Vulnerability Assessment Report of

Lancer API

March 2023

**About L&T Technology Services:**

L&T Technology Services Limited (LTTS) is a global leader in Engineering and R&D (ER&D) services. With 399 patents filed for 51 of the Global Top 100 ER&D spenders. Our innovations speak for itself – World’s 1st Autonomous Welding Robot, Solar ‘Connectivity’ Drone, and the Smartest Campus in the World, to name a few. LTTS expertise in engineering design, product development, smart manufacturing, and digitalization touches every area of our lives. With 49 Innovation and R&D design centers globally, we specialize in disruptive technology spaces such as 5G, Artificial Intelligence, Collaborative Robots, Digital Factory, and Autonomous Transport. **LTTS is a publicly listed subsidiary of Larsen & Toubro Limited, the $18 billion Indian conglomerate operating in over 30 countries.**

**Document History:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Var. Rel.**  **No.** | **Release Date** | **Prepared By.**  **Prepared. Dt.** | **Reviewed By**  **Reviewed Dt.** | **Approved By**  **Approval Dt.** | **Remark/Revision Details** |
| 1.0 | 19-03-2021 | Manvitha  Haripriya | Sunil MC |  |  |
| 19-03-2023 | 19-03-2021 |  |

**Table of Contents**

[1. Overview of the project 4](#_Toc130164943)

[2. Vulnerabilities explained in detail. 7](#_Toc130164944)

[**3.1 Broken Functional Level Authorization (Lancer Marketing)** 7](#_Toc130164945)

[**3.2 Unauthorized access (Lancer Manager)** 9](#_Toc130164946)

[**3.3 User Enumeration(Login & Forgot Password)** 9](#_Toc130164946)

[**3.4 Vulnerable API KEY** 12](#_Toc130164947)

[**3.5 Weak Lockout Mechanism** 13](#_Toc130164948)

[3. Abbreviation 15](#_Toc130164949)

[4. Appendix 15](#_Toc130164950)

# Overview of the project

This is an Azure IoT based solution focused on real time device monitoring. This collects data from devices via IOT hub. Interactive dashboards show all collected data in charts. Alerts can be set to notify the users in case of any Device disconnected or other failure via configurable notification rule. This data is also stored for trend data, reporting and analytics. Build new features & services on cloud for enhanced customer experience.

**Objective of the security assessment:**

As a part of this engagement a holistic approach was taken to conduct the Vulnerability Assessment and Penetration Testing on Lancer APIs. During the engagement High, Medium, and Low severity issues were identified with respect to Lancer APIs.

**Approach**

The following approach was taken to make sure the target site is assessed against OWASP Top 10 Vulnerabilities from all possible security perspectives:

Manual Penetration Testing in tandem with penetration testing tools. Some of the tools which were used are listed below:

|  |  |
| --- | --- |
| Target site URL | [http://demoui.azurewebsites.net/](http://demoui.azurewebsites.net/" \t "_blank" \o "http://demoui.azurewebsites.net/)/LancerUserManagement  [http://demoui.azurewebsites.net/](http://demoui.azurewebsites.net/" \t "_blank" \o "http://demoui.azurewebsites.net/)/assetmanagement  https:// demoui.azurewebsites.net /LancerAnalytics/  <http://demoui.azurewebsites.net//mediamanagement>  <http://demoui.azurewebsites.net/>/deviceManagement/v1  [http://demoui.azurewebsites.net/](http://demoui.azurewebsites.net/" \t "_blank" \o "http://demoui.azurewebsites.net/)/LancerNotifications  <http://demoui.azurewebsites.net/>/BrandManagement |
| Browser | Chrome, Firefox |
| Tools | BURP, Postman, RESTAPI |

* + 2. Key Security Policies

OWASP top 10 listed vulnerabilities were used as a reference framework. The following key security aspects were checked:

* Broken object level authorization
* Broken authentication
* Excessive data exposure
* Lack of resources and rate limiting
* Broken function level authorization
* Mass assignment
* Security misconfiguration
* Injection
* Improper assets management
* Insufficient logging and monitoring

