YILDIZ TEKNIK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

TCP/IP ORTAMINDA ÖRNEK İKİ SERVER UYGULAMASI

FTP SERVER VE BOOTP SERVER

Bilgisayar Mühendisi Esin Köran

F.B.E Bilgisayar Bilimleri Mühendisliği Anabilim Dalında hazırlanan

YÜKSEK LİSANS TEZİ

Tez Danışmanı : Y. Doç. Dr. Tülay Tinli

İSTANBUL, 1996

İÇİNDEKİLER

BÖLÜM 3

BÖLÜM 4

BÖLÜM 5

ŞEKİL LİSTESİ

KISALTMALAR LİSTESİ

NLM : NetWare Loadable Modules

TCP : Transmission Control Protocol

UDP : User Datagram Protocol

IP : Internet Protocol

FTP : File Transfer Protocol

BOOTP : Bootstrap Protocol

TEŞEKKÜR

Tez çalışmamın başlangıcından tamamlanmasına kadar geçen süre boyunca gerekli araştırma ve huzurlu çalışma ortamını sağlayan hocam Prof. M. Yahya Karslıgil'e ve hocam Y.Doç Tülay Tinli'ye, destekleyici ve yönlendirici fikirleriyle yol gösteren hocam Arş. Grv. A. Gökhan Yavuz'a ve çalışmamın her aşamasında desteklerini esirgemeyen araştırma görevlisi arkadaşlarıma tüm kalbimle teşekkür ederim.

ÖZET

Bu tez çalışmasında, globalleşme sürecinde önem kazanan internet ortamında hizmet verebilecek, TCP/IP protokol ailesinin üyelerinden File Transfer Protocol ve UDP/IP protokol ailesinin üyelerinden Bootstrap Protocol uygulama programları, client - server modeli kullanılarak gerçekleştirilmiştir.FTP Server ve BOOTP Server olarak adlandırılan bu programlar, NetWare 3.12 işletim sisteminde, Novell Software Development Kits ve Watcom C Compiler kullanılarak geliştirilmiştir. Her iki program NetWare Loadable Modules yapısında olduğundan, NetWare işletim sistemi tarafından sistemin bir modülü olarak görülürler. NetWare işletim sisteminin multithread program yapısını desteklemesi, dagıtık uygulamaların geliştirilmesi için elverişli bir ortam sunmaktadır. NLM C arayüzünde tanımlanmış pek çok fonksiyon tasarımcıya yardımcı olmaktadır.

Client - server haberleşme yöntemi olarak, Berkeley Unix 4.2BSD soket yapısı kullanılmıştır. Soket yapısı Internet Domain'inde, farklı sistemler arası haberleşmeye kolaylıklar getirmektedir. FTP Server programı stream türü soketleri, BOOTP server programı datagram türü soketleri kullanırlar. File Transfer Protocol, farklı makinalar arası dosya transferini sağlamak üzere geliştirilmiştir. Bir sistemde çalışan kullanıcılar bir diğer sisteme bu protokol yardımıyla erişerek, iki yönlü dosya transferi yapabilirler. BOOTP protokolünde, disksiz bir client makina hardware adresini göndererek server'dan IP adresini iste. Server kendi veritabanından hardware adresine karşı düşen IP adresi ile birlikte yerel network kaynakları hakkında client'abilgi gönderir. Bu kaynaklar, domain name server, gateway, domain name, hardware type, template host gibi bilgileri içermektedir.

File Transfer Protokulü ve Bootstrap Protokolü standartları için Request For Comments dökümanlarından yararlanılmıştır.

ABSTRACT

In this thesis work, application programs of File Transfer Protocol from TCP/IP protocol family and Bootstrap Protocol from UDP/IP protocol family, that will serve in the internet environment has been realized by using client-server programming model. Those programs are named as FTP Server and BOOTP Server and developed under NetWare 3.12 operating system via Novell Software Development Kits and Watcom C Compiler. Since both programs are designed as NetWare Loadable Modules, they are seen as modules of NetWare operating system. NetWare operating system supports multithreaded programing which is necessary for designing distributed applications. Besides NLM C interface helps the designer with numerous functions.

The basic building block for client - server communication is the socket. The Berkeley Unix 4.2BSD release introduced new facilities for interprocess communication and networking. FTP Server program uses stream type sockets, while BOOTP Server program uses datagram type sockets. The objectives of FTP are, to promote sharing of files (computer programs/data), to encourage indirect or imlicit use of remote computers and to shield a user from variations in file storage systems among hosts, besides to transfer data reliably and efficiently. Bootstrap protocol, allows a diskless client machine to discover its own IP address, to configure itself dynamically and the assignment of local network resources.

BÖLÜM 1

1.1 Genel Giriş

Berkeley Unix 4.2BSD versiyonu prosesler arası haberleşme (interprocess communication -IPC) ve networking için yeni olanaklar getirmiştir. Bu olanakların temelindeki fikir, dosya tanımlayıcı kullanarak I/O işleminin yapılmasıdır. Bütün proseslerin data dosyalarını okumak ve yazmak için, zaten dosya tanımlayıcı kullanıyor olmaları bu fikrin dayanak noktasıdır.Daha önce signal ve pipe yöntemleri IPC amacıyla kullanılmaktaydı, fakat bu yöntemlerin yetenekleri sınırlıydı. Örnegin pipe tekniğinde iki proses arası tek yönde I/O yapılabilirdi.

BSD soket tipi, pipe'larin iki yönlu I/O yapabilen türleridir. Soketler , Internet domain ve Unix domain'den tek bir tanesi için kullanılabilirler. Unix domain'inde, aynı dosya sistemindeki iki prosesin haberleşmesine izin verilirken, Internet domain'inde farklı makinalar arası haberleşme mümkündür. Internet protokolleri (TCP, IP, UDP) kullanılarak, iki türlü haberleşme saglanabilir.: stream ve datagram. Transmission Control Protocol'un desteklediği stream türü haberleşmede, iki soket arası bağlantı kurularak, güvenli ,hatasız ve sıralı haberleşme sağlanır.User Datagram Protocol'un desteklediği datagram türü haberleşmede, soketler arası bağlantı yoktur ve her datagram gideceği adresi kendi içersinde taşır.

Bu çalışm iki aşamadan oluşmaktadır. Birinci aşamada , client-server modelinde tasarlanan, Internet uygulama protokollerinden olan ve TCP'nin üzerinde çalışan File Transfer Protokolu (FTP) için, server programı geliştirilmiştir. İkinci aşamada , yine client-server modelinde tasarlanan, UDP'nin üzerinde çalışan Boot Strap Protokolu (BOOTP) için, server program geliştirilmiştir. Her iki uygulamadada client-server arası haberleşme amacı ile BSD soketleri kullanılmıştır. FTP server uygulamasında stream tipi soketlerden yararlanılırken, BOOTP server uygulamasında datagram tipi soketler kullanılmıştır. File Transfer Protokolunun özelliği gereği, dosya transfer işleminin verimli ve güvenilir olabilmesi icin, TCP tercih edilmiştir. Boot Strap Protokolunun yapısı gereği, tek-servis erişimi (single-service access - SAS) ile gerekli olan bilgiler sağlanabildiğinden, UDP tercih edilmiştir.

Protokolleri uygulama programları, NetWare 3.12 işletim sistemi ortamında geliştirilmiştir. NetWare işletim sistemi multithread programlamayı desteklediğinden, dağıtık uygulamalar icin uygun bir ortam sunmaktadır. Programlar, Yüklenebilir NetWare Modulleri olarak tasarlandığından, işletim sisteminin bir parçası olarak kabul edilirler.

1.2 Yüklenebilir NetWare Modülleri

Yüklenebilir NetWare Modulleri (NetWare Loadable Modules - NLMs), Netware işletim sisteminin temel yapı taşlarını olusturan programlardır. Bu programlar NetWare server belleğinde işletim sistemi ile çalişmak üzere tasarlanırlar. NLM'ler belleğe yüklendiklerinde işletim sisteminin bir parçası olduklarından, herhangi bir servis protokolü kullanmadan, doğrudan NetWare servislerine erişebilirler. NetWare'in sağladıgı pek cok temel servis NLM'ler sayesinde gerçekleşmektedir. NLM'ler NetWare işletim sistemine açıklık, modülerlik ve esneklik kazandırırlar.

