**T.C.**

**YILDIZ TEKNİK ÜNİVERSİTESİ**

**FEN BİLİMLERİ ENSTİTÜSÜ**

# DAĞITIK VERİTABANI SİSTEMLERİNDE GERÇEK ZAMANLI VERİ GÜVENLİĞİ: BİLGİ AKIŞ DENETİMİ

**Çiğdem BAKIR**

DOKTORA TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Programı

Danışman

Dr. Öğr. Üyesi. Mustafa Utku KALAY

Aralık, 2020

**T.C.**

**YILDIZ TEKNİK ÜNİVERSİTESİ**

**FEN BİLİMLERİ ENSTİTÜSÜ**

# DAĞITIK VERİTABANI SİSTEMLERİNDE GERÇEK ZAMANLI VERİ GÜVENLİĞİ: BİLGİ AKIŞ DENETİMİ

Çiğdem BAKIR tarafından hazırlanan tez çalışması çalışması 04.12.2020 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı, Bilgisayar Mühendisliği Programı **DOKTORA TEZİ** olarak kabul edilmiştir.

Dr. Öğr. Üyesi. Mustafa Utku KALAY

Yıldız Teknik Üniversitesi

Danışman

**Jüri Üyeleri**

Dr. Öğr. Üyesi. Mustafa Utku KALAY, Danışman
Yıldız Teknik Üniversitesi _____

Doç. Dr. Mehmet Sıddık AKTAŞ, Üye
Yıldız Teknik Üniversitesi _____

Prof. Dr. Oya KALIPSIZ, Üye
Yıldız Teknik Üniversitesi _____

Doç. Dr. Atakan KURT, Üye
İstanbul Üniversitesi _____

Prof. Dr. Selim AKYOKUŞ, Üye
İstanbul Medipol Üniversitesi _____

Danışmanım Dr. Öğr. Üye. Mustafa Utku KALAY sorumluluğunda tarafımca hazırlanan Dağıtık Veritabanı Sistemlerinde Gerçek Zamanlı Veri Güvenliği: Bilgi Akış Dnetimi başlıklı çalışmada veri toplama ve veri kullanımında gerekli yasal izinleri aldığımı, diğer kaynaklardan aldığım bilgileri ana metin ve referanslarda eksiksiz gösterdiğimi, araştırma verilerine ve sonuçlarına ilişkin çarpıtma ve/veya sahtecilik yapmadığımı, çalışmam süresince bilimsel araştırma ve etik ilkelerine uygun davrandığımı beyan ederim. Beyanımın aksinin ispatı halinde her türlü yasal sonucu kabul ederim

Çiğdem BAKIR

İmza

*Aileme*

*Doç.Dr Veli HAKKOYMAZ anısına*

# TEŞEKKÜR

Tez aşamasında benimle bilgi birikimini paylaşan, motive eden ve desteğini benden esirgemeyen sevgili danışman hocam Dr. Öğr. Üyesi Mustafa Utku KALAY ve Doç. Dr. Veli HAKKOYMAZ'a teşekkürlerimi sunarım. Tez izleme komitemde yer alan ve değerli yorumlarıyla katkıda bulunan Doç. Dr. Mehmet Sıddık AKTAŞ ve Doç. Dr. Atakan KURT'a teşekkürlerimi ayrıca belirtmek isterim.

Doktora aşamasına gelene kadar beni okutan ve yetiştiren sevgili anne ve babam Beyhan BAKIR, Şerif BAKIR'a ve her zaman yanımda olan kardeşlerime teşekkür ederim. Hepiniz iyi ki varsınız. Ayrıca Yıldız Teknik Üniversitesi Bilgisayar Mühendisliği Bölümündeki tüm hocalarım ile Erzincan Binali Yıldırım Üniversitesi Bilgi işlem Daire Başkanlığı'ndaki çalışma arkadaşlarıma ve bu zorlu süreçte bana yardımcı olan başta Mehmet GÜÇLÜ olmak üzere tüm arkadaşlarıma katkılarından ve desteklerinden dolayı teşekkür ederim.

Bu tezi, maddi ve manevi hiçbir fedakarlıktan kaçınmayıp desteklerini hiçbir zaman esirgemeyen sevgili aileme ve merhum hocam Doç.Dr Veli HAKKOYMAZ'a ithaf ediyorum.

Çiğdem BAKIR

# İÇİNDEKİLER

# SİMGE LİSTESİ

| | |
|---|---|
| n | Düğüm Sayısı |
| T | Eşik Değer |
| i | Nesne |
| c | Sabit Sayı |
| L | Zincir Uzunluğu |

# KISALTMA LİSTESİ

| | |
|---|---|
| DAC | Dicretionary Access Control |
| MAC | Mandatory Access Control |
| RBAC | Rol Based Access Control |
| RSA | Rivest- Shamir-Adleman |

# ŞEKİL LİSTESİ

# TABLO LİSTESİ

# ÖZET

## Dağıtık Veritabanı Sistemlerinde Gerçek Zamanlı Veri Güvenliği: Bilgi Akış Denetimi

Çiğdem BAKIR

Bilgisayar Mühendisliği Anabilim Dalı

Doktora Tezi

Danışman: Dr. Öğr. Üyesi. Mustafa Utku KALAY

Veri güvenliği, verinin yetkisiz kişilerce kullanılması, değiştirilmesi ve yayılmasının önlenmesini amaçlar. Bu çalışmada, dağıtık veritabanlarında bilgi akış denetimi ile veri gizliliği ve kullanıcıların veri mahremiyetini sağlamak amaçlanmıştır. Kullanıcılarla ilgili çeşitli kriterler dikkate alınarak kullanıcı saldırılarının tespit edilmesi amacıyla ortak bir model oluşturulmuştur. Ayrıca, veri akış denetimi ile gizliliği muhafaza edecek dağıtık etiket modeli tanıtılır. Bu model aktör, nesne ve etiketten oluşur. Nesne sahibi bir aktördür ve sahip olduğu veriyi sistemdeki başka aktörlerce paylaşmak durumundadır. Aktörler nesneleri etiketleyerek veri gönderimini sağlar. Etiket aktörler tarafından verilen kişisel güvenlik politikası ifadeleridir. Her aktör diğerlerinden bağımsız bir şekilde kendi güvenlik ve gizlilik politikasını belirler. Etiket aracılığıyla, güvenli olmayan ulaşım kanallarında, akış kontrolü, sistemde bulunan tüm aktörlerin veri gizliliğini sağlar. Veri nesnesi, güvenli olmayan aktör ve ortamlarda güvenli bir şekilde yayılır ve paylaşılır. Bu çalışmada; ilk kez 1998 yılında Myer tarafından tanıtılmış olan dağıtık etiket

modelinin dağıtık veritabanında gerçekleştirilen tüm işlemler (okuma, yazma, güncelleme, silme) için uygulanarak daha pratik ve esnek bir şekilde gerçekleştirilmesi sağlanmıştır. Bu model aktör, nesne ve etiketten oluşur. Literatürde sadece okuma veya sadece yazma işlemleri ayrı etiket kullanılarak gerçekleştirilmiştir. Aynı anda tüm işlemler için veri güvenliğini sağlayacak bir yapı geliştirilmemiştir. Bizim çalışmamızda ise nesne üzerinde gerçekleştirilen tüm işlemler için tek etiket kullanılır. Tek etikete bakılarak hangi işlem ile aktörler arasında nasıl bir yetkilendirme ve erişim denetimi yapılacağı gösterilir. Böylelikle literatüde gerçekleştirilen çalışmaların aksine dağıtık veritabanında gerçekleştirilen tüm işlemler için veri gizliliği, veri bütünlüğü ve veri tutarlılığı sağlanmıştır. Önerilen tek etiket modelin sonuçları deneysel çalışma ile gösterilmiştir. Ayrıca yol kısaltma ile kaynak düğümden hedefe geçen nesnelerin oluşturduğu uzun düğüm zinciri kırılarak nesnelere hızlı erişim yapılır, erişim maliyeti azalır. Bu sonuç, dağıtık ortam modellenerek, deneysel çalışma ile de gösterilmiştir.

**Anahtar Kelimeler:** Dağıtık sistemler, yol kısaltma, etiketleme modeli, gizlilik, mahremiyet

# ABSTRACT

## Real Time Data Security in Distributed Database Systems: Information Flow Control

Çiğdem BAKIR

Department of Computer Engineering

Doctor of Philosophy Thesis

Advisor: Assist. Prof. Dr. Mustafa Utku KALAY

Data security aims to prevent the use, modification, and spread of data by unauthorized people. In this study, our purpose is to provide data privacy and confidentiality with information flow control in distributed databases. The common model is formed in order to detect the user attacks by means of the various criteria performed by the users. Also, a decentralized label model is introduced that maintains confidentiality including privacy, with data flow control. This model consists of the actor, object and label. The owners of the objects are actors and they need to share their data objects with others. Actors label data objects and then send them out. A label contains policy statements of data security issued by each of the owners. Each owner sets its own security and privacy policy independently of other owners. The confidentiality of data in unsecured transport channels is ensured for all actors in the system by means of labels while data is in flow. Data objects are spread and shared securely among actors within unsecured environments. In this study, by applying the distributed label model, which was first introduced by Myer in 1998, for all transactions performed in the distributed database (read, write, update, delete), it is provided to perform in a more practical and flexible way. This

model consists of actor, object and label. In the literature, only reading or only writing transactions were carried out using separate labels. A structure that ensures data security for all transactions at the same time has not been developed. As for that in our study, a single label is used for all transactions performed on the object. By looking at the single label, it is shown that how to perform an authorization and access control between which transaction and actors. Thus, on the contrary for the studies carried out in the literature, the data confidentiality, data integrity and data consistency were ensured for all transactions performed in the distributed database. The results of the proposed single label model have been demonstrated by experimental study. In addition, with the path compression, the long node chain that is formed while data objects are passing between the source node and the destination is broken, so that the objects are retriewed fast and the cost of access is reduced. This result is shown with experimental study by modeling the distributed environment.

**Keywords:** Distributed systems, path compression, labeling model, confidentiality, privacy

# 1
# GİRİŞ

Dağıtık veritabanı sistemlerinde veri güvenliği, verinin yetkisiz kişilerce kullanılması, değiştirilmesi ve yayılmasının önlenmesini gerektirir. Teknolojinin hızla gelişmesiyle birlikte bankacılık, sağlık, e-ticaret ve iletişim gibi birçok alanda veri güvenliği önemli bir sorun haline gelmiştir. Bu sorunların çözülmesi amacıyla bilgi akış denetimi ve erişim denetimi gibi tedbirler kullanılmaktadır[1].

Veri gizliliği (confidentiality), bir verinin sadece yetkisi olan aktörlerce kullanılarak onun üzerinde okuma ve yazma gibi işlemleri yapabilmesini ifade eder. Kişisel verinin korunması (mahremiyet) ile veri gizliliği arasında ortak bazı noktalar da bulunmaktadır[2]. Ancak, mahremiyet (privacy) ve gizlilik aynı kavramlar değildir. Literatürde pek çok makalede bu konu tartışılmıştır [3,4,5,6] . Mahremiyet gizliliğe göre daha karmaşık bir kavram olup odağında insan vardır. Gizlilik haberleşme güvenliğinde kullanılan odağında veri olan şifreleme yöntemlerinin kullanıldığı kriptolojinin bir çalışma alanıdır[7]. Diğer bir ifadeyle, mahremiyetin korunması demek, kişisel veya kurumsal hassas, önemli verinin, onu kötüye kullanabilecek aktörlerin eline geçmesinin engellenmesidir[8,9,10].

Organizasyonların finansal verileri, hastanelerdeki hastaların tanı ve tedavi süreciyle ilgili kişisel bilgiler, bankalardaki müşterilerle ilgili kredi kart bilgileri ya da sanayide ürün tasarım bilgileri hassas ve gizli veri örnekleridir. Bu bilgilerin kaynağında, aktörler arası sirkülasyonunda ve hedefte korunması gerekir[11].

Örnek olarak, iki tıp merkezindeki hasta kayıtlarının tutulması ile ilgili bir senaryoyu ele alalım (Bkz. Şekil 1.3): Amaç, kayıtların her iki merkezde hızlı ve güvenli bir şekilde paylaşılmasıdır. Bir merkez veriyi güncellemesi durumunda, her iki merkezden bakıldığında birbiriyle eşit hasta kaydının gözlenmesi gerekmektedir. Her tıp merkezinde bir hastanın ad, soyad, TC kimlik numarası, doğum tarihi, doğum yeri, kan grubu, cinsiyet, telefon, adres gibi genel bilgilerinin yanında tanı, tedavi süreci, geçmiş sağlık bulguları, kullandığı ilaçlar, laboratuvar

raporları, radyoloji raporları, geçirdiği ameliyatlar, kronik rahatsızlıklar, bulaşıcı hastalıklar, gebelik durumu vb. sağlıkla ilgili kişiye özel gizli kalması gereken bilgileri vardır. Ayrıca hastalarla ilgili genel ve gizli bilgilerin yanı sıra tıp merkezinin kayıt verileri bulunmaktadır[12,13]. Verilen bu örnek, ortak bir bilgi nesnesi ve buna katkı yapan aktörleri göstermektedir. Başlıca aktörler hasta, doktor ve tıp merkezi personeli olarak görülebilir. Yani, bahsedilen aktörler bu bilgi nesnesine ortaklaşa sahiptir diyebiliriz[14,15,16]. Hastalarla ilgili eksik bilgiler, tedavi sürecinin güncellenmesi gibi sorunları çözebilmek için her merkez kendi kayıtlarında değişiklikler yapar[17,18,19]. Bir merkez diğer merkezden hasta ile ilgili bilgilere erişmek istediğinde erişmek istediği bilgileri yerel güvenlik politikalarına göre alır ve gönderir. Bilgi akış denetimi ile bu politikaların gerçeklenmesi sağlanır. Hastalarla ilgili kayıtlar, her iki merkezde yetkili olanlara gösterilmeli, diğerleri engellenmelidir [20,21,22].

