The X-CSRF Signature header could not be validated from Query Editor

Last updated by | Subbu Kandhaswamy | Aug 29, 2022 at 3:02 PM PDT

Contents

The X-CSRF Signature header could not be validated from Query Editor

Issue

Error connecting from inside Azure, Resource URI of the Azure VM/Cloud Service: **Query Editor**.

Cause

This is created and validated to prevent a certain type of attack against your Azure SQL Servers. Specifically, some web browsers can save your passwords which might then allow an attacker who doesn't know the password to issue queries using the remembered password. In order to prevent this type of attack, known as Cross Site Request Forgery (CSRF), we attach this little bit of extra data, called the "CSRF Signature". This signature proves that the credentials were known at the time of the request, not just remembered by the browser.

Error

'The X-CSRF-Signature header could not be validated.'

Triage

Customers may be reporting connectivity issues to the WebQueryEndpoint. This can be caused by the following categories of issues:

- Client side issues, such as browser request filtering.
- Issues in transit, such as misconfigured HTTP proxies or firewall appliances.
- Server side issues, such as performance problems, downstream connectivity issues, etc.

It is necessary to determine if the customer's request has reached the application.
First, obtain the server fully qualified domain name and region and run the following query:

```
MonWebQueryEndpoint
| where TIMESTAMP >= ago(1d)
| where host_name == "<server_fqdn>"
| where event startswith "waf_"
```

This will show successful and failed HTTP connections. If the query does not contain the expected rows, it is most likely a client networking or proxy issue.

If the HTTP status code from the query indicates success, consider client-side timeouts or application issues.

If the HTTP status code from the query indicates failure, copy a request ID from a representative request and proceed to mitigation.

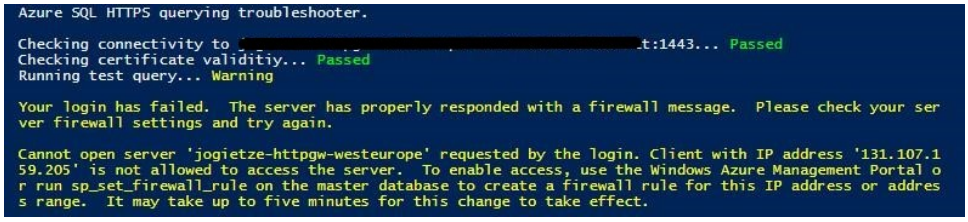
The following query shows number of customers impacted by connectivity issues in order to assess impact:

```
ExtEvents
| where area contains "QueryEditorBlade.signIn" and message contains "This may indicate an issue with your net
| summarize dcount(userId) by bin(TIMESTAMP, 1d)
| render timechart
```

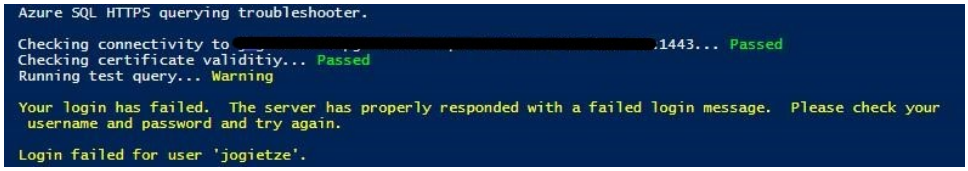
If the customer's request is not reaching the endpoint, the customer may use this script to troubleshoot client-side connectivity issues: <<TroubleshootHttpsQuerying.ps1>>

The tool will attempt to diagnose connectivity problems. Here are some examples of problems that will be diagnosed:

(No Problems, all tests succeeded.)

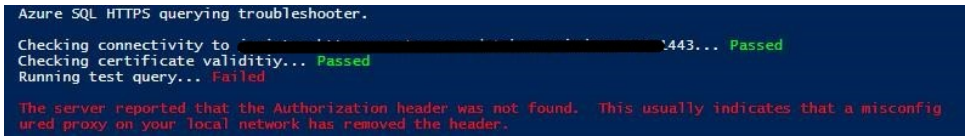


(Endpoint is available, customer needs to configure server-side firewall settings.)

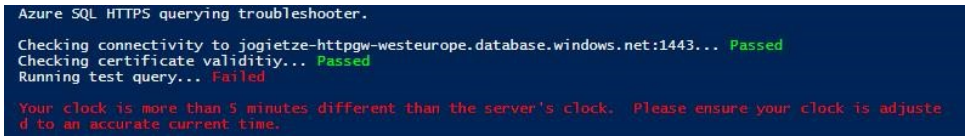


(Endpoint is available, customer needs to supply valid credentials.)

