Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем и программной инженерии

Лабораторная работа №1

по дисциплине "Информационные сети"

ТЕМА РАБОТЫ:

БАЗОВАЯ НАСТРОЙКА СЕТЕВЫХ СРЕДСТВ ОС

Выполнил:

студент гр. ПРИ-120

Парахин К.В.

Принял:

Курочкин С.В.

Владимир 2022 г.

Цель работы:

Получить навыки базовой настройки сетевых средств ОС Windows

Вариант 8

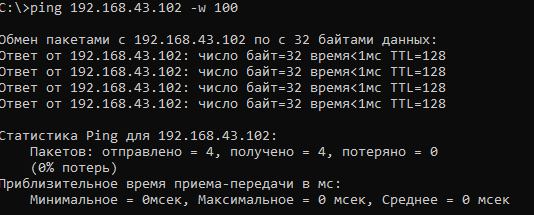
Выполнение заданий:

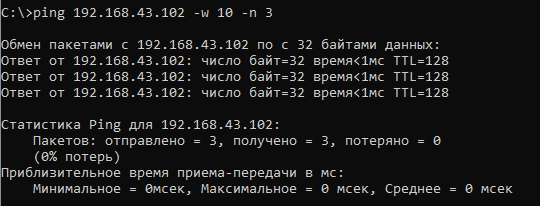
Тестирование и вызов основных утилит с использованием разных параметров:

