ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ   
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»

Кафедра вычислительных систем

**КУРСОВОЙ ПРОЕКТ**

по дисциплине

“ Сетевое программирование”

Реферат по теме «IPv6»

|  |  |
| --- | --- |
| Выполнил студент | Костыль Валерия Юрьевна |
|  | Ф.И.О. |

|  |  |
| --- | --- |
| Группы | ИП-015 |
|  |  |

|  |  |  |
| --- | --- | --- |
| Работу принял |  | ассистент кафедры ВС Третьяков Глеб Николаевич |
|  | подпись |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Защищена |  | Оценка |  |
|  |  |  |  |

Новосибирск – 2023

Оглавление

[Введение 3](#_Toc135809142)

[Адресация протокола 3](#_Toc135809143)

[Формат заголовка 6](#_Toc135809144)

[Проблемы IPv6 и пути их решения 7](#_Toc135809145)

[Взаимодействие IPv6 и IPv4 8](#_Toc135809146)

[IPv6 в России. 10](#_Toc135809147)

[Заключение 12](#_Toc135809148)

[Лабораторная работа №2: Параллельный (мультипроцессный) сервер 12](#_Toc135809149)

[Server.cpp 12](#_Toc135809150)

[Client.cpp 14](#_Toc135809151)

# 

# Введение

IPv6 (англ. Internet Protocol version 6) - это новая версия протокола IP, призванная решить проблемы, с которыми столкнулась предыдущая версия (IPv4) при её использовании в Интернете, за счет использования адреса длиной 128 бит вместо 32. В настоящее время протокол IPv6 уже используется во многих сетях по всему миру и поддерживается практически всем современным программным обеспечением (таким, как Windows XP и старше, Linux, FreeBSD или MacOS), но пока ещё не получил широкого распространения в Интернете, где преимущественно используется IPv4. Протокол был разработан IETF еще в 1996 году.

# Адресация протокола

Система адресации IPv6 существенно отличается от системы адресации IPv4.

Адреса назначения и источника в IPv6 имеют большую длину: 128 бит или 16 байт. Это дает возможность пронумеровать огромное количество узлов: 340 282 366 920 938 463 463 374 607 431 762 211 456 или более 300 млн адресов на каждого жителя Земли. Выбранная длина IP-адреса должна надолго снять проблему дефицита IP-адресов. Кроме того, в версии IPv6 предполагается использование протокола динамической выдачи адресов по протоколу DHCP, позволяющее разделять одни и те же адреса между большим количеством узлов сети.

Сравните с 4 294 967 296 возможных IPv4 адресов, включая технические - мультикаст, зарезервированные под локальные сети группы адресов etc. Использование NAT-серверов, подменяющих внутренние адреса узлов сети одним собственным IP-адресом, также стало результатом необходимости снижения потребности в IP-адресах.

Однако, главной целью изменения системы адресации было не механическое увеличение разрядности адреса, а обеспечение возможности увеличения числа уровней иерархии в адресе. Вместо прежних двух уровней (номер сети и номер узла) в IPv6 предлагается использовать 5 уровней, включая двухуровневую идентификацию провайдеров, и трехуровневую - абонентов сети.

Предполагается также, что младшие 6 байт, которые содержат идентификатор узла, представляют собой МАС-адрес сетевого адаптера (как это уже давно делается в протоколе IPX), что обеспечит возможность автоконфигурации стека.

В версии IPv6 не вводятся классы адресов сетей, вместо этого предполагается использовать бесклассовую технологию CIDR (Classless Inter-Domain Routing). Эта технология заключается в назначении каждому провайдеру непрерывного диапазона в пространстве IP-адресов. При таком подходе все адреса сетей каждого провайдера имеют общий префикс, так что маршрутизация на магистралях Internet может осуществляться на основе префиксов, а не полных адресов всех сетей конечных абонентов. Локализация адресов позволяет уменьшить объем таблиц в маршрутизаторах всех уровней, а, следовательно, ускорить работу маршрутизаторов и повысить пропускную способность Internet. Деление IP-адреса на номер сети и номер узла в технологии CIDR происходит не на основе нескольких старших бит (класса сети А, В или С), а на основе маски переменной длины, назначаемой провайдером.

Технология CIDR уже успешно используется в текущей версии IPv4 и поддерживается такими протоколами маршрутизации как OSPF, RIP-2, BGP4. Предполагается , что эти же протоколы будут работать и с IPv6.

