Post Context deadline exceeded (Client.Timeout exceeded while awaiting headers)

Asked 2 years, 4 months ago Modified 10 months ago Viewed 54k times

10

I am running a simple query using `Clickhouse HTTP Interface`

```
package main

import (
"fmt"
"net/http"
"time"
)

func main() {

    url := "http://localhost:8123" + "?query=select%201"
    req, _ := http.NewRequest("Post", url, nil)

    // set headers
    req.Header.Set("X-ClickHouse-User", "user")      //user
    req.Header.Set("X-ClickHouse-Key", "password") //password

    client := &http.Client{
        Timeout: 5 * time.Second,
    }
    resp, err := client.Do(req)
    if err != nil {
        fmt.Println(err.Error())
    } else {
        fmt.Println("Success")
    }
}
```

It is giving me this error `Post "http://localhost:8123?query=select%201": context deadline exceeded (Client.Timeout exceeded while awaiting headers)`

Note: If I use insert query instead of this select query it will give me the same error but also will insert the data correctly.

Any help would be appreciated.

Thanks

http go clickhouse clickhouse-go

Share Improve this question Follow edited Nov 9, 2022 at 9:10 asked Nov 9, 2022 at 8:31

Zunnurain Badar
1,042 ● 2 ● 10 ● 34

Probably is because you are setting the timeout on your side and the server timeout is longer than yours. In consequence the server may send a response after the 5sec that you are waiting for. – Marcel Kohls
Nov 9, 2022 at 11:17

1 @MarcelKohls I sent the same request from the postman, Nodejs and it worked. it takes less that a sec to respond – Zunnurain Badar Nov 9, 2022 at 12:32

1 Answer

Sorted by: Highest score (default)

6

I did a quick sample here using your code, and as I suspected, locally this error only happens when the server times out. Maybe postman is doing something that hides this behavior.

Also, don't forget that the http.Client timeout is for the entire request-response cycle that is constituted up of Dialer, TLS Handshake, Request Header, Request Body, Response Header and Response Body timeouts.

Please take a look in the POC below and change the `timeout()` for different values to see the success (4- sec) or fail (5+ sec).

```
package main

import (
    "fmt"
    "log"
    "net/http"
    "time"
)


func timeout(w http.ResponseWriter, req *http.Request) {
    time.Sleep(5 * time.Second)
}

func server() {
    go func() {
        http.HandleFunc("/", timeout)
        log.Fatal(http.ListenAndServe(":8123", nil))
    }()
}

func main() {
    server()

        url := "http://localhost:8123" + "?query=select%201"
    req, _ := http.NewRequest("Post", url, nil)

    // set headers
    req.Header.Set("X-ClickHouse-User", "user")      //user
    req.Header.Set("X-ClickHouse-Key", "password") //password
    client := &http.Client{
        Timeout: 5 * time.Second,
    }
    resp, err := client.Do(req)
    if err != nil {
        fmt.Println(err.Error())
    } else {
        fmt.Println("Success", resp)
    }
}
```

you can also try to make use of specific http.Client timeouts as for example:

```
c := &http.Client{
    Transport: &http.Transport{
        Dial: (&net.Dialer{
                Timeout:   30 * time.Second,
                KeepAlive: 30 * time.Second,
        }).Dial,
        TLSHandshakeTimeout:   10 * time.Second,
        ResponseHeaderTimeout: 10 * time.Second,
        ExpectContinueTimeout: 1 * time.Second,
    }
}
```

Some references that may help you: https://blog.cloudflare.com/the-complete-guide-to-golang-net-http-timeouts/ https://itnext.io/http-request-timeouts-in-go-for-beginners-fe6445137c90 https://gosamples.dev/http-client-timeout/

Share Improve this answer Follow

answered Nov 9, 2022 at 16:04
Marcel Kohls
1,897 ● 20 ● 31

Very helpful !! I was earlier setting overall `Timeout` in the `Client` struct. This did not take care off the scenario when we are uploading large files on slow internet. Instead now shifted to using `ResponseHeaderTimeout` in `Transport` struct – Madhur Bhaiya Dec 21, 2024 at 9:29

Start asking to get answers

Find the answer to your question by asking.

Ask question

Explore related questions

http go clickhouse clickhouse-go

See similar questions with these tags.