1. Hostname

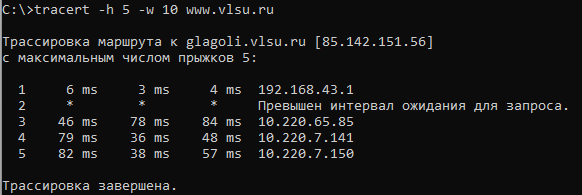


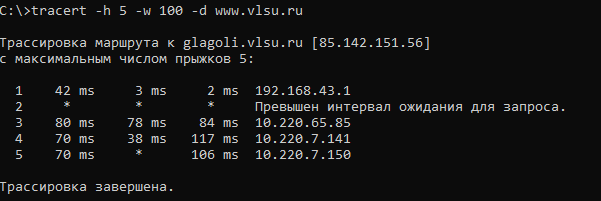
1. Ping



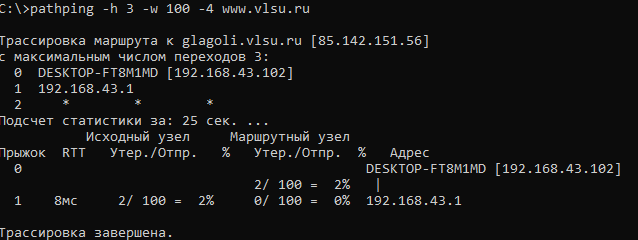


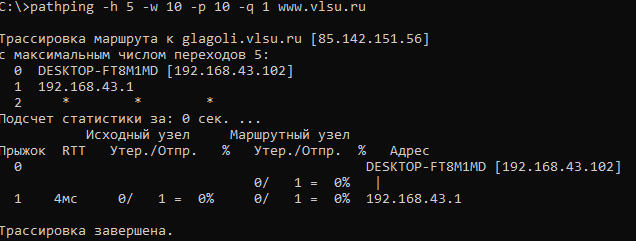
1. Tracert



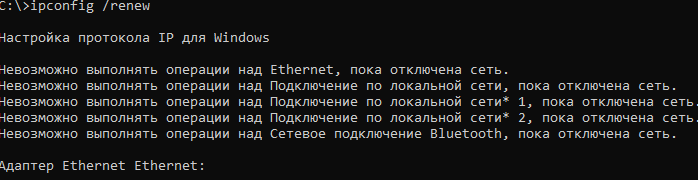


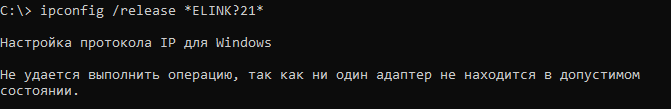
1. Pathping



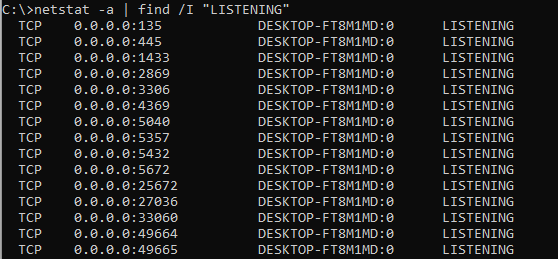


1. Ipconfig

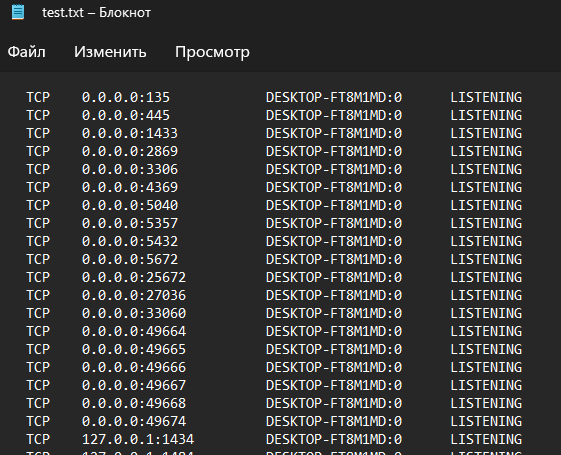




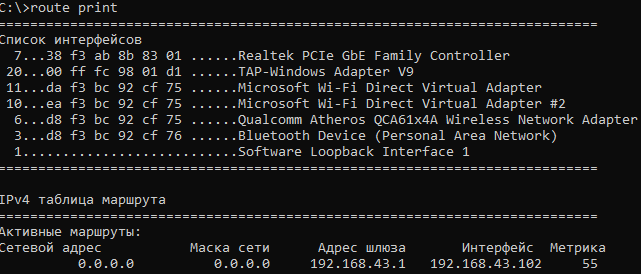
1. Netstat







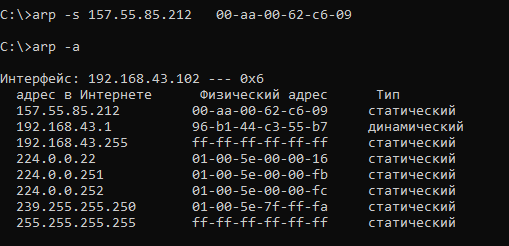
1. Route

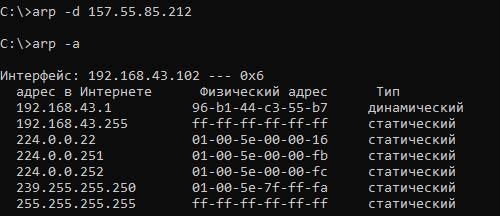




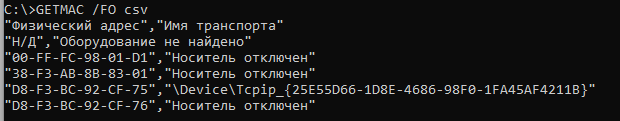


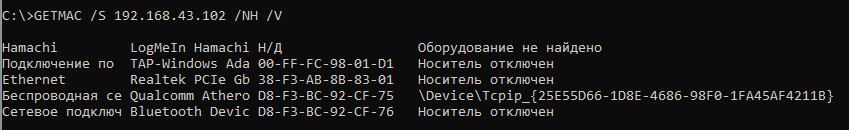
1. Arp



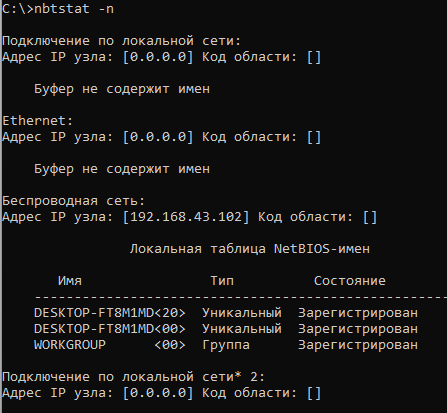


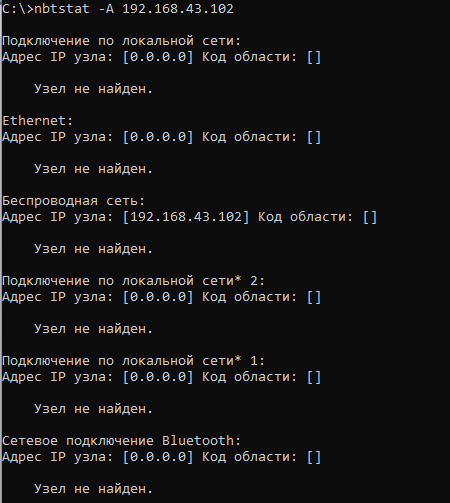
1. GETMAC



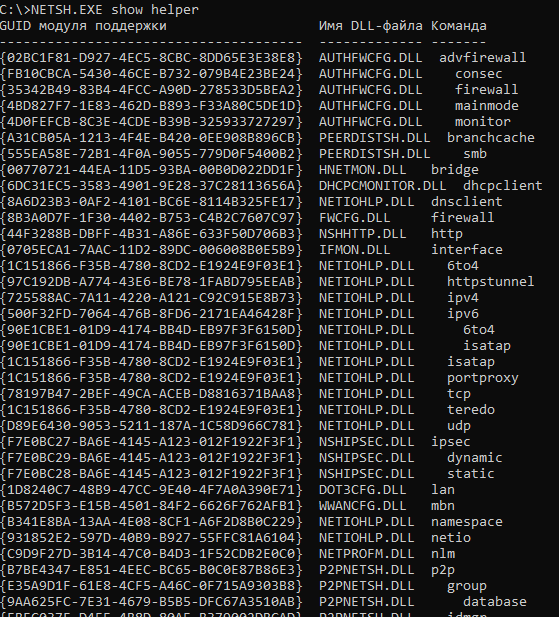


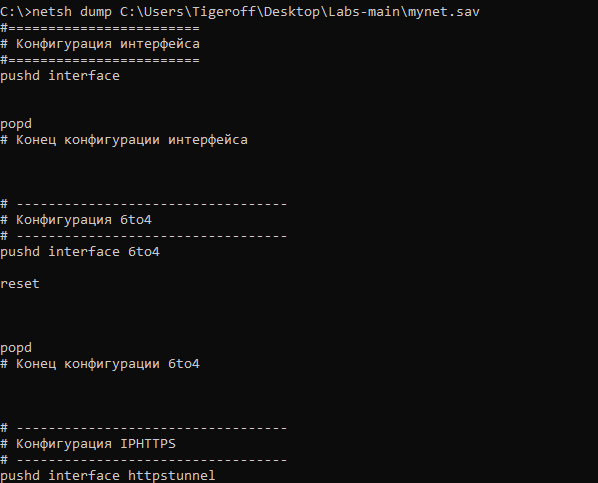
1. Nbtstat





1. Netsh.exe

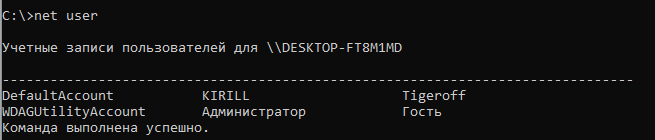






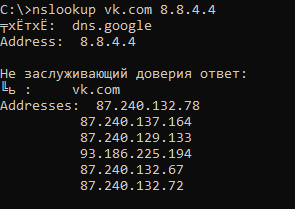
1. Net.exe

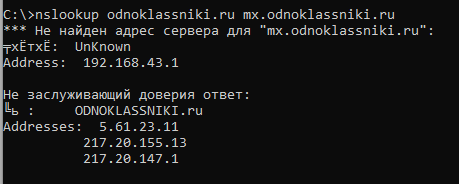






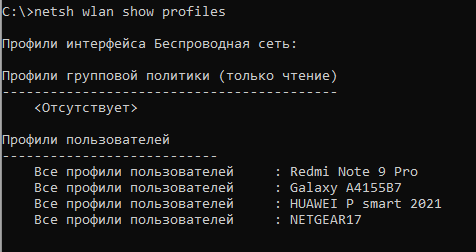
1. Nslookup

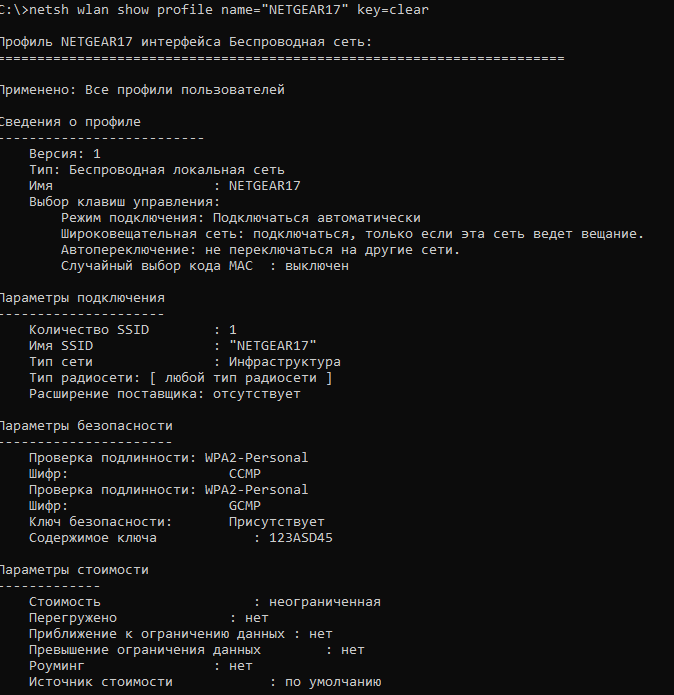




Выполнение некоторых базовых действий (опциональное задание\*)