### 1.1 Literatür Özeti

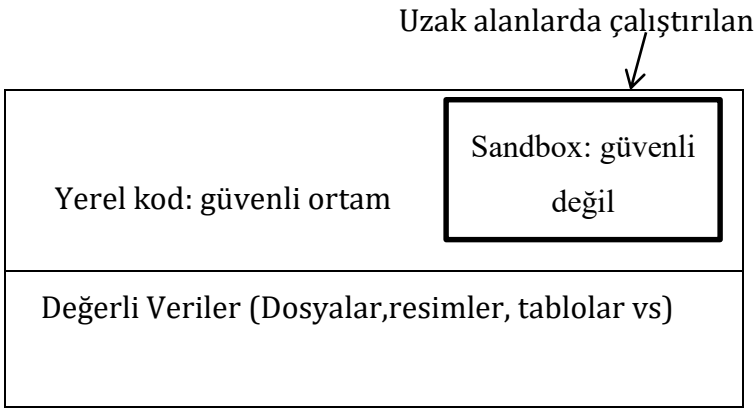
Bir dağıtık hesaplama ortamı, graf veri yapısı ile modellenebilir. Graf, düğümler ve bu düğümleri birbirine bağlayan kenarlar kümesinden oluşan bir veri yapısıdır[23,24]. Eğer G graf ise, bunun tanımı;

$G = (V,E)$

$V(G) = \{v_1, v_2, \ldots, v_N\}$

$E(G) = \{e_1, e_2, \ldots, e_M\}$ olmak üzere $E \subseteq VxV$'dir.

Dağıtık ortam, saklama (storage), çalışan (worker) ve yayın (dissemination) olmak üzere üç çeşit düğüm ile gerçekleştirilir[25]. Kenarlar, bir veri nesnesinin bir düğümden diğerine geçişini gösterir. Saklama düğümü, nesneleri kalıcı bir şekilde saklar. Yayın düğümü kendisinden, nesne istediğinde nesnenin kopyalanmasını sağlar. Çalışan ve yayın düğümünün, nesneleri almaya yetkisi olup olmadığına bakar (gizlilik politikası ile). Çalışan düğüm, programları çalıştırır. Yayın düğüm, ise sık kulanılan nesneleri gruplar halinde saklar[26,27].




**Şekil 1.1** Java güvenlik modeli

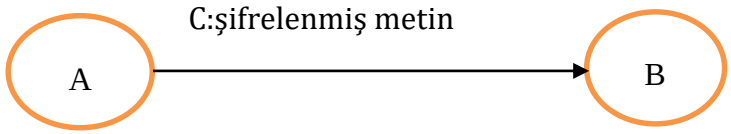
Güvenli olmayan kodların yüklenmesine karşı her program bazı güvenlik önlemleri alır. Örneğin, Java uzak sitelerden kod indirilmesine izin verir. Bu, bazı gizli bilgilerin bu sitelere aktarılmasına neden olabilir. Kullanıcılar Java'da applet (kod parçacığı) yükleyerek kendi verilerini hesaplar[28]. Kod parçacığı, Java kodlarının, Internet üzerinden yayınlanıp tarayıcı içerisinde çalışmasını sağlar. Ancak kullanıcı bunu indirdiğinde, çeşitli gizli dosyalara erişebilir ve çeşitli sitelere bu bilgilerin aktarılmasına sebep olabilir. Bu tür kullanıcılardan gelen kod parçacıklarından korunmak gerekir. Java bu risklere karşılık 'sandbox' güvenlik modelini kullanmıştır. Bu güvenlik modeli Şekil 1.1'de gösterilmiştir. Bu model daha çok yerel kodların çalışmasında güvenilirdir. Yerel uygulamalar, güvenilir olmayan kullanıcılara verilerin paylaşılmasını önler. Ancak çok büyük uygulamalardaki ve uzak alanlarda çalıştırılan güvenli olmayan kod parçacıklarının çalışmasında veri sızıntısına sebep olduğu için, sınırlı alanlarda kullanılan bir güvenlik modelidir[28].

Veri tabanlarında veri güvenliğini sağlamak için dört temel denetim mekanizması önerilmiştir. Bunlar, erişim denetimi (access control), çıkarsama denetimi (inference control), akış denetimi (flow control) ve veri şifreleme (data encryption) yöntemidir. Bizim çalışmamız, veri güvenliği için akış denetimi ile ilgilidir[29].

Erişim denetimi, isteğe bağlı erişim denetimi (discretionary access control, DAC), zorunlu erişim denetimi (mandatory access control, MAC) ve rol tabanlı erişim denetimi (rol based access control, RBAC) olmak üzere üç farklı yöntemle

gerçekleştirilir[30]. DAC, veri sahibi tarafından, veriye erişim hakkı ver veya bu hakkı kaldır şeklinde tanımlamayı ifade eder. Bu, isteğe bağlı yetkilendirme yapması nedeniyle esnek olmayan, yetki var ya da yok şeklinde katı bir güvenlik modelidir. [31,32]. MAC ise veriye erişim sağlanması için duyarlılık düzeyi (sensitivity level) kullanır. Kullanıcı veri ve nesneleri piramidal küme içeren çeşitli güvenlik seviyelerine böler. Bu güvenlik seviyelerine göre bilgi akışı sağlanır (örneğin; top secret, secret, classified, unclassified gibi). MAC merkezi ve zorunlu erişim seviyelerine dayalı, endüstri ve şirketlerden ziyade özellikle, devlet için askeri ve istihbarat uygulamalarına yönelik bir güvenlik modelidir[33]. RBAC ise kullanıcılara işletmelerdeki görev ve sorumluluklarına göre roller tanımlar. Kullanıcılar kendilerine atanan bu rollerin yetkilerini kullanır. Erişim işlemleri bu rollere göre yapılır. Aynı role sahip tüm kullanıcılar bu rolün sahip olduğu tüm işlemleri gerçekleştirir[34].

Erişim denetimi, bilgi akış denetimi ile kapsanan problemleri çözmede yetersiz kalır. Çünkü bu teknikler sadece aktörün yetkisini ifade eder[35]. Veri nesnelerinin yetkisiz kişilerin eline geçmesini önleme, yetkisiz kişilere iletilmesini sağlayan ulaşım kanallarının denetim altına alınmasını gerektirir[36]. Gizlilik ihlalini önlemek için, dağıtık ortamda güvensiz düğümlerin olmasına rağmen, veri nesnelerinin güvenli bir şekilde paylaşılması, fonksiyon ve hesaplamaların gerçekleştirilmesi gerekir[37]. Veri gizliliği, alt düzey güvenlik yöntemi olan şifreleme ile de gerçekleştirilmektedir. Şifreleme, verilerin bir anahtar ile farklı bir biçime dönüştürülmesine denir. Verilerin bir göndericiden bir alıcıya ulaşmasından sonra, alıcı şifrelenmiş metni düz metne dönüştürür. Örneğin, RSA (Rivest-Shamir-Adleman) şifreleme yöntemi için public ve private anahtarlar kullanılmaktadır. Public anahtar herkesçe bilinir. Ancak private anahtar kişiye özeldir. Şekil 1.2'de bir metnin bir düğümden diğerine şifrelenerek nasıl aktarıldığını göstermektedir. M düz metni, C ise şifrelenmiş metni ifade eder. Düğüm A'dan düğüm B'ye C şifrelenmiş metnin iletildiğini kabul edelim. Bu metnin çözülebilmesi (decription) için düğüm B'de $Private_A$ anahtarı (A'ya ait private anahtar) olmalıdır. Bu yoksa, B düğümü C'den M'ye geçiş yapamaz. Şifrelenmiş metnin içeriğinin çözülebilmesi için B düğümünün $Private_A$ anahtarını bir biçimde edinmesi gereklidir[38].




$C = Enc(M, Public_A, Private_A)$ $M = Decrip(C, Public_A, Private_A)$

**Şekil 1.2** Düğümler arası şifreli metin aktarımı

Ancak, alt düzey güvenlik tekniği olan şifreleme tekniği, dağıtık ortamlarda anahtar dağıtım problemi ortaya çıkarır. Anahtar yani şifrenin kullanıcılara dağıtılması, kendi başına bir güvenlik problemidir[39].

Bu tezde, anahtar olarak etiket kullanılmıştır ve etiketler aracılığıyla veri erişimi denetlenmektedir. Merkezi yerine tüm aktörlerin güvenlik gereksinimini tanımladığı dağıtık güvenlik modeli önerilmiştir [40]. Bizim modelde birden fazla kaynaktan sağlanan veriler o sistemdeki bazı aktörlerin ortak verisi olarak kabul edilmiştir[41]. Bu veri ancak veri sahiplerinin ortak onayı alınarak serbest bırakılır. Diğer teknikler güvenli kullanıcılar, güvenli nesneler yani güvenli ortamlarda bilgi akışını sağlarken; etiket modeli tekniği birbirine güvenmeyen aktörler yani güvenilir olmayan ortamlarda da bilgi akışı denetimi yapar. Bu modelde her kullanıcı, kendi güvenlik politikasını diğerlerinden bağımsız olarak belirler.

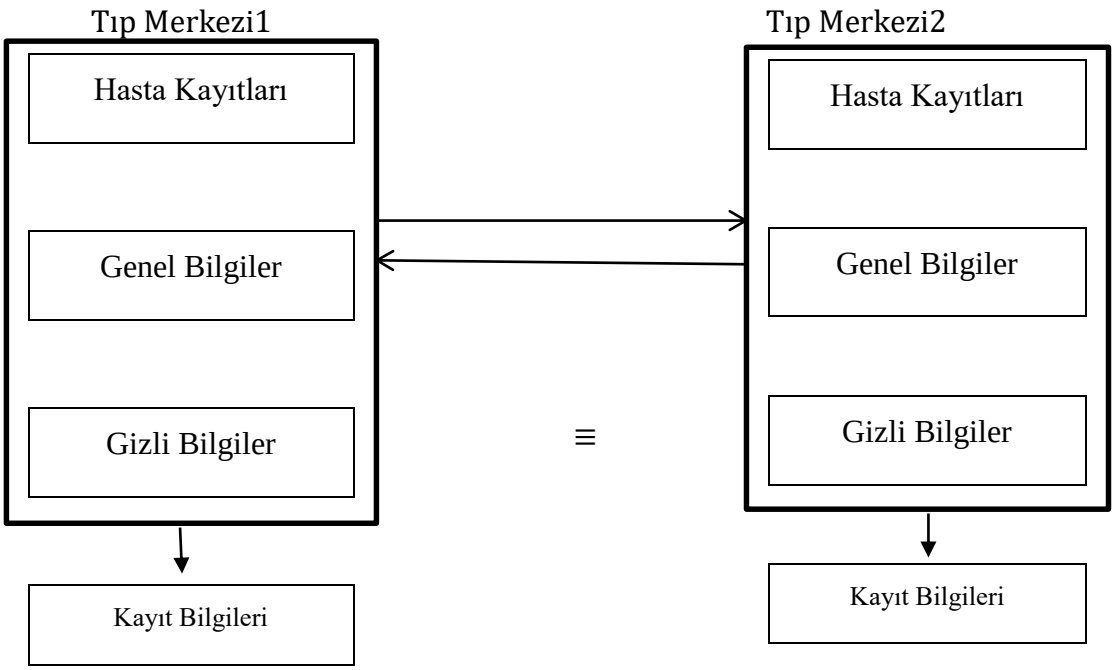
## 1.2 Tezin Amacı

Bu tezin amacı, verilerin, dağıtık ortamda farklı kullanıcıların erişmesine izin veren ve aynı zamanda gizliliğin korunmasını sağlayan yöntem geliştirmektir. Birden çok aktörün ortak eriştiği bir veriye, yetkisiz erişimi engelleyecek yöntemlerin araştırılması amaçlanmıştır. Veri sızıntısı, bilgi akış denetimi yapılmadığında ortaya çıkan bir durumdur. Bunun, muhtemelen, kişisel verilerin korunması yasasının ihlalinden, ulusal güvenliğin tehlikeye atılmasına değin bir dizi istenmeyen neticeleri olacaktır[5]. Yetkisiz, beklenmeyen ve niyet edilmeden yapılan erişim ve bunun üçüncü kişilere sızdırılması, mahçupiyete, kurum içinde olması elzem olan halkın güven duygusunun kaybına ya da kuruma karşı yasal işlemlerin başlatılmasına neden olacaktır[6].

Örnek olarak, hasta verisi ve vergi formundaki veri, birçok aktör tarafından veri işlemek veya hesaplama yapmak amacıyla erişilmek durumundadır. Erişimden kasıt

okuma ve yazma gibi işlemlerdir. Bu erişim sağlanırken, verinin aynı zamanda korunması gibi spesifik bir problemin çözümü önemli bir bilimsel katkıdır. Çalışma bu yönde bir ilk adım olmayı amaçlamıştır.

Mahremiyet, insan haklarının en önemli maddesidir [7]. Özellikle de hasta mahremiyeti, hastaların tüm kişilerden saklamak istediği tanı ve tedavilerini içerir. Bu sağlık verilerinin tutulduğu kayıtların, ilgisiz kişiler tarafından kolayca ulaşılabilir olması, hastanın hakkının çiğnenmesine sebep olur. Hatta bu veriler yayılabilir, başka kişilere satılabilir ya da kişilere şantaj malzemesi olarak kullanılabilir. Hasta mahremiyetini korumak, hasta haklarını güvence altına almak gerekir[8].




**Şekil 1.3** İki tıp merkezi arasındaki bilgi paylaşımı

Spesifik bir örnek olarak, tıp merkezlerinde sağlık verilerinin hangi kullanıcılara hangi yetkilerle verileceği ve erişim yetkilerinin belirlenmesi önemlidir. Hem içerdeki personellerin hem de dışarıdaki yetkili/yetkisiz kullanıcıların hastaların sağlık verilerine erişim işlemleri denetime tabidir. Bu denetim, hastanın izin verdiği bilgilere, yetki verdiği aktörler tarafından erişilmesini sağlar. Ancak bu denetimin tam olarak yapılamaması yetkisiz üçüncü kişilerin bu verilere erişimi ve kullanımına ya da verilerin yayılması gibi sorunlara yol açar[9]. Bu çalışmadaki etiketleme

modeli ile hasta kayıtları verisinin gizliliği sağlanmış olacaktır. Diğer bir deyişle, her hastanın, kişisel sağlık verisinin güvenlik yönetimini, gerçekleştirmek hedefi de etiketleme modeli ile yakalanacaktır.

