

# SAAS CASE STUDY

**Industry**

SAAS

## Client Background

With over 6000 customers in more than 40 countries, the company is a customer-focused, quickly developing, 100% remote product company. Deloitte has ranked them among India's top 10 fastest-growing IT companies, as well as among the top 100 in Asia Pacific, for the past two years. They're profitable, and venture investors from India and the United States have put money into them.

## Business Challenges

- Protecting the application from being abused to distribute malware.
- Safeguarding 16 million + sales and support agents.
- Prevent supplier product listing data from being accessed or changed.

## Environment

**6K+** Covered Approx. **6000 +** Customers.

**40+** Operates around **40+** countries.

**16M+** High Impact Communication tools for **16 million +** sales and support agents.

## Solution

The company approached Kratikal's Managed Security Services division to identify technical and logical vulnerabilities that may be present in their Web Application Penetration testing. To come up with risk mitigation suggestions based on successful exploitation of these vulnerabilities.

## Approach

- The tests were conducted in accordance with the best practices available in the industry such as Open Web Application Security Project (OWASP).
- Various manual scripts were executed to test their web application
- The test was conducted as a Black Box exercise.
- The test was carried out in a Dummy environment.



## Major Findings

- Stored XSS occurs when data collected from users is not adequately screened, allowing malicious code to run on the website.
- Insecure Direct Object Reference (IDOR) occurs when an application exposes a reference to an internal implementation Object, exposing the identity and format utilized in the storage backend.
- An iframe Injection attack took place, in which iframe tags were placed into a page or other measures were taken to compromise the site visitors' computers.
- Excessive data exposure occurs when a program fails to safeguard sensitive data.
- On the production environment, a .GIT file was discovered that collects sensitive information by requesting the secret metadata directory that version management tool Git produces.
- HTML injection occurs when a user has control over an input point and can inject arbitrary HTML code into a vulnerable web page.
- In the case of forgotten passwords, there are no rate limits, causing the password to be re-entered many times.
- There was no security policy in place to allow for the generation of backup files.
- The WordPress CMS version was out of date, resulting in an XSS problem.
- An outdated version of jQuery, Moments, JS, Slick JS was executed
- The old session was active at the server end even after the Logging off.
- Application was Vulnerable to HTTP Parameter Pollution and Content Spoofing

## Risks

- The user's privacy is jeopardized because the attacker has access to the details of the other team members.
- Users are routed to other malicious websites that are utilized in Phishing and other types of attacks.
- Some APIs expose client email addresses and mobile phone numbers, providing a security concern.
- When combined with other vulnerabilities, GIT Repository files can be beneficial to the attacker.
- Stored XSS allows an attacker to insert malicious code onto a susceptible page.



## Impact- (Impact on the company)

- The corporation could face a lot of financial losses, suffer lawsuits, and have its brand image ruined
- The attacker might utilize the application's capability to redirect users to malicious websites and install malware such as ransomware on their computers and networks.
- Leaking sensitive client information from their application regarding their clients and users can result in massive fines under numerous compliances and local cyber laws in the countries where they work.
- If a successful cyber-attack occurs, you may be locked out of your company's critical databases, with attackers demanding a large ransom to regain access.



## Recommendations

- For stored XSS, never inject untrusted data into HTML style parameter values unless it's in an allowed area.
- Ifor IDOR, utilize a hash function or values to implement access control regulations.
- Accepting URLs from accepted domains as a solution to Iframe Injections; nevertheless, do not use user input for URLs.
- It is advised that no sensitive information be made public and that directory listings be restricted in the web server setup.
- Access to the .GIT file should be restricted considering the sensitivity of design and configuration information before it is published online.
- No sensitive directories should be included in the robots.txt file.
- It is suggested that you configure your web server to prevent.DS Store files from being downloaded.
- To avoid data loss and account takeovers, we advised them to change their application flows.
- The application's vulnerabilities were documented in detail, with explanations of the issue, its cause, and how to fix it.
- For database security and server design, we advised the company on advanced controls and cryptographic techniques (such as obfuscation techniques).


## Kratikal Privacy commitment

Kratikal is dedicated to safeguarding your company from advanced threats, such as data leakage. For this reason, we do not reveal the names of our case study participants.

 +91 9289192210

 sales@kratikal.com

 www.kratikal.com

 A-130, 2<sup>nd</sup> Floor, Sector 63, Noida,  
Uttar Pradesh, India-201307