

36

## SQL Injection in █████

Share:     State  Resolved (Closed)Disclosed **August 19, 2019 5:51pm +0530**Reported To [U.S. Dept Of Defense](#)

Weakness SQL Injection

Severity  High (7 ~ 8.9)Participants   

Visibility Disclosed (Full)

[Collapse](#)

SUMMARY BY ARINERRON2

See the writeup at <https://aaronesau.com/blog/posts/5> 

TIMELINE · EXPORT

[arinerron2](#) submitted a report to [U.S. Dept Of Defense](#).

Apr 1st (6 months ago)

### Summary


There is an SQL injection vulnerability in █████ in the /█████/recruiter/updapp.aspx page, exploitable through the `app\_id` form parameter.

### Impact

An attacker could use this vulnerability to control the content in the database, exfiltrate information, and obtain remote code execution.

### Step-by-step Reproduction Instructions

1. Visit <https://█████/Gateway/sso.aspx> and sign in. Note that any user can create a user (and any privilege level works for this vulnerability as long as a user is signed in), so this should be considered an unauthenticated vulnerability.
2. With the Network tab of devtools open, visit <https://█████/recruiter/updapp.aspx>
3. Replay the GET request that returned the HTTP 500 error as a POST request with the body `app_id='`. This can be done by right clicking on the request, copying it as cURL, pasting the command in terminal, and appending `-k -X POST --data "app_id='"`.
4. Notice in the response, there is an error: `ORA-01756: quoted string not properly terminated`. This is because the single apostrophe (`'`) caused the SQL query to be syntactically invalid.
5. Replay the request in the same way as shown in #3, but with the body `app_id=''` (this time append `-k -X POST --data "app_id=''"` to the cURL command).
6. Notice in the response, there is an error: `ORA-01722: invalid number`. This is because the double apostrophes (`''`) did not cause the SQL query to be syntactically invalid, but because apostrophes are not numbers, they caused a different error.
7. Repeat step #3 as many times as you like. An odd number of apostrophes (`'`) will cause the SQL query to fail because it is syntactically invalid, and an even number will cause it to fail because it is valid, but apostrophes are not numbers.

I did not want to exploit this to get remote code execution because this is a live production system, but to get RCE, simply execute an SQL query that writes the file at <https://raw.githubusercontent.com/danielmiessler/SecLists/master/Web-Shells/laudanum-0.8/asp/shell.aspx>  to

D:\█████\shell.aspx using the `INTO OUTFILE` syntax, then visit <https://█████/shell.aspx>

### Suggested Mitigation/Remediation Actions

Sanitize everything in the SQL query (use prepared statements), and validate the data before putting it in the query.

Note: I wouldn't have been able to find this vulnerability if it wasn't for the fact that verbose error pages were enabled. Because they were, it leaked source code, and I could see that the SQL injection vulnerability existed before testing.

### Impact

An attacker could use this vulnerability to control the content in the database, exfiltrate information, and obtain remote code execution.



BOT: [U.S. Dept Of Defense](#) posted a comment.  
Greetings from the Department of Defense (DoD),

Apr 1st (6 months ago)

Thank you for supporting the DoD Vulnerability Disclosure Program (VDP).

By submitting this report, you acknowledge understanding of, and agreement to, the DoD Vulnerability Disclosure Policy as detailed at [@DeptofDefense](#).

The VDP Team will review your report to ensure compliance with the DoD Vulnerability Disclosure Policy. If your report is determined to be out-of-scope, it will be closed without action.

We will attempt to validate in-scope vulnerability reports and may request additional information from you if necessary. We will forward reports with validated vulnerabilities to DoD system owners for their action.

Our goal is to provide you with status updates not less than every two weeks until the reported vulnerability is resolved.

Regards,

The VDP Team



[ag3nt-j1](#) updated the severity to High.

Apr 3rd (6 months ago)



[ag3nt-j1](#) changed the status to ○ **Triaged**.  
Greetings,

Apr 3rd (6 months ago)

We have validated the vulnerability you reported and are preparing to forward this report to the affected DoD system owner for resolution.

Thank you for bringing this vulnerability to our attention!

We will endeavor to answer any questions the system owners may have regarding this report; however, there is a possibility we will need to contact you if they require more information to resolve the vulnerability.

You will receive another status update after we have confirmed your report has been resolved by the system owner. If you have any questions, please let me know.

Thanks again for supporting the DoD Vulnerability Disclosure Program.

Regards,

The VDP Team



[ag3nt-z3](#) closed the report and changed the status to ○ **Resolved**.  
Good news!

Apr 11th (6 months ago)

The vulnerability you reported has been resolved and this report is now closed. If you have any further questions or disagree that the report is resolved, please let us know.

Thank you for your time and effort to improve the security of the DoD information network.

Regards,

The VDP Team



[arinerron2](#) requested to disclose this report.

Apr 11th (6 months ago)



[ag3nt-j1](#) agreed to disclose this report.

Aug 19th (2 months ago)



This report has been disclosed.

Aug 19th (2 months ago)

