|  |  |  |
| --- | --- | --- |
| 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)** | <https://epass.apcfss.in/DulhanPrint.do>  <https://epass.apcfss.in/GiriPutrikaPrint.do>  <https://epass.apcfss.in/OverseasRegEdit.do>  <https://epass.apcfss.in/SkillUpgradationPrint.do>  <https://epass.apcfss.in/UploadOverseasMarksMemo.do>  <https://epass.apcfss.in/OverseasRegEditForKapu.do>  https://epass.apcfss.in/OverseasRegEdit.do | |
| **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 URL and enter a Boolean based true condition in field as shown below.  mainsite.JPG  Untitled.png  **Step 2**: It is observed that, the application displayed the data as shown below.  Untitled.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** |  | |