|  |  |  |
| --- | --- | --- |
| 1. **Vulnerability Name** | **SQL Injection** | **Risk Rating**: High |
| **Description** | SQL Injection is an attack technique used to exploit applications that construct SQL statements from user supplied input. When successful, the attacker is able to change the logic of SQL statements executed against the database. | |
| **Affected Path(s)** | http://apttss.apcfss.in/login.htm;jsessionid=386F89ECCBE60668BE6A3FD62770C1F4 | |
| **Impact** | A wide range of damaging attacks can be delivered via SQL injection, including reading or modifying critical application data, interfering with application logic, escalating privileges within the database and taking control of the database server. | |
| **Evidence/Proof of Concept**  Step 1: Access the 1st URL and navigate to the TTSS Dashboard as shown in below image .enter a Boolean based true condition in hall ticket number field as shown below.  #1 Capture.JPG  **Step 2:** now in TTSS Dashboard window select search employee and enter the data in treasure id and phone number fields and capture the request as show in below image.  Annotation 2019-08-05 091923.png  Annotation 2019-08-05 104707.png  **Step 4:** now incaptured request insert the boolean based malicious sql payload in treasure id and phone number as shown in below image  **Annotation 2019-08-05 094826.png**  **Step 5:** After injecting the malicious payload the application is displaying the data as shown in below image  #3 Capture3.JPG  Annotation 2019-08-05 094918.png | | |
| **Recommendation** | 1. This is a critical vulnerability to have on a web application and should be addressed immediately. User controllable data should be validated before any queries are performed on the database using the data. Blacklisting is an approach which consists of checking the input data for malicious characters but a more effective approach is white listing. White listing consists of only allowing certain characters to be submitted. For example checking if data submitted is alphanumeric and rejecting the request if it is not. Many libraries exist, such as built-in libraries for programming languages and open-source libraries, which can assist you in preventing this vulnerability. | |
| **Management Comments** |  | |