Bu tez, dağıtık etiketleme modelini tanıtır. Bu modelin bilimsel katkısı ise, böyle birçok paydaşların kullanımına açık veriler sadece yetkili aktörlerce rahatça kullanılırken, yetkisiz üçüncü aktörlerce kullanılmasına izni vermemesidir. Aynı zamanda ortak olarak kullanılan kaynakların bilgi sızıntısına sebep olmadan kullanılmasını sağlayan yöntemlerin araştırılmasına katkı sağlamaktır. Ayrıca veriye erişimde zaman ve hesaplama maliyetini de azaltmak gerekir. Yol kısaltma (path compression) yöntemi ile veriye daha hızlı erişilmekte, erişim maliyeti düşürülmektedir.

Yani, bu tez kapsamında, dağıtık veritabanlarında bilgi akış denetimi ile veri gizliliğini muhafaza edecek dağıtık etiket modeli anlatılmaktadır.

Yapılan diğer çalışmalardan farkı, etiketleme modeli ile güvenilir olmayan aktörler ve ortamlarda veri gizliliğini hedeflemesidir. Verilere verilen etiketler aracılığıyla her aktör kendi güvenlik politikasını diğer aktörlerden bağımsız biçimde belirleyebilir ve diğer aktörlerden seçtiklerini yetkilendirir.

Erişim yetkilerinin gözden geçirilmesi konusu bilgi güvenliği başlığı altındaki zahmetli çalışmalardandır. Bu konuda yapılan çalışmalar kurumlar içerisinde zaman ve iş gücü maliyeti yüksek çalışmalardır. Özellikle karmaşık veri tabanı yapıları içerisinde erişim yetkilerinin gözden geçirilmesi her zaman maliyet etkin bir şekilde yapılamamakta, bu nedenle ihmal edilebilmekte veya yeterince nitelikli biçimde gerçekleştirilmemektedir [42,43]. Bizim çalışmamızda erişim denetimi ve yetkilendirme aktörlerin isteğine uygun veri sızıntısına sebep olmadan bilgi akış denetimi gözeterek sağlanır. Aktörler pratik ve esnek bir şekilde kendi güvenlik, gizlilik ve bütünlük politikalarını oluşturabiliyor. Aynı zamanda işleri bittiğinde rahatlıkla bu politikaları değiştirebiliyor ya da tamamen silebiliyor. Bu da çalışmamızda çalışma zamanında ve hem statik hem de dinamik bir şekilde gerçekleştiriliyor. Çoklu nesne ortamlarında, güvenli olmayan birçok aktörün erişim sağladığı ortamlarda rahatlıkla aktörlerin kendi belirlediği güvenlik politikalarıyla

verilerini koruması amaçlanmıştır. Diğer çalışmalardan farklı olarak veri gizliliği, veri bütünlüğü ve veri tutarlılığını birlikte sağlamasıdır.

Schultz ve arkadaşları, kullanıcıların veri erişimlerini otomatik olarak takip edilmesini sağlayan bir platform geliştirmişlerdir. Bir kullanıcı sisteme her bir işlem için ayrı ayrı giriş yaptığından dolayı yetki kontrolü yeniden gerçekleştirilir. Kullanıcı her aşamada yetki kontrolü yapmak zorunda kalır. Sadece bir aşamada kontrol yapmadığı takdirde veri gizliliği ihlal edilir. Bu durum yetkinin otomatik olarak takip edilebilmesi ihtiyacını doğurmaktadır[44]. Bizim çalışmamızda ise hem yetki verilmesi hem de yetkinin geri alınması için ayrı bir kontrole ihtiyaç bulunmamaktadır. Çalışmamızda okuma, yazma, güncelleme, silme gibi her bir işlem için ayrı bir yetkilendirme ya da erişim denetimine gerek yoktur. Ayrıca kötü niyetli aktörlerin veriye erişimlerinin takipleri ile bilgi ifşasının önüne geçilmeye çalışılmıştır.

## 1.3 Hipotez

Bu çalışma ile dağıtık etiketleme modeli tanıtılmıştır. Bu modelin bilimsel katkısı birçok kullanıcının erişimine açık olan veriler yetkili aktörlerce rahat kullanılırken, yetkisiz üçüncü aktörlerde kullanılmasına izin verilmemesidir. Diğer çalışmalardan farklı olarak, güvenilir olmayan ortam ve aktörlerde veri gizliliği korunmaktadır. Ayrıca dağıtık ortam modellemesi yapılarak nesneye erişimde etkinlik sağlayan yol kısaltma algoritması önerilmiştir. Kendisi de çalışma zamanı (runtime) yönünden etkin bir algoritmadır. Dağıtık ortam simülasyonu gerçekleştirilmiş ve deneysel çalışma ile önerdiğimiz algoritmanın etkinliği ve başarısı gösterilmiştir. Veriye erişimde, erişim maliyeti azaltılmış ve veriye hızlı erişim yapılmıştır. Kısa olarak, çalışmamız ile dağıtık ortamda veriye hem güvenli hem de hızlı erişim yapılmaktadır.

## 1.4 Veri Gizliliği ve Mahremiyet

Veri gizliliği (confidentiality), hassas ve kıymetli olduğu değerlendirilen verilerin yetkisiz kişilerin eline geçmesinin önlenmesi ve sadece uygun erişim izni verilen kullanıcıların onun üzerinde okuma, yazma gibi işlemlerini gerçekleştirebilmesidir. Başka bir deyişle, bir bilgiye kimlerin erişebileceğinin kontrol altına alınmasıdır. Bir

banka veya şirketteki verilerin yetkisiz personel tarafından görülmesi ve üzerinde işlem yapılması veri gizliliğinin ihlal edilmesine örnek olarak gösterilebilir. Veri gizliliği, temel olarak iletişim ve haberleşme güvenliğinde kullanılan ve odağında veri olan farklı şifreleme yöntemlerinin kullanıldığı kriptolojinin bir çalışma alanıdır[10].

Mahremiyet (privacy) ise, veri gizliliğini de içine alan daha geniş ve kompleks bir kavram olmakla beraber odağında insan vardır. Mahremiyet, kişisel/hassas verilerin toplanmasında, paylaşılmasında ve saklanmasında kişinin kimliğini korumak için veriye erişimin sınırlandırılmasıdır. Diğer bir ifadeyle, kişi hak ve özgürlüklerine uygun olarak kendi, göre verisinin istediği kadarını ve istediği yönde uygun kişilere açık etmesi ve diğer kişilerden uzak tutma isteğidir[11].

Örnek olarak, akıllı telefonlar üzerinden bir uygulama ile kullanıcılar diyelim bir müze sistemine farklı yerlerden erişim sağlamak istesin. Bu tür bir müze uygulamasında genel olarak müzeyi ziyaret eden kullanıcılar, müze yönetimi, müzede sergilenen sanat eserleri ve müzenin çevresel bilgileri, bir veritabanında farklı servisler tarafından toplanır ve saklanır. Varsayalım bir kullanıcı bir bilet satın aldığında bu kullanıcının adı, telefon numarası, cinsiyeti, posta kodu, bilet türü, ödeme türü gibi birtakım bilgiler müze sistemi tarafından kaydedilsin ve saklansın. Ayrıca kullanıcılar müze ziyaretlerini gerçekleştirirken, müzedeki sanat eserlerine zarar oluşturmamak için çevredeki sıcaklık, nem, partikül madde, gürültü miktarı çeşitli algılayıcı sensörler tarafından tespit edilir ve veritabanına kaydedilir. Bu müze verilerinin toplanması, paylaşılması ve saklanmasında birtakım mahremiyet ihlalleri ortaya çıkar. Daha somut olmak için, örneğin, bir saldırgan tüm müze ziyaretçilerinin telefon numaralarına ulaşabilir ya da bilet türüne göre (sağlıkçı, asker, denizci, havacı, kamu personeli vs) kişilerin hangi sektörde hangi görevde oldukları bulunabilir. Ayrıca ödeme bilgileri çeşitli reklam şirketlerine verilebilir ve bu şirketler kullanıcılara birtakım istenmeyen gelişigüzel mesajlar gönderebilir. Bu belirtilenler, mahremiyet ihlallerine somut örnek olarak verilmiştir[8].

Mahremiyet ihlallerini önlemek için yaygın olarak şifreleme ve anonimleştirme gibi metotlar kullanılır[9]. Anonimleştirme, kişilerin kimliklerini gizleyerek mahremiyeti korumayı sağlama amaçlı bir metotdur. Anonimlik (kimliksizleştirme),

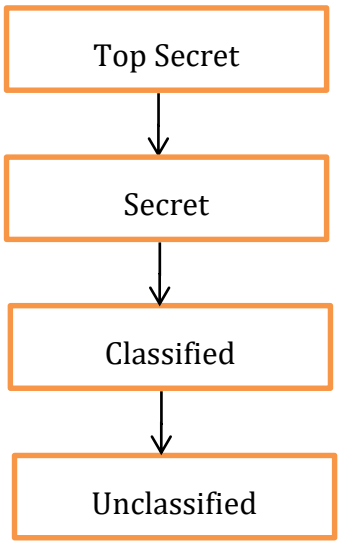
veriyi genelleyerek (generalization) veya gizleyerek (suppression) korumayı amaçlar. Yer değiştirme, şifreleme ya da silme ile yapılır. Bir veri sahibi, bireysel kimlik bilgilerini göstermeden bu veriyi aktarmak istediğinde diğer kullanıcılar tarafından kimlik bilgilerinin kime ait olduğu anlaşılamıyorsa bu veri korumuş olur. Örneğin; anonim tekniği ile müze uygulamasında kişinin yaşı gizlenmek istendiğinde, kişinin yaş alanı bir aralık ile (örneğin, 20-30 yaş arası, 45'ten büyük gibi) daha belirsiz bir ifadeyle değiştirilebilir. Ya da gün, ay ve yıl olarak kaydedilen müze ziyaretçilerinin doğum yılı kısmına * konularak doğum yılı kısmı gizlenir. Müze ziyaretçilerinin yer bilgisini gizlemek ya da ziyaretçilerin arasındaki mesafeyi gizlemek için anonim tekniği kullanılabilir. Ziyaretçilerin ödeme türü alanına bakarak kaç erkek ve kadının nakit ve kredi kartıyla ödeme yaptığı, %'lik oranı gibi birtakım istatiksel veriler ile gösterilebilir. Bu verileri korumak amacıyla bir eşik değer belirlenebilir. Bu eşik değerden küçük olanlar hassas veriler olarak kabul edilir. Ayrıca müze ziyaretçisinin doğum tarihi, posta kodu ve cinsiyeti gibi birtakım kişisel bilgileri birleştirilerek kişinin kimliğinin bulunması anonim tekniği ile engellenir.

## 1.5 Tanımlar

Bu bölümde, çalışmada kullandığımız bazı tanımlar verilecektir[42]:

- *Erişim denetimi*, bir veriye kimlerin erişeceğini ve bu kişilerin veri üzerinde hangi işlemleri yapabileceğini gösterir.
- *Veri akış denetimi*, bilgilerin yetkisiz kişilerin eline geçmesini önlemek için gizli kanalların kontrol edilmesidir. Verinin yetkisiz kişilere iletilmesine imkan tanıyan ulaşım kanallarını tıkayarak bilgi sızıntısının önüne geçer.
- Declassification (downgrading), verinin güvenlik seviyesinin üst güvenlik seviyesinden daha alt güvenlik seviyesine düşürülmesidir. Veri bir alt güvenlik seviyesine aktarılarak, gizlilik derecesi azaltılmış olur.
- Etiket, bir dizi güvenlik politikasından oluşur. Her güvenlik politikası, veri sahiplerinden bir ortağı ve bu ortağın veriyi kimlere kullandıracağını (örn. okuyucu olarak) gösteren bir ifadedir.

- Mahremiyet, kurum ve kullanıcıların kendi verilerinin kim tarafından, ne zaman ve ne kadarını kullanacağı olgusudur. Kullanıcıların izinleri doğrultusunda veri erişiminin sınırlandırılmasıdır.
- Yeniden etiketleme, etiketin güvenlik seviyesi azalmamak şartıyla, etikette güvenlik politikalarında değişiklik yapmaktır.
- Çok katlı güvenlik seviyesi, kullanıcı ve bilgi nesneleri için farklı piramidal seviyelerin tanımlanması ve bu seviyelere bu aktör ve nesnelerin atanmasıdır. Şekil 1.4'de 4 tane farklı güvenlik seviyesi gösterilmiştir. Top secret, en üst güvenlik seviyesi iken; unclassified, en alt güvenlik seviyesini ifade eder.




**Şekil 1.4** Güvenlik seviyeleri

# 2
# VERİTABANI GÜVENLİĞİ

Gelişen teknoloji sayesinde verilere anlık erişim ve verilerin gönderimi oldukça kolaylaşmıştır. Günümüzde Internet kullanımının hızla artması sonucu veri gizliliği, erişim sorunları ve veri güvenliği ile alakalı birçok problemler meydana gelmiştir. Özellikle de güvenlik alanında yeni bir saldırı ya da tehditle ile karşılaşılmaktadır. Ayrıca kötü niyetli saldırganların zarar verebilecekleri daha çok sistemi ve buna bağlı olarak daha fazla veriyi ele geçirme olasılığını doğurmuştur.. Bu saldırılar ve tehditler karşısında banka, eğitim, sağlık gibi çeşitli yerlerdeki özel ve gizli verilerin korunması gerekmektedir. Örneğin, kredi kartı ve kimlik kartı gibi kritik bilgileri tutan kurumlar için veritabanı güvenliği büyük önem taşır.