Техника CIDR помогает также решить известную проблему фрагментации адресного пространства IPv4. Например, очень редко абонент использует все 254 адреса сети класса С или 65 534 адреса сети класса В. Часть адресов узлов пропадает. Требование оплаты каждого адреса узла поможет пользователю решиться на перенумерацию, с тем, чтобы получить ровно столько адресов, сколько ему нужно.

Как и в версии IPv4, в IPv6 вводится несколько типов адресов.

Unicast - индивидуальный адрес. Такой адрес определяет отдельный узел - компьютер или порт маршрутизатора. Индивидуальные адреса подразделяются на несколько подтипов:

Global - основной тип адресов в Internet;

Link-local и Site-local - используются в сетях, не подключенных к Internet. Поле идентификатора провайдера заполняется нулями, что дает возможность при подключении к Internet сохранять эти адреса;

Compatible - обеспечивают совместимость с адресами IPv4, IPX, NSAP.

Multicast (one-to-many) - адрес набора узлов, возможно в различных физических сетях. Копии пакета должны быть доставлены каждому узлу набора, используя аппаратные возможности групповой или широковещательной доставки, если это возможно. В протоколе IPv6 отсутствует понятие широковещательного адреса (broadcast), эти адреса реализуются с помощью адресов multicast. Поддержка мультивещания - важная часть стратегии модернизации протокола IP, так как это свойство необходимо для поддержки экономного в отношении пропускной способности сети распространения аудио- и видео- информации многочисленным абонентам.

Anycast (one-to-nearest) - адрес набора узлов. Обозначает группу узлов, которые имеют общий адрес, но в отличие от мультивещания пакет должен быть доставлен любому, лучше ближайшему, члену группы. Этот тип адресов используется для того, чтобы абонент мог без особых сложностей обеспечить прохождение своего трафика через маршрутизаторы определенных провайдеров. В IPv6 предполагается широкое использование маршрутизации от источника (Source Routing), при которой узел-источник задает полный маршрут прохождения пакета через сети. Такая техника освобождает маршрутизаторы от работы по просмотру адресных таблиц при выборе следующего маршрутизатора и тем самым повышает пропускную способность Internet. В последовательности адресов, задаваемых узлом-отправителем по алгоритму Source Routing, наряду с адресами маршрутизаторов типа unicast, можно использовать адреса anycast, которые определяют все маршрутизаторы одного провайдера.

Синтаксически адрес anycast не отличим от адреса unicast. Схема назначения адресов состоит в следующем. Каждому порту маршрутизатора наряду с уникальным адресом присваивается еще один, общий для всех портов и маршрутизаторов данного провайдера адрес, который и является anycast-адресом.

Для обеспечения плавного перехода от версии IPv4 к версии IPv6 введен специальный тип адресов - IPv4-compatible. Такие адреса содержат нули в старших 96 разрядах, а в младшие 32 разряда помещается 4-х байтовый адрес версии IPv4. Такие адреса легко могут транслироваться в обе стороны. Это позволит на начальном этапе внедрения IPv6 решить проблему совместимости частей Internet, работающих по IPv6, с частями Internet, пока поддерживающими только версию IPv4. Для этого узлам в "островках" IPv6 будут присваиваться адреса типа IPv4-compatible. Для передачи трафика IPv6 через те части Internet, маршрутизаторы которых пока не поддерживают версию IPv6, будет использоваться техника туннелирования - пришедший пакет IPv6 будет упаковываться пограничным маршрутизатором в пакет формата IPv4, при этом в качестве адреса будет использоваться младшая часть адреса из пакета IPv6.

# Формат заголовка

Главной целью изменения формата заголовка в IPv6 было снижение накладных расходов, то есть уменьшение объема служебной информации, передаваемой с каждым пакетом. Для этого в новом IP было введены понятия основного и дополнительного заголовков. Основной заголовок присутствует всегда, а дополнительные являются опциональными.

В предложениях по IPv6 фигурируют пока следующие типы дополнительных заголовков:

Routing - заголовок для указания полного маршрута при маршрутизации от источника (Source Routing)

Fragmentation - содержит информацию о фрагментации IP-пакета. Поле обрабатывается только в конечных узлах.

Authentication - заголовок содержит информацию, необходимую для аутентификации конечных узлов и обеспечения целостности содержимого IP-пакетов.

Encapsulation - заголовок содержит информацию, необходимую для обеспечения конфиденциальности передаваемых данных путем выполнения шифрации и дешифрации.

Hop-by-Hop Option - специальные параметры, используемые при выполнении обработки пакетов по алгоритму Hop-by-Hop

Destination Options - заголовок содержит дополнительную информацию для узла назначения.

