INF 443
Dağıtık Sistemler ve Uygulamaları
Proje Tanımı

Serhan Daniş

10 Aralık 2014

İçindekiler

1 Giriş

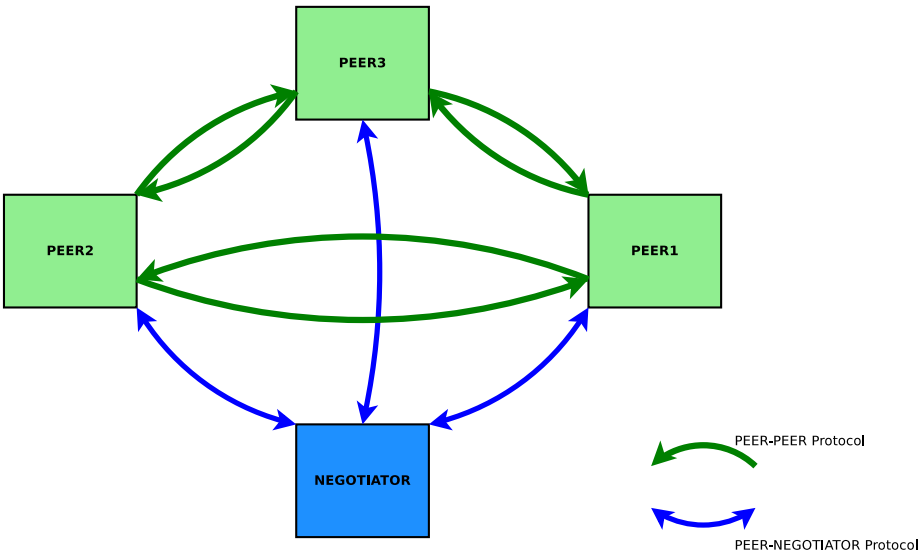
Projede peer-to-peer tabanlı bir dosya paylaşım platformu tasarlamanız ve oluşturmanız beklenmektedir.

Oluşturulacak sistem aşağıdaki yetkinliklerde olmalıdır:

- Bütün sistem iki çeşit yazılımdan oluşacaktır: `PEER` ve `NEGOTIATOR`
- İki farklı bağlantı şekli olacaktır: `PEER-PEER` ve `NEGOTIATOR-PEER`
- `PEER-PEER` bağlantısında, amacı dosya paylaşımı olan bir protokol bulunacak.
- `NEGOTIATOR-PEER` bağlantısının amacı ise `PEER` buluşturma olacaktır.

2 Eş sistem (Peer)

`PEER` ya da eş sistem, kullanıcıların sisteme dahil olabilmek için kullanacakları uç sistem olacaktır. Aşağıda tanımlanmaktadır:



Şekil 1: p2p dosya paylaşım sistemi

Her eş üzerinde hem sunucu hem de istemci tarafları bulunacaktır. Eş üzerinde çalışan sunucu (`PEER_SERVER`) tarafından başka eşlerden gelen sorgulara cevap verebiliyor olup, istemci (`PEER_CLIENT`) ise benzeri şekilde başka eşlere sorgu yapabilmelidir.

2.1 Dosyalama

- Her eşin kontrol ettiği bir paylaşım dizini olacaktır: (`SHAREDIR`).
- Bir eş diğer bir eş üzerindeki dosyaları ve dosya parçalarını sorgulayabilecektir.
- Gönderilmek istenen dosyalar parça parça gönderilecektir: `CHUNK`.
- Parça büyüklükleri bütün sistemde önceden belirlenmiş bir büyüklük olmalıdır: `CHUNKSIZE`.

2.2 Arabirim

Eş yazılımları kullanıcıya en az dosya aratma ve dosya indirme seçeneklerini sağlayacak bir görsel arabirime sahip olmalıdır. Bunun için daha önceki IRC sistemi ödevlerinde kullandığınız QT tabanı kullanılabilir.

3 Arabulucu sistem

Her eş, büyük sisteme dahil olabilmek için öncelikle arabulucu-sunucuya (`NEGOTIATOR`) bağlanacaktır. Arabulucu-sunucu, bağlanan eşe diğer eşlerin bağlantı bilgilerini iletecektir (`CONNECT_POINT`). Dolayısıyla her bağlanan eş, arabulucu-sunucuya kaydolup kendi bağlantı adresini ve portunu sağlamalıdır. Arabulucu-sunucusu belirli zaman aralıklarıyla (`UPDATE_INTERVAL`) eşlere kendisine bağlı olan eşlerin bağlantı adresi listesini iletmelidir veya her eş belirli zamanlarda bu bilgiyi istemelidir.

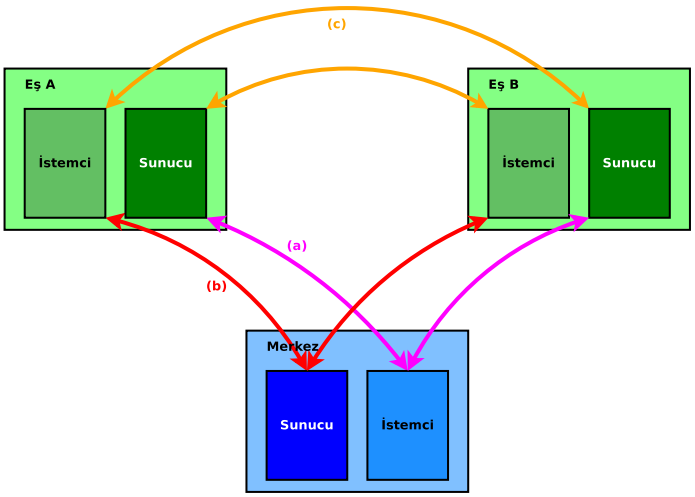
Arabulucu üzerinden dosya alışverişi yapılmayacaktır.

Yeni bir eş bağlandığında ve kendisini kaydettiğinde (`REGISTER`), arabulucu-sunucunun bir eş gibi o eşe bağlanmaya çalışıp `[PEER_IP, PEER_PORT]` ikisilinde bir problem olup olmadığını kontrol etmesi gerekir.

Arabulucu-sunucu belirli zaman aralıklarında kendi üzerindeki bağlantı adres listesini (`CONNECT_POINT_LIST`) güncelleme çalışması yapacaktır.

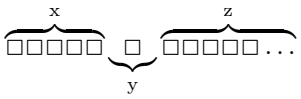
4 Protokol Mesajları

Tasarlanan sistemde toplamda üç bağlantı çeşidi bulunacaktır: (a) Arabulucu-istemci, eş-sunucu; (b) Eş-istemci, arabulucu-sunucu; (c) eş-istemci, eş-sunucu.



Şekil 2: p2p protokol çeşitleri

Protokol mesajları metin tabanlı olup, uzunlukları sabit değildir. En başında 5 ASCII karakterinden oluşan bir komut (x) bulunmakta, ardından gelen bir boşluk karakteri (y), komutları parametrelerden (z) ayırmaktadır. Parametreler her komut anlatılırken beraberinde açıklanacaktır. Her kare (□) bir karakteri belirtecek şekilde olmak üzere aşağıdaki şekildeki gibi olacaktır:



Aşağıdaki bölümlerde mesaj tipleri kısaca tanımlanmıştır. İstekler paragraf başı olası cevaplar da paragraf içinde yazılmıştır. Yanlış bir mesaj yapısında `CMDER` cevabı döndürülecektir.

4.1 Arabulucu-istemci ↔ Eş-sunucu (a)

- `HELLO`
  - `SALUT <type>`
- `CLOSE`
  - `BUBYE`

4.2 Eş-istemci ↔ Arabulucu-sunucu (b)

- Bölüm 4.1'dekileri olduğu gibi kullanır.
- `REGME <ip>:<port>`
  - `REGWA`
  - `REGOK <time>`
  - `REGER`
- `GETNL <num>`
  - `NLIST BEGIN`
    `<ip>:<port>:<time>`
    `<ip>:<port>:<time>`
    `...`
    `NLIST END`
  - `REGER`

4.3 Eş-istemci ↔ Eş-sunucu (c)

- Bölüm 4.2'dekileri olduğu gibi kullanır.
- `FINDF <filename>`
  - `NAMEY BEGIN`
    `<filename>:<md5sum>:<filesize>`
    `<filename>:<md5sum>:<filesize>`
    `...`
    `NAMEY END`

    - NAMEN <filename>
    - REGER

- FINDM <md5sum>
    - MSUMY <md5sum>
    - MSUMN <md5sum>
    - REGER

- FINDC <md5sum>:<num>
    - CHNKY <md5sum>:<num>
    - CHNKN <md5sum>:<num>
    - REGER

- GETCH <md5sum>:<num>
    - CHUNK <md5sum>:<num>:BEGIN
      <chunkdata>
      CHUNK <md5sum>:<num>:END
    - CHNKN <md5sum>:<num>
    - REGER

5 Protokol detayları

Sistemdeki bütün istemciler ve sunucular arasındaki protokol soru-cevap (senkron) şeklinde olacaktır. Sunucular, istemcileri sorgu yapmadığı sürece bilgi vermeyecektir. Dolayısıyla bu istemcilerin asenkron veri beklemesi gerekmemektedir.

5.1 Arabulucu-istemci ↔ Eş-sunucu (a)

Arabulucu-istemcinin işi sadece eşlerin sunucularını test etmek oluduğu için küçük bir kelime haznesine sahiptirler. Bağlantı yaptıktan sonra HELLO ve CLOSE bilirler. Asıl amaçları HELLO cevabı düzgün geldiğinde bağlantı yaptıkları eş-sunucusunun bilgilerini onaylamaktır.

Tanımadıkları bir cevap karşılığında CMDER döndüreceklerdir.

HELLO komutu parametre almaz, alması önemsenmez. Karşılığında eş-sunucusunun vermesi gereken cevap `SALUT` komutuyla başlar, bir boşluktan sonra kendisinin hangi tipte olduğunu belirtir. `<type>` değişkeni iki değer alabilir:

`<type>` $\in$ {`N, P`}

CLOSE komutu parametre almaz, alması önemsenmez. Karşılığında eş-sunucudan beklenen cevap `BUBYE` komutudur.

5.2 Eş-istemci $\leftrightarrow$ Arabulucu-sunucu (b)

Eş-istemcilerinin arabulucu sunucudan beklentileri diğer eşlerin bağlantı bilgileridir. Dolayısıyla protokol bu bilgileri almak üzere tasarlanmıştır. **Bu bölüm bir önceki bölümdeki protokol isteklerini ve cevaplarını da içerir**. Tekrar yazma gereği duyulmamıştır.

REGME <ip>:<port> (register me) komutu iki parametre alır: `<ip>` eş-sunucusunun çalıştığı IP adresi ve `<port>` aynı sunucunun port numarasıdır. Bu değerlerin geçerli IP adresi ve port numaraları olmaları gerekmektedir.

Eş-istemci bu bilgileri bağlandığı düğümlere gönderir. Düğümler bu bilgileri aldıkları gibi `REGWA` cevabı döndürür ve kaydetme işleminin devam ettiğini bildirirler. Aynı zamanda istemcileriyle verilen bilgiler üzerinden bağlantılar test edilir. Bu durumda düğümün `CONNECT_POINT_LIST`'ine ilgili girdi yapılıp karşısına `W` yazılır.

Test başarılı sonuçlanırsa, yani ters bağlantı yapılabiliyor ise kaydetme başarılıdır ve ilgili haneye `S` yazılır ve yanına kayıt zamanı `<time>` eklenir. Test başarısız ise `CONNECT_POINT`, `CONNECT_POINT_LIST`'ten silinir.

Eğer arabulucu-sunucu hemen cevap verebiliyor ve cevabı hayır ise cevap olarak parametresiz `REGER` döner.

Eğer arabulucu-sunucu bağlantı bilgisine sahipse cevap olarak `REGOK <time>` döner. Buradaki `<time>`, POSIX TIME olup, eş-istemcinin en son test edildiği zamanı ifade eder.

GETNL <num> (get node list) en fazla bir parametre alır. Parametre gönderilecek düğüm sayısını ifade eder. Parametre verilmediği durumlarda bütün düğüm bilgisi çekilmeye çalışılır.