NetWare isletim sistemi, network servislerinin optimizasyonu için tasarlenmış özel amaclı bir işletim sistemidir. Özellikle client/server uygulamaları için verimli olması dikkate alınarak tasarlanmıştır. Print server, database server, communications server gibi uygulamalar veya paylasılan kaynaklar için servis sunan bütün uygulamalar NetWare ortamında verimli çalışırlar.

NetWare'in verimli olmasinin nedenlerinden biri de nonpreemptive sekilde çalışmasıdır. Bunun anlamı uygulamalar ve işletim sistemi, başka uygulamalar tarafından kesilmeksizin çalışabilirler. Gerçekten de, eğer bir NLM CPU'nun kontrolunu periyodik olarak birakmaz ise, diger NLM'lerin çalışma şansı olmayacaktır. O halde bu nonpreemptive ortamda, uygulamalar sistem kaynaklarını kullanmakta bencil davranmamalıdırlar. NOS nonpreemptive oldugu için, NLM'lerin çalışma şekillerini gözlemez, böylece verimliliği artar.

NetWare isletim sistemi, CPU'nun protected modunda çalışır. Protected modunda, bellek sürekli bir adres dizisi şeklinde adreslendiğinden (flat memory model) bellek yönetimi esnek ve kolay olmaktadır. Protected modun bir diğer avantajıda çok sayıda programı eş zamanlı (concurrent) çalıştırabilmesidir (multitasking). NetWare ortamında, her task/process bir **thread** olarak adlandırılır. Multithreaded programlama NetWare avantajlarını en verimli şekilde kullanabilmektedir.

1.3 Client - Server Modeli

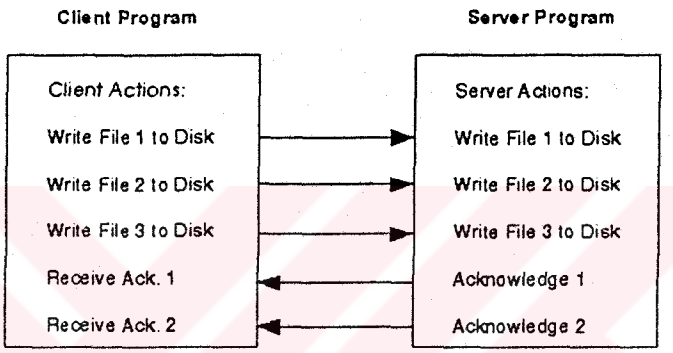
NetWare üzerinde çalışan dağıtık uygulamaların tasarımında yaygın olarak client-server modeli kullanılmaktadır. Bu modelde client program, server programdan servis isteğinde (request) bulunur. Server program gelen servis isteklerine cevap (reply) gönderir.
NLM'lerde kullanılan client-server modelin genel karakteristiği aşağıdaki gibi sıralanabilir :
- Server program file server'da NLM olarak ver alır.

- Server program servislerini başlatarak, değişkenlerini hazırlayarak, client'dan gelecek istekleri bekler.
- Client program bir servise ihtiyaç duydugu zaman, ilgili server'i bularak, istekte bulunur.
- Client - server arasındaki haberleşme istek/cevap çifti şeklinde gerçekleşir. İstekte bulunan client, cevap üreten ise server'dır.

1.3.1 Client-Server Haberleşmesi

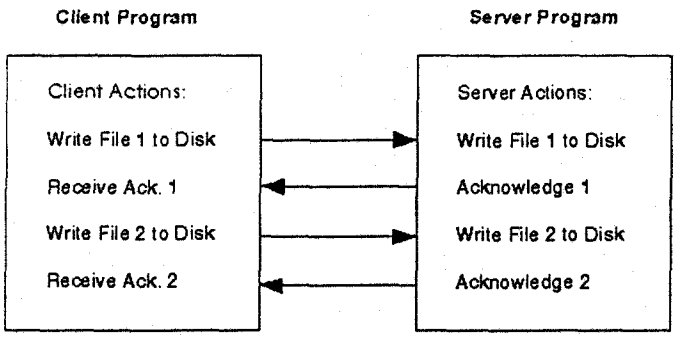
Client-server haberleşmesi iki şekilde olabilir : asenkron haberleşme ve senkron haberleşme.

- Asenkron Haberleşme : Client program servis isteklerini gönderdikten sonra, server'dan cevap gelinceye kadar çalışmaya devam eder.



Şekil 1.1. : Client -server arası asenkron haberleşme

- Senkron Haberleşme : Client program, servis isteğinde bulunduktan sonra, server'dan cevap gelinceye kadar çalışmasını durdurur ve bekler.



Sekil 1.2 : Client-server arası senkron haberleşme

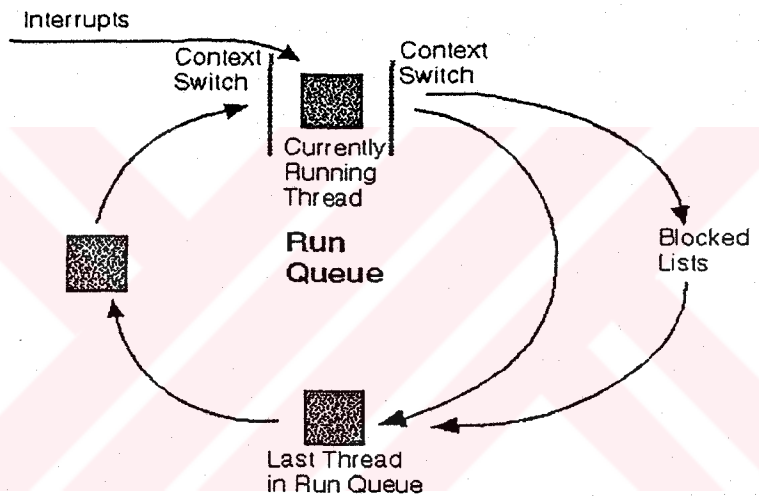
Bu tez çalışmasında client-server arası haberleşme yöntemi olarak senkron haberleşme uygulanmıştır.

1.4 Thread'ler

Netware işletim sistemi, NLM'lerin çoklu-thread içermelerine izin vermektedir. Her NLM'in, main() fonksiyonunu yürüten en az bir thread'i vardir. Herhangi bir anda çalışan tek bir thread vardır ve CPU kontrolu elindedir. Bu durum, thread çalışmasını bitirinceye kadar veye CPU kontrolunu bıraktıracak (relinquish) bir blocking fonksiyon çağırıncaya kadar devam eder. Aktif olarak çalışan bir thread'e hiç bir thread müdahale edemez, sadece hardware interrupt çalışan thread'i geçici olarak kesebilir.

1.4.1 NetWare 3.x İşletim Sisteminde Thread Yönetimi

NetWare 3.x işletim sisteminde thread yönetimi 4.0 versiyonundan farklıdır.

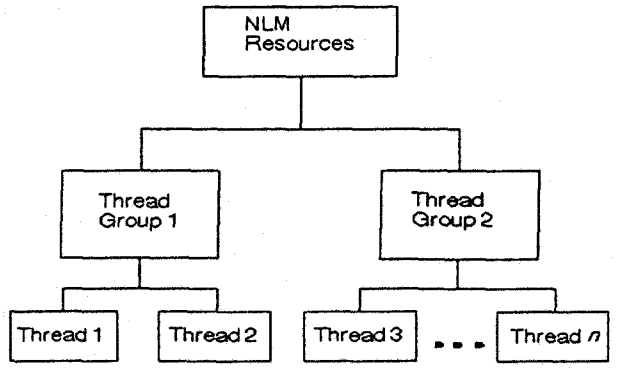


Şekil 1.3 : NetWare 3.x işletim sisteminde thread yönetimi.