Поскольку маршрутизаторы обрабатывают только основные заголовки (почти все дополнительные заголовки обрабатываются только в конечных узлах.), то это увеличивает их производительность и тем самым пропускную способность сети. Напомним, что в IPv4 все опции обрабатываются маршрутизаторами.

Наличие большого количества дополнительных необязательных параметров позволяет расширить функциональность протокола IP.

# Проблемы IPv6 и пути их решения

Ученые осознали ограниченность адресного пространства в IPv4 еще в начале 90х годов. Тогда и был разработан протокол IPv6, призванный прийти на смену старой технологии. Кроме существенного увеличения количества уникальных IP-адресов (в IPv6 оно доведено практически до бесконечности), протокол обладает и рядом других преимуществ перед IPv4. Например, в нем предусмотрена возможность шифрования данных и поддержка ряда параметров, которые повышают качество мультимедийных трансляций. Кроме того, IPv6 упрощает маршрутизацию в Интернете и позволяет ускорить обмен данными между устройствами.

На первый взгляд, новый протокол обладает рядом существенных преимуществ перед IPv4. Однако до сих пор скорость его внедрения продолжает оставаться низкой. По данным Форума IPv6, только семь из 21 крупнейших интернет-провайдеров предпринимают шаги, необходимые для полноценного перехода к использованию новой технологии.

Во многом это связано с тем, что до сих пор не разработан оптимальный способ внедрения IPv6. Большинство существующих сегодня сетей, да и весь современный Интернет в целом основаны на протоколе IPv4, а "взаимопонимание" старой и новой технологий невозможно без использования специальных технологий. Переход к IPv6, по сути, означает построение новой инфраструктуры Всемирной Паутины. Это - весьма длительный процесс, требующий значительных временных затрат и масштабных финансовых вложений. А поскольку полноценное внедрение IPv6 не может произойти моментально, необходимо обеспечить надежное взаимодействие новой технологии с IPv4 и DNS, которая также лежит в основе Интернета.

# Взаимодействие IPv6 и IPv4

На сегодняшний день известно три основных способа обеспечения взаимодействия IPv6 и IPv4: туннелирование, двойной стек и трансляция протоколов.

Туннелирование

Суть туннелирования состоит в том, что пакет данных IPv6 внедряется (инкапсулируется) в поле данных пакета IPv4. Получившийся в результате этой операции пакет IPv4 содержит в себе два заголовка, IPv6 и IPv4, и может передаваться через обычные IPv4-сети. Он доставляется к узлу (хосту) декапсуляции, где заголовок IPv4 отбрасывается, а данные передаются к устройству, использующему IPv6. В зависимости от того, где происходит инкапсуляция и декапсуляция, выделяют следующие виды туннелирования:

"Маршрутизатор - Маршрутизатор";

"Хост - Маршрутизатор";

"Маршрутизатор - Хост".

Туннель, используемый в процессе передачи данных, должен иметь точку входа и точку выхода. Точка входа находится на границе инфраструктуры IPv4, а потому определить ее достаточно легко.

Двойной стек

Реализация технологии двойного стека предполагает поддержку устройством (узлом), имеющим доступ в Интернет, и протокола IPv6, и протокола IPv4.

При этом адрес IPv4, приписанный узлу, должен быть уникальным. Поэтому к моменту исчерпания адресного пространства в IPv4 важно, чтобы внедрение новой технологии находилось на таком уровне, который позволял бы большинству устройств взаимодействовать друг с другом, используя только IPv6-адреса.

Трансляция протоколов

Трансляция протоколов заключается в преобразовании пакетов одной версии протокола в пакеты другой по определенным правилам. Она может осуществляться несколькими способами.

Первый из них состоит в использовании протокол-шлюзов, которые размещаются на границах между IPv6-сетями и IPv4-сетями. Кроме того, трансляция может осуществляться с помощью транспортного ретранслятора, который обрабатывает в передаваемом пакете данных IP-заголовки и заголовки транспортного уровня.

Наконец, трансляция протоколов возможна путем их преобразования на прикладном уровне через прокси-сервер.

Таким образом, существует несколько приемлемых способов совмещения протоколов IPv4 и IPv6. Однако интернет-сообщество пока еще не решило, какой из них наиболее удобен и позволит обеспечить безболезненный переход к использованию новой технологии.

# IPv6 в России.

В мире пока немного публичных IPv6-сетей и все они находятся в Европе и в США, непосредственно получить такой адрес у провайдера в России практически невозможно, но существуют специальные службы, предоставляющие всем желающим IPv6-подсети для свободного использования через специальные механизмы тунеллирования.