Veritabanları bilgi sistemlerinin merkezinde yer alan, yüksek seviyede güvenliğin sağlanması gerektiği kurumların ya da bireylerin hassas ve özel bilgilerinin tutulduğu ortamlardır. Veritabanı güvenliğinin gizlilik, bütünlük ve erişebilirlik olmak üzere üç temel öğesi vardır. Gizlilik, verilerin yetkisiz kişiler tarafından görülmesinin önlenmesidir. Bütünlük, verilerin yetkisiz kişiler tarafından değiştirilmesinin önüne geçilmesidir. Erişebilirlik ise, verilerin yetkili kişiler tarafından ulaşılabilmesinin sağlanmasıdır. Veri güvenliği bu üç temel unsurun birlikte sağlanmasıyla oluşur. Ancak virüsler, truva atı, solucanlar, DOS saldırıları, SQL enjeksiyonu , erişim yetkisi olmayan kullanıcıların yetkisiz erişimi, yeterli bilgiye sahip olmayan personeller veri güvenliğini olumsuz etkilemektedir. Özellikle de insan odaklı saldırılar veri güvenli açısından büyük bir risk taşır.

Literatürde veri güvenliğini sağlamak için birçok çalışma yapılmıştır[13,14,15,16]. Bu çalışmaların büyük bir çoğunluğu kullanıcıların yaptıkları saldırılar sonrası yapılacak işlemleri içerir. Bizim çalışmamızda ise, bu çalışmalardan farklı olarak olay kayıtlarında saldırı oluşmadan önceki adımları gösterir. Olabilecek bir saldırıya karşı önlem alınması için riskli kullanıcı ve kullanıcı gruplarını belirlenmiştir. Kısacası, e-ticaret, sağlık, banka gibi çeşitli kurum ve kuruluşlarda sisteme tehdit

unsuru oluşturabilecek riskli kullanıcılar bulunmuştur. Böylelikle risk analizini gerçekleştirebilecek saldırı tahmin sistemi oluşturulmuştur.

Günümüzde yetkisiz erişim, kötü niyetli kullanıcılar, çeşitli yöntemler kullanarak saldırılar yaparak sistemi ele geçirerek özel ve gizli bilgilere ulaşma sıkça karşılaşılan problemler olduğu için çalışmamızda veritabanı gizliliğini ele aldık. Çalışmamız, *veri güvenliği sorunu oluşmadan önce* kullanıcıların çeşitli hal ve hareketlerine bakarak riskli kullanıcıları tahmin etme ve olası saldırılara karşı önlem almayı amaçlıyor.

Bu tez kapsamının ilk bölümünde, veritabanına aktarılan log (olay) kayıtlarındaki çeşitli bilgilere bakarak riskli kullanıcıları belirlenmeye çalıştık. Her kurum ve kuruluşta kullanılabilecek ve geçerli olabilecek bir model geliştirdik. Ayrıca olay kayıtlarındaki yapılan işlemler, veriler , IP adresleri ve roller gibi özellikler değerlendirilerek yetkili/yetkisiz kullanıcılar belirleyerek bilgi güvenliğinin sağlanmasını amaçladık.

Bu tez kapsamında, bir bankadan gerçek olay kayıtları alınmıştır. Bu olay kayıtları üzerinde birtakım normalize teknikleri kullanılarak veriler anlamlı hale getirilmiştir. Bir sonraki aşamada ise, düzenli hale getirilen verilerin birtakım özellikleri alınarak çeşitli kurallar oluşturulmuştur. En son aşamada ise, bu kurallar kullanarak bir model gerçekleştirilerek kullanıcılar risk durumlarına göre çeşitli gruplara ayrılmıştır. Önerdiğimiz model ile sınıflandırdığımız riskli kullanıcıları önceden belirlenen riskli kullanıcılar ile karşılaştırarak önerdiğimiz modelin başarısını ve doğruluğunu hesapladık[17,18].

Literatürde saldırı tahmin etme için istatiksle bir yöntem olan Quickprop sinir ağları kullanılmıştır [19]. Kullanılan veri seti üzerinde erişime izin verilmeyen kullanıcılar tespit edilmiştir. Ancak uzun vadede gerçekleştirilen kusurlu ve anormal davranışları tespit edememiştir. Ayrıca yetkisiz kullanıcıları belirlemek için genetik algoritma kullanılmıştır[20]. Ancak bu çalışma tüm saldırılar için bir çözüm oluşturmamıştır. Diğer kullanılan bir yöntem de Saklı Markov Modelidir[21]. Ancak bu çalışmada uzun vadede gerçekleşebilecek saldırı türlerini bulmada etkili olmamıştır. Bizim çalışmamızın literatürden farklı uzun vadede

gerçekleştirilebilecek olası tüm saldırı türlerini belirlemede kurum içi çalışan personele yardımcı olabilecek bir sistem geliştirmektir.

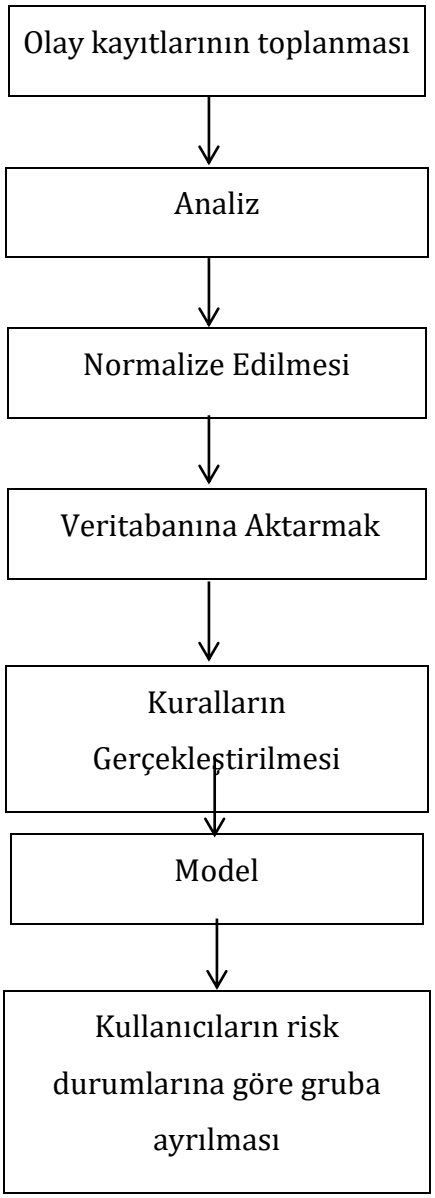
## 2.1 Bilgi Güvenliği için Olay Kayıtları

Olay kayıtları veritabanında gerçekleşen verileri tutmak, istendiğinde kontrol etmek ve sistemde oluşan hataları tutar. Veritabanında gerçekleşen işlemler bu kayıtlarda tutulur ve tüm işlemler burda değerlendirmeye alınır. Olay kayıtlarında hangi kullanıcının hangi nesne üzerinde hangi işlemi (okuma, yazma vs.) gerçekleştirdiği, ne kadarını kullandığı, hangi IP adresinden girdiği gibi kıstaslar tutulur. Bizim çalışmamızda veriler nesneler olarak ifade edilir.

Olay kayıtları nesneleri kalıcı bir şekilde saklar. Çalışan düğüm veya yayın düğümü kendisinden nesne istediğinde nesnenin kopyalanmasını sağlar. Çalışan düğümün ve yayın düğümün nesneleri almasına yetkisi olup olmadığına bakar(güvenlik ve gizlilik politikası ile). Aynı zamanda çalışan düğümde nesne güncellemelerinin yapılıp yapılmadığını kontrol eder (bütünlük politikası ile). Ayrıca olay kayıtlarına bakarak, kusurlu kullanım ve çeşitli saldırıların tespiti gerçekleştirilir. Saldırı tespiti için nesneler ve nesneler üzerinde gerçekleştirilen işlemler ele alınır. Olay yönetimi için Cobit gibi standartlar kullanılmıştır [7]. Ayrıca olay analizi yapabilmek için çeşitli açık kaynak kodlu yazılımlar mevcuttur.

## 2.2 Sistem Tasarımı

Sistem tasarımı Şekil 2.1'de gösterilmiştir. Bizim çalışmamız birkaç adımdan oluşmaktadır. Sistemde gerçekleşen tüm işlemler veritabanında tutulur, temizlenerek ön işlem gerçekleştirilir, belli kıstasla alınarak kurallar oluşturulur ve bu kurallar kullanılarak model geliştirilir. Geliştirdiğimiz model ile birlikte kullanıcılar risk durumlarına göre sınıflandırılmıştır.

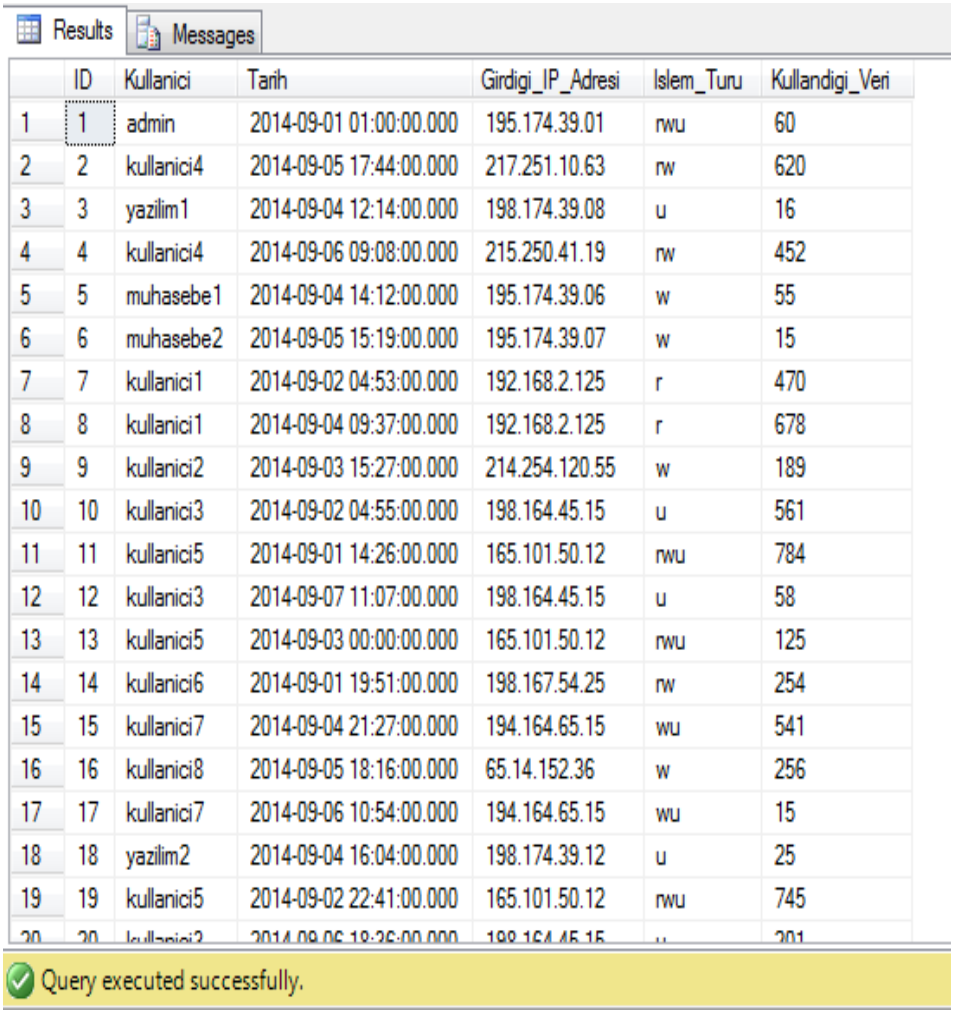



**Şekil 2.1** Sistem tasarımı

## 2.3 Olay Kayıtlarının Normalize Edilmesi

Bizim çalışmamızda, gerçek bir veri seti kullanılmıştır. Bir bankadan son 1 ay içerisinde alınan yaklaşık 3000 kullanıcı ve 10000 işlem üzerinde çalışma yapılmıştır. Bu alınan olay kayıtları çeşitli veri madenciliği teknikleri ile filtreden geçirilerek temizlenmiştir ve normalize edilmiştir. Şekil 2.2'de normalize edilen olay kayıtları gösterilmiştir. Böylelikle kullanılmayacak olan veriler olay kayıtlarından çıkarılarak veriler düzenli hale getirilmiştir. Normalizasyon işlemi olay kayıtlarından anlamlı bilgiler çıkararak verilerin istediğimiz formatta

kullanılmasını sağlamak için gerçekleştirilen filtreleme, temizleme ve raporlama işlemidir. Bu çalışmada kullanıcıların gerçekleştirdikleri işlemler, kullandıkları miktarlar, türler ve IP adresler gibi kriterler alınarak model gerçekleştirilmiştir.