NetWare 3.x işletim sisteminde çalışmayı bekleyen thread'ler Çalışma Kuyruğuna yerleştirilirler. Bu kuyruk FIFO mantığında çalışır. Kaynaklara erişmeyi bekleyen thread'ler Blok Listesine yerleştirilerek, CPU kontrolünü Çalışma Kuyruğundaki diger thread'lere bırakırlar. İhtiyaç duyduğu kaynağa erişebildiği zaman, thread Blok Listesinden alınarak Çalışma Kuyruğunun sonuna yerleştirilir. Thread, Çalışan Thread durumuna her girdiğinde veya Çalışan Thread durumundan bir başka duruma geçtiğinde, **context switch** gerçekleşir.

1.4.2 Thread Grupları

Bir thread grubu, programcı tarafından tanımlanan bir veya daha fazla thread'den oluşur. Bir thread grubundaki thread'ler aynı context'i paylaşırlar.



Sekil 1.4 : Multithread NLM yapısı.

Yukarıdaki şekilde multithread'li bir NLM konfigürasyonu görülmektedir. Thread1 ve Thread2, Thread Grup1 adlı aynı gruba aittir. Numaralandırılmış diğer thread'ler Thread Grup2'ye aittir. Bunun anlamı, Thread1 ve Thead2 aynı thread-grup-seviyesi context bilgisini paylaşırlar. Thread3 ile threadn arası thread'ler farklı bir context paylaşırlar.

1.4.3 Multithreaded Programlama

Multihread yapısı dagıtık uygulamalarda kullanılır. Örneğin, bir client-server NLM, servis verdiği her client için farklı bir thread grubu kullanır. Bu yöntem NLM'in çok sayıda client'a eş-zamanlı servis vermesini sağlar. Multithread yapısını kullanmanın avantajlarını aşağıdaki şekilde sıralayabiliriz.

- Modül yapısı ile kodun basitleştirilmesi : İşlemleri thread'ler şeklinde ayırmak, programların kolayca anlaşılmasını, bakımının yapılabilmesini sağlar.
- İş üretiminin artırılması : NLM'i çok sayıda thread'e bölerek, CPU'nun boş kaldığı süre azaltılabilir. I/O istekleri sırasında CPU bloke olmak yerine, işletim sistemi kontrolü bir diğer thread'e devreder.
- Cevap süresini kısaltır :Server, thread'ler arası geçiş yapabildiğinden (tek bir thread'in CPU kullanımını tekeline almasını önleyebildiğinden), client'lar isteklere daha hızlı cevap alabilirler. Böylece bir client'tan gelen uzun bir I/O-yoğun istek sırasında, diger bir client'ın isteği yerine getirilebilir.

- Thread grupları sayesinde çoklu context kullanılabilir: Aynı thread grubundaki thread'ler, aynı context'i paylaşacağından (örneğin Current Working Directory, current connection gibi) programcıya kolay çözümler sunacaktır.

Multithread yapısını kullanmanın avantajı performans ve verimliliğin artmasıdır. Mulltithread uygulamalarda, process'ler sistem kaynaklarının kullanılmasında daha eşit zamana hak kazanırlar. Ayrıca her thread diğerlerinden bağımsız olarak çalışabilmektedir.

1.5 NLM'lerde Context Kavramı

NLM C arayüzü, çalışan her NLM için bir context sağlar. Context üç seviyeden oluşmaktadır: thread-level context, thread-group-level context ve NLM-level context. Her context seviyesinde farklı bilgiler yer alır. Bu bilgiler, değişkenler, ekranlar, işaretçiler, sayıcılar, referanslar, kontrol bilgisi ve parametreler şeklinde olabilir.Adı geçen context seviyelerini inceleyecek olursak:

- Thread-level context : Bir NLM içindeki en özel context bilgilerini içerir. Her thread'in context'i sadece kendisine aittir. Bir thread'deki dataya bir diğer thread referans veremez.

-Thread-group-level context : Bir thread grubundaki bütün thread'ler aynı context'i paylaşırlar. Bir thread'in thread-group verisine yaptığı herhangi bir değişiklik, gruptaki diğer bütün thread'leri de etkiler. Farklı thread gruplarındaki değişiklikler birbirini etkilemez.

-NLM-level context : Bir NLM'deki bütün thread'ler ve thread-grupları tarafından paylaşılan bir seviyedir. Bu seviyede verilerin tek bir değeri vardır, dataya yapılan değişiklikler NLM'deki bütün thread ve thread-gruplarını etkiler.

1.6 Senkronizasyon Servisleri

Senkronizasyon servisi sunan fonksiyonlar, uygulamaların network dosyalarına ve diğer kaynaklara erişim koordinasyonunu yapar. Bu servisler iki kategoriye ayrılır: kilitleme (locking) ve semaforlar.

- Kilitleme : Bu yöntem ile dosya türünde bir kaynağa sadece bir thread'in erişmesi sağlanır. Thread'ler dosya adını ve boyutunu bir log tablosuna yazarak ve daha sonra tek bir lock çağrısı ile tablodaki bütün kaynakları kilitleyebilirler. Log tablosu kullanmanın yararı, bir bütün olarak hepsinin birden kilitlenip veya kilitlenmemesini sağlamasıdır. Böylece deadlock durumu önlenmiş olur.

- Semaforlar : Kilitleme yönteminde tek bir thread'in kaynağa erişmesi mümkünken, semaforlar network kaynaklarına erişebilecek thread sayısına bir limit getirirler. Semafor kullanarak bir kaynağın kullanıcı sayısı sınırlandırılmış olur. Semaforlar dosya ve device'lar için kullanılırlar.

Iki tür semafor vardır :

- Network semaforları : Network üzerindeki server'lara ve iş-istasyonlarına ait kaynaklar için kullanılırlar.

-Yerel semaforlar : Tek bir server'a ait kaynaklar için kullanılırlar.

Bir NLM içindeki threadler arası haberleşme için yerel semaforlar kullanılır.

1.7 NLM'in Kaynakları Bırakması

NLM'ler calışmaları bittiğinde aldıkları kaynakları (bellek, soket,ekran, device, semafor gibi) bırakmaktan sorumludurlar. Program sona ererken bütün kaynaklar işletim sistemine devredilmelidir. NLM sonlanırken, NetWare ve NLM C arayüzü, NLM'in kullandıgı kaynakları özgür bırakırlar. Fakat, yerel semaforları NLM'in kendisi kapatmalıdır. Programdan çıkişta yerel semaforların kapatılmaması, file server'in çökmesine neden olur.

1.8 NLM'in Sonlanması

NLM C arayüzü fonksiyonları kullanılarak, NLM sona erme işlemini kendisi kontrol edebilir. Bu amaca yönelik üç mekanizma vardır :

- Kendi kendini sona erdirme .
- NLM'in unload edilmesi.
- Beklenmedik bir durum sonunda normal olmayan sona erme.

BÖLÜM 2

2.1 BSD Socket Yapısı

BSD Soket yapısı, TCP/IP tabanlı uygulamaların NetWare ortamında yapılabilmesine yardımcı olur. BSD soketlerinin UNIX dosya sistemi arayüzüne benzemektedir. Bir disk dosyası açmak yerine programda bir soket yaratılmakta ve bu soketin I/O tanımlayıcısı (handle), bir disk dosyasıymış gibi okuma,yazma ve soketi kapama amacıyla kullanılmaktadır.

Program tarafından yaratılan her soketin bir network adresi vardır. Data göndermek için, program hedef soketin adresini sağlamalıdır. Soketleri kullanacak bir network uygulama programının izlemesi gereken adımlar aşağıdaki gibi olmalıdır :

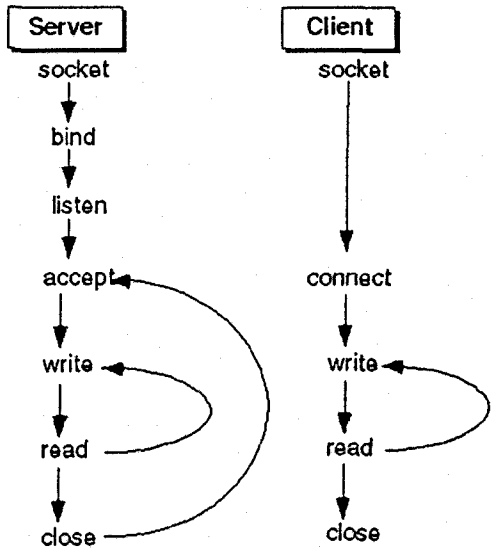
1- Bir soketi yaratmak ve onu belirli bir protokol (TCP veya UDP) ile birleştirmek: Bu amaçla hem client hem de server program **socket** fonksiyonunu kullanırlar.