Eğer kayıt işlemi başarılı `(S)` durumda değilse bu komuta cevap olarak `REGER` döndürülür.

Başarılı durumlarda cevap olarak `NLIST BEGIN` satırı gönderilir ve ardından her bir satır `<ip>:<port>:<time>` olacak şekilnde düğüm listesi gönderilir.

Buradaki değişkenler sırasıyla bağlantı düğümlerinin IP adresi, port numarası ve en son test edilme zamanlarını ifade eder. Her satır sonu satır sonu karakteriyle, `\n`, biter. Varsa, sayı (`<num>`) sonuna ulaşıncaya kadar satırlar gönderilmeye devam eder. En sonda `NLIST END` gönderilmesi beklenir. Bu gönderildiğinde listenin sonlandığı anlaşılır. Liste sonuna ulaşılmadan başka bir yapı gönderilirse istemci bağlantıyı kapatır.

5.3 Eş-istemci ↔ Eş-sunucu (c)

Eşler arasındaki bağlantı dosya temin etmeye yöneliktir. **Bu bölüm önceki iki bölümdeki protokol isteklerini ve cevaplarını da içerir**. Tekrar yazılmasına gerek duyulmamıştır.

`FINDF <filename>` (find file) tam olarak bir parametre alır. Parametre aranacak dosya adını ifade eder.

Bu fonksiyon kayıtlı olmayı gerektirir. Kayıt işlemi başarısız durumda ise cevap olarak `REGER` döndürülür.

Başarılı durumlarda cevap olarak `NAMEY BEGIN` satırı gönderilir ve ardından her bir satır `<filename>:<md5sum>:<filesize>` olacak şekilde dosya listeri gönderilir. Değişkenler sırasıyla (benzer) dosya ismi, dosyanın hash fonksiyonundan geçirilmiş parmak izi ve dosyanın boyutudur. Her satır sonu satır sonu karakteriyle, `\n`, biter. En sonda `NAMEY END` gönderilmesi beklenir. Bu gönderildiğinde listenin sonlandığı anlaşılır. Liste sonuna ulaşılmadan başka bir yapı gönderilirse istemci bağlantıyı kapatır.

Aranan dosya hiçbir şeye benzetilemediyse `NAMEN <filename>` cevabı verilir.

`FINDM <md5sum>` (find fingerprint) tam olarak bir parametre alır. Parametre dosya parmak izini ifade eder.

Bu fonksiyon kayıt olmayı gerektirir. Kayıt işlemi başarısız durumda ise cevap olarak `REGER` döndürülür.

Bu parmak izli bir dosya bulunuyorsa `MSUMY <md5sum>`, bulunmuyorsa `MSUMN <md5sum>` cevabı gönderilir.

`FINDC <md5sum>:<num>` (find chunk) tam olarak iki parametre alır. Parametreler sırasıyla dosya parmak izini ve dosya parçası numarasını ifade eder.

Bu fonksiyon kayıtlı olmayı gerektirir. Kayıt işlemi başarısız durumda ise cevap olarak `REGER` döndürülür.

Eş-sunucusuna bir dosya parçası mevcut mu diye sormaya yarar.

Dosya parçası varsa CHNKY <md5sum>:<num>, yoksa CHNKN <md5sum>:<num> cevapları gönderilir.

GETCH <md5sum>:<num> (get chunk) tam olarak iki parametre alır. Parametreler sırasıyla dosya parmak izini ve dosya parçası numarasını ifade eder.

Bu fonksiyon kayıtlı olmayı gerektirir. Kayıt işlemi başarısız durumda ise cevap olarak REGER döndürülür.

Eş-sunucusundan bir dosya parçasını çekmeye yarar.

Eğer dosya parçası mevcutsa CHUNK <md5sum>:<num>:BEGIN cevabı gönderilir. Sonuna satır sonu karakteri konur ve ardından boyutu önceden bilinen (CHUNKSIZE) <data> gönderilir. Gönderim sonunda bir satır sonu karakteri ardından CHUNK <md5sum>:<num>:END beklenir.

Dosya parçası yoksa CHNKN <md5sum>:<num> cevapları gönderilir.

6 Anahtar kelimeler

- NEGOTIATOR: Bir eşin diğer eş bilgilerini çekebileceği ve kendisini kaydedeceği sunucu.
- PEER: Kullanıcıların dosya paylaşım sistemine dahil olabilmesi için kullanacakları (eş) uç yazılım.
- PEER_CLIENT: PEER'ın istek gönderen tarafı.
- PEER_NEGOTIATOR: PEER'ın bağlantı dinleyici tarafı.
- PEER_IP: Eşin başka bir eşten bağlantı bekleyeceği IP adresi.
- PEER_PORT: Eşin başka bir eşten bağlantı bekleyeceği port numarası.
- CONNECT_POINT: Bir eşin diğer eşlerin bağlanabilmesi için NEGOTIATOR'e göndereceği ve dinliyor olacağı [PEER_IP, PEER_PORT] ikilisi.
- CONNECT_POINT_LIST: Bir eş veya sunucunun bildiği CONNECT_POINT listesi.
- SHAREDIR: PEER sistemindeki paylaşılan dosyaların bulunduğu dizin.
- UPDATE_INTERVAL: CONNECT_POINT_LIST'in güncellenme aralığı.
- CHUNK: Dosya parçası.
- CHUNKSIZE: Dosya parçası boyutu.
- REGISTER: Eşin sunucuya bağlanıp kendisini kaydetmesi.
- FILE: Paylaşılan dosya.

7 Problemler ve çeşitli senaryolar

7.1 Sisteme bağlanma

Kullanıcının öncelikle sisteme giriş yapması gerekiyor. Bunun için kendindeki eş programı açıyor ve bildiği arabulucu-sunucusu adresine bağlanıyor. Kullanıcı başarıyla bağlanabilirse kendi bağlantı bilgisini sunucuya aktarıyor. Bu bağlantı bilgisine göre arabulucu ters bir test yapacaktır. Bağlantı problemsiz kurulur ve karşı tarafında bir eş-sunucusunun çalıştığından emin olur. Üzerindeki `CONNECT_POINT_LIST`'teki ilgili bağlantı alanına onay verip bağlantı kuran eşi kaydetmiş olur.

7.2 Bağlantı listesi isteme

Kullanıcı eşinin arabulucu-sunucuya bağlantı kurmasının arkasında diğer eş sistemlerin bağlantı adreslerini alma ihtiyacı yatıyor. Dolayısıyla arabulucu-sunucuya ilk bağlantıyı kurduğu zaman, arabulucu-sunucudan başka bağlı eşlerin bağlantı bilgilerini istiyor. Bu bağlantı isteme işlemi eş programında arkada devamlı çalışan bir thread altında yapılabilir. Belirli aralıklarda kullanıcı eş programı arabulucu-sunucuya ve diğer eşlere bağlanıp bu bilgiyi güncelleyebilir.

7.3 Dosya sorgulama

Kullanıcımız bir dosya indirmek istesin. İndirilecek dosyamızın adı da `DOSYA.avi` olsun. Bu durumda `DOSYA.avi` dosyası ismine benzerlik gösteren dosyaların listesine ihtiyaç olacaktır. Bu listeyi de her bir bağlı eşe sorarak elde edebiliriz. Her eşe tek tek bu dosya ismini sorgulatan bir komutla liste göndermesi istenecektir. Eşler de cevap olarak `<filename>:<md5sum>:<filesize>` şeklinde satırlardan oluşan bir liste döndürecektir.

7.4 Dosya indirme ve parça kontrolü

Gelen liste içinde kullanıcı hangi dosyayı indirmek istiyorsa onun parmak izini, (`FILEMD5`) kullanacaktır. Kendi bilgisayarının paylaşım dizininde bu isimde bir dosya açıp parçaları bunun içine yazacaktır. Eşin aynı zamanda parça kontrolü de yapması gerekmektedir. Bunun için aynı dizinde `.chunk` uzantılı bir dosya içinde parça bilgisini tutabilir veya daha kolay oluyorsa paylaşım dizininden farklı bir dizin içinde bu bilgi dosya adı altında tutulabilir.