Управление профилями беспроводных сетей





Выполнение индивидуального задания

Написать программу на любом языке программирования для взаимодействия пользователя со следующими утилитами: утилита IPConfig, Ping, Tracert.

К написанной программе добавить следующие утилиты: ARP, NSLOOKUP (Вариант 8)

Листинг программы на языке C#:

using DnsClient;

using System.Diagnostics;

using System.Net;

using System.Net.NetworkInformation;

using System.Text;

namespace ProtocolUtilites

{

public class IPInfo

{

public static void PingRoute(string ipAddressOrHostName, int timeout)

{

using var ping = new Ping();

PingReply reply = ping.Send(ipAddressOrHostName, timeout);

Console.WriteLine("Ping:");

if (reply.Status == IPStatus.Success)

{

var msg = @$"Status: {reply.Status} IP Address: {reply.Address} Time:{reply.RoundtripTime} ms";

Console.WriteLine(msg);

}

else

Console.WriteLine(reply.Status);

}

public static void TraceRoute(string ipAddressOrHostName)

{

PingReply? pingReply = null;

StringBuilder traceResults = new StringBuilder();

using (Ping pingSender = new Ping())

{

PingOptions pingOptions = new PingOptions();

Stopwatch stopWatch = new Stopwatch();

byte[] bytes = new byte[32];

pingOptions.DontFragment = true;

pingOptions.Ttl = 1;

int maxHops = 30;

traceResults.AppendLine(string.Format("Tracing route to {0} over a maximum of {1} hops:", ipAddressOrHostName, maxHops));

traceResults.AppendLine("Trace may take some time...");

traceResults.AppendLine();

Console.WriteLine("TraceResults:");

for (int i = 1; i < maxHops + 1; i++)

{

stopWatch.Reset();

stopWatch.Start();

pingReply = pingSender.Send(ipAddressOrHostName, 1000, new byte[32], pingOptions);

stopWatch.Stop();

if (pingReply.Status != IPStatus.TtlExpired && pingReply.Status != IPStatus.Success)

traceResults.AppendLine(string.Format("{0} \t{1}", i, pingReply.Status.ToString()));

else

{

IPHostEntry @ipHost = Dns.Resolve(pingReply.Address.ToString());

if (ipHost.HostName != pingReply.Address.ToString())

traceResults.AppendLine(string.Format("{0}\t{1} ms\t{2} \t[{3}]", i, stopWatch.ElapsedMilliseconds, pingReply.Address, ipHost.HostName));

else

traceResults.AppendLine(string.Format("{0}\t{1} ms\t{2}", i, stopWatch.ElapsedMilliseconds, pingReply.Address));

}

if (pingReply.Status == IPStatus.Success)

{

traceResults.AppendLine();

traceResults.AppendLine("Trace complete.");

Console.WriteLine("{0}", traceResults.ToString());

break;

}

pingOptions.Ttl++;

}

}}

public static void GetARPResult()

{

Process? p = null;

string output = string.Empty;

try

{

p = Process.Start(new ProcessStartInfo("arp", "-a")

{

CreateNoWindow = true,

UseShellExecute = false,

RedirectStandardOutput = true

});

output = p!.StandardOutput.ReadToEnd();

p.Close();

}

catch (Exception ex)

{

throw new Exception("IPInfo: Error Retrieving 'arp -a' Results", ex);

}

finally

{

if (p is not null)

p.Close();

}

Console.WriteLine("ARP Result:");

Console.WriteLine(output);

}

public static void NsLookup(string webSite)

{

var lookup = new LookupClient();

var result = lookup.Query(webSite, QueryType.NS);

foreach (var nsRecord in result.Answers.NsRecords())

Console.WriteLine(nsRecord.NSDName);

Console.WriteLine("NS Lookup Result:");

Console.WriteLine("Adresses for " + webSite + " :");

}

private static string IPConfig()

{

Console.WriteLine("IPConfig:");

string sHostName = Dns.GetHostName();

IPHostEntry ipE = Dns.GetHostByName(sHostName);

IPAddress[] IpA = ipE.AddressList;

for (int i = 0; i < IpA.Length; i++)

{

Console.WriteLine("IP Address {0}: {1} ", i, IpA[i].ToString());

}

return IpA[2].ToString();

}

protected static string? ipAddressOrHostName;

public static void Main()

{

ipAddressOrHostName = IPConfig();

PingRoute(ipAddressOrHostName, timeout: 100);

Console.WriteLine("------------------");

TraceRoute(ipAddressOrHostName);

Console.WriteLine("------------------");

GetARPResult();

Console.WriteLine("------------------");

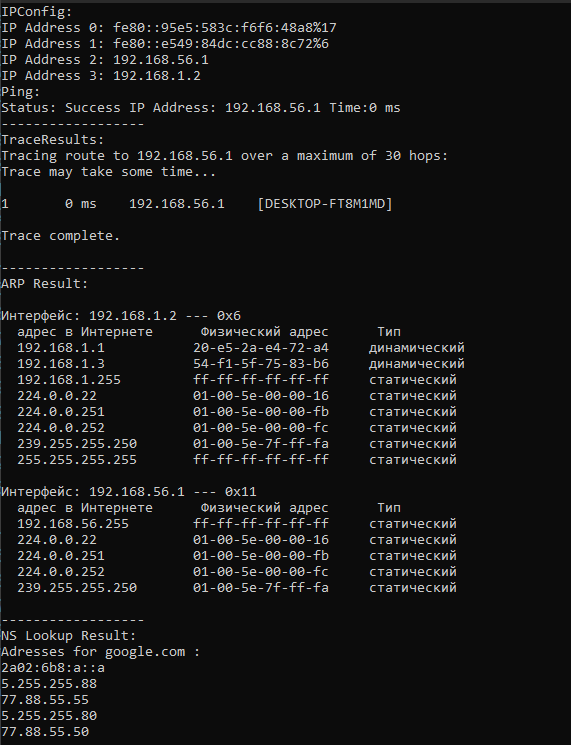
NsLookup("google.com");

}

}

}

Тестирование программы:



Вывод

В результате выполнения работы я получил навыки базовой настройки сетевых средств ОС Windows, на практике ознакомился с вызовом и использованием основных сетевых утилит.

Кроме этого, благодаря полученным знаниям написал программу на алгоритмическом языке с использованием сетевых библиотек, благодаря которой осуществил выполнение некоторых их этих утилит вручную с помощью сетевых библиотек.