Results | Messages

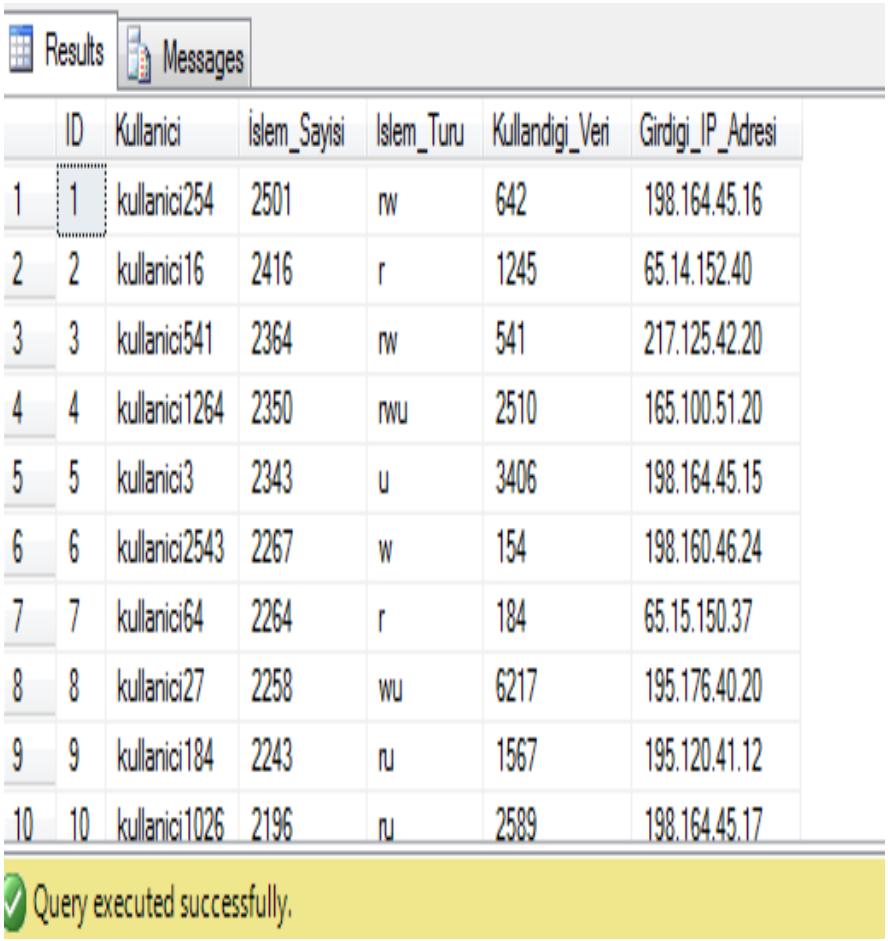
| | ID | Kullanici | Tarih | Girdigi_IP_Adresi | Islem_Turu | Kullandigi_Veri |
|---|---|---|---|---|---|---|
| 1 | 1 | admin | 2014-09-01 01:00:00.000 | 195.174.39.01 | rwu | 60 |
| 2 | 2 | kullanici4 | 2014-09-05 17:44:00.000 | 217.251.10.63 | rw | 620 |
| 3 | 3 | yazilim1 | 2014-09-04 12:14:00.000 | 198.174.39.08 | u | 16 |
| 4 | 4 | kullanici4 | 2014-09-06 09:08:00.000 | 215.250.41.19 | rw | 452 |
| 5 | 5 | muhasebe1 | 2014-09-04 14:12:00.000 | 195.174.39.06 | w | 55 |
| 6 | 6 | muhasebe2 | 2014-09-05 15:19:00.000 | 195.174.39.07 | w | 15 |
| 7 | 7 | kullanici1 | 2014-09-02 04:53:00.000 | 192.168.2.125 | r | 470 |
| 8 | 8 | kullanici1 | 2014-09-04 09:37:00.000 | 192.168.2.125 | r | 678 |
| 9 | 9 | kullanici2 | 2014-09-03 15:27:00.000 | 214.254.120.55 | w | 189 |
| 10 | 10 | kullanici3 | 2014-09-02 04:55:00.000 | 198.164.45.15 | u | 561 |
| 11 | 11 | kullanici5 | 2014-09-01 14:26:00.000 | 165.101.50.12 | rwu | 784 |
| 12 | 12 | kullanici3 | 2014-09-07 11:07:00.000 | 198.164.45.15 | u | 58 |
| 13 | 13 | kullanici5 | 2014-09-03 00:00:00.000 | 165.101.50.12 | rwu | 125 |
| 14 | 14 | kullanici6 | 2014-09-01 19:51:00.000 | 198.167.54.25 | rw | 254 |
| 15 | 15 | kullanici7 | 2014-09-04 21:27:00.000 | 194.164.65.15 | wu | 541 |
| 16 | 16 | kullanici8 | 2014-09-05 18:16:00.000 | 65.14.152.36 | w | 256 |
| 17 | 17 | kullanici7 | 2014-09-06 10:54:00.000 | 194.164.65.15 | wu | 15 |
| 18 | 18 | yazilim2 | 2014-09-04 16:04:00.000 | 198.174.39.12 | u | 25 |
| 19 | 19 | kullanici5 | 2014-09-02 22:41:00.000 | 165.101.50.12 | rwu | 745 |
| 20 | 20 | kullanici2 | 2014-09-06 18:26:00.000 | 198.164.45.15 | u | 201 |

Query executed successfully.

**Şekil 2.2** Normalize sonucu olay kayıtları

### 2.4. Olay Kayıtlarının Veritabanına Eklenmesi

Şekil 2.2'de gösterilen olay kayıtlarında kullanılan çeşitli kıstaslar Şekil 2.3'de gösterilmiştir.




**Şekil 2.3** Olay kayıtlarının veritabanına eklenmesi

## 2.5 Modelin Gerçekleştirilmesi

Bizim çalışamamızda diğer çalışmalardan farklı olarak birden fazla özellik kullanılmıştır. Kullanıcıların yaptıkları işlemler, kullandıkları veri miktarı, IP adresleri, sistemde kalma miktarları çalışmamızda kullandığımız özelliklerdir. Bütün platformlarda kolay bir şekilde kullanılabilecek ortak bir model ile güvenlik analizi gerçekleştirilmiştir. Böylelikle kasten ya da kaza ile sistemi tehlikeye sokabilecek saldırgan kullanıcıları risk durumlarına göre belirleyerek bu kullanıcıları yetkisi doğrultusunda erişim izinleri tanımlanmıştır.

## 2.6 Kuralların Oluşturulması

Risk durumunun hesaplanması için aşağıdaki kurallar dikkate alınmıştır:

1) Sistem Kullanımı: Kullanıcının sistemde kalma durumuna göre aşağıdaki kurallar oluşturulmuştur: Sisteme,

- 1 ile 10 arası girenlerin ağırlıkları 0,
- 11 ile 20 arası girenlerin ağırlıkları 1,
- 21 ile 50 arası girenlerin ağırlıkları 2,
- 51 ile 100 arası girenlerin ağırlıkları 3,
- 101 ile 250 arası girenlerin ağırlıkları 4,
- 251 ve daha fazla girrenlerin ağırlıkları 5,

olarak değerlendirilmiştir.

2) Kullanılan veri miktarı: Kullanıcıların kullandıkları verilerin miktarına göre;

- 0 ile 50 KB arası ağırlık 0,
- 51 ile 100 KB arası ağırlık 1,
- 101 ile 150 KB arası ağırlık 2,
- 151 ile 500 KB arası ağırlık 3,
- 501 KB ve üzeri ağırlık 4,

olarak değerlendirilmiştir.

3) Kullanıcı Yetkileri: Kullanıcıların sistemdeki yetkilerine göre;

- Read için ağırlık1,
- Write için ağırlık 2,
- Update için ağırlık 3,

olarak değerlendirilmiştir.

4) IP Adres Sıklığı: Sisteme en fazla giren 100 IP adresi bulunmuştur.

- Bu 100 IP adresinden girmeyenlerin ağırlıkları 0,

- Bu 100 IP adresinden girenlerin ağırlıkları 1,

olarak değerlendirilmiştir.

### 2.7 Kullanıcıların Kümelenmesi

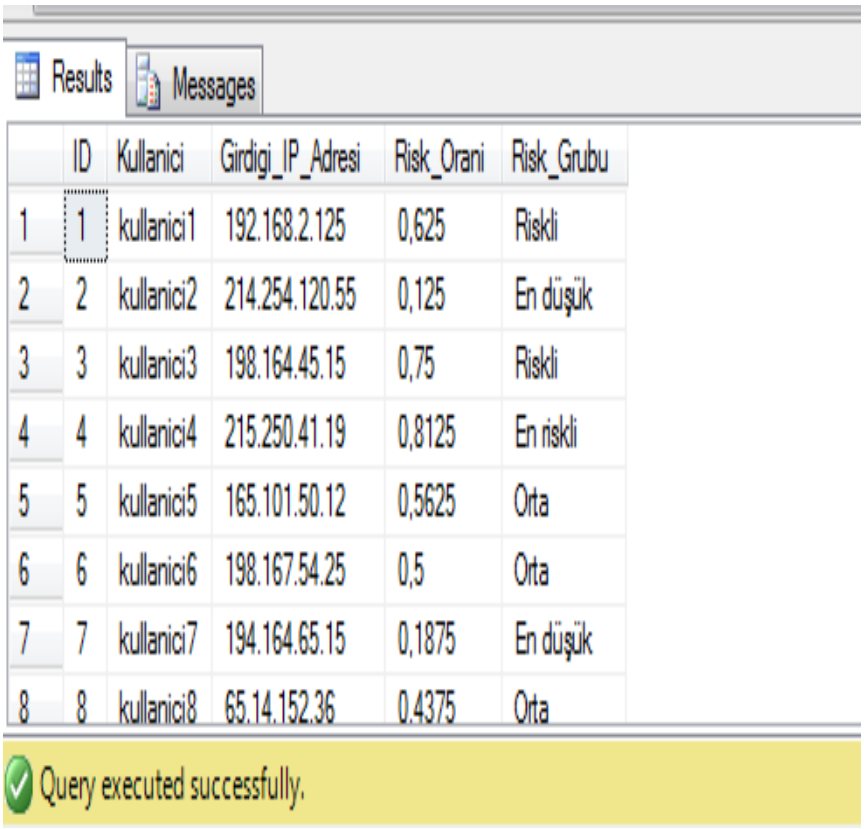
Kurallar oluşturulduktan sonra risk oranının hesaplanabilmesi için eşitlik 2.1 kullanılmıştır:

$$Risk\ Oranı = \sum_{i=1}^{n} (w_{a,i}\ , w_{b,i}, w_{c,i}, w_{d,i}, w_{e,i})/T \qquad (2.1)$$

Bu hesaplamada, $w_{a,i}$ **a** özelliğinin (sisteme giriş sayısı), $w_{b,i}$ **b** özelliğinin (kullanılan veri miktarı), $w_{c,i}$ **c** özelliğinin (kullanıcılara verilen yetki), $w_{d,i}$ **d** özelliğinin (IP adresi), $w_{e,i}$ ise **e** özelliğinin (sistemi yöneten ve gerçekleyen kullanıcıların rolleri) ağırlığını gösterir. **n** sistemde bulunan kullanıcı sayısını gösterirken, **T** ise herbir kriter içerisinde en güçlü ağırlığa sahip kuralların oluşturduğu toplam ağırlığı ifade eder. Yukarıdaki formül hesaplandığında risk oranı 0-1 arasında değerler alır. Bu formüle göre risk sınıflandırılması;

Risk Oranı =
- 0-0.2 ise *en düşük riskli*
- 0.21-0.4 ise *düşük riskli*
- 0.41-0.6 ise *orta riskli*
- 0.61-0.8 ise *riskli*
- 0.81-1.0 ise *en riskli*

olmak üzere kullanıcılar 5 grupta sınıflandırılır. Riskli kullanıcılar ileride oluşabilecek saldırılara karşı önlem için Şekil 2.4'deki gibi veritabanına aktarılmıştır.




**Şekil 2.4** Kullanıcıların risk durumları

## 2.8 Sonuç

Bizim çalışmamızda riskli kullanıcı ve kullanıcı gruplarını belirleyen bir risk analizi yapılmıştır. Kullancıların çeşitli kurallara göre risk oranları hesaplanmıştır ve bu orana göre kullanıcılar sınıflandırılmıştır. Olası yapılabilecek saldırıların önlenmesi başlıca hedefimizdir. Böylelikle kurum ve kuruluşlarda güvenlikle alakalı çalışan personele yardımcı olabilecek bir model geliştirilmiştir. İlerde daha fazla krite ele alınarak çalışmamızın eksik yönleri giderilmesi hedeflenmiştir.

Ayrıca kötü niyetli kullanıcılar veritabanı güvenliğini tehdit edebilecek unsurlar oluşturduğundan kullanıcılar arasında bilgi akış denetiminin sağlanması gerekmektedir. Bu sebeple tezin diğer bir aşamasında veri sızıntısına sebep olmamak için bilgi akış denetimini sağlayacak bir model oluşturduk. Bilgi akış denetimini dağıtık veritabanlarında gerçekleştirdik. Bu modeli diğer bölümlerde açıklayıcı olarak verdik.

# 3
# DAĞITIK ORTAM

Bir dağıtık hesaplama ortamı, graf veri yapısı ile modellenebilir. Graf, düğümler ve bu düğümleri birbirine bağlayan kenarlar kümesinden oluşan bir veri yapısıdır[14]. Eğer G graf ise, bunun tanımı;

G = (V,E)

V(G) = {v1, v2, …, vN}

E(G) = {e1, e2, …, eM} olmak üzere E⊆VxV'dir.

Dağıtık ortam, saklama (storage), çalışan (worker) ve yayın (dissemination) olmak üzere üç çeşit düğüm ile gerçekleştirilir[15]. Kenarlar, bir veri nesnesinin bir düğümden diğerine geçişini gösterir. Her makine her düğümden bir ya da birden çok düğüm barındırır. Bu sayede, düğümden düğüme veri gönderimi (fonksiyon/hesaplama) gerçekleştirilir. Farklı düğümler ile veri ve fonksiyon gönderimi daha hızlı bir şekilde yapılır. Ayrıca çoklu düğümlerde nesneler üzerinde okuma ve yazma işlemleri gerçekleştirilir.

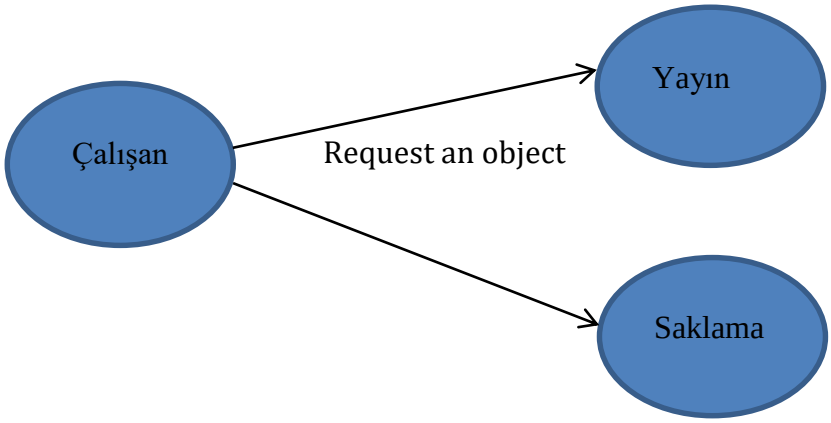
- Veri gönderimi (data shipping) verinin bulunduğu konumdan (düğüm) hesaplamanın yapıldığı konuma (düğüm) gönderilmesidir. Kopyalanmış nesnelerin değerlerinin de hesaplanması için veri iletilir. Tüm işlemler istemcide gerçekleştirilir.
- Fonksiyon gönderimi (function shipping) hesaplamaların bulunduğu düğümden verinin bulunduğu düğüme gönderilmesidir. Tüm işlemler ve sorgular sunucuda gerçekleştirilir.

