention.
Moderate	Some inconvenience to the user community. The ability to provide a service is severely compromised. Moderate effort required to implement an alternative solution. Public criticism from key stakeholders regarding the organisation's services or activities.
Major	Noticeable impact on user community. Some core services unavailable. Potential for serious distress or minor injury. Sustained criticism from majority of key stakeholders on suitability of organisation in its current form.
Catastrophic	Community unable to function without significant support. Key technologies no longer available and no viable alternative exists. Potential for major injury or fatalities. Irreparable damage to relationships with key stakeholders and potential for organisation to cease operating in current form.

The resultant risk rating is detailed in the following risk matrix:

	Rare	**Unlikely**	**Possible**	**Likely**	**Almost Certain**
Insignificant	Very Low	Very Low	Very Low	Low	Low
Minor	Very Low	Low	Low	Low	Low
Moderate	Low	Medium	Medium	Medium	Medium
Major	Medium	Medium	High	High	High
Catastrophic	High	High	Extreme	Extreme	Extreme

Appendix V – Revision History

Version	Date	Name	Revision Comment
0.1	14 Aug 2020	Harold Zang	Initial report draft
0.2	14 Aug 2020	Sarath Nair	Internal report review
0.3	14 Aug 2020	Jamie Ooi	Client report release
1.0	21 Aug 2020	Jamie Ooi	Final client release