**Summary of Findings**

The graph below shows a summary of the number of vulnerabilities found for each impact level for the API testing. Vulnerabilities found are addressed according to priority, findings, analysis, and recommendations from the assessment.

|  |  |  |
| --- | --- | --- |
| **Sr. No** | **Title** | **Risk Rating** |
| 1 | **Broken Functional Level Authorization** | **Medium** |
| 2 | **Unauthorized access** | **Medium** |
| 3 | **Vulnerable API KEY** | **Medium** |
| 4 | **User Enumeration** | **Low** |
| 5 | **Weak Lockout Mechanism** | **Low** |

# Vulnerabilities explained in detail.

|  |  |  |  |
| --- | --- | --- | --- |
| **3.****1 Broken Functional Level Authorization (Lancer Marketing)** | | | |
| **Impact** | Medium | **Risk Rating** | Medium |
| **Ease of Exploit** | Easy |
| **Likelihood** | Medium |
| **Category** | API5:2021 Broken function level authorization | | |
| **URL/Impacted system** | <http://demoui.azurewebsites.net/>LancerNotifications/AddUserNotificationPreference | | |
| **Description:**  Lancer marketing is low privilege user unauthorized to access notifications but able to access and add notifications by breaking function-level authorization this is a security vulnerability that occurs when an API does not properly enforce access controls for its functions or resources. This can allow an attacker to gain unauthorized access to sensitive functionality or data by exploiting flaws in the authorization mechanism. | | | |
| **Impact**: | | | |
| BFLA vulnerability in this application scenario allows a low privilege user lancer marketing can add notifications get added notifications. This could lead to data breaches, data manipulation, other malicious activities. | | | |
|  | | | |
| **Recommendation** | | | |
| * Broken function level authorization can be mitigated by implementing proper authorization mechanism such as role-based access control and only allow operations to users belonging to the appropriate group or role. * Deny access by default unless a resource is meant to be made available to the public. * Properly design and test authorization. | | | |
| **How to recreate the Security defect** | | | |
| * Log in with a low privilege user as Lancer marketing. * The low privilege user Lancer marketing should not have access but able to add and save the notification preferences. | | | |
| **Evidence** | | | |
| Graphical user interface, text, application, email  Description automatically generated  Graphical user interface, text, application  Description automatically generated | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **3.2 Unauthorized access (Lancer Manager)** | | | |
| **Impact** | Medium | **Risk Rating** | Medium |
| **Ease of Exploit** | Easy |
| **Likelihood** | Medium |
| **Category** | API1:2021 Broken object level authorization | | |
| **URL/Impacted system** | https://azdevlancerapims.azureapi.net/LancerUserManagement/user/GetById | | |
| **Description:**  Lancer manager is a low privilege user unauthorized to view higher level admin details but here able to view admin sensitive information. | | | |
|  | | | |
| **Impact**:  Unauthorized user can access some resources and delete the user accounts. | | | |
|  | | | |
| **Recommendation** | | | |
| * Force all requests to go through access control checks. Make sure that all requests pass through an access control verification layer of some kind. Deny access by default unless a resource is meant to be made available to the general public. Properly design and test authorization. | | | |
| **How to recreate the Security defect** | | | |
| * Log in with a low privilege user as manager. * The low privilege user manager can able to view the details of a high privileged user Lancer admin in Users List. | | | |
| **Evidence** | | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **3.3 User Enumeration (Login & Forgot Password)** | | | |
| **Impact** | Medium | **Risk Rating** | Low |
| **Ease of Exploit** | Easy |
| **Likelihood** | Low |
| **Category** | API1:2021 Broken authentication | | |
| **URL/Impacted system** | http://demoui.azurewebsites.net/LancerUserManagement/Authentication | | |
| **Description** | | | |
| In this application scenario when attempting to login and forgot password with wrong credentials it is showing generic error messages. This is a security weakness that allows an attacker to determine the existence of user accounts or user names (through generic error messages) by attempting to register or login to the API with different combinations of credentials. This vulnerability can be exploited to gain access to sensitive data and perform unauthorized actions or escalated privileges. | | | |
| **Impact** | | | |
| An attacker may use the contents of error messages to help launch web application attack through bruteforce. Using this type of error messages hacker can use enumeration technique to pick up valid username. | | | |
| **Recommendation** | | | |
| * One other way to block user enumeration is with a [web application firewall (WAF)](https://www.rapid7.com/fundamentals/web-application-firewalls/). * A good WAF will detect and block single IP address making many of these requests. Some WAFs will drop these requests entirely, others will issue a negative response, regardless of whether the request is valid. | | | |
| **How to recreate the Security defect** | | | |
| * Attempt to login with wrong credentials and capture the request and send. * Observe the response it contains generic error messages which includes any sensitive information about its environment, users, or associated data. | | | |
| **Evidence** | | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **3.4 Vulnerable API KEY** | | | |
| **Impact** | Medium | **Risk Rating** | Medium |
| **Ease of Exploit** | Easy |
| **Likelihood** | Medium |
| **Category** | API7:2021 Security misconfiguration | | |
| **URL/Impacted system** | http://demoui.azurewebsites.net | | |
| **Description** | | | |
| In this application scenario observed Google map API feature through that found one API key. This API key used to access google maps services it is vulnerable as it is public & unprotected. This is the security issue any one can use the key and leads to financial loss to the owner. | | | |
| **Impact** | | | |
| An attacker use vulnerable google map API key and potentially access sensitive data, launch attacks, or incur significantly financial costs to the owner of the API key Hacker may use the API key to generate large volumes of API requests, which can result in financial charges to owner of the key, or use the key to access sensitive information about users or locations. | | | |
| **Recommendation** | | | |
| To Secure the API keys keep them confidential, restricting access to them and using tools google cloud IAM to manage access control to resources and regularly review and rotate API keys to prevent any unauthorized use. | | | |
| **How to recreate the Security defect** | | | |
| * Identified google map feature in <http://demoui.azurewebsites.net> * This API map feature accessed using chunk.js then it hits to server there able to see Vulnerable API key and it is scanned there it is public not protected. | | | |
| **Evidence** | | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **3.5 Weak Lockout Mechanism (Have unlimited login attempts)** | | | |
| **Impact** | Low | **Risk Rating** | LOW |
| **Ease of Exploit** | Easy |
| **Likelihood** | Low |
| **Category** | A07:2021Identification and Authentication Failures | | |
| **URL/Impacted system** | http://demoui.azurewebsites.net/LancerUserManagement/Authentication | | |
| **Description** | | | |
| Lancer web portal have unlimited login attempts after certain wrong password attempts also it is allowing to login with correct credentials. This leads to password guessing attacks. | | | |
| **Impact** | | | |
| Weak lock out mechanism can also lead to DOS (Denial -of- Service) attack where an attacker repeatedly attempts to login to an account using incorrect credentials, causing the account to be locked out and preventing legitimate users from accessing it. | | | |
| **Recommendation** | | | |
| * Limit the number of login attempts allowed before an account is locked out. * Lockout the user for a reasonable amount of time. * Notify the user when their account has been locked out and provide a way to unlock it. * Use CAPTCHAS or other techniques to prevent automated brute force attacks. | | | |
| **How to recreate the Security defect** | | | |
| * Capture Login request and attempt to login with an incorrect password for certain number of attempts. The account have not locked. after certain number of attempts able login with correct credentials as lockout mechanism is not there. | | | |
| **Evidence** | | | |
| **Graphical user interface, text, application  Description automatically generated** | | | |

# Abbreviation

|  |  |
| --- | --- |
| APP | Application |
| HTML | Hyper Text Mark-up Language |
| HTTP | Hypertext transfer protocol |
| LTTS | Larsen & Toubro Technology Services |
| VAPT | Vulnerability Assessment and Penetration Testing |
| SOP | Same Origin Policy |
| OWASP | Open Web Application Security Project |
| IDOR | Insecure direct object references |
| SOP | Same Origin Policy |
| CRLF | Carriage Return Line Feed |
| CSP | Content Security Policy |
| CORS | Cross-origin resource sharing |
| XSS | Cross-site Scripting |
| URL | Uniform Resource Locator |
| XSS | Cross Site Scripting |
| XXE | XML External Entities |