DAĞITIK SİSTEMLER VE UYGULAMALARI
GRUP 3
PROJE RAPORU

M. Yavuz Bahçeci	10401567
Mert Yazan	13401648
Beyza Cankuş	14401672
Ayşenur Kılıç	14401673
Emre Toraman	16401733

İÇİNDEKİLER

1. Protokol Tasarımının Yapılması
2. Aracı Eşin Çalıştırılması
   2.1. Temel Sınıfların Oluşturulması
      2.1.1. Logların Tutulması
      2.1.2. Sunucu Tarafı
      2.1.3. İstemci Tarafı
   2.2. Protokol Mesajlarına Göre Verilerin Ayrıştırılması
      2.2.1. Mesaj Ayrıştırıcısı
      2.2.2. Sunucu Ayrıştırıcısı
      2.2.3. İstemci Ayrıştırıcısı
   2.3. Yararlanılacak Fonksiyonların Oluşturulması
3. Yayıncı Eşin Çalıştırılması
4. Grafik Arabirimin Oluşturulması

1. Protokol Tasarımının Yapılması

Protokol tasarımında aracı ve yayıncı eşlerin ihtiyaçlarına göre gerekli protokol mesajları ve bu mesajlara gelecek cevapların anlaşılabilirlik için 4 karakterli olması gerektiğine karar verilmiştir. Tüm mesajların parametreleri ve parametrelerin boyutları, açıklamaları ile beraber Şekil 1'de gösterilmektedir.

Protokol	Parametre1	Parametre2	Parametre3	Parametre4	Parametre5	Response	Parametre6	Açıklama
HELO	UUID(48bit)	IP(32bit)	Port(16bit)	Type(1bit)	Name(59bytes)	WAIT	UUID	Kayit Kontrol
						REJN		Nick kabul edilmedi
						WLCM	UUID	Kayitli kullanici onayi
SUID						AUID	UUID	Baglanti Kontrol
LIST						LSTO	List(N*312bytes)	Liste sorgulama
						AUTH		Peer kayitli degil
PUBR	Pub Key(256bytes)					PUBO	Pub Key	Pub Key Degisimi
						AUTH		Peer kayitli degil
PUBC	txt(54bytes)	Signedtxt(54bytes)				PUBV		Public Key kontrolu (5 karakterli string)
						PUBD		Public key reddedildi.
						AUTH		Peer kayitli degil
MBLR	N(28bytes)					MBLO	Blist(N*245bytes)	Mikro blog istegi gönderme
						AUTH		Peer kayitli degil
						BLCK		Peer bloklanmis
SUBS						SUBO		Abonelik istegi gonderme
						AUTH		Peer kayitli degil
						BLCK		Peer bloklanmis
UNSB						UNSO		Abonelikten cikma
						AUTH		Peer kayitli degil
						BLCK		Peer bloklanmis
TWTS	text(245bytes)					TWTO		yayin gönderme
						AUTH		Peer kayitli degil
						BLCK		Peer bloklanmis
BANN								Peer'i bloklama
						AUTH		Peer kayitli degil
						BLCK		Peer bloklanmis
UBAN						UBAO		Blok kaldirma
						AUTH		Peer kayitli degil
						BLCK		Peer bloklanmis
PRIV	mesaj(245bytes)					PRIO		Ozel mesaj gönderme
						AUTH		Peer kayitli degil
						BLCK		Peer bloklanmis
						ERRP		parametre hatasi
						ERRG		genel hata

(Şekil 1)

"HELO" protokol mesajı her bağlantı öncesinde gönderilmesi gereken, eşlerin birbirlerini tanımasını sağlayan mesajdır. Bu mesajı gönderen eş bağlantı kurduğu tarafa kendi kimliğini, adresini, eş tipini ve ismini gönderir. Kimliğini "UUID" parametresi, adresini IP ve Port parametreleri ile belirtir. Burada gönderdiği "UUID" verisi eşin bilgisayarının MAC adresine göre oluşturulan 48 bitlik "integer" tipindedir. 10 karakterden oluşacak isim verisinin boyutunun 59 bytes olmasının nedeni, Python dilinin "string" tipleri için büyük boyutta yer ayırmasından kaynaklanır. Bu mesaja dönecek cevap, eş daha önceden kayıt olduysa "WLCM" mesajı, ilk defa bağlantı oluşuyorsa "WAIT" mesajı ardından "SUID" mesajının gönderilip cevabının alınmasıyla kayıt tamamlandığında "WLCM" mesajı olacaktır. Sistemde kullanıcı adları benzersiz olmalıdır. Bu

yüzden eğer kullanıcı adı daha önceden başka bir eş tarafından alınmış ise “REJN” cevabı döndürülecektir.
“SUID” protokol mesajı, “HELO” mesajıyla yeni bir bağlantı isteği gönderen eşe yollanacak bir bağlantı kontrol mesajıdır. Mesajın parametresi yoktur. “AUID” cevabı ile karşı tarafın “UUID” bilgisi döner. Böylece doğru bağlantı kurulduğu anlaşılır.
“LIST” Protokol mesajı, belirli bir dakikada bir otomatik olarak eşin listesinde kayıtlı olan eşlere gönderilerek tüm eşlerdeki kullanıcı listelerinin güncel tutulmasını sağlayan mesajdır. İsteği gönderen eş, karşı eşin kişi listesinde kayıtlı değilse “AUTH” hata mesajı ile karşılaşırken, kayıtlı olduğu durumda ise “LSTO” onaylama mesajıyla beraber karşı eşin listesini parametre olarak alır.
“PUBR” mesajı karşı eşten açık anahtarını isteme mesajıdır. Bu mesajı yollayan eş kendi açık anahtarını da göndererek anahtar değiş tokuşunu sağlamış olur. Ancak henüz kayıt olmamış bir eş “AUTH” hata mesajıyla karşılaşacaktır.
“PUBC” mesajı, eşlerin aldıkları açık anahtarların doğruluğunu kontrol etmeleri için gereken bir mesajdır. Alınan açık anahtar ile 5 karakterden oluşan bir metin şifrelenir ve bu metin orijinal hali ile beraber karşı eşe yollanır. Karşı eş kendi kapalı anahtarı ile şifrelenen mesajı çözüp orijinal metinle aynı olduğunu anlarsa “PUBV” mesajını göndererek anahtarı doğrular. Eğer aynı metine ulaşamaz ise “PUBD” hatası, eş kayıtlı değil ise “AUTH” hatası gönderir.
Tabloda verilen diğer mesajlarda ayrıca bloklama durumu için de hata mesajları yer alır. Mikro blog isteği, özel mesaj gönderme gibi durumlarda isteği yollanan eşin bloklanmış kullanıcılar listesinde olup olmadığı kontrol edilerek işlem yapılır.
Tablonun son iki satırında ise tüm bu mesajları kapsayacak hata durumları tanımlanmıştır. Eksik parametreli gönderilen mesajlar için “ERRP” hatası, genel hata durumları için ise “ERRG” hatası gönderilir.

2. Aracı Eşin Çalıştırılması

2.1. Temel Sınıfların Oluşturulması

Aracı eş için genel olarak sunucu ve istemci adında iki tip Thread bulunmaktadır. Aracı programı başlatıldığında ana istemci thread ve ana sunucu için bir soket yaratılır. Bu soket sürekli dinler durumdadır. Gelen bağlantı isteklerini kabul eder ve her bağlantı için ayrı sunucu threadleri yaratır. Başlangıçta yaratılan istemci de okuma ve yazma istemci threadlerini yaratmadan sorumludur.

2.1.1. Logların Tutulması

Yeni bir nesne yaratılması, bağlantı kurulması, hata meydana gelmesi gibi durumların bilgisini kaydetmek üzere “loggerThread” sınıfı oluşturulmuştur. Bu sınıfın kodları ayrı bir dosyada yer alır ve kullanılacağı diğer proje dosyalarına eklenerek çalıştırabilir durumdadır. Thread’ler arası iletişim log kuyruğu ile sağlanır. Gerekli yerlerde log mesajları oluşturulur ve bu kuyruğa eklenerek “loggerThread” tarafından okunur hale gelir. Ardından bu Thread, mesajların başına zaman bilgisi ekleyerek oluşturulan “log.txt ”dosyasına kaydeder. Böylece programın içinde her seferinde zaman fonksiyonları yazılmasına gerek kalmamıştır.
Log dosyasının okunabilirliğini sağlamak böylece hataların tespitini kolaylaştırmak amacıyla programda oluşacak bütün nesnelere ayrı bir isim verilmiştir. Bu isimler dosya türlerinin yanına artan sayılar ekleyerek oluşturulmuştur.
Yayıncı eşler tarafından gerçekleşecek işlemlerin durumları için de log tutulmuştur. (Abonelik, mikro blog isteği, mesaj gönderimleri gibi)

2.1.2. Sunucu Tarafı

Aracı programının çalıştırılmasıyla tüm bağlantı isteklerini dinlemeye başlayan sokete herhangi bir bağlantı isteği gelmesiyle o isteğe özel bir “ServerThread” yaratılır. Gelen bağlantının adresi ile oluşan yeni soket nesnesi bu server thread nesnesine parametre olarak gönderilir. Ana sunucu açılacak sunucu threadler ile “ServerReaderQueue” kuyruğu sayesinde haberleşir. Gelen bağlantı sonrası oluşan tuple yapısı bu kuyruğa yazılır ve sunucu threadleri tarafından okunur. Böylece soket bilgileri bir kenarda tutularak cevap döndürülebilinir hale getirilmiş olur. Kuyruktan okunacak mesajlar “inc_parser_server” fonksiyonu yardımı ile ayrıştırılır ve incelenir. Bu bölüm sunucu ayrıştırıcısı kısmında detaylı anlatılacaktır. Gerekli işlemlerden geçen mesajın cevabı burada elimizde bilgileri olan sokete gönderilir.

2.1.3. İstemci Tarafı

Program başlangıcında yaratılan ana “clientThread” devamlı “clientSenderQueue” isimli kuyruğu dinler ve kuyruğa herhangi bir şey yazıldığında yazılanı sözlük yapısında alıp yeni bir “clientSender” thread’i oluşturur. Bu thread’in görevi kuyruk vasıtasıyla kendisine ulaştırılan sözlük yapısını çözümleyip içerisinden çıkan adrese yeni bir soket oluşturarak bağlanıp yine içerisinden çıkan mesajı bu soketten yollamaktır. Ardından bu

mesaja gelecek cevabı dinlemek için bir “clientReader” thread yaratılır. Yaratılırken soket bilgisi de “clientReaderQueue” isimli kuyruğa yazılır. Böylece “clientReader” çalıştığında bu kuyruktan soket bilgisini alarak gelecek cevabı alır ve mesajın türüne göre işlemlerini gerçekleştirir. Örneğin “AUID” mesajı gelirse, bu mesajı sunucuya iletmek için “clientToServerQueue” kuyruğuna yazar ve bir “clientToServer” thread’i yaratır. Bu thread ise mesajla gelen “UUID”’nin sözlüktekiyle aynı olup olmadığını kontrol eder ve aynıysa “WLCM” mesajını “AUID” mesajıyla gelen bağlantı bilgilerini kullanarak soketten iletir. Aynı değil ise hata mesajı gönderip soketi kapatır. Bu kısım Sunucu Ayrıştırıcısı kısmında anlatılan “SUID” mesajlarının devamıdır.
Aracı eşler arası belirli aralıklarla yapılacak liste sorgulama işlemleri için program çalıştırıldığında “clientDictThread” isimli bir thread yaratılır. Bu thread dakikada bir listesindeki tüm eşlere liste sorgulama yapıp bu bilgileri bir sözlük yapısına yazar. Eşlerle “HELO” mesajları ile bağlantı başlatma işi “list_control” fonksiyonu ile yapılır. Buna cevap olarak gelecek “WLCM” mesajları ise “LIST” isteklerini tetikler. Böylece kişi sözlükleri güncellenmiş olur.

2.2. Protokol Mesajlarına Göre Verilerin Ayrıştırılması

2.2.1. Mesaj Ayrıştırıcısı

Mesaj ayrıştırıcısı yani “parser” fonksiyonu mesajları aralarındaki boşluklarına göre parçalayıp mesaj türüne göre sözlük yapıları oluşturan bir fonksiyondur. Ayrıştırılma sonrası ilk kısım “command” olarak belirlenir ve buna göre sözlük yapısının parçaları belirlenir. Sözlük yapılarına mesaj ve parametreleri dışında alternatif cevap mesajları da eklenir. Daha sonraki kısımlardaki kontrollere göre cevap belirlenecektir. Bu fonksiyonun dönüş değeri de oluşacak bu sözlük yapılarıdır.

2.2.2. Sunucu Ayrıştırıcısı

Sunucu ayrıştırıcısına yani “inc_parser_server” fonksiyonuna gelen mesaj ilk olarak “parser” mesaj ayrıştırıcısına yollanır ve dönüş değeri olan sözlük yapısı alınır. Sözlük yapısındaki “cmd” anahtarı ile protokol mesajlarının isimlerine göre ne yapılacağına karar verilir.
Örneğin gelen mesaj “HELO” mesajı ise sözlük yapısındaki “UUID”nin sunucunun kayıtlı kişiler sözlüğünde bulunup bulunmadığına bakılır. Bulunuyorsa “WLCM” cevabı yanına “UUID” eklenerek sunucuya bağlantı

başlatma soketinden gönderilmek üzere yollanır ve ardından kişinin ip, port bilgisi ve en son aktif olma durumu sunucunun kişiler sözlüğünde güncellenir. Bulunmuyorsa ise “WAIT” cevabı aynı şekilde soketten gönderilmek üzere sunucuya iletilir. Ayrıca kayıt işlemine devam etmek için “SUID” mesajı oluşturup “clientSenderQueue”ya yazar.
Gelen mesajın “WLCM”, “LIST”, “SUID” olduğu durumlarda da cevap oluşturulup sunucuya döndürülür.
Yukarıdaki durumlar “parser” sonucu durumun “OK” olarak döndüğü durumlardır. Bunun dışındaki durumlarda sokete hata mesajı yazılarak gönderilir ve soket kapatılır.

2.2.3. İstemci Ayrıştırıcısı

İstemciye gelen ve istemcinin karşıya yolladığı, mesaj ayrıştırıcısı ile sözlük yapısına çevrilmiş mesaj içeriklerinin sonuçlarını sisteme kaydetmek ya da gerekli log mesajlarının tutulması işlerini gerçekleştirmek amacıyla “out_parser_client” ve “inc_parser_client” isimlerinde iki fonksiyon yazılmıştır. Örneğin gönderilen bir “PUBR” mesajına cevap olarak “PUBO” onayı gelirse “inc_parser_client” fonksiyonunda, gelen bu mesajın içeriğindeki “UUID”’ye göre “pubkey_dict” sözlüğüne o “UUID”nin gösterdiği bir alan oluşturulup yine mesajın içeriğinden gelecek açık anahtar bu alana yazılır. Ardından sonraki oturumlarda kullanılmak üzere açık anahtarların tutulduğu bir metin dosyasına yazılır. Böylece gelen açık anahtarın kayıt işlemi gerçekleşir.

2.3. Yararlanılacak Fonksiyonların Oluşturulması

“util_functions” isimli script içerisinde proje boyunca işe yarayacak genel amaçlı fonksiyonlar oluşturulmuştur. Asimetrik şifreleme kullanılan mesajlaşmalar için açık anahtar ve kapalı anahtar yaratma fonksiyonları yayıncı eş için yaratılmıştır. Anahtarlar yaratılıp bir dosyaya yazılır ve gerekli durumlarda oradan çekilip kullanılabilir. Mesajların şifrelenmesi ve çözülmesi ile ilgilenen fonksiyonlarda da bu dosyada yer alır.
Programın çalışması boyunca birçok sözlük oluşturulur ve çeşitli eklemeler sonucu sözlükler değişebilir. Bu değişikliklerin sonraki oturumlarda kullanılabilmesi için dosyaya sözlük yapısının yazılması ve bu dosyalardan okuma yapılmasını sağlayan “appendToDictionaryFile” ve “readFromDictionaryFile” isimlerinde iki fonksiyon kullanılır. Ayrıca liste sorgulama istekleri sonucu gelecek yeni kişi bilgilerini de mevcut kişiler

sözlüğüne ekleyebilmek adına “mergeTwoDict” isimli fonksiyon oluşturulmuştur.

3. Yayıncı Eşin Çalıştırılması

Yayıncı eş aracı eşin yapacağı tüm işleri yapabilecek bir eştir. Bu sebeple, daha önce oluşturulan aracı eşin içeriği “blogger” isimli yeni bir dosyaya kopyalanmış ve üzerine ek fonksiyonlar yazılmıştır. Mesaj gönderme, engellendirme bildirimi public key isteği gibi mesajlar için ayrıştırıcılara eklemeler yapılmıştır.

4. Grafik Arabirimin Oluşturulması

Grafik tasarımın düzenlenmesi için PyQt5 Designer kullanılmıştır. Önyüz main fonksiyonu çalıştığı anda bağlantı işlemlerini otomatik olarak gerçekleştirmektedir. Ancak connect butonuna bastıktan sonra çalışan “connect_to_twitter” fonksiyonu kişinin nickname’ini referans olarak alır ve onun için bağlantı işlemlerini gerçekleştirmek üzere “out_parser_client”’a uygun mesajı gönderir.
“subscribe”, “share_context”, “unblock_user”,”refresh_feed”,”send_message” fonksiyonları da yine kullanıcı girişi yapıldıktan sonra çalışabilmekte olup parser’larına doğru mesajı gönderdikten sonra gereken response’u karşıdan alıp içeride oluşmuş listwidgetları o alınan bilgilere göre güncellemektedir. Sayfa güncelleme işleminin sürekli olması için bu görev dagitik_twitter_main.py ‘ın içindeki bir thread ‘ e verilmiştir.