Например, компания www.hexago.com (Канада) предоставляет доступ в свою сеть freenet6.net, позволяя уже сейчас пользоваться всеми прелестями IPv6 протокола.

Еще в ноябре 1999 года RIPE NCC (Регистратор IP адресов в Европе) выдал блок адресов SubTLA (Sub Top Label Aggregation Block) одному из крупнейших российских Интернет сервис-провайдеров - академической исследовательской сети FREEnet. Это позволило ему уже в ближайшее время начать предоставление клиентам сети услуги по IPv6.

А в 2003 году Российский НИИ Развития Общественных Сетей (РосНИИРОС), являющийся оператором опорной научно-образовательной сети RBNet, предоставил возможность абонентам сети воспользоваться всеми преимуществами расширенной функциональности новой, шестой версии протокола IP (IPv6). Это стало возможным благодаря заключению пиринговых соглашений с сетью GEANT (Стокгольм), сетями Abilene (Internet2) и ASNET.

Ввели также поддержку IPv6 и RBNet (2001 год), RUNnet (2003 год), 2004 г. - ELTEL и NLine.

Для интеграции российских IPv6-сетей в глобальную IPv6-инфраструктуру в 2000 году в Москве была создана система IPv6 eXchange, позволяющая обмениваться IPv6-трафиком любым телекоммуникационным операторам. Впервые связь с зарубежными сетями была реализована в апреле 2001 года. В настоящее время пиринговые соглашения заключены с сетями GEANT, Abilene и ASNET (через Starlight - Чикаго), а также соглашение о сотрудничестве с NORDUNet, что обеспечивает взаимодействие российских и международных научно-образовательных сетей с использованием протокола IPv6. При переходе к новой версии протокола IP на магистральных участках научно-образовательных сетей реализована архитектура с двойным стеком IPv4/IPv6 для обеспечения обратной совместимости с доминирующим сейчас в Интернете протоколом IPv4. Это позволяет расширять абонентскую базу.

В декабре 2004 года "Корбина Телеком", российский универсальный оператор связи, приступила к реализации в Москве проекта "Интернет2", одной из отличительных особенностей которого является использование протокола передачи данных IPv6.

Все больше разработчиков телекоммуникационных услуг и оборудования включают поддержку IPv6 в свою продукцию. Так, например операторы мобильных сетей готовятся к внедрению IPv6 в свою структуру. Это связано с планируемым в ближайшем будущем ростом числа мобильных приборов с выходом в Интернет, дальнейшим ростом числа коммуникаторов и телефонов с возможностями игровой станции, интернет-радио, навигаторов итп.

Особенность современного этапа внедрения новой версии Интернет-протокола в России - это завершение периода ее апробации в научно-образовательных сетях и переход к промышленному использованию в сетях общего пользования. Поэтому в России наиболее подготовленным к переходу на IPv6 сегментом является образование. Созданы научно-образовательные сети (FREENet, RBNet, RUNNet). Среди таких некоммерческих сетей можно выделить проект MIRNet при сети МГУ. Это проект сотрудничества России и США в сфере разработки новых сетевых решений.

На сегодняшний день по количеству выделенных префиксов (блоков IPv6-адресов) Россия занимает 19-е место в Европе и 29-е место в мире.

# Заключение

Делая вывод из всего, можно сказать, что после перехода на IPv6 останутся, конечно, сторонники и у IPv4 - от этого не уйти, но со временем IPv6 станет основным протоколом и преобразит весь Интернет. Так же, как и для IPv4 сейчас, будут созданы программные и аппаратные средства для его поддержки и усовершенствования.

# Лабораторная работа №2: Параллельный (мультипроцессный) сервер

## Server.cpp

#include<stdio.h>

#include<iostream>

#include<stdlib.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<arpa/inet.h>

#include<netdb.h>

#include<unistd.h>

#include<string.h>

#include<wait.h>

#include<signal.h>

#define BUFFER\_SIZE 1024

using namespace std;

void ZombieHandler(int s)

{

    int sig;

    while(wait3(&sig,WNOHANG,(struct *rusage\**)0)>=0);

}

int main()

{

*//При прекращении дочернего процесса вызваем функцию для остановки процесса*

    signal(SIGCHLD,ZombieHandler);

    pid\_t child\_pid=0;

    int sockTCP,sockClient;

    unsigned int Length;

    struct *sockaddr\_in* serverAddr, clientAddr;

    int msgLength;

    char message[BUFFER\_SIZE];

    sockTCP=socket(AF\_INET, SOCK\_STREAM, 0);

    if(sockTCP<0)

    {

        printf("Open UDPsocket error");

        exit(1);

    }

    bzero((char\*) &serverAddr, sizeof(serverAddr));

    serverAddr.sin\_family=AF\_INET;

    serverAddr.sin\_addr.s\_addr=htonl(INADDR\_ANY);

    serverAddr.sin\_port=0;

    if(bind(sockTCP,(sockaddr\*) &serverAddr,sizeof(serverAddr)))

    {

        printf("Bind error");

        exit(1);

    }

    Length=sizeof(serverAddr);

    if(getsockname(sockTCP,(sockaddr\*) &serverAddr,&Length))

    {

        printf("getsockname() error");

        exit(1);

    }

*//Ожидаем запрос связи с передаваемыми аргументами протокола и количества одновременных запросов*

    listen(sockTCP,5);

    cout<<"PORT server:"<<ntohs(serverAddr.sin\_port)<<endl;

    cout<<"IP server:"<<inet\_ntoa(serverAddr.sin\_addr)<<endl;

    while(1)

    {

        Length=sizeof(clientAddr);

*//Ожидаем запрос на сервер и передаем его в сокет клиента*

        if((sockClient=accept(sockTCP,(sockaddr\*) &clientAddr,&Length))<0)

        {

            printf("accept() error");

            exit(1);

        }

*//Создаем новый дочерний процесс*

        child\_pid=fork();

        if(child\_pid!=0)

        {

*//Закрываем сокет для прослушивания*

            close(sockTCP);

            while(1)

            {

                bzero(message,BUFFER\_SIZE);

                msgLength=recv(sockClient,message,BUFFER\_SIZE,0);

                if(msgLength<0)

                {

                    printf("recvfrom() error");

                    exit(1);

                }

*//Закрытие сокета при окончании отправки сообщений от клиента*

                else if(msgLength==0)

                {

                    cout<<endl<<"PORT "<<ntohs(clientAddr.sin\_port)<<" close!"<<endl<<endl;

                    close(sockClient);

                    break;

                }

                cout<<"(Process: "<<getpid()<<" IP: "<<inet\_ntoa(clientAddr.sin\_addr)<<" PORT: "<<ntohs(clientAddr.sin\_port)<<")"<<endl;

                cout<<message<<endl<<endl;

                message[1]='+';

                if(send(sockClient,message,BUFFER\_SIZE,0)<0)

                {

                    printf("sendto() error");

                    exit(1);

                }

            }

            return 0;

        }

    }

    close(sockTCP);

    return 0;

}

## Client.cpp

#include<stdio.h>

#include<iostream>

#include<stdlib.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<arpa/inet.h>

#include<netdb.h>

#include<unistd.h>

#include<string.h>

using namespace std;

#define BUFFER\_SIZE 1024

#define COUNT 24

int main(int argc, char*\** argv[])

{

    int sockTCP;

    int Number;

    unsigned int Length;

    struct *sockaddr\_in* serverAddr;

    struct *hostent* *\**hp;

    char message[BUFFER\_SIZE];

    if(argc<4)

    {

        printf("Too few arguments");

        exit(1);

    }

    sockTCP=socket(AF\_INET, SOCK\_STREAM, 0);

    if(sockTCP<0)

    {

        printf("Open UDPsocket error");

        exit(1);

    }

    bzero((char\*) &serverAddr, sizeof(serverAddr));

    serverAddr.sin\_family=AF\_INET;

    hp=gethostbyname(argv[1]);

    bcopy(hp->h\_addr, &serverAddr.sin\_addr,hp->h\_length);

    serverAddr.sin\_port=htons(atoi(argv[2]));

    if(connect(sockTCP,(sockaddr\*)&serverAddr,sizeof(serverAddr))<0)

    {

        printf("Connect Error");

        exit(1);

    }

    Number=atoi(argv[3]);

    for(int i=0;i<COUNT;i++)

    {

        sleep(Number);

        if(send(sockTCP,argv[3],strlen(argv[3]),0)<0)

        {

            printf("sendto() error");

            exit(1);

        }

        bzero(message,BUFFER\_SIZE);

        Length=sizeof(serverAddr);

        if(recv(sockTCP,message,BUFFER\_SIZE,0)<0)

        {

            printf("recvfrom() error");

            exit(1);

        }

        cout<<message<<endl;

    }

    printf("Ok end.\n");

    close(sockTCP);

    return 0;

}