2- Yaratılan soketi, lokal host'un IP adresi ile ve lokal soketin port numarasi ile birleştirmek: Bu amaçla hem client hemde server program **bind** fonksiyonunu kullanırlar.

3- Soket'i ve lokal port'u, remote host'un IP adresi ile ve remote soketin port numarası ile birleştirmek: Bu amaçla client program **connect** fonksiyonunu kullanırken, server program **listen** ve **accept** fonksiyonlarını kullanır.

4-Datayı transfer etmek: Bu amaçla hem client, hemde server program **read, write, sendto, recvfrom** fonksiyonlarını kullanırlar.

5- Lokal ve remote soketlerin kapatılması: Bu amaçla hem client hemde server program **close** fonksiyonunu kullanırlar.



Şekil 2.1 : Berkeley Soketleri Programlama Modeli.

2.2 Soket Tipleri

Bütün network programları, stream veya datagram soket tiplerini kullanırlar:

- Strem soketleri, sıralı, güvenilir, ve connection oriented hizmet sunarlar, TCP protokolü ile kullanılırlar.
- Datagram soketleri, sınırlı uzunlukta, bağlantısız ve güvenilir olmayan bir hizmet sunarlar, UDP protokolü ile kullanılırlar.

2.3 Soket Fonksiyonlarının Kullanımı

Server program açısından soket fonksiyonlarının kullanımı bu bölümde ayrıntılı olarak yer almaktadır .

2.3.1 Soketin Yaratılması

Bir host'tan diğer bir host'a data gönderebilmek için lokal host'ta bir soket yaratılmalıdır. Bu amaçla kullanılan **soket** fonksiyonunu iki gorevi vardır :

- Soketi yaratma ve kaynakları hazır hale getirme,
-Bir NetWare file handle açarak, bunu soket kaynakları ile bağlama, böylece soketler de device'lara tanınan işletim sistemi servislerinden yararlanabileceklerdir.

Soket fonksiyonunun formatı : *s = socket (domain, type, protocol)*

s : fonksiyonun geri döndürdüğü soket file handle.

domain : protokol ailesini gösterir, NetWare ortamında degeri PF_INET olmalıdır.

type : soket tipinin TCP veya UDP olduğunu gösterir.TCP icin SOCK_STREAM, UDP için SOCK_DGRAM olmalıdır

protokol : soket icin kullanılan belirli bir protokolü gösterir.

2.3.2 Network Adresinin Bind Edilmesi

Bind fonksiyonu ile network adresi, soket file handle ile bağlanır.

Bind fonksiyonunun formatı : *status = bind (s, addr, addrlen)*

status : fonksiyonun geri döndürdüğü durum kodu.

s : soket file handle (soket fonksiyonu ile elde edilen)

addr: TCP ve UDP soketlerinde sockaddr_in yapısını işaret eder. Bu yapı PF_INET domain'indeki network adresini içerir.Network adresi, IP domainini ve port numarasını kapsamaktadır

sockaddr_in yapısı :

```
struct in_addr
{
        unsigned long s_addr ;
}

struct sockaddr_in
{
        short                   sin_famiily;
        unsigned  short         sin_port;
        struct    in_addr       sin_addr;
        char                    sin_zero[8];
}
```

addrlen : addr ile işaret edilen sockaddr_in yapısının boyutunu byte cinsinden gösterir.

2.3.3 Bağlantı İsteğinin Alınması

Listen fonksiyonu ile soketin, gelen bağlantı (connection) isteklerini alması saglanır.

Listen fonksiyonunun formatı : *status = listen (s, backlog)*

status : fonksiyonun sonucunun başarılı olup olmadığını gösterir.

s: soket file handle (soket fonksiyonu ile elde edilen)

backlog : bir kuyrukta tutulabilecek bağlantı isteklerinin kabul (accept) edlilmeden önce, kaç tane olabileceğini gösterir. Accept fonksiyonuna yapılan her çağrı, bu kuyruktan bir bağlantı isteğini siler.

2.3.4 Bağlantı İsteğinin Kabul Edilmesi

Accept fonksiyonu ana soketin kuyruğunda bekleyen bağlantı isteği için yeni bir soket yaratır.
Accept fonksiyonunun formatı : *s2 = accept (s, addr,addrlen)*

s2 : accept fonksiyonu s soketi ile aynı özellikleri taşıyan yeni bir soket yaratarak, bu sokete s2 handle değerini verir.Bu handle artık bu baglantı için yapılacak bütün I/O işlemlerinde kullanılacaktır. Böylece tek bir soket listen edilerek, çok sayıda accept yapılabilmektedir. Her accept işlemi ile kuyruktan bir istek silinmekte ve yeni isteklerin kabulü sağlanmaktadır.
addr : client programın internet adresını ve port numarasını içeren sockaddr_in yapısını işaret eder.
addrlen : addr'nin byte cinsinden boyutunu gösterir.

2.3.5 Data Transferi

Baglantı kurulduktan sonra, data transferi artık iki yönlü yapılabilir.
Bu amaçla kullanılan **read** ve **write** fonksiyonlarının formatı :
rc = write (s, buf, sizeof(buf))
rc = read (s, buf, sizeof(buf))
rc : read için, soketten buf 'a kaç byte data transfer edildiğini gösterir.
write için, buf ' tan sokete kaç byte data transfer edildiğini gösterir.
s : soket file handle (soket fonksiyonu ile elde edilen)
buf : read için, soketten okunan datanın yerleştirildiği buffer.
write için, sokete yazılacak datanın bulunduğu buffer.

2.3.6 Baglantının Kapatılması

Programın tasarımına göre, client veya server bağlantıyı önce kapayabilir.Örneğin sokete yazarken, yazma fonksiyonunun sıfır sonucunu döndürmesi, karşı tarafın soketi kapattığını gösterir, o halde diğer tarafta bağlantıyı kapatmalıdır.

2.4 Port Numaraları

Makinalar arası mesaj gönderileceği zaman, bu mesajlar hedef makinanın protokol ilgilisine (protocol handler) gönderilirler. Protokol ilgilisinin görevi, adresi yorumlayarak, mesajın hangi sokete ait

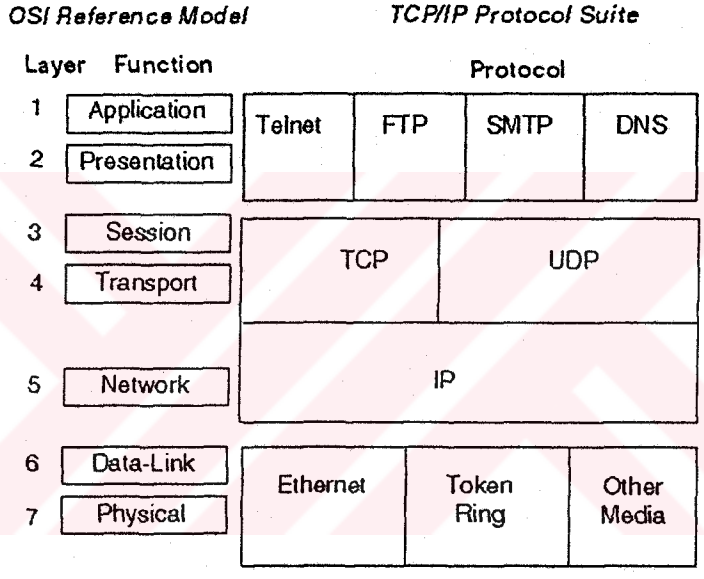
olduguna karar vermektir. Soket bir port ile bağlantılıdır ve protokolu yerine getiren fonksiyonlar port'u da dağıtım noktası (delivery slot) olarak kullanırlar.