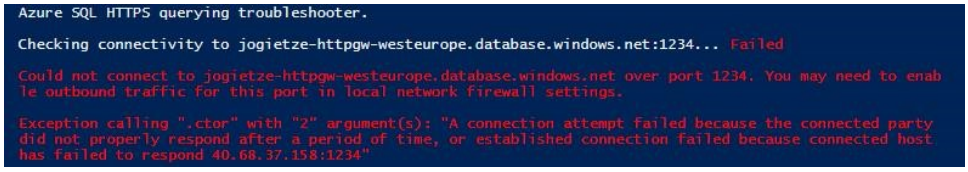
(Proxy has changed the cert, but the test passed. Traffic is being monitored by a trusted third party.)



(The Authorization header was removed.)



(Clock synchronization issue.)



(Port blocked or total service outage. If the service is available, the customer needs to enable outbound traffic on the port.)

Mitigation

To determine the failure cause, obtain a request_id and use the following query to get more information:

```
MonWebQueryEndpoint
| where TIMESTAMP >= ago(1d)
| where request_id == "<request_id>"
| where event == "web_query_endpoint_trace"
```

Requests will proceed through the following steps at the start of the request:

1. Validate the format of the authorization header.
2. **The Authorization header was not found**. This indicates that the request is missing the authorization header. To mitigate, the customer will need to specify the Authorization header on the request.
3. **The specified authorization scheme is not supported**. This indicates that the request attempted to use an authorization scheme other than Basic or Bearer. Only basic and bearer are supported. To mitigate, the customer should switch authorization schemes.
4. **The specified host name did not pass the validation procedure**. As a security measure against relay attacks, the host name must match the public SSL certificate. To mitigate, the customer will need to specify a canonicalized host header including the server's fully qualified domain name.
5. Validate the CSRF signature if present, and require its presence for Basic auth.
6. **The X-CSRF-Signature header could not be validated**. The CSRF signature value specified did not conform to the expected format or did not match the expected value. To mitigate, the customer will need to fix signature generation to match the expected signature validation procedure.

7. **Basic authentication must be accompanied by a X-CSRF-Signature header with a valid signature**. The request contained a Basic authorization header, but no CSRF protection header. CSRF protection is necessary because User Agents MAY store basic credentials. To mitigate this, the customer will need to include the CSRF header on all requests with Basic authentication.
8. Obtain the client IP address.
9. **The client IP address could not be determined**. The client's original IP address was not found or did not conform to the format of an IP address.
10. Establish a TDS connection with the instance.
11. **The connection timed out**. The application's configured connection timeout was reached attempting to connect to the target instance. If the root cause is not clear, copy the client connection ID and escalate to the performance team.
12. **The connection attempt was unsuccessful.{exception}** Consult the included exception details to determine the root cause. If the exception is not helpful or further investigation is warranted, copy the client connection ID and escalate to engineering.

Messaging to customers

If customers detected that port 1443 on their side cannot be open, we should provide them with the following message:

> Thank you for reporting this issue. As pointed in the documentation, in order to use the Query Editor functionality in the Azure portal, customers have to open ports 443 and 1443 on their local network for outbound HTTPS traffic. This is a new requirement due to recent changes of the underlying components powering this experience to a more secure and robust implementation. Modern web browsers are extremely capable application programming platforms, but they do not today support connecting directly to an Azure SQL database. That gap used to be bridged via an intermediate service which has been deprecated and replaced with a Web Query HTTPS API endpoint. This new API is more secure as it honors your SQL server's firewall configuration.

> A very common question is why does port 1443 need to be open on the customer's network when port 443 is commonly used for HTTP/HTTPS traffic. This design choice was made in order to avoid throttling of other HTTP traffic on this port, such as any Azure SQL manageability HTTP request. This is a temporary state, as we are planning more changes to this end point that will eventually securely eliminate this requirement and only use port 443. We apologize for any inconvenience this is causing in the meantime and we appreciate your patience as we work towards providing you with rich functionality in the most reliable and secure way. The best way to work around this temporary problem for any customer who cannot open port 1443 is to use one of our client tools for querying databases, such as SSMS.

Classification

Root Cause: Azure SQL DB v2\Connectivity\Login Errors\Firewall errors and misconfigurations

How good have you found this content?