

682

SQL Injection Extracts Starbucks Enterprise Accounting, Financial, Payroll Database

Share:

State ○ Resolved (Closed)Disclosed **August 6, 2019 11:21am +0530**Reported To **Starbucks**Asset Other non domain specific items
(Other)

Weakness SQL Injection

Bounty \$4,000

Severity 🔴 Critical (9.3)

Participants

Visibility Disclosed (Limited)

[Collapse](#)

SUMMARY BY STARBUCKS



As described in the Hacker Summary, @spaceraccoon discovered a SQL Injection vulnerability in a web service backed by Microsoft Dynamics AX. @spaceraccoon demonstrated that the flaw was exploitable via XML-formatted HTTP payload requests to the server. We appreciate @spaceraccoon's clear and thorough report, which helped us quickly and effectively triage the report and remediate the vulnerability.

SUMMARY BY SPACERACCOON



False Starts

I first came across the endpoint via typical subdomain enumeration. On the surface, it looked like an extremely promising target: a simple HTML file upload form. I began by testing for unrestricted file uploads with PHP shells and such, but it quickly became clear from the verbose error messages that while the files were being sent to the server, they were being processed as XML files and were not saved on the server.

Fortunately, the error messages helped me craft a properly-formatted XML file that was accepted by the server. It appeared to be some kind of accounting database entry as it expected nodes like `MainAccount`, `Credit`, `Debit`, `Invoice` and so on. Moreover, the error messages included references to [Microsoft Dynamics AX](#), an enterprise financial/accounting software platform. At this point, I started testing for XXE attacks. However, it appeared that external entities were blocked and despite multiple attempts at bypasses, I could only achieve a "Billion Laughs" attack that would result in denial of service. This wasn't good enough, so after several more days of trying, I eventually moved on to other targets.

The Lightbulb Moment

More than a month later, I revisited the target. Fortunately, it was still online. This time, I suspected that if the XML input was being entered into a database, I should test for SQL injections. In particular, the `MainAccount` looked promising because it accepted a numerical ID like `<MainAccount>123456</MainAccount>` and was perhaps used in a `WHERE` SQL query.

However, it appeared that the apostrophe `'` was being properly escaped. After a bit of testing, I realized my mistake: the XML format prohibits some characters, including the apostrophe. To include them, you have to enter apostrophes as escaped entities. Instead of `<MainAccount>123456'</MainAccount>`, I had to use `<MainAccount>123456'</MainAccount>`. The server immediately returned a `database error` message - I was on the right path!

With a bit more manual testing, I realized it was possible to craft a time-based SQL injection. I then switched to `sqlmap` with the `--tamper htmleencode` flag to automate my attack. After a few minutes of anxious waiting, `sqlmap` confirmed the exploit and returned the database version: `Microsoft SQL Server 2012`. I was in!

Assessing Impact

So I had an SQL injection - but what if the database was unused or negligible? I decided to test for three things: the type of data in the database, the amount of data, and the recency of data. However, I quickly met a roadblock: as an enterprise database, Microsoft Dynamics AX was massive: a quick check revealed that the database had thousands of tables. I had to find and focus on the main table, but where should I begin?






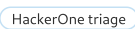






















Fortunately, Microsoft provides documentation online about Dynamics AX. After a bit of research, I found the default main table and the relevant columns. A few minutes later, the answers came in. There were almost a million entries up till the previous year that included real accounting information. Zaheck!! I immediately stopped testing and wrote my report.

Takeaways

- 1. Don't get tunnel vision. Just because it's a file upload, don't focus solely on uploading a reverse shell. Just because it accepts XML files, don't focus solely on XXE.
- 2. Take notes and revisit old targets. Many of my best bugs were from vulnerabilities I found after revisiting an old endpoint. Taking time off can reveal new attack paths.
- 3. Assess impact after the initial exploit, but don't go too far.

Thank you Starbucks team and Hackerone triagers for responding quickly and communicating so well!

TIMELINE · EXPORT

| | | |
|---|--|-------------------------------------|
|  | spaceraccoon submitted a report to Starbucks. | Apr 8th (about 1 year ago) |
|  | spaceraccoon changed the report title. | Apr 8th (about 1 year ago) |
|  | spaceraccoon posted a comment. | Apr 8th (about 1 year ago) |
|  | spaceraccoon posted a comment. | Apr 8th (about 1 year ago) |
|  | glassofbeer  posted a comment. | Apr 9th (about 1 year ago) |
|  | spaceraccoon posted a comment. | Apr 9th (about 1 year ago) |
|  | glassofbeer  changed the weakness. | Apr 9th (about 1 year ago) |
|  | glassofbeer  updated the severity. | Apr 9th (about 1 year ago) |
|  | spaceraccoon posted a comment. | Updated Apr 9th (about 1 year ago) |
|  | spaceraccoon posted a comment. | Apr 9th (about 1 year ago) |
|  | shadegrown changed the scope from Other assets to Other non domain specific items. | Apr 9th (about 1 year ago) |
|  | shadegrown changed the status to  Triaged . | Apr 9th (about 1 year ago) |
|  | spaceraccoon posted a comment. | Updated Apr 10th (about 1 year ago) |
|  | shadegrown posted a comment. | Apr 11th (about 1 year ago) |
|  | Starbucks rewarded spaceraccoon with a \$4,000 bounty. | Apr 11th (about 1 year ago) |
|  | spaceraccoon posted a comment. | Apr 12th (about 1 year ago) |
|  | shadegrown closed the report and changed the status to  Resolved . | Apr 15th (about 1 year ago) |
|  | spaceraccoon posted a comment. | Apr 16th (about 1 year ago) |
|  | spaceraccoon posted a comment. | Apr 16th (about 1 year ago) |
|  | spaceraccoon posted a comment. | Apr 16th (about 1 year ago) |
|  | spaceraccoon requested to disclose this report. | Jul 13th (11 months ago) |
|  | spaceraccoon posted a comment. | Aug 5th (10 months ago) |
|  | shadegrown agreed to disclose this report. | Aug 6th (10 months ago) |



This report has been disclosed.

Aug 6th (10 months ago)