7.5 Dosya parçası sorgulama

Dosya indirme ihtiyacı duyduğunda eş program indireceği parçaları kendi belirleyecektir. Belirli parçayı bağlı olduğu eşlere soracaktır. Varsa parçayı isteyecektir. Parça istenmesi durumunda karşı eş parçayı gönderecektir. Kullanıcı eş programı gelen parçayı dosyanın doğru yerine yazacaktır.

7.6 Paylaşım dizini düzenleme

Kullanıcı eş sistemi çalışmaya başladıktan sonra kendisindeki paylaştığı dosyaları, md5sumlarını ve dosya parçalarının durumunu biliyor hale gelmelidir. Mesela `DOSYA.mp3` diye bir dosya paylaşım dizininde bulunuyor olsun. Eğer bu dosya henüz tamamlanmamış ise dosya dizinde `.<md5sum>` gibi bir dosya şeklinde durur. Bu durumda nokta ile başlayan dosyalar dışarıya paylaşılmamalıdır.

7.7 Parçalı dosya problemi

Dosyalar bütün halinde tutulacağı, ama parça parça geleceği için parçaları ayrı bir yerde tutmak yerine dosyaya yazmak en kolayı olacaktır. Ancak bu durumda dosya tamamlanmadan tekrar paylaşıma açılması söz konusu olabilir. Bu olasılığı engellemek için, dosya tamamlanana kadar dosyayla ilgili bilgileri tutacak bir bilgi dosyası düzenlenebilir. Bu dosya altında da parçaların durumu tutulabilir. Eğer bu bilgi varsa o zaman program dosyanın henüz tamamlanmadığını anlayabilir. Böyle bir dosya `.<md5sum>.chunk` şeklinde olabilir. İçinde de sırasıyla dosya parça listesi durur, yeni bir parça dosyaya yazılınca listeden o parça çıkarılır. Dosyanın tamanlanması bu dosyada yeni bir girdi olmaması durumunda anlaşılabilir.

7.8 Farklı eşlerde farklı isimde aynı dosyalar

Dosya isimleri dosyaların isimlerinde ve isim aranırken kullanılacaktır. İndirme senaryosu içinde dosyanın parmak izi (md5sum) kullanılacaktır. Bu parmak izi dosyanın tamamı aynıysa aynı olur. Tek bitlik bir değişiklikte parmak izi de farklı olacaktır. Dosya isteme parmak izi üzerinden yapılırsa aynı dosya farklı eşlerden parça parça alınabilir.

7.9 Bağlantı listesi güncellemeleri

Bağlantı listesi, `CONNECT_POINT_LIST`, içinde eş-sunucularının IP, port, kayıt zamanı ve kendi açısından durumlarını tutacaktır. Bu liste diğer düğümler kullanılarak güncellenebilir. Ayrıca her düğüm belirli bir zaman sonra kendi bağlantı listesindeki bağlantıları kontrol edip, bağlantı listesini güncelleyecektir.

8 Ödev - P2P

Bu ödevde arabulucu sistem ve eş sistem yazılımlarını geliştireceksiniz. Ayrıca aşağıdaki örnek belgeler altındaki RFC1 benzeri bir belge hazırlamanız gerekiyor.

8.1 Örnek belgeler

- RFC1: SWECHAT Application System-Wide Requirements Specification –v0.2
- RFC2 : Technical Specifications for SWE544 Chat Project Protocol, SWECHAT Protocol –v0.3

8.2 Teslim edilecekler

1. Son teslim zamanı: 22.12.2014 - 00:00 (TSİ)
2. Proje gösterimi: 26.12.2014 - 11:00 (TSİ)
3. Projeleri uni.gsu.edu.tr'deki ders sayfasında ilgili alana gönderilecek.
4. Bir `<soyad>_proje.zip` isimli sıkıştırılmış dosya içinde en az 3 tane dosya olacak.
5. Projeyi rfc gibi raporlayacağınız bir pdf dosyası: "`rfc_proje.pdf`".
6. Sunucu ve istemci için birer python dosyası.

Not: Laboratuvar kitapçığında yazılı "Ödev Kuralları" (Bölüm 0.4) projede de aynen geçerlidir.