### 3.1 Saklama Düğümü

Nesneleri kalıcı bir şekilde saklar. Çalışan düğüm veya yayın düğümü kendisinden nesne istediğinde nesnenin kopyalanmasını sağlar. Çalışan düğümün ve yayın düğümün nesneleri almasına yetkisi olup olmadığına bakar(güvenlik ve gizlilik

politikası ile). Aynı zamanda çalışan düğümde nesne güncellemelerinin yapılıp yapılmadığını kontrol eder(bütünlük politikası ile).

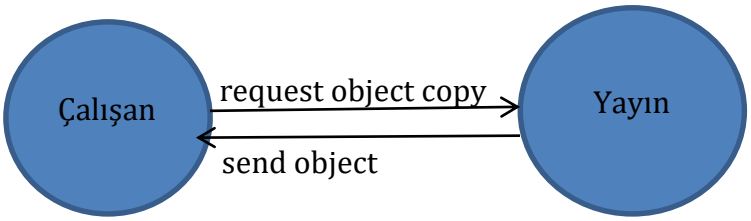
Şekil 3.1'de okuma ve yazma yapabilmek için çalışan düğüm saklama ve yayın düğümden nesne isteğinde bulunur. Bu isteği gerçekleştirmek için saklama düğümü şifrelenerek etiketlenen nesneyi gizlilik politikasına göre inceler. Güvenlik politikasına göre sadece istekte bulunan çalışan düğümün nesneyi görmesine izin verir. Bu iki düğüm arasındaki nesne aktarımı şifresiz yeni düz metin (plaintext) olarak sağlanır. Tüm düğümler birer aktör (principal) olarak kabul edilir. Her saklama düğümü, düğümler arasındaki etkileşimi takip eder. Bu etkileşim aktörler arasındaki nesne yetkilendirmesini gösteren yetki mekanizmasını (delegation mechanism) oluşturur.




**Şekil 3.1** Çalışan, saklama ve yayın düğümü arasındaki nesne aktarımı

Yayın düğümünde bir nesne tutulduğu gibi nesne grupları (object group) bilgisi de tutulur. Nesne grupları birbiriyle ilişkili olan nesneleri ifade eder. Şekil 3.2'de çalışan düğüm yayın düğümden bir nesne üzerinden istekte bulunabilir. Çalışan düğüm yayın düğümden bu isteğinin gerçekleştirilmesini bekler. Bu durumda bu nesneyle ilişkili tüm nesne grupları yayın düğümden aktarılır. Yayın düğümde verilerin nesne grubu olarak tutulmasının nedeni sık kullanılan nesneleri belirleyerek, performansı arttırmak ve maliyeti düşürmektir. Tüm nesne gruplarının erişim denetimi, güvenlik ve gizlilik politikaları aynı olduğundan gizlik ve bütünlüğün sağlanması için aynı davranır. Bir çalışan düğüm bir nesneyi aldığında nesnenin kopyasını kullanarak hesaplamaları gerçekleştirir. Yapılan

transaction işlemleri sonucunda nesnelerin durumunu değiştirerek nesnelerin güncellenmesini sağlar.



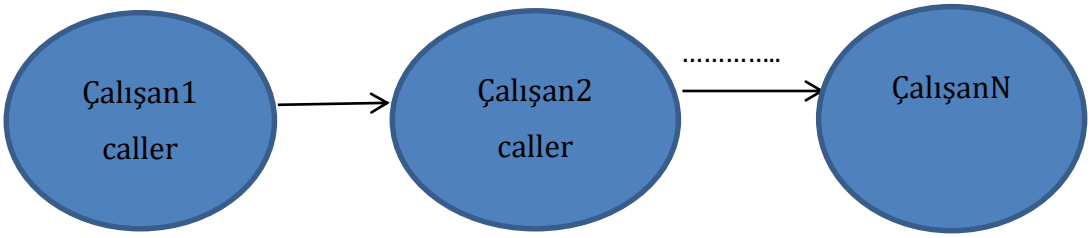

**Şekil 3.2** Çalışan ve yayın düğümü arasındaki nesne aktarımı

## 3.2 Çalışan Düğüm

Görevi programları çalıştırır. Programların çalıştırılması veri ve fonksiyon gönderimi olmak üzere iki şekilde gerçekleştirilir. Nesneleri saklama düğümünden alır. Veri gönderimini (fonksiyon/hesaplama) ve fonksiyon gönderimini gerçekleştirir. Okuma ve yazma işlemleri sonrasında değişen nesnelerin durumunu günceller. Uzaktan çağrı metotları (remote call method) ile fonksiyon hesaplamalarını diğer çalışan düğümlere iletir.

Çalışan ve saklama düğümleri arasındaki veri gönderiminin (data shipping) gerçekleştirilmesi Şekil 3.1'de gösterilmiştir. Bir çalışan düğüm bir saklama düğümünden ya da yayın düğümünden ihtiyaç duyduğu nesneleri almak için istekte bulunur. Saklama düğümü ve yayın düğümü bu nesne ya da nesne gruplarının maliyetine ya da güvenlik politikalarına bakarak çalışan düğüme iletilip iletilmeyeceğine karar verir. Eğer nesnenin maliyeti kabul edilen hesaplama maliyetinden düşükse ya da kendi güvenlik ve gizlilik politikasına uygunsa nesne çalışan düğüme iletilir.

Programlar çoklu düğümlerde dağıtık bir şekilde gerçekleştirildiği için genelde fonksiyon gönderimi tercih edilir. Uzaktan çağrı metotları (remote call method) kullanılarak birden fazla çalışan düğümde fonksiyon gönderimi gerçekleştirilir. Şekil 3.3'de birden fazla çalışan düğümün eş zamanlı ve multithreaded olarak nasıl çalıştığı gösterilmiştir. Çalışan düğüm arayan (caller) ya da aranan (callee) düğüm olabilir. Uzaktan çağrı metodunun diğer bir kullanımı da çalışan ve saklama düğümlerinin aynı yerde ve aynı zamanda çalışmasıdır. Bu şekilde kullanım düşük maliyetli erişimi ve performansı sağlar.

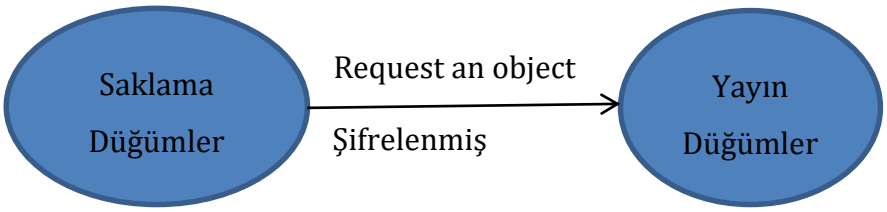



**Şekil 3.3** Çoklu çalışan düğümlerin eş zamanlı olarak istekte bulunması

### 3.3 Yayın Düğüm

Nesneler düğümler tarafından bir anahtar ile şifrelenir(encription). Bu şifrelenmiş nesne güvenlik ve gizlilik politikaları ile çalışan düğümler arasında yayılır. Ölçeklemeyi (scability) sağlamak için, saklama düğümü nesnelerin kopyasını yayın düğüme gönderir. Sık kullanılan popüler veriler nesne grupları halinde halinde saklama düğümü yerine yayın düğümde tutulur. Böylelikle performans sağlanarak maliyet azaltılmış olunur. Çalışan düğümler okuma ve yazma yapabilmek, veri ve fonksiyon gönderimini sağlamak için nesnelerin kopyasını yayın düğümünden ister. Bu istek Şekil 3.4'de gösterilmiştir.

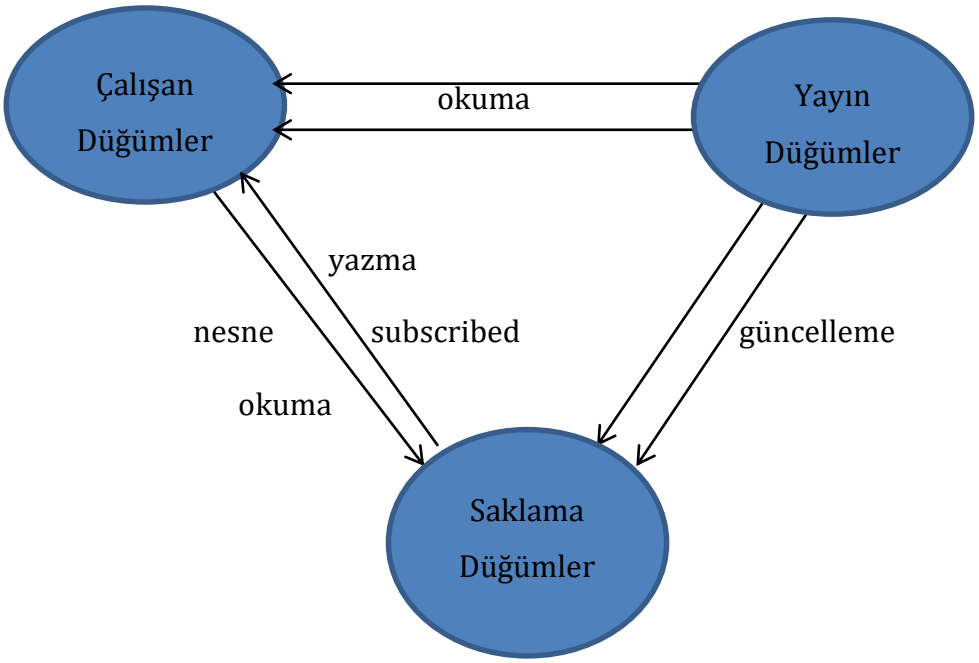
Saklama düğümleri şifrelenmiş nesneleri Şekil 3.4'te gösterildiği gibi yayın düğümden alır. Alınan şifrelenmiş nesnelerin şifreleme anahtarı düğümlerde gözükmez. Bu sebeple şifrelenmiş nesnelerin alınması güvenlik açığına sebep olmaz.




**Şekil 3.4** Nesne gruplarının yayın düğüme iletilmesi

Şekil 3.5'de nesne gruplarının güncellenmesi gösterilmektedir. Nesne güncellemesini sağlamak için çalışan ve yayın düğüm saklama düğümünden nesneyi alır. Böylelikle nesne üzerinde okuma ve yazma işlemlerini, veri gönderimini ve fonksiyon gönderimini gerçekleştirilebilir. Herhangi bir nesne güncellendiğinde saklama düğümü de bu nesneyi günceller. Güncellenen nesne üzerinde çalışan düğüm okuma ve yazma işlemlerini gerçekleştirir. Çalışan düğüm yayın düğümden

nesne üzerinde okuma ve yazma yapabilmek için istekte bulunur. Yayın düğüm bilgi akışını güvenlik ve gizlilik politikalarına göre kontrol eder. Eğer yetki varsa çalışan düğüm nesneyi yayın düğümden alır ve gerekli işlemleri gerçekleştirir[5].



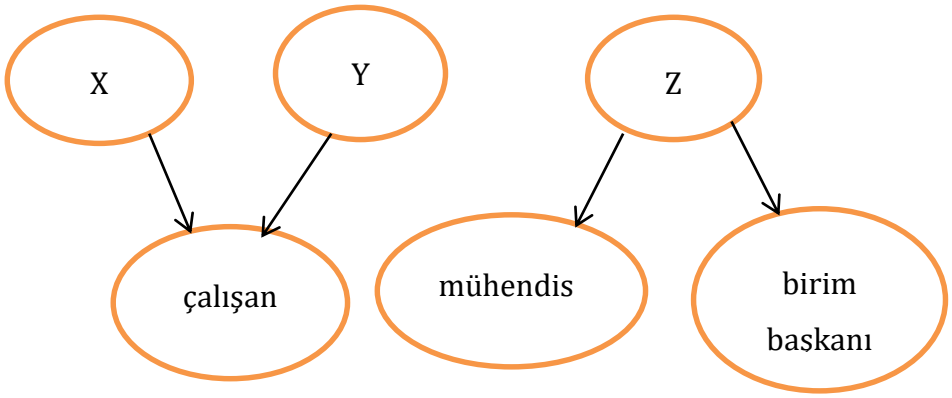

**Şekil 3.5** Bilgi akışında güvenliğin üç çeşit düğüm ile sağlanması

# 4
# DAĞITIK ETİKET MODELİ

Dağıtık etiket modeli farklı aktör, nesne ve etiketlerden oluşur[45].

## 4.1 Aktör

Aktör, veri sahipleri ile veriler üzerinde yetki alma ve verme gibi işlemleri gerçekleştiren kullanıcı ya da kullanıcı gruplarını içerir. Etiket, aktörler tarafından verilen güvenlik politikaları listesinden oluşur. Her aktör veri gizliliği için verilerini etiketler. Yani, her bir veri nesnesi ile eşlenik bir etiket tanımlanır. Ayrıca her aktör ayrı ayrı bu güvenlik politikalarını güvenli bir şekilde değiştirme yetkisine sahiptir [46]. Güvenilir olmayan aktörler ve ortamlar için bu model geliştirilmiştir. Her aktör birbirinden bağımsız şekilde kendi politikasını değiştirerek yeniden etiketleme yapar(relabeling). Güvenli yeniden etiketleme (safe relabeling) yapabilmek için tüm aktörlerin güvenlik politikalarını 'emin' bir işlem ile etiketlemesi gerekir[47].



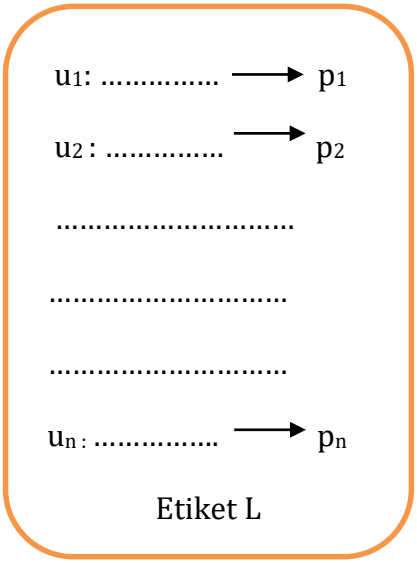

**Şekil 4.1** Aktör hiyerarşisine örnekler

Dağıtık etiket modelinde aktörler sahibi oldukları verileri değiştiren grup ya da rollerden oluşur[48]. Aktörler diğer aktör ya da aktör gruplarına kendi verilerini okumasına izin verir. Bu izin verme işlemi aktör hiyerarşisinde gösterilir. Şekil 4.1'de örnek bir aktör hiyerarşisi gösterilmiştir. Bu şekilde X ve Y çalışan grubunun temsilcileridir. Çalışan Z ise mühendis ve birim başkanı olmak üzere iki yetki ve

görevi vardır. Aktör hiyerarşisindeki izin verme işlemi geçişlidir. Örneğin; X'in Y aktörüne yetki vermesini X→Y şeklinde gösterelim. Eğer X→Y ve Y→Z ise X→Z vardır.

### 4.2 Etiket

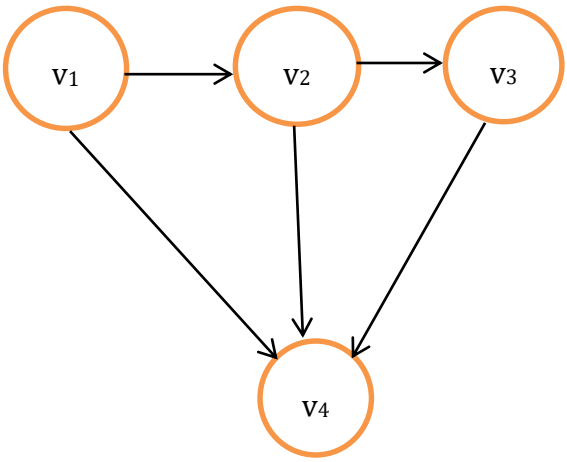
Etiket, bir verinin korunması için oluşturulan politikalar topluluğudur[49]. Şekil 4.2'de bir etiketin içeriği gösterilmiştir. Burada $u_1$, $u_2$,..., $u_n$ sistemde bulunan aktörlerden veri nesnesinin sahiplerini gösterirken; $p_1$, $p_2$,..., $p_n$ yani L etiketindeki herbir içerik tanımı ise ilgili aktörün bu ortak veriye ilişkin güvenlik politikasını göstermektedir. Veri nesnesinin sahibi her aktör etikette kendi politikasını belirler. Bundan sonra, herhangi bir aktör bu veri nesnesini etiketiyle beraber diğer aktörlere gönderir.




**Şekil 4.2** Bir veri nesnesi için etiket örneği

### 4.3 Etiketin Graf ile Modellenmesi

Bir etiket, bir graf ile gösterilebilir. G grafı için belirlenen etiket $L_G$ olsun.




**Şekil 4.3** Etiketi modelleyen bir graf G

Literatürde yapılan daha önceki çalışmalarda [15,16] nesne üzerinde gerçekleştirilen herbir işlem (okuma, yazma) için ayrı bir etiket kullanılmıştır ve sadece okuma ile yazma işlemi yapılmıştır.

Okuma işlemi için, $L_G$={owner:readers} olmak üzere iki kısımdan oluşur. Buradaki owner (veri sahibi) etiketlenen nesnenin sahibi olan aktörleri, readers(okuyucu) ise veri sahipleri tarafından kendisine okuma yetkisi verilen aktörleri göstermektedir.

Yazma işlemi için, $L_G$={owner:writers} olmak üzere iki kısımdan oluşur. Buradaki owner (veri sahibi) etiketlenen nesnenin sahibi olan aktörleri, writers(yazıcı) ise veri sahipleri tarafından kendisine yazma yetkisi verilen aktörleri göstermektedir.

Şekil 4.3'deki graf G ile gösterilen etiketi, $L_G$, yazım biçiminde aşağıdaki şekilde verebiliriz:

$$L_G=\{ v_1:v_2,v_4;\ v_2:v_3,\ v_4;\ v_3:v_4;\ v_4:\ \} \tag{4.1}$$

Bir etiketi oluştururken kullanılan noktalı virgül politikaları birbirinden ayırır. Buna göre, $L_G$ etiketinde {$v_1$:$v_2$,$v_4$}, {$v_2$:$v_3$, $v_4$}, {$v_3$:$v_4$} ve { $v_4$: } olmak üzere dört politika vardır. $v_1$, $v_2$, $v_3$ ve $v_4$ $L_G$ etiketinin ait olduğu veri nesnesinin sahiplerini; $v_2$, $v_3$ ve $v_4$ ise okuyucularını gösterir.

İlk politika;

$v_1 \rightarrow v_1$ , $v_1 \rightarrow v_2$ , $v_1 \rightarrow v_4$ kenarları ile ifade edilmiştir. Bunun anlamı, $v_1$ aktörü $v_1$ ,$v_2$ ve $v_4$ aktörlerine verisini okuyabilmesi için izin veriyor.

İkinci politika;

$v_2 \rightarrow v_2$, $v_2 \rightarrow v_3$, $v_2 \rightarrow v_4$ kenarları ile ifade edilmiştir. Bunun anlamı, $v_2$ aktörü ise $v_2$ ,$v_3$ ve $v_4$ aktörüne verisini okuyabilmesi için izin veriyor.

Üçüncü politika;

$v_3 \rightarrow v_3$, $v_3 \rightarrow v_4$ kenarları ile ifade edilmiştir. Bunun anlamı, $v_3$ aktörü $v_3$ ve $v_4$ aktörüne verisini okuyabilmesi için izin veriyor.

Son politika;

$v_4 \rightarrow v_4$ kenarı ile ifade edilmiştir. Bunun anlamı, $v_4$ kendinden başka kimseye verisini okuma izni vermiyor.

Bir aktörün, bir veriyi okuyabilmesi için; o aktörün etikette veri sahipleri arasında olması ya da tüm politikalarda okuyucular içerisinde bulunması gerekir (ortak okuyucu olarak). Örnek verecek olursak; yukardaki $L_G$ ile etiketlenen veriyi; $v_1$, $v_2$, $v_3$ ve $v_4$ bu etiketin veri sahipleri olduğu için; bu dört politikanın ortak okuyucusu $v_4$ olduğundan, bu veriyi okumaya yetkilidir.

Bir etikette hiçbir satır bulunmuyorsa yani hiçbir politika yoksa tüm aktörler bu veriyi okur. Yani, $L_G = \{\}$ ise bu veriyi tüm aktörler okur.

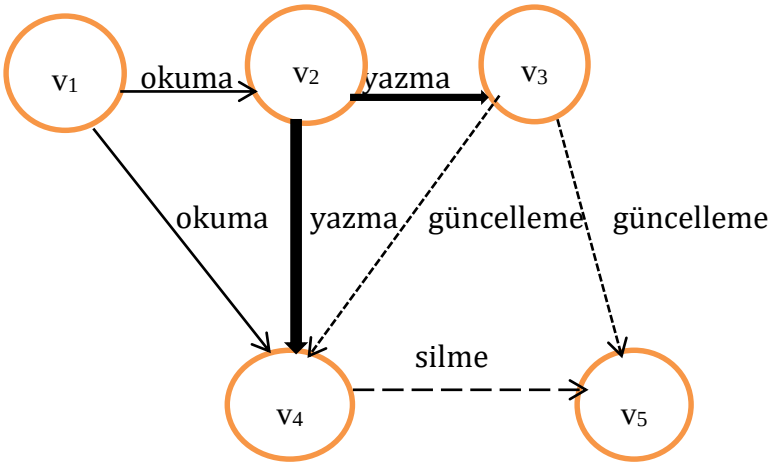
$L_G=\{v_4: \}$ etiketi $L_G=\{v_4: v_4\}$ ile aynı anlama gelir. Etiketlenen verinin okuyucu kümesi yoksa bu veri sadece sahibi olan aktör tarafından okunur.

### 4.4 Önerilen Model

Literatürde yapılan daha önceki çalışmalarda [15,16] nesne üzerinde gerçekleştirilen herbir işlem (okuma, yazma) için ayrı bir etiket kullanılmıştır ve sadece okuma ile yazma işlemi yapılmıştır. Bizim tarafımızdan önerilen çalışmada ise nesne üzerinde gerçekleştirilen tüm işlemler (okuma, yazma, güncelleme, silme) tek etiket kullanarak gerçekleştirilir. Böylelikle tek etikete bakılarak hangi işlem ile aktörler arasında nasıl bir yetkilendirme biçimi olduğu gösterilir.

Bir etiket, bir graf ile gösterilebilir. G grafı için belirlenen etiket $L_G$ olsun. Dağıtık veritabanında hangi işlemin yapılacağı okun şekline göre belirlenir. Okuma, yazma, güncelleme ve silme işlemlerinin her biri için farklı bir ok kullanılır. Böylelikle tek

etiket ile hem daha pratik hem de daha güvenli bir yetkilendirme ve erişim işlemi gerçekleştirilir.




**Şekil 4.4** Modellenen bir graf G

Literatürde [15,16], $L_G$={owner:readers} ve $L_G$={owner:writers} olmak üzere 2 farklı etiket kullanılmıştır. Ancak bizim çalışmamızda, $L_G$ etiketi $L_G$={owner:readers:writers:updates:deletes} olmak üzere beş kısımdan oluşur. Buradaki owner (veri sahibi) etiketlenen nesnenin sahibi olan aktörleri, readers(okuyucu) ise veri sahipleri tarafından kendisine okuma işlemi için yetki verilen aktörleri, writers (yazıcı) ise veri sahipleri tarafından kendisine yazma işlemi için yetki verilen aktörleri, updates (güncelleyici) ise veri sahipleri tarafından kendisine güncelleme işlemi için yetki verilen aktörleri, deletes (silici) ise veri sahipleri tarafından kendisine silme işlemi için yetki verilen aktörleri göstermektedir. Şekil 4.4'deki graf G ile gösterilen etiketi, $L_G$, yazım biçiminde aşağıdaki şekilde verebiliriz:

$$L_G=\{ v_1:v_2,v_4;\ v_2:v_3,\ v_4;\ v_3:v_4,\ v_5;\ v_4:v_5,\ v_5 \} \qquad (4.2)$$

Bir etiketi oluştururken kullanılan noktalı virgül politikaları birbirinden ayırır. Buna göre, $L_G$ etiketinde {$v_1$:$v_2$,$v_4$}, {$v_2$:$v_3$, $v_4$}, {$v_3$:$v_4$, $v_5$}, { $v_4$: $v_5$} ve { $v_5$: } olmak üzere beş politika vardır. $v_1$, $v_2$, $v_3$ ve $v_4$ $L_G$ etiketinin ait olduğu veri nesnesinin sahiplerini; $v_2$, $v_3$, $v_4$ ve $v_5$ ise veri sahipleri tarafından nesne üzerinde çeşitli işlemler (okuma, yazma, güncelleme, silme) için yetki verilen aktörleri gösterir.

İlk politika nesne üzerinde okuma işlemini göstersin:

$v_1 \rightarrow v_1$ , $v_1 \rightarrow v_2$ , $v_1 \rightarrow v_4$ kenarları ile ifade edilmiştir. Bunun anlamı, $v_1$ aktörü $v_1$ ,$v_2$ ve $v_4$ aktörlerine verisini okuyabilmesi için izin veriyor.

İkinci politika nesne üzerinde yazma işlemini göstersin:

$v_2 \rightarrow v_2$, $v_2 \rightarrow v_3$, $v_2 \rightarrow v_4$ kenarları ile ifade edilmiştir. Bunun anlamı, $v_2$ aktörü ise $v_2$ ,$v_3$ ve $v_4$ aktörüne verisini yazması için izin veriyor.

Üçüncü politika nesne üzerinde güncelleme işlemini göstersin:

$v_3 \rightarrow v_3$, $v_3 \rightarrow v_4$, $v_3 \rightarrow v_5$ kenarları ile ifade edilmiştir. Bunun anlamı, $v_3$ aktörü $v_3$, $v_4$ ve $v_5$ aktörüne verisini güncelleyebilmesi için izin veriyor.

Dördüncü politika nesne üzerinde silme işlemini göstersin:

$v_4 \rightarrow v_4$, $v_4 \rightarrow v_5$ kenarı ile ifade edilmiştir. Bunun anlamı, $v_4$ aktörü $v_4$ ve $v_5$ aktörüne verisini silmesi için izin veriyor.

Son politika;

$v_5 \rightarrow v_5$ kenarı ile ifade edilmiştir. Bunun anlamı, $v_5$ kendinden başka kimseye verisi üzerinde hiçbir işlem yapma yetkisi vermiyor.

### 4.5 Aktörler arası Veri Transferi

Bir veri nesnesinin $u_i$ aktöründen $u_j$ aktörüne aktarımı Şekil 4.6'da gösterilmiştir. Bu sırada, $u_j$ aktörünün $u_i$ aktöründen gelen veriyi alıp üzerinde çeşitli işlemler (okuma, yazma, güncelleme, silme) yapabilmesi için, $u_j$ aktörünün bu verinin etiketi olan L'de bir politikanın veri sahibi ya da yetki verilen tüm listeler içerisinde yer alması gerekir. Bunu aşağıdaki koşul ile ifade ediyoruz.

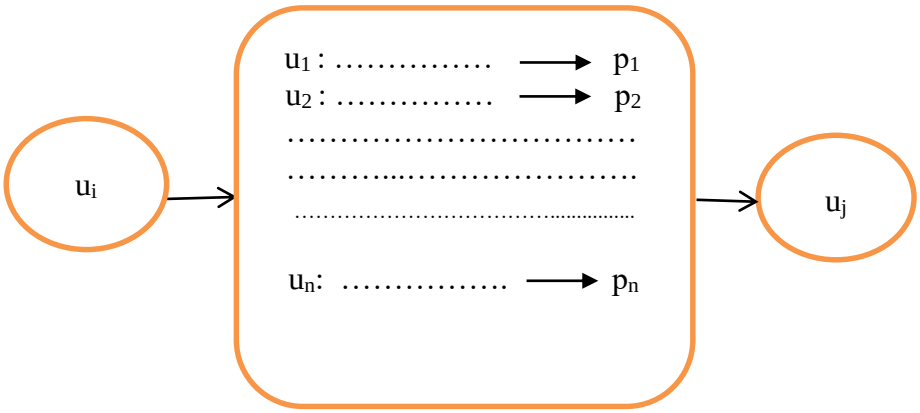
**Veri Üzerinde İşlem Yapma Koşulu:**

$i \neq j$ olmak üzere, $u_j$ aktörünün L etiketli veri transferi koşulunu Şekil 4.5'deki gibi ifade edebiliriz:

```
if {1≤ i ≤ n; ∀i ui ∈ readeri[L], writeri[L], updateri[L], deletei[L]} or {1≤ i ≤ n; ∃i, ui ∈
owneri[L]}
  {
      uj has permission to read,write, update and delete data w/ label L
  }
else
  {
      uj has no permission to read,write, update and delete data w/ label L
  }
```

**Şekil 4.5** Veri transferi kodu

Alınan bir verinin iletilip iletilmeyeceği bu koşul ile denetlenecektir. Eğer bu koşulu sağlamıyorsa, $u_j$ bu veri üzerinde işlem yapamaz. Ancak bir uçtan diğer uca veri nesnesi transferinde aracılık yapmış olur (Şekil 4.6).

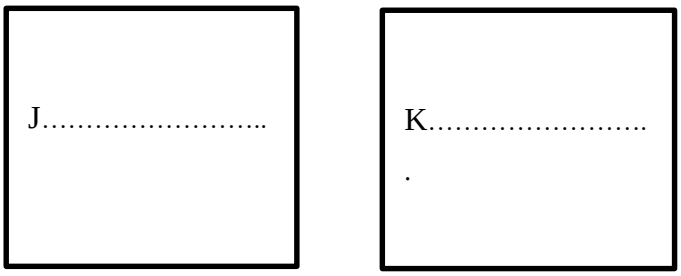



**Şekil 4.6** Aktörler arası veri transferi

### 4.6 Kısıtlama ile Yeniden Etiketleme

Her veri nesnesi bir etikete sahiptir. Bir nesneye yeni bir değer atandığı vakit, bu nesnenin aldığı değeri etiketinde de göstermek gerekir. Eski etiketteki tüm politikalara yeni kısıtlamalar koyarak yeni etiket değeri belirlenir. Nesnenin yeni etiket değeri en az eski etiket değeri kadar kısıtlayıcı olmalıdır. Bu, kısıtlama ile yeniden etiketleme kuralıdır.

$L_1 \subset L_2$ ifadesinin anlamı $L_1$ etiketindeki politikaların $L_2$ etiketindeki politikalara eşit olması gerekir ya da $L_2$ etiketi $L_1$ etiketindeki politikalara ek olarak başka politikalar da içerebilir. Kısacası; $L_2$ etiketi en az $L_1$ etiketi kadar kısıtlama ya da daha fazla kısıtlama içerir.




**Şekil 4.7** Yeniden etiketleme

Şekil 4.7'de **J**, $L_1$ etiketinin politikası, **K** ise $L_2$ etiketinin politikası olmak üzere $L_1$ 'den $L_2$ 'ye yeniden etiketleme ile geçiş yapabilmek için gerekli olan kural eşitlik 4.3'de verilmiştir[48,49]:

$$L_1 \subset L_2 \rightarrow \text{owner}(K)=\text{owner}(J) \text{ ve readers}(K) \subseteq \text{readers}(J) \qquad (4.3)$$

Kısıtlama ile yeniden etiketlemeye örnekler verelim:

$L_1$ (eski etiket) $\subseteq$ $L_2$ (yeni etiket) sağlayan örnekler;

**Örnek 1:** {X:Y,Z} $\subseteq$ {X:Y}. $L_1$ etiketinin Y ve Z aktörleri okuyucuları iken; $L_2$ etiketi Z okuyucusunu kaldırarak verisini sadece Y aktörünün görmesine izin vermiştir.(okuyucu kaldırma ile yeniden etiketleme)

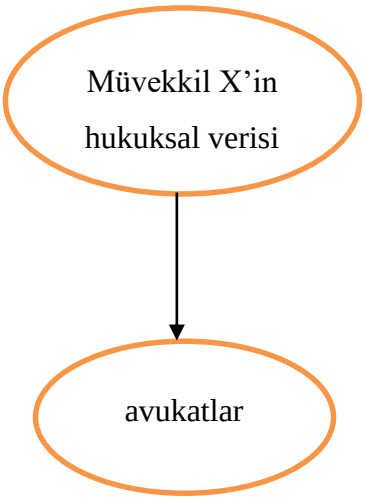
**Örnek 2:** {X:Y}$\subseteq$ {X: ,Z:T}. $L_2$ etiketi Y okuyucusunu kaldırmış ve {Z:T} yeni politikasını ekleyerek kısıtlamayı arttırmıştır.(okuyucu kaldırma ve politika ekleme)

**Örnek 3:** {X:Y,Z}$\subseteq$ {X:Y;X:Z}. $L_1$ ve $L_2$ eşit kısıtlama içerir.

Etiketteki veri sahipleri kendi politikalarını silme ya da okuyucu kaldırma gibi çeşitli kısıtlayıcı işlemler ile verinin yayılımını kontrol eder. Bu yeniden etiketlemenin amacı güvenli declassification gerçekleştirmektir. Veri sahibinin eklediği okuyucu, diğer veri sahipleri tarafından da eklenmesi halinde, bu biçimde yeniden etiketlenen veri, bu okuyucu tarafından okunur[50,51].

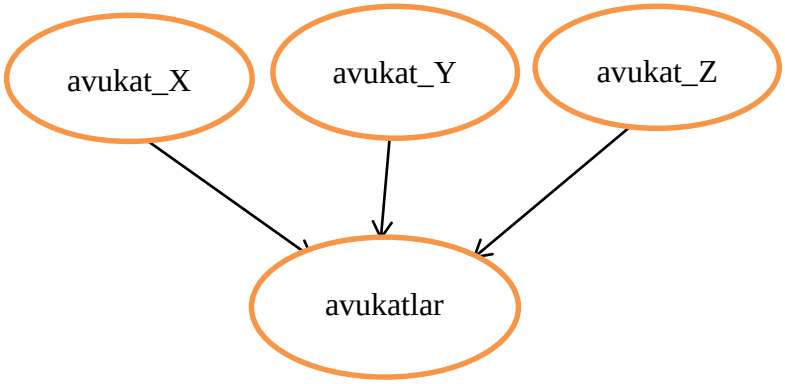
### 4.7 Yeniden Etiketleme Aksiyonları

Veri, declassification ya da downgrading adı verilen üst güvenlik seviyesinden alt güvenlik seviyesine geçerken yeniden etiketlenir. Yeniden etiketleme ile veri daha az kısıtlayıcı ve daha az sıkı hale getirilir.




**Şekil 4.8** Örnek bir aktör hiyerarşisi

Şekil 4.8'deki aktör hiyerarşisi ile müvekkil_X aktörü avukatlar grubuna hukuksal verisini okuması için izin verir.




**Şekil 4.9** Aktör hiyerarşisine örnekler

Şekil 4.9'da bir avukatlık bürosundaki aktör hiyerarşisi gösterilmiştir. Avukatlar aktörü bir grubu oluşturur ve bu grup içerisindeki tüm aktörler (avukat_X, avukat_Y, avukat_Z) bu grubun birer üyesidir. Bu avukatlar bu grubun yetki ve sorumluluğunu kullanır.

Aktör hiyerarşisi göz önünde bulundurarak adım adım 5 farklı yeniden etiketleme aksiyonu aşağıda verilmiştir:

**Aksiyon 1: Okuyucu kaldırma:** Bir etiketten okuyucu kaldırarak etiketlenen veriyi daha kısıtlayıcı hale getirebiliriz. Şekil 4.7'deki aktör hiyerarşişinde müvekkil_X hukuksal verisini avukatlar grubundaki avukat_X ve avukat_Y avukatlarının görüp değerlendirmesi için verisini {müvekkil_X: avukat_X,avukat_Y} ile etiketlemiş olsun. Ancak daha sonra avukat_Y avukatını okuyucu listesinden kaldırarak kendisini sadece avukat_X'in savunmasını isteyebilir. Bu durumu {müvekkil_X: avukat_X} etiketi ile sağlar.

**Aksiyon 2: Politika ekleme:** Bir etikete yeni bir politika ekleyerek veri daha sıkı korunmuş hale gelir. Yani, veri daha kısıtlanmış olur. Şekil 4.7'deki aktör hiyerarşi göz önüne alınırsa {müvekkil_X:avukat_X} ile etiketlenen veriye {avukat_X: avukat_Z} politikası eklenerek elde edilen {müvekkil_X: avukat_X; avukat_X:avukat_Z } yeni etiketi ile veri kısıtlanmış olur.

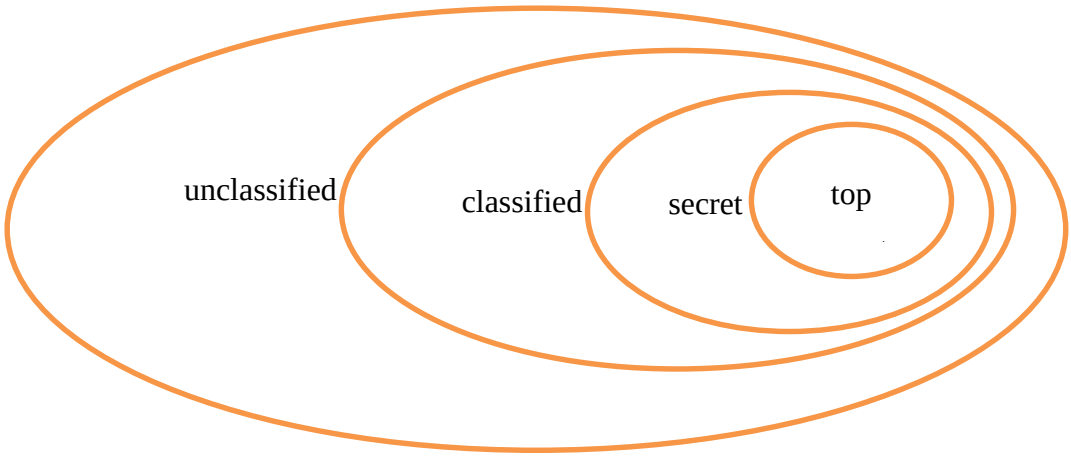
**Aksiyon 3: Okuyucu ekleme:** X veri sahibi, Y ve Z ise X aktörü tarafından kendisine yetki verilen aktörler olsun. Y ve Z aktörleri etikete okuyucu eklenerek veri yeniden etiketlenir. Bir verinin sahibi tarafından vekalet verilen aktörlerin etikete okuyucu olarak eklenmesi ile verinin yeniden etiketlenmesi olayıdır. avukat_X, avukat_Y ve avukat_Z avukatlar grubunun üyeleridir ve bu grup adına hareket ederler. {müvekkil_X: müvekkil_X, avukatlar} etiketi ile müvekkil_X'in hukuksal verileri avukatlar grubunda bulunan tüm avukatlar tarafından görülür. Ancak müvekkil_X kendisini avukat_X'in savunmasını isteyebilir. Bu sebeple eski etikete avukat_X okuyucu olarak eklenerek verilerin sadece avukat_X tarafından görülmesi sağlanır. Yeni oluşan {müvekkil_X: müvekkil_X,avukat_X} etiketi ile veri eski etikete göre daha kısıtlanmış hale gelir.

**Aksiyon 4: Veri sahiplerini değiştirme:** Bir verinin sahibi tarafından kendisine vekalet verilen aktörler etiketlenen veriyi declassification yapabilir. müvekkil_X→avukat_X vekalet verme işlemi Şekil 4.7 ve Şekil 4.8'deki aktör hiyerarşisinde gösterilmiştir. müvekkil_X aktörü avukat_X aktörüne vekalet verdiğinden {müvekkil_X: müvekkil_X} ile etiketlenen verinin sahibi avukat_X olarak

değiştirilebilir. {avukat_X: müvekkil_X} yeni etiketi ile veri yeniden etiketlenmiş olur.

**Aksiyon 5: Kendi kendine otorite verme:** L etiketi {X:Y} olsun. X veri sahibi Y ise bu etiketin okuyucusudur. Ayrıca X aktörü tarafından Z ve T aktörlerine vekalet verilsin. Bu vekalet verilen aktörler {X:Y} ile etiketlenen veriye okuyucu olarak eklenir. Böylelikle yeni oluşan {X:Y,Z,T} etiketi ile veri daha kısıtlanmış olur. Bir avukatlık bürosunda müvekkilin yetki verdiği avukatlar yeni etikete okuyucu olarak eklenerek müvekkili temsil edebilir. {müvekkil_X:avukat_X} etiketi ile müvekkil_X avukat_X'e kendi davasının yürütmesi için yetki verir. Şekil 4.7 ve Şekil 4.8'deki aktör hiyerarşisi göz önünde tutulursa müvekkil_X avukatlar grubundaki tüm avukatlara vekalet verir. Eski etikete avukat_Y ve avukat_Z okuyucu olarak eklenerek yeni oluşan {müvekkil_X:avukat_X, avukat_Y , avukat_Z} etiketi ile veri daha kısıtlanmış hale gelir.

Yukarıda verilen yeniden etiketleme kuralları herhangi bir proses ile gerçekleştirilebilir. Çünkü bu kurallar güvenlik seviyesini bozmadan yeniden etiketlenir. Örneğin; çok seviyeli güvenlik modelinde her kullanıcı kendi verilerini korumak için unclassified, classified, secret ve top secret olmak üzere kendi verisini bu sınıflardan birine dahil eder ve yeniden etiketleme yapar. Çok seviyeli güvenlik sınıfları Şekil 4.10'da gösterilmiştir. Bu dört güvenlik sınıfı aktör olarak ifade edilebilir. Herbir güvenlik sınıfındaki herbir aktör kendi sınıfı adına hareket eder ve bu aktörler kendi sınıfındaki tüm prosesleri görebilir. Yeniden etiketleme çalışma anındaki aktör hiyerarşisine bağlıdır.




**Şekil 4.10** Çoklu güvenlik sınıfları

# 5
# DAĞITIK ETİKET MODELİ UYGULAMA ÖRNEKