BLGM 343 – DENEY 7*
UDP İLE VERİ İLETİŞİMİ

Amaçlar

1. UDP protokolünün öğrenilmesi
2. Ağ programlamada kullanacağımız sistem komutlarının öğrenilmesi
3. Ağ programlamanın kavranması

UDP ile veri alışverişi

UDP bağlantı sağlanmadan kullanılan bir protokoldür, yani, istemci sunucuya direkt olarak veriyi göndermeyi dener. Verinin sunucuya ulaşıp ulaşmadığının tespiti mümkün değildir. UDP veri alışverişinde kesin bir istemci/sunucu ayrımı yoktur. Ancak, veri transferini başlatan taraf istemci olarak adlandırılır. UDP ile veri iletişimi kurabilmek için ilk önce bir soket oluşturulmalı (**socket**), gelen bilgiler için adres yapısı oluşturulmalı (**sockaddr_in**) ve bu oluşturulan adres oluşturulmuş olan sokete bağlanmalıdır (**bind**). Bu işlemlerin ardından veri okunmaya çalışılabilir (**recvfrom**) veya verilen bir hedef adrese (**sockaddr_in**) gönderilebilir (**sendto**). İletişim tamamlandığı zaman açılan soket kapatılmalıdır (**close**).

socket()

soket() sistem çağrısına iki parametre ile istenilen soket tipi belirtilir. Internet Protocol için AF_INET, UDP bağlatısı için SOCK_DGRAM (datagram soketi) kullanılması gerekmektedir. Eğer döndürülen değer negatifse, bir hata çıktığını belirtmektedir. Eğer herhangi bir sorun yoksa bu sistem çağrısı soket tanımlayıcısı döndürülür. Örnek:

```
int soket = socket(AF_INET, SOCK_DGRAM, 0);
```

bind()

bind() sistem çağrısı verilen soket tanımlayıcısını verilen yerel adres ve kapı numarasına bağlar. Parametre olarak önce soket tanımlayıcısı, sonra yerel adres tanımlayıcısı (protokol, adres ve kapı

* BLGM 343 dersi için Güz 2013/2014 döneminde Gürcü Öz ve Cem Kalyoncu tarafından hazırlanmıştır

numarasını içerir) ve soket adres tanımlayısıcının boyutunu kabul eder. Soket adres tanımlayıcı olarak sockaddr_in (soket adresi, internet) kullanacağız. Adres veya kapı numarası bizim için önemli değilse **INADDR_ANY** değerini kullanabiliriz. Çağrı esnasında hata çıkarsa sonuç olarak -1 döndürülür. En sık karşılaşılan hata belirtilen kapının numarasının kullanılıyor olmasıdır. Eğer **close()** çağrısı kullanılmamışsa, bir program kapandıktan bir süre sonra o programın kullandığı kapı numarası serbest duruma düşer.

```
struct sockaddr_in sunucu = {};

sunucu.sin_family = AF_INET;
sunucu.sin_addr.s_addr=htonl(INADDR_ANY);
sunucu.sin_port=htons(32000);

bind(soket, (struct sockaddr *)&sunucu, sizeof(sockaddr_in));
```

sendto()

Gönderilmek istenilen bir veri, herhangi bir hedefe **sendto()** sistem çağrısıyla gönderilebilir. Bu çağrı hedef adres bilgisini almaktadır. Hedef adres bilgisi elle doldurulabileceği gibi **recvfrom()** sistem çağrısı tarafından da gönderilebilir. Aşağıda adresi ve kapı numarası bilinen bir bilgisayara veri gönderim örneği mevcuttur.

```
struct sockaddr_in sunucu={};

sunucu.sin_family = AF_INET;
sunucu.sin_addr.s_addr=inet_addr("127.0.0.1");
sunucu.sin_port=htons(32000);

char data[]="merhaba";
sendto(soket, data, strlen(data)+1, 0,
          (struct sockaddr *)&sunucu, sizeof(sunucu));
```

recvfrom()

recvfrom() sistem çağrısı verilen adres ve kapı numarasından veri alımını sağlar. Eğer bu bağlantı noktasında veri varsa, bu veri derhal programa iletilir, eğer veri yoksa sistem veri gelinceye kadar bekler. Veri alındığı zaman recvfrom() sistem çağrısı verilen adres bilgisini (**sockaddr_in**), göndericinin bilgileriyle doldurur. Böylece bu adres bilgisi kullanılarak, veriyi gönderen programa veri iletimi yapılabilir. Aşağıdaki örnek gelen veriyi ekrana yazdırıp, veriyi gönderen programa "Alındı" yazısını iletmektedir.

```
char mesaj[1024];
int mesaj
int istemci_uzunlugu;
struct sockaddr_in istemci={};

istemci_uzunlugu = sizeof(istemci);

mesaj_uzunlugu = recvfrom(soket, mesaj,1024, 0,
     (struct sockaddr *)&istemci, &istemci_uzunlugu);

sendto(soket, "Alındı",6, 0,
     (struct sockaddr *)&istemci, sizeof(istemci));
```

close()

Açılan bir soket sistemde sorun yaratmaması için düzgün bir şekilde kapatılmalıdır.

Deney

Bu deneyde basit bir sunucu/istemci sistemi yazmanız istenmektedir.

Bu sistemde sunucu sürekli olarak gelen verileri beklemelidir. Veri alan sunucu, mesajı ekrana yazıdırmalıdır. Bunun ardından veriyi değiştirmeden göndericiye geri göndermelidir. Sunucuyu CTRL+C ile kapatabilirsiniz.

İstemci çalıştığı zaman kullanıcıdan bir mesaj okumalı, ve bunu yerel makinada (ip numarası: 127.0.0.1) bulunan sunucuya aktarmalıdır. Daha sonra, sunucudan gelecek olan veriyi beklemeli, ve bu veriyi ekrana yazdırırarak çıkmalıdır. İstemciyi çalıştırmadan önce sunucuyu çalıştırmanız gerektiğini unutmayınız.

Ek çalışma: Sunucuya gelen SIGINT mesajını algılayarak, sunucunun soketi düzgün şekilde kapatarak çıkmasını sağlayınız. Bu durumdan emin olmak için ekrana sunucunun çıkıyor olduğuna dair bir mesaj yazdırınız.

Sorular

1. Bu deneyin amacı nedir?

2. UDP ve TCP soketleri arasındaki farklar nelerdir?

3. sockaddr_in veri yapısının içeriği nedir?

4. Bir IPv4 adresinin uzunluğu ne kadardır?

5. Noktalı olarak yazılmış bir IP adresi sockaddr_in veri yapısının bir elemanına eşitlenmeden önce tam sayı formuna nasıl çevirilir?

6. Bu programda kullanılan sistem çağrıları ve parametreleri nelerdir?

7. Eğer socket() sistem çağrısı başarılıysa döndürdüğü sonuç nedir?

8. Sunucu verinin geldiği istemcinin adresine nasıl ulaşabilir? Bu adres ne amaçla kullanılabilir?