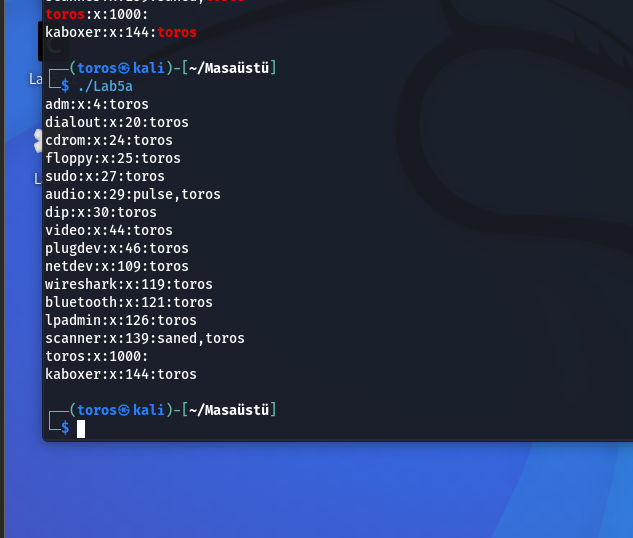


metin, ekran görüntüsü, yazılım, tasarım içeren bir resim

Açıklama otomatik olarak oluşturuldu

İki çıktı arasında belirgin bir fark gözlemlenmiyor. Her ikisi de "toros" kelimesini içeren grupları gösteriyor ve benzer sıralama ve grup düzenini sergiliyor. İki çıktı da aynı grupları içeriyor ve görünüşe göre aynı sıralamayı koruyor. "toros" kelimesini içeren gruplar, program çıktısı ile getent group | grep toros komutunun çıktısı arasında uyumlu bir şekilde yer alıyor.



Sistemdeki bir grup genellikle bir grup adını, grubun numarasını (gid), ve o gruba üye olan kullanıcıları içerir. Örneğin, "root" grup adıyla, "x" grup numarasıyla ve "root" kullanıcısının üye olduğu bir grup bu çıktıda bulunmaktadır. Gruplar, sistemdeki kullanıcıları düzenlemek ve belirli izinlere sahip kullanıcıları gruplandırmak için kullanılır.

Kali kelimesini içeren gruplar genellikle "toros" kullanıcısının bu gruplara üye olduğunu gösterir. Örneğin, "toros" kullanıcısı "audio" grubuna üyedir. Bu, "toros" kullanıcısının sesle ilgili işlemleri gerçekleştirebilmesi için gerekli izinlere sahip olduğunu gösterir.

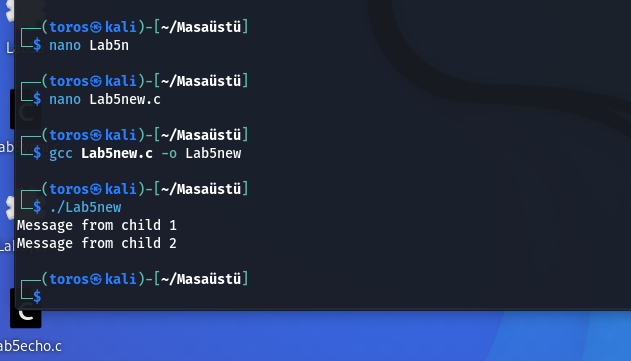
Bu çıktı, kullanıcıların sistemdeki grup üyeliklerini anlamak için kullanılır. Kullanıcılar, birçok farklı gruba üye olabilir ve bu gruplar, kullanıcının sahip olduğu izinleri ve yetenekleri belirler. Örneğin, belirli bir gruba üye olmak, o gruptaki dosyalara ve kaynaklara erişim yetkisi anlamına gelir.

Özel isimlendirilmiş gruplar, örneğin "\_gvm", "\_gophish", "mosquitto" gibi, genellikle belirli uygulamalar veya hizmetlerle ilişkilidir. Örneğin, "\_gvm" grupları, Güvenlik Açığı Yönetimi (GVM) yazılımıyla ilgili olabilir. Bu tür gruplar, belirli bir hizmet veya uygulamanın kullanıcılarını yönetmek ve bu kullanıcılara özel izinleri sağlamak için oluşturulmuştur.

"getent" komutu, sistemdeki grup bilgilerini getirir. "grep" komutu ise bu grupları filtreler ve sadece "toros" kelimesini içeren grupları gösterir. Bu, sistem yöneticilerinin veya kullanıcıların belirli bir kelimeyi içeren grupları hızlıca bulmalarına yardımcı olur.

Bu bilgi çıktısı, sistemdeki kullanıcıların hangi gruplara üye olduğunu anlamak ve sistemdeki grupları denetlemek amacıyla kullanılabilir. Gruplar, dosya ve dizinlere erişim izinlerini yönetmek, ağ paylaşımlarını düzenlemek ve diğer sistem kaynaklarına erişimi kontrol etmek için önemli bir rol oynar. Bu bilgiler, genel sistem güvenliği ve kullanıcı yönetimi açısından kritik bir öneme sahiptir. Gruplar, belirli yetkilere sahip kullanıcıları bir araya getirerek kaynaklara güvenli ve organize bir şekilde erişmelerini sağlar. Bu nedenle, grup üyelikleri ve grup yapıları, sistem yöneticileri için önemli bir inceleme ve düzenleme alanıdır.

Kod başarıyla çalıştı ve ilk süreç belirtilen kelimeleri bir pipe yazdı. İkinci süreç ise bu kelimeleri okuyarak ekrana yazdırdı. İlk süreç tarafından yazılan kelimeler, ikinci süreç tarafından alınıp ekrana basıldı. Bu pipe iletişimi, süreçler arasında veri iletimini sağlar, paralel veya dağıtık hesaplamalarda kullanılır.



İlk iki çocuk süreci, bir pipe sırasıyla "Mesaj 1 çocuktan." ve "Mesaj 2 çocuktan." mesajlarını yazdı. Üçüncü çocuk süreci ise bu mesajları pipedan okuyarak ekrana yazdırdı. Gönderilen mesajların arasında herhangi bir bölünme olmadığı gözlemlendi, yani mesajlar birbiri ardına ve tamamen aktarıldı.

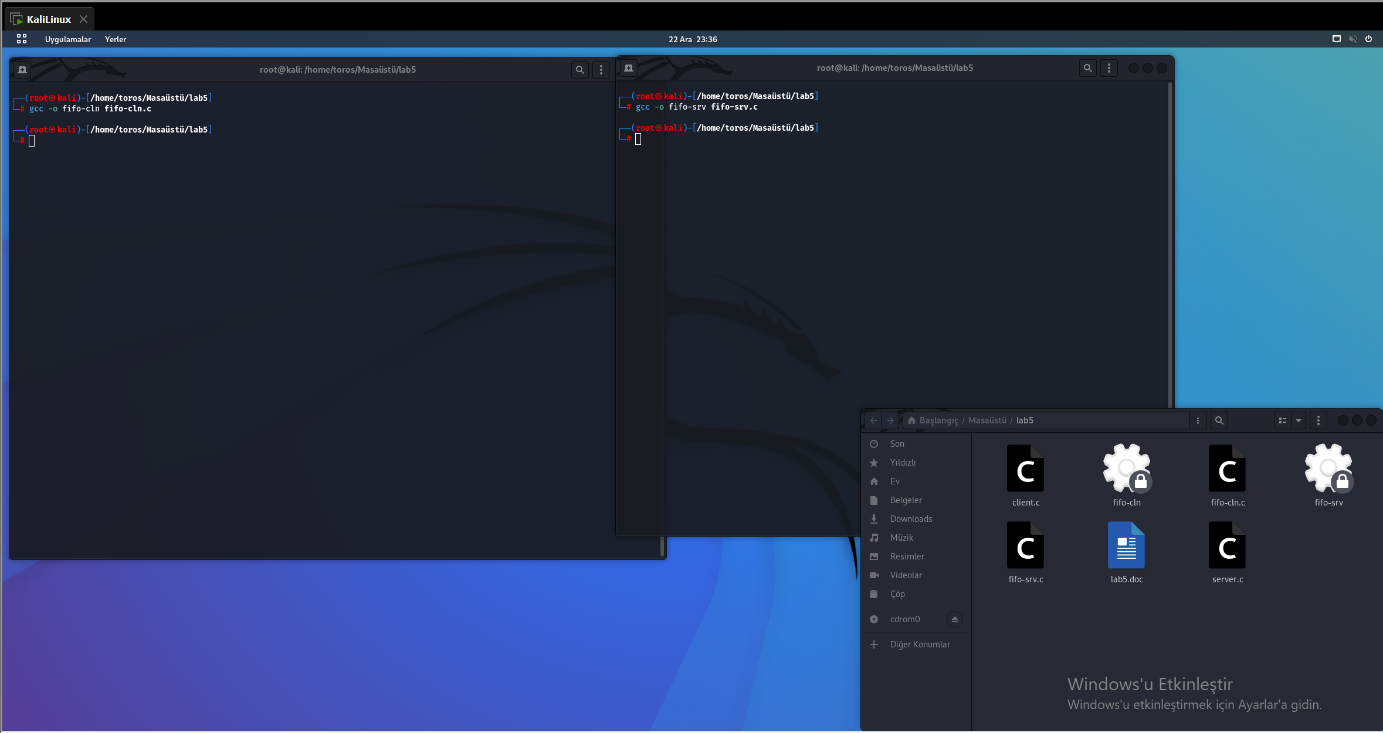
Bu örnek, pipe kullanarak çocuk süreçler arasında veri paylaşımını gösteriyor. Her çocuk sürecin bağımsız bir şekilde çalıştığı ve pipe'ın verileri doğru bir şekilde aktardığı görülüyor. Bu tür bir iletişim mekanizması, süreçler arasında veri iletimi sağlamak ve koordineli çalışmalarını sağlamak için sıkça kullanılır. Çocuk süreçler arasında böyle bir boru kullanarak iletişim, paralel işlemler ve veri paylaşımı konularında etkili bir yol sunar.

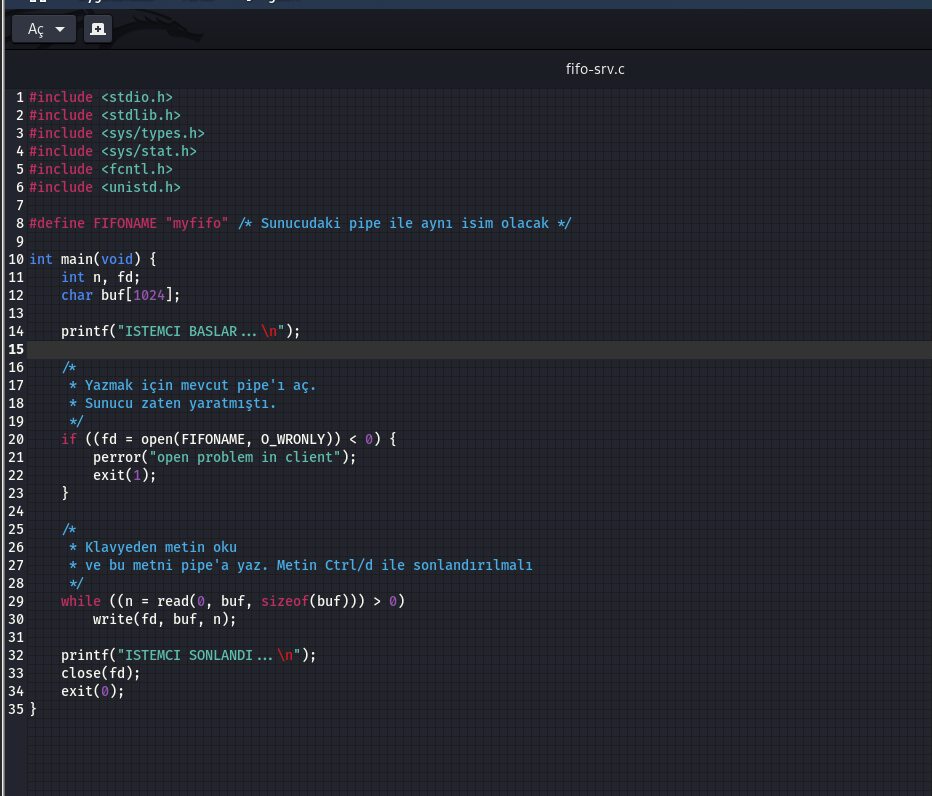
metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

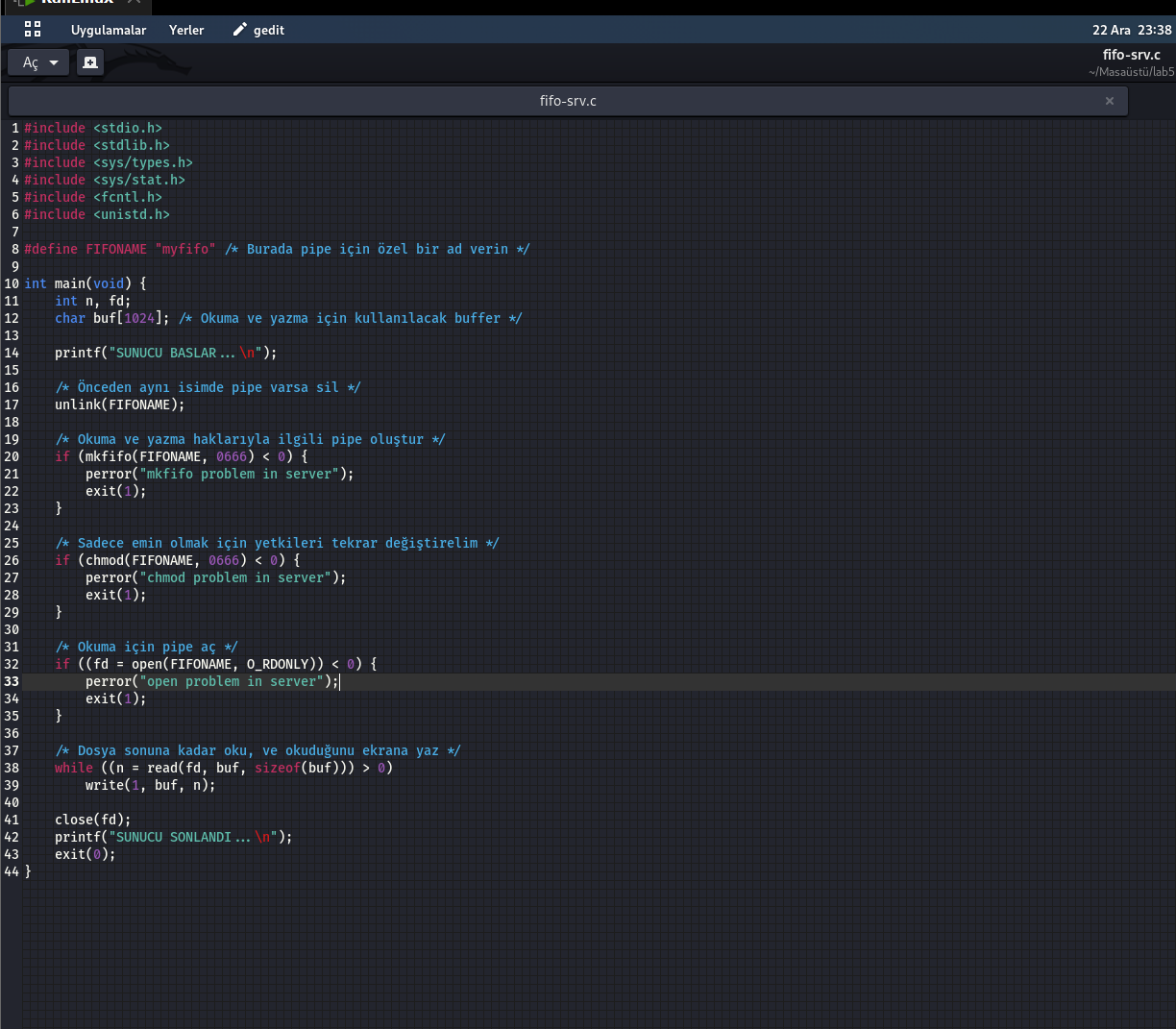
Açıklama otomatik olarak oluşturuldu

Formun Üstü

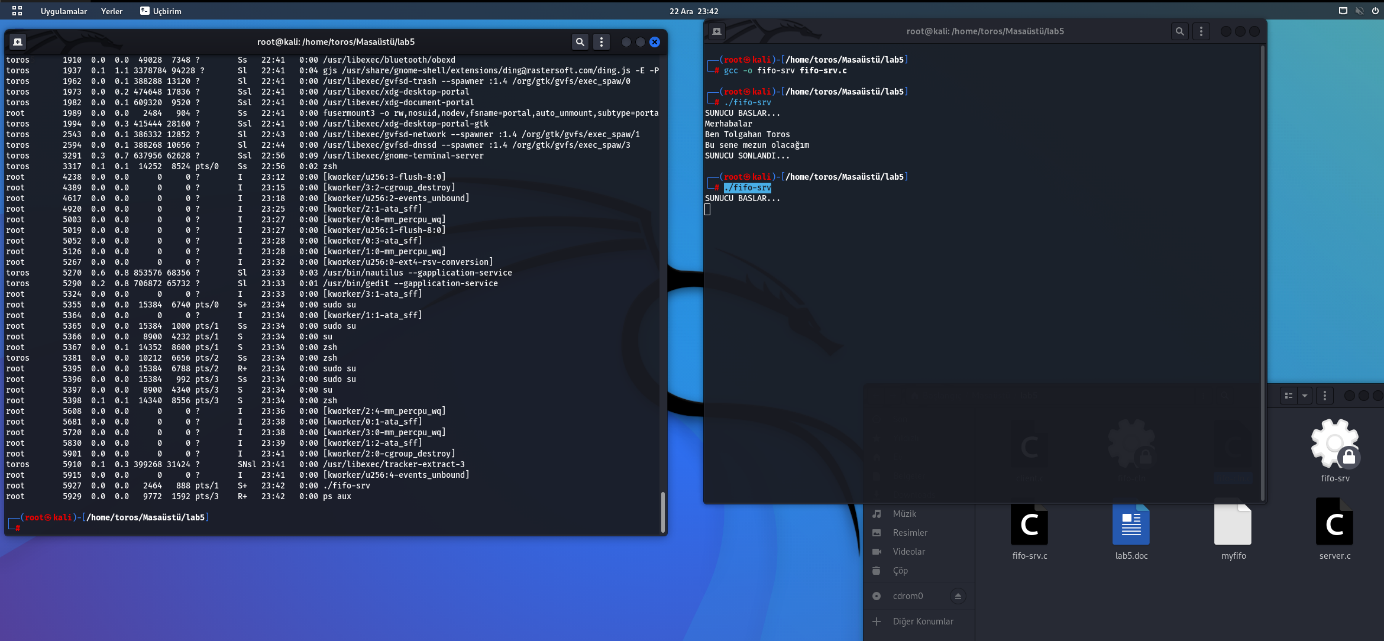
Bölüm 2

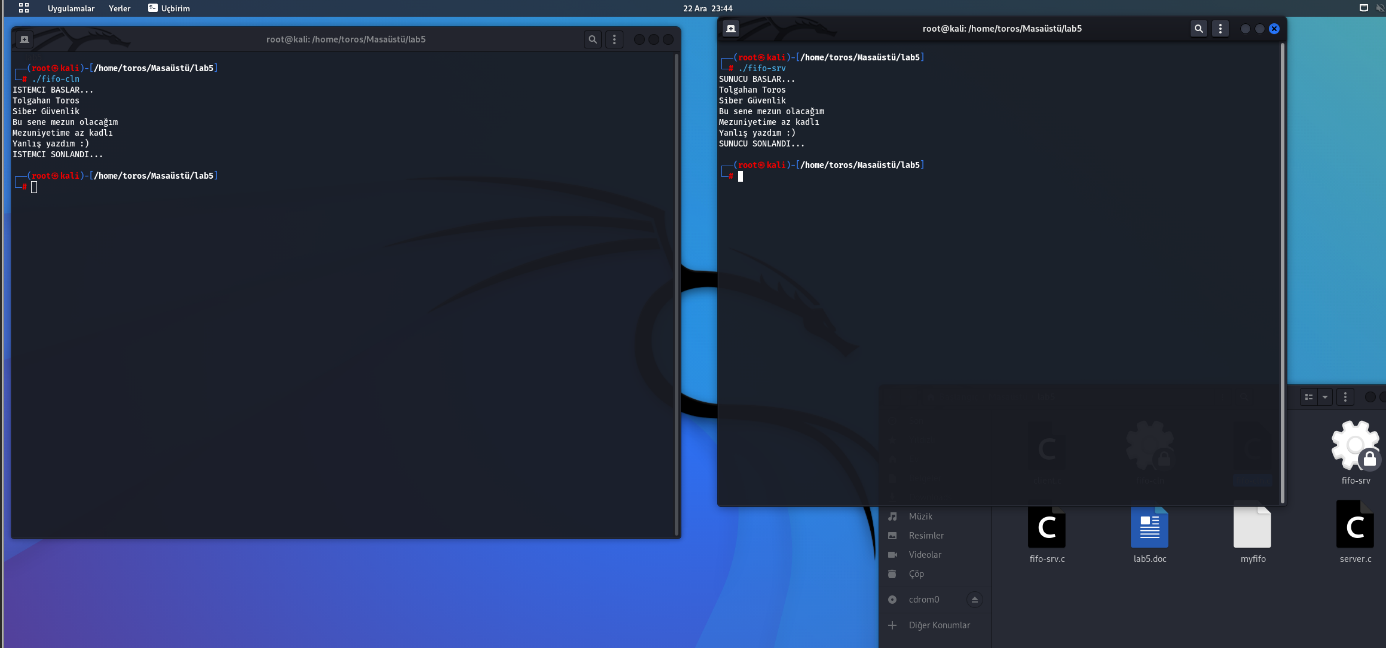






metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu



metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, multimedya yazılımı, elektronik donanım içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

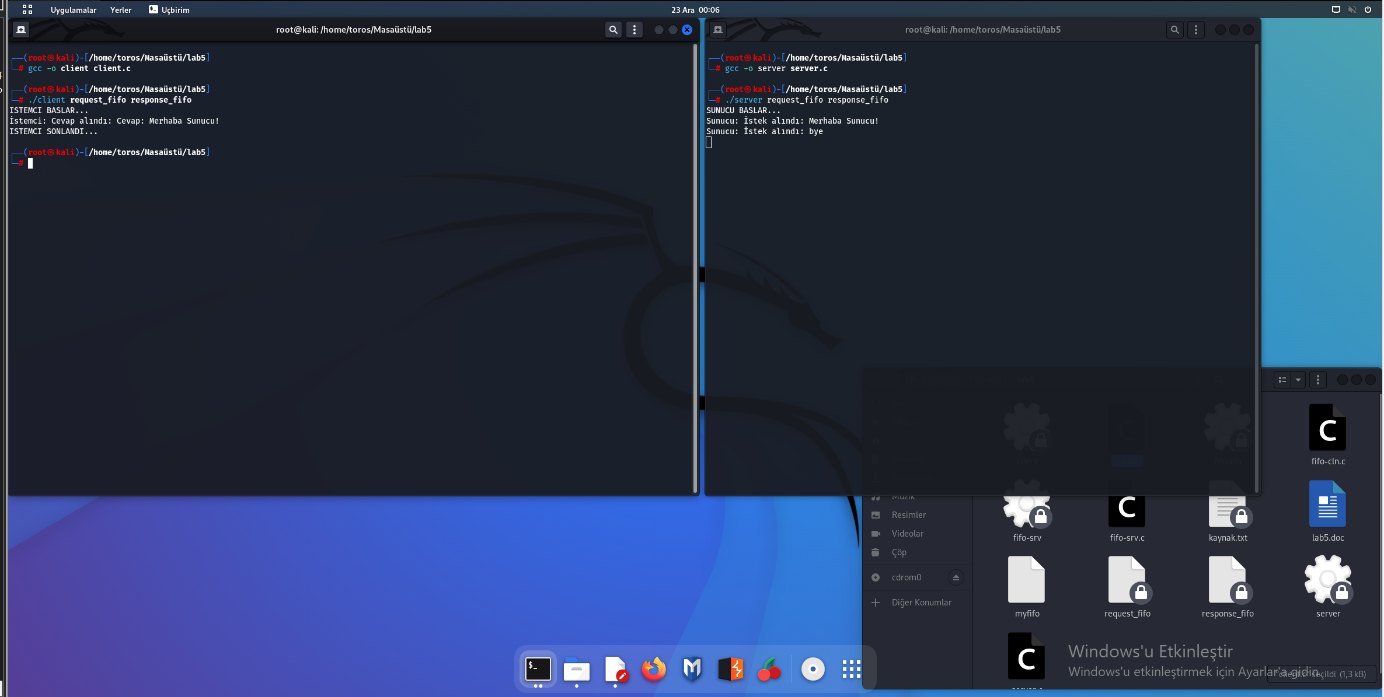
elektronik donanım, metin, ekran görüntüsü, ekran, görüntüleme içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, ekran, görüntüleme, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bu program, bir sunucu ve istemci arasında iki yönlü iletişimi gerçekleştiren bir yapı üzerine kuruludur. Sunucu tarafından başlatılan program, parametre olarak aldığı iki pipe ismi ile iki yönlü veri akışını sağlar. Sunucu, istemciden bir istek mesajı bekler ve bu isteğe yanıt olarak bir mesaj gönderir. İstemci, sunucuya istek gönderir ve sunucudan gelen yanıtı alarak işlemlerini gerçekleştirir. İstemci, işini tamamladığını belirten "bye" mesajını sunucuya gönderir ve sunucu da bu mesajı aldıktan sonra kullanılan pipeları silerek iletişimi sonlandırır. Bu şekilde, sunucu ve istemci arasında düzenli bir iletişim sağlanır ve program sonlandırılırken pipelar temizlenir.



**server.c** adlı programın içeriği şu şekildedir:

1. **FIFO İsimleri:** **REQUEST\_FIFO** ve **RESPONSE\_FIFO** isimlerinde iki adet FIFO tanımlanmıştır. Sunucu, istemciden gelen istekleri **REQUEST\_FIFO** üzerinden alır ve cevapları da **RESPONSE\_FIFO** üzerinden gönderir.
2. **FIFO Oluşturma:** **mkfifo** fonksiyonu kullanılarak **REQUEST\_FIFO** ve **RESPONSE\_FIFO** boruları oluşturulur. Bu borular, iletişim için kullanılır.
3. **Dosya İşlemleri:** Sunucu, **REQUEST\_FIFO**'yu okuma (**O\_RDONLY**) ve **RESPONSE\_FIFO**'yu yazma (**O\_WRONLY**) modunda açar.
4. **İstekleri ve Cevapları İşleme:** Sonsuz bir döngü içinde sunucu, **REQUEST\_FIFO**'dan gelen istekleri okur, bu isteklere cevap verir ve cevapları da **RESPONSE\_FIFO**'ya yazar.
5. **FIFO Silme:** Sunucu, işini tamamladıktan sonra **REQUEST\_FIFO** ve **RESPONSE\_FIFO**'yu siler ve programı sonlandırır.

**client.c** adlı programın özellikleri şu şekildedir:

1. **FIFO İsimleri:** **REQUEST\_FIFO** ve **RESPONSE\_FIFO** adlarında iki FIFO tanımlanır. İstemci, sunucuya istekleri **REQUEST\_FIFO** üzerinden gönderir ve sunucudan gelen cevapları **RESPONSE\_FIFO** üzerinden okur.
2. **FIFO Oluşturma:** **mkfifo** fonksiyonu kullanılarak **REQUEST\_FIFO** ve **RESPONSE\_FIFO** boruları oluşturulur.
3. **Dosya İşlemleri:** İstemci, **REQUEST\_FIFO**'yu yazma (**O\_WRONLY**) ve **RESPONSE\_FIFO**'yu okuma (**O\_RDONLY**) modunda açar.
4. **İstek Gönderme ve Cevap Alma:** İstemci, sunucuya bir istek gönderir (**REQUEST\_FIFO**'ya yazar) ve ardından sunucudan gelen cevabı alır (**RESPONSE\_FIFO**'dan okur).
5. **FIFO Silme:** İstemci, işini tamamladıktan sonra **REQUEST\_FIFO** ve **RESPONSE\_FIFO**'yu siler ve programı sonlandırır.
6. **Uyarı:** İstemci, sunucuya **bye** mesajını gönderir ancak sunucu programında bu mesaj işlenmemiştir. Daha gelişmiş bir uygulama için bu tür durumların ele alınması önemlidir.

Her iki program da, gelen isteği basit bir cevapla yanıtlayan temel bir iletişim mekanizması sunmaktadır. Gerçek uygulamalarda, bu temel mekanizmalar genellikle daha karmaşık işlemler ve güvenlik kontrolleri ile birleştirilir.