**Отчет по лабораторной работе № 3.2**

Тарасов А. Т.

Группа ИУ9-31Б

**Задача 1:** Реализовать приложение посылающее ICMP пакеты к заданному хосту и выводящее результаты ответа.

**Задача 2:**  Реализовать ICMP-флуд используя горутины

http://golang-book.ru/chapter-10-concurrency.html

**Задача 3:** Реализовать трассировку до заданного хоста.

Исходный код программ

**ICMP Flood**

package main

import (

    "fmt"

    "github.com/sparrc/go-ping"

    "log"

    "runtime"

    "flag"

    "time"

)

type chs chan string

var (

    url     = flag.String("url", "ya.ru", "Url")

)

func main() {

    flag.Parse()

    runtime.GOMAXPROCS(2)

    result := make(chs)

    for i := 0; i < 5; i++ {

        go func(i int) {

            fmt.Println(i)

            result.StartPinging(i, \*url)

        }(i)

    }

    for i := 0; i < 5; i++ {

        select {

        case res := <-result:

            fmt.Println(res)

        case <-time.After(10 \* time.Second):

            fmt.Println("Timed out")

            return

        }

    }

}

func (result \*chs) StartPinging(i int, addr string) {

    fmt.Println("pinging...")

    pinger, err := ping.NewPinger(addr)

    if err != nil {

        log.Println(err)

    }

    pinger.OnRecv = func(pkt \*ping.Packet) {

        fmt.Printf("%d bytes from %s: icmp\_seq=%d time=%v\n",

            pkt.Nbytes, pkt.IPAddr, pkt.Seq, pkt.Rtt)

    }

    pinger.OnFinish = func(stats \*ping.Statistics) {

        fmt.Printf("\n--- %s ping statistics ---\n", stats.Addr)

        fmt.Printf("%d packets transmitted, %d packets received, %v%%               packet loss\n",

            stats.PacketsSent, stats.PacketsRecv, stats.PacketLoss)

        fmt.Printf("round-trip min/avg/max/stddev = %v/%v/%v/%v\n",

            stats.MinRtt, stats.AvgRtt, stats.MaxRtt, stats.StdDevRtt)

    }

    pinger.Count = 1

    pinger.SetPrivileged(true)

    pinger.Run()

    \*result <- fmt.Sprint(i, ": ", pinger.Statistics())

}

Трассировка до заданного хоста

package main

import (

    "fmt"

    "flag"

    "github.com/lixiangzhong/traceroute"

)

var (

    url     = flag.String("url", "ya.ru", "Url")

)

func main() {

    flag.Parse()

    t := traceroute.New(\*url)

    result, err := t.Do()

    if err != nil {

        fmt.Println(err)

        return

    }

    for \_, v := range result {

        fmt.Println(v)

    }

}