Bir port numarası, protokol'un ilgili mesajları koydugu bir mailbox olarak düşünülebilir. 1 -1023 arası tanımlanmış "well-known" port numaraları belirli protokollere ayrılmıştır. Örneğin, 20 ve 21 numaralı portlar File Transfer Protokolu için, 67 ve 68 numaralı portlar Boot Strap Protokolü için ayrılmıştır.

BÖLÜM 3

3.1 Protokol Aileleri

BSD soket yapısı, Internet Protocol (IP) ailesi protokollerini destekler. IP ailesi aşağıdaki protokolleri kapsar: Internet Control Message Protocol (ICMP), Transmission Control Protocol (TCP), User Datagram Protocol (UDP), Internet Protocol (IP), Address Resolution Protocol (ARP). Aşağıdaki şekilde görüldüğü gibi, IP protokolu, hem TCP'nin hem de UDP'nin altında bulunur ve farklı medya içeren networkler arası end-to-end data dağıtımını saglar.



Şekil 3.1 : OSI modeli ile TCP/IP protokol modeli.

TCP/IP bilgisayarların birlikte uyum içersinde çalışarak, network üzerindeki kaynakları paylaşabilmeleri için geliştirilen protokoller grubudur. Bu protokollerin bazıları, pek çok uygulama için gerekli olan alt seviye fonksiyonlarını yerine getirirler.: IP, TCP, UDP. Diğer protokollerinde servis sunmaya yönelik belirli görevleri vardır : FTP, TELNET, EMAIL.

Bu tez çalışmasında NetWare işletim sistemi üzerinde, TCP/IP uygulama protokollerinden olan FTP ve BOOTP gerçekleştirilmiştir.

3.2 Transmission Control Protocol (TCP)

TCP, güvenilir, iki yönlü, circuit-oriented, sıralı veri aktarımını destekleyen bir protokoldür. TCP protokolü stream soket tipini destekler. Her TCP soketinin adresi tektir ve host IP adresi ile TCP port numarasını içerir. Karşılıklı iki soket arası baglantı kurulduktan sonra , bu baglantıyı tanımlayan dört eleman vardır:

- Lokal host'un IP adresi
- Lokal soketin port numarası
- Remotee host'un IP adresi
- Remote soketin port numarası

Bu çalışmada File Transfer Protokolunun uygulandığı server programında TCP protokolü kullanılmıştır.

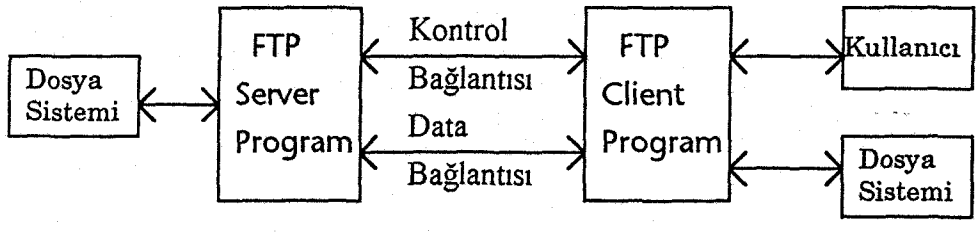
3.3 User Datagram Protocol (UDP)

UDP'de, verinin iki yönlü message-oriented aktarımı sağlanırken, güvenilir, sıralı ve undublicated olması garantisi yoktur. UDP protokolü datagram soket tipini destekler, bu soketler baglantısız yapıdadır. Uygulamada kısa süreli istek/cevap etkileşimi söz konusu ise, datanın garantili ve sıralı dağıtımı gerekli değilse UDP protokolu tercih edilmelidir.

Bu çalışmada Bootstarp Protokolunun uygulandığı server programında UDP protokolu kullanılmıştır.

3.4 File Transfer Protocol (FTP)

FTP, farklı makinalar arası dosya transferini sağlar. Bir sistemde çalışan kullanıcılar, bir diğer sisteme erişerek, dosya alıp, dosya gönderebilirler. Dosya transfer işleminin verimli ve güvenilir olabilmesi için FTP, TCP'nin üzerinde çalışır.



Şekil 3.2 : FTP Modeli

Bir FTP Client'in, FTP servisinden yararlanabilmesi için, FTP Client programını çalıştırması gerekir. Bu amaçla kullanıcılar "ftp" komutunu kullanırlar. Öncelikle FTP Client ile FTP Server arasında Kontrol Bağlantısı kurulur. Bu bağlantıyı kurma isteği client'dan gelir. Kontrol bağlantısı üzerinden FTP komutları (requests) ve bunlara FTP server'in verdiği cevaplar (replies) gönderilir. Verinin karşılıklı olarak aktarımı için ihtiyaç duyuldukça Data Bağlantısı kurulur, istenilen yönde data transferi yapılır ve bu bağlantı sona erdirilir.

Client ve server arasindaki bir FTP session boyunca, Kontrol Bağlantısı devamlı mevcuttur, ama Data Bağlantısı ihtiyaç duyuldukça kurulur. Client'in gönderdiği FTP komutları , kurulacak olan Data Bağlantısının parametrelerini belirtir (data port'u, transfer modu gibi). Kontrol Bağlantısının kapatılması client'in isteği ile olur. Tanımlanmış portlar arasında, FTP Kontrol Bağlantısı için 21 numaralı port, Data Bağlantısı için 20 numaralı port ayrılmıştır. Kontrol Bağlantısı ve Data Bağlantısı client port ile server port arası kurulan TCP bağlantısı şeklindedir.

Client program, FTP komutlarını göndermekten ve bunlara gelen cevapları yorumlamaktan sorumludur. Server program, gelen komutları yorumlayarak, bunlara cevap göndermekten ve Data Bağlantısı kurmaktan sorumludur.

3.4.1 Data Bağlantısının Kurulması

Client'in Kontrol Bağlantısı üzerinden gönderdiği komutun, bir transfer komutu olması halinde (get/retr, put/stor) server'in izlediği prosedür aşağıdaki gibidir :

- Kullanıcı bir transfer komutu çalıştırır (get/put).
- Client program Data Bağlantısının kurulabilmesi amacıyla , PORT komutunu server programa gönderir.
- Server program PORT komutunu onayladığını gösteren 200 numaralı cevabı clent'a gönderir.
- Client program PORT için olumlu cevabı alınca, transfer komutunu Kontrol Bağlantısı üzerinden server'a gönderir.
- Bu arada server program Data Bağlantısını istenilen adres ile 20 numaralı port üzerinden kurar.
- Server program transfer komutunu alınca, komutun amacına göre bilgiyi 20 numaralı port'a yazar (get) veya bu port'tan okur (put).
- Server program data Bağlantısını kapatmak amacıyla 20 numaralı port'un açıldığı soketi kapatır.

3.5 FTP Komutları

3.5.1 Erişimi Kontrol Eden Komutlar

Kullanıcı Adı (USER)

FTP Kontrol Bağlantısı kurulduktan sonra Client'in göndermesi gereken ilk komuttur. Arguman olarak kullanıcı adı verilir. Kullanıcı adı server'in dosya sistemine erişirken ihtiyaç duyduğu bir bilgidir. Bu komut bağlantı sırasında tekrar kullanılarak farklı bir kullanıcı için server'dan hizmet alınabilir.

Şifre (PASS)

Arguman olarak kullanıcının şifresi verilir. Server'a erişim için, doğru olarak verilmesi gerekir, aksi halde login işlemi yapılamaz.

Çalışılan Directory'nin Değiştirilmesi (CWD)

Arguman olarak pathname girilir. Bu komut ile kullanıcının farklı directory'lere erişmesi sağlanır.

Ana Directory'e Geçiş (CDUP)

CWD komutunun bir başka türüdür. Kullanıcının farklı bir directory'den, kendi home directory'sine dönmesini sağlar.

Çıkış (QUIT)

FTP programını sona erdirerek, server'in Kontrol Bağlantısını kapatmasını sağlar.

3.5.2 Parametre Aktarım Komutları

Veri aktarımı için kullanılan parametrelerin bilinen (default) değerleri vardır. Bu değerler değiştirileceği zaman aşağıdaki komutlar kullanılır.

Data Port (PORT)

Arguman olarak Data Bağlantısında kullanılacak port numarası, ve client'in IP adresi gönderilir. Bu arguman 32-bit Internet host adresini ve 16-bit TCP port adresini içerir.

Verinin Tipi (TYPE)

Arguman olarak datanın ne şekilde olacağı verilir. Arguman değeri ASCII veya binary (image) olabilir.

3.5.3 FTP Servis Komutları

Kullanıcının hangi yönde dosya transfer isteğinin olduğunu gösteren komut grubudur.

Client'a Data Alma (RETR)
Arguman olarak pathname ve dosya adı verilir. Pathname verilmediği zaman, çalışılan directory (current working directory) göz önünde tutulur. Adı verilen dosyanın bir kopyası server'dan client'a gönderilir.

Server'a Data Alma (STOR)
Arguman olarak pathname ve dosya adı verilir. Pathname verilmediği zaman, çalışılan directory (current working directory) göz önünde tutulur. Adı verilen dosyanın bir kopyasının client'dan server'a gönderilmesini sağlar. Gönderilen dosya server'da mevcut ise ve yazma izni varsa eskisinin üzerine yazılır, mevcut değilse yeni bir dosya oluşturulur.

Listeleme (LIST)
Arguman olarak pathname veya wildcard (*,?) karakterleri kullanılarak çeşitli listeleme şartları verilebilir. İstenilen directory listesinin, server'dan client'a gönderilmesini sağlar. Arguman kullanılmaması, çalışılan directory listesini ifade eder. Bu komut sırasında data tipi ASCII olmalıdır.

Calışılan Directory Adı (PWD)
Çalışılan directory adının ilgili cevap numarası ile Kontrol Bağlantısı üzerinden gönderilmesini sağlar.

Sistem Adı (SYST)
Server'daki işletim sistemi hakkında bilgi verir.

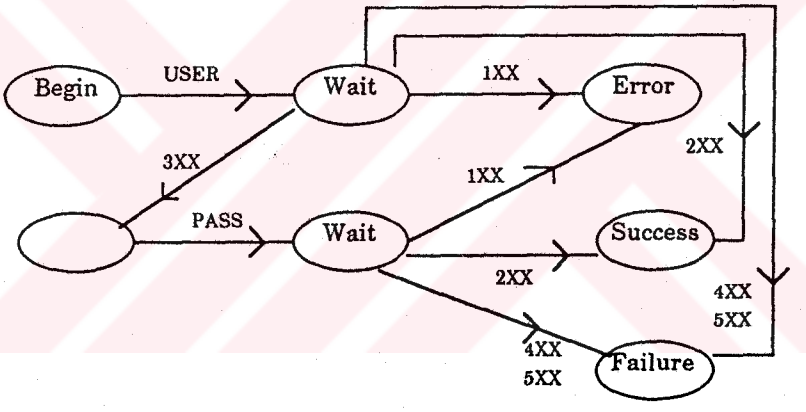
3.6 FTP Cevapları

FTP komutlarına karşı server'in gönderdiği cevapların amacı, dosya transferı için senkronozasyonu sağlarken ve her zaman için client'in server'in durumundan haberdar olmasını sağlamaktır. Client bir FTP komutu gönderdiğinde , server bu komuta karşılık gelen cevabı (reply) ilgili cevap numarası ile gönderir.

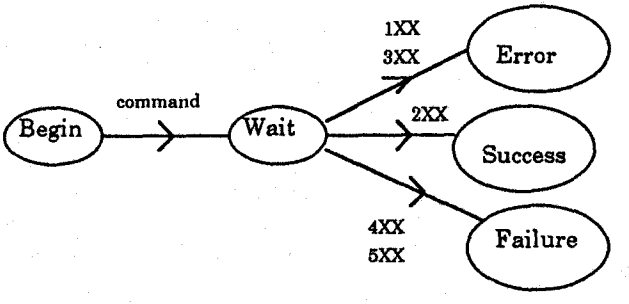
Başka bir komut göndermeden önce, client bir cevap gelmesini bekler. Bazı komutlar birden fazla cevap gerektirebilir. Örnegin, RETR komutu için ilk cevap ile Data Bağlantısının kurulduğu bildirilir, dosya transfer edilir ve ikinci bir cevap ile Data Bağlantısının sona erdiği client'a bildirilir.

Komut-cevap sıralamasının detayı , aşağıdaki durum diyagramlarında görülmektedir. WAIT durumu client'ın cevap için beklediği durumu simgeler. Her komut için üç olası durum vardır : Success, Failure, Error. Server'in gönderdiği cevaplar, 1-5 arası gruplara ayrılmıştır :

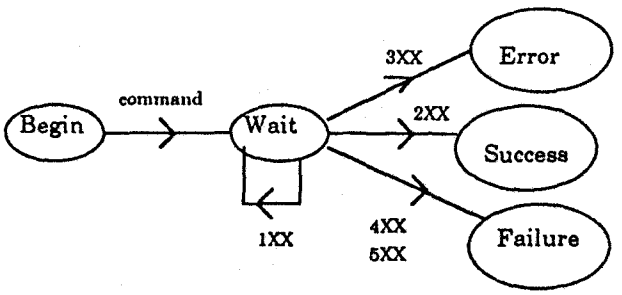
- 1XX grubu cevaplar, istenen işlemin başlatıldığını gösterir.
- 2XX grubu cevaplar, istenen işlem yerine getirildi, yeni bir istek başlatılabilir.
- 3XX grubu cevaplar , istenen işlemin başlatıldığını fakat tamamlanabilmesi için client'ın yeni bir komut göndermesi gerektiğini gösterir.
- 4XX grubu cevaplar, komutun kabul edilmediğini, ve istenen işlemin yapılmadığını gösterir, ama hata durumu geçicidir, client tekrar isteği tekrarlayabilir.
- 5XX grubu cevaplar, komutun kabul edilmediğini ve istenen işlemin yapılmadığını gösterir.



Şekil 3.3 : Login işlemi için sonlu durum makinası.(USER, PASS)



Şekil 3.4 :Parametre aktarım komutları için sonlu durum makinası.(PORT, PWD, TYPE, QUIT)



Şekil 3.5 : Servis komutları için sonlu durum makinası. (LIST, RETR, STOR)

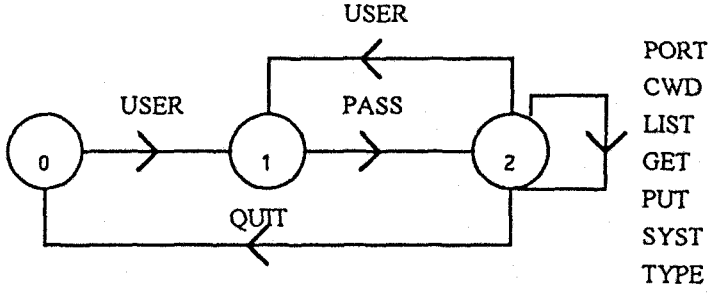
3.7 Tipik Bir FTP Senaryosu

Kullanıcının Girdiği Komutar	Yapılan İşlem
ftp fs0 <CR>	fs0 makinası ile Kontrol Bağlantısı kurulur.
	<------220 Service ready<CRLF>
userid : esin<CR>	USER esin <CRLF>------->
	<------331 User name ok, need password <CRLF>
password : abc<CR>	PASS abc <CRLF>------->
	<------230 User logged in <CRLF>
retr test.doc<CR>	PORT 193,140,4,17,1234 <CRLF>------->
	<------200 Command ok <CRLF>
	RETR test.doc <CRLF>------->
	<------150 Open data connection<CRLF>
	data transferi yapilir
	<------226 Closing data connection<CRLF>
quit<CR>	QUIT<CRLF>
	<------221 Service closing control connection<CRLF>

BÖLÜM 4

4.1 FTP Uygulaması

Bu çalışmada client'tan gelen istekleri server programın değerlendirme şekli, aşağıdaki durum makinasında (finite state machine) gösterildiği gibidir.



Şekil 4.1 : FTP Sonlu Durum Makinası

0. Durum : Kullanıcı "FTP" komutunu bağlanmak istediği FTP server'ın adresi ile çağırmıştır. FTP Kontrol Bağlantısı kurulduktan sonra, bu aşamada login işlemi başlamaktadır. Öncelikle kullanıcıya "userid" sorulacak, kullanıcı file server'da tanımlı bir kullanıcı adı verince, 1. Duruma geçilecektir.

1. Durum : Kullanıcıya "password" sorulacak, doğru şifre girildiğinde de, login işlemi tamamlanarak, kullanıcı FTP server'dan servis almaya başlayacaktır.

2. Durum : Kullanıcı amacına yönelik olarak, FTP servislerinden gerekli komutları kullanarak yararlanabilecektir. Bu durumda gelen bir USER komutu, 1. Duruma geçerek, login işlemini yeniden başlatacaktır. Bu durumda gelen bir QUIT komutu client-server arası session'u sona erdirecektir.

FTP server programında uygulanan yöntem aşağıdaki şekilde listelenebilir :

1 - FTP konfigürasyon dosyası okunarak parametreler elde edilir.

2 - Client'la haberleşmeyi amacıyla soket işlemleri (socket, bind) yerine getirilir. Port adresi olarak 21 kullanılır.

3 - Oluşturulan soket listen() fonksiyonu ile dinlenerek , gelen bağlantı istekleri kuyrukta tutulur.

4 - Accept() fonksiyonu ile bir client baglantı istegi kabul edildiğinde, bu client ile ilgilenecek bir thread-grubu oluşturulur. ve kuyruktaki diğer client'larla ilgilenilir.

5 - Program supervisor tarafından unload edildiğinde izlemesi gereken prosedür, atexit() fonksiyonu ile tanımlanır.

6 - Artık client'tan gelecek bütün istekler ilgili thread grubu tarafından karşılanacağından bu istekler bir sonlu durum makinası ile değerlendirilerek her istek için gerekli işlemler yapılarak client'a cevap gönderilecektir.

7 - Thread-grup-level context kavramı sayesinde, farklı client'larin istekleri birbirini etkilemeyecektir. Örneğin, bir client için directory değiştirildiğinde, diğerleri bu durumdan etkilenmezler.

8 - Client'dan Data Bağlantısı isteği geldiği zaman, yeni bir soket açılarak, baglantı 20 numaralı port üzerinden, bilgiler istenilen yönde aktarılarak, işlem sona erdiğinde de soket kapatılacaktır.

9 - LIST isteği geldiğinde, directory bilgileri UNIX formatına göre listelenirler. Dosyanın kullanım hakları, sahibi, oluşturulduğu tarih, boyutu ve adı listelenir.

10 - USER komutu ile kullanıcı adı gönderildiğinde, kullanıcının file server'da tanımlı olup olmadığı (bir bindery object olup olmadığı) ve şifresinin doğruluğu kontrol edilerek, FTP hizmetinin kullanımı sağlanır.

11 - Client'dan beş dakika süre ile bir istek gelmez ise, server bu client'la olan Kontrol Bağlantısını kapatacak ve client'i 421 numaralı reply ile idle-time limitinin dolduğu konusunda haberdar edecektir.

12 - Client'in aldığı hizmetler bir log dosyasında saklanırlar.

13 - Client QUIT komutu gönderdiği zaman, sadece ilgili thread grubu sona erer ama server diğer client'lara hizmet etmeye devam eder.

14 - Program ancak supervisor tarafından konsoldan unload edildiğinde, hizmet sunmaya son verecektir.

4.2 Konfigürasyon Programı

FTP Server programının konfigürasyonu bir diğer NLM ile yapılmaktadır. FTP Server Utility adı verilen bu program menu-tabanlı olması nedeniyle, kullanıcıyı yönlendirmektedir. Program iki ana fonksiyondan oluşur :

1 - Parametrelerin değer alması (Set Parameters)

2 - FTP log dosyasının görüntülenmesi (View FTP Log File)

Birinci bölümde FTP server programı için parametrelerin default değerleri verilmektedir.

İkinci bölümde FTP servisinden yararlanan kullanıcılarla ilgili bilgilerin tutulduğu dosya sergilenmektedir.

4.2.1 Parametreler

Max Number of Sessions : Aynı anda FTP Server'in hizmet verebileceği client sayısını gösterir. Soket fonksiyonlarından olan Listen fonksiyonunda, backlog argumanının değerine karşılık düşmektedir.
Anonymous User Access : File Server üzerinde kullanıcı tanımları olmayan, misafir kullanıcıların FTP servisinden yararlanabilmeleri bu parametrenin olumlu olmasına bağlıdır.
Default User's Home Directory : Kullanıcıların erişim hakkının olduğu directory adı, absolute olarak bu parametrede verilir. Volume adı ve root directory adı verilmelidir.
Anonymous User's Home Directory : Misafir kullanıcının hangi directory'i kullanacağı bu parametre ile verilir.
Intruder Detection : Sistemi kullanım yetkisi olmayan kullanıcıların saptanması, bu parametrenin olumlu verilmesiyle sağlanır.
Number of Unsuccessfull Attempts : Aynı kullanıcının login işlemini üstüste bu parametre sayısı kadar hatalı yapması ile bu kullanıcının intruder olduğuna karar verilir ve bir log dosyasında kullanıcının adı ve adresi tutulur.
Log Level : FTP servisinden yararlananlarla ilgili bilgilerin tutulup tutulmaması bu parametrenin değerine bağlıdır (statistics / none).

4.2.2 Log Dosyası

Log dosyasında tutulan bilgiler :
- FTP servisinin başladığı tarih ve saat,
- FTP servisinin verildiği kullanıcının adresi,
- Retrieve edilen dosyaların sayısı,
- Stor edilen dosyaların sayısı,
- FTP servisinin sona erdiği tarih ve saat.

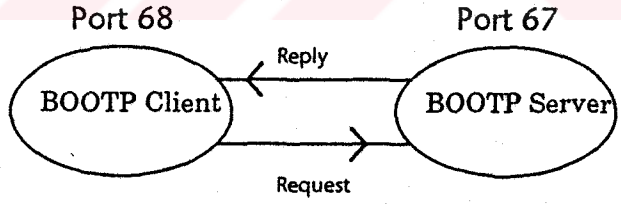
BÖLÜM 5

5.1 BOOTSTRAP Protololü

Bir IP/UDP protokolü olan Boot Strap Protokolü (BOOTP) disksiz bir client' makinanın hardware adresini kullanarak, server'dan IP adresini istemesi amacıyla kullanılır.

Protokolün özellikleri :

- Client-server arası tek bir paket değiş-tokuşu yapılır.
- Client bir cevap alıncaya kadar timeout süresi boyunca istekte bulunur.
- Her iki yöndede aynı paket formatı kullanılır.
- Client BOOTREQUEST paketi gönderirken, server BOOTREPLY paketi gönderir.
- BOOTREQUEST paketinde client'ın hardware adresi, BOOTREPLY paketinde client'ın IP adresi vardır.
- Server'ın, hardware adreslerine karşılık gelen IP adreslerini tuttuğu bir veritabanı vardır.
- BOOTP iki port numarası kullanır , BOOTP Client için 68 numaralı port, BOOTP Server için 67 numaralı port kullanılır.
- Client, hedef port olarak BOOTP Server (67) portunu kullanarak isteklerini gönderir.
- Genellikle, client isteklerini yayın (broadcast) şeklinde gönderir.
- Yumurta/tavuk problemi : eger, client IP adresini bilmiyorsa, server bu client'a datagram nasıl gönderebilir? Bu sorun, server'in uygun olan interface üzerindeki IP Broadcast adresini kullanarak BOOTREPLY göndermesi ile çözülür.



Şekil 5.1 : BOOTP client-server haberleşmesi.

5.2 BOOTP Paketinin Formatı

Alan Adı	Byte	Açıklama
OP	1	Paketin tipi : BOOTREQUEST = 1 , BOOTREPLY = 2
HTYPE	1	Hardware adres tipi: 10MB Ethernet = 1
HLEN	1	Hardware adres uzunluğu : 10MB Ethernet = 6

HOPS	1	(secimlik) Gateway'ler tarafından kullanılır.
XID	4	Transaction ID, istek-cevap çiftinin uyumu için kullanılır.
SECS	2	Client'ın istek göndermeye başladığından beri gecen süre.
-----	2	Kullanılmayan alan.
CIADDR	4	Client IP adresi.
YIADDR	4	'Your' (client) IP adresi, server tarafından BOOTREPLY ile gönderilir..
SIADDR	4	Server IP adresi, server tarafından BOOTREPLY ile gönderilir.
GIADDR	4	Gateway IP adresi.
CHADDR	16	Client hardware adresi, client tarafından BOOTREQUEST ile gönderilir.
SNAME	64	(secimlik) Server host adı.
FILE	128	(seçimlik) Boot dosya adı.
VEND	64	Vendor specific alan.

5.3 Client'ın İstek Göndermesi

Client istek gönderirken, IP Destination adresi olarak, Broadcast adresini (255.255.255.255) kullanır. Source port olarak, BOOTP Client, destination port olarak BOOTP Server portlarını kullanır. OP alanının değeri 1'dir. HTYPE ve HLEN alanları, bu uygulamanın yapıldığı sistem için 1 ve 6 değerleriyle doldurulur. XID alanı random bir transaction id değeri ile doldurulur. SECS alanı, client'ın ilk isteği gönderdiğinden beri geçen süreyi gösterir. CHADDR alanı client'ın hardware adresi ile doldurulur. VEND alanı 4 byte 'magic number' (hexadecimal olarak : 63, 82, 53, 63) ile doldurularak, kullanıcı adı gönderilir.

5.4 Server'in İstekleri Alması

Client IP adres alanı CIADDR, sıfır olduğu için, client IP adresini bilmiyor demektir. O halde server, client'ın hardware adresini veritabanında arayıp, eğer karşılığı yoksa paket iptal (discard) edilmeli. Eğer hardware adresi veritabanında mevcut ise, buna karşılık düşen IP adresi, Your IP Address YIADDR alanına yazılır. OP alanı BOOTREPLY değerini alır.
VEND alanı, client'ın gönserdiği 4Byte magic number değerine ek olarak, vendor tarafından tanımlanan resource bilgilerini de taşımalıdır. Bu bilgilerin ne olabileceği aşağıdaki listede tanımlanmıştır :

Etiket	Adı	Uzunluğu
0	Pad	0
1	Subnet Mask	4

2	Time Zone	4
3	Gateways	Değişken
4	Time Server	Değişken
5	Name Server	Değişken
6	Domain Server	Değişken
7	Log Server	Değişken
8	Quotes Server	Değişken
9	LPR Server	Değişken
10	Impress Server	Değişken
11	RLP Server	Değişken
12	Hostname	Değişken

UDP destination port, BOOTP Client seçilir. Eğer gatway adresi sıfır değilse (GIADDR), UDP destination port, BOOTP Server seçilirve paket GIADDR adresine gönderilir.

5.5 BOOTP Server Uygulaması

BOOTP server programının izlediği adımlar aşağıdaki gibi sıralanabilir :

1 - Veritabanı dosyası okunur ve parse edilerek gerekli alan bilgileri elde edilir. Bu alanların en önemlisi, hardware adresinin olduğu alandır. Çünkü daha sonra client'dan istek geldiğinde veritabanından ilgili bilgileri bulmak için anahtar olarak kullanılacaktır.

2 - Client'la haberleşmeyi sağlamak amacıyla soket işlemleri (socket, bind) yerine getirilir ve port adresi olarak 67 seçilir.

3 - Recvfrom() fonksiyonu ile client'in gönderdiği BOOTREQUEST paketi okunur.

4 - BOOTREQUEST paketinden , client hardware adres (CIADDR) alanı alınarak veri tabanında bu bilginin mevcudiyeti kontrol edilir. Eger mevcut ise ilgili alan bilgileriyle BOOTREPLY paketi hazırlanarak , soket üzerinden bu kez 68 numaralı port ve ilgili segment adresi kullanılarak sendto() fonksiyonu ile client'a gönderilir.

Aşağıda örnek BOOTREQUEST ve BOOTREPLY paketleri görülmektedir :

BOOTREQUEST

1,1,6,0,6c,2f,9c,35,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,c0,6c,25,98,35,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,63,82,53,63,c,4,45,53,49,4e,ff,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

BOOTREPLY

2,1,6,0,6c,2f,9c,35,0,0,0,0,0,0,0,c1,8c,4,f,c1,8c,4,1,0,0,0,0,0,c0,6c,25,98,35,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,63,82,53,63,1,4,ff,ff,ff,0,3,4,c1,8c,4,2,6,4,c1,8c,2,1,c,16,
68,65,69,64,69,2e,63,65,2e,79,69,6c,64,69,7a,2e,65,64,75,2e,74,72,ff,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

5.6 Örnek BOOTP Server Veritabanı

Veritabanı belirli kurallara göre oluşturularak, server programın bu bilgileri değerlendirmesi sağlanır. Veritabanında bir kayda ait alanlar ':' ile birbirlerinden ayrılırlar. Kısaltmalarla gösreilen alnların adları şu şekildedir : tc : template host, sm : subnet mask, gw : gateways, ds : domain name servers, ht : hardware type, dn : domain name, sg : segment address/name, ip : host IP address, ha : hardware address.

```
ComputerCenter.tc:\
  :sm=255.255.255.0:\
  :gw=193.140.4.2:\
  :ds=193.140.2.1:\
  :ht=ETHERNET_II:\
ComputerCenter:\
  :tc=ComputerCenter.tc:\
  :dn=ce.yildiz.edu.tr:\
  :sg=193.140.4.0:
carmen.ce.yildiz.edu.tr:\
  :tc=ComputerCenter.tc:\
  :ip=193.140.4.17:\
  :ha=080717050000:\
  :sg=ComputerCenter:
```

SONUÇLAR

Bu tez çalısmasıyla, kominikasyon konusundaki çalışmalarıma bir yenisini daha eklemiş oluyorum. Lisans projemde SNA ve SDLC protokolleri ve Data Link Layer konusunda çalışmıştım. Bu çalışmada ise daha üst seviyelerle ilgilendim. Novell NetWare ortamında her türlü altyapının sağlanmış olması, uygulama geliştirmeye yönelik, Software Development Kits ve NetWare C Interface araçlarının kullanılabilmesiye verimli bir çalışma gerçekleşmiştir. Ayrıca client - server uygulamaları tanımak ve bölümümüzde ihtiyaç duyulan TCP/IP protokol ailesinin üyelerinden File Transfer Protocol ve UDP/IP ailesinin üyelerinden Bootstrap Protocol için server uygulamalarını geliştirmek, test ederek kullanıma sunmak mümkün olmuştur.

Bu tez çalışması ile bölümümüzün Professional Novell Developer ünvanı altında geliştirdiği ürün yelpazesine iki ürün daha katılmıştır.

KAYNAKLAR

1 - Novell, NetWare NLM Library Reference Volume I - II, Edition 1.1

2 - Novell, NetWare NLM Development and Tools Overview, Edition 1.1

3 - Day, Koonts, Marshall, Novell's Guide to NetWare 4.0 NLM Programming, Novell Press, 1993

4 - Postel, Reynolds, RFC 959 File Transfer Protocol, October 1985

5 - Craft, Gilmore, RFC 951 Bootstrap Protocol, September 1985

6 - Prindeville, RFC 1048 Bootp Vendor Information Extensions, February 1988

ÖZGEÇMİŞ

Doğum Tarihi : 17.04.1966

Doğum Yeri : Siverek - Şanlıurfa

1977 - 1984 : Kadıköy Anadolu Lisesi

1984 - 1986 : YTÜ Meslek Yüksekokulu Bilgisayar Programcılığı Bölümü

1989 - 1993 : YTÜ Elektrik Elektronik Fakültesi, Bilgisayar Bilimleri ve Mühendisliği Bölümü

1993 - : YTÜ Fen Bilimleri Enstitüsü, Bilgisayar Bilimleri ve Mühendisliği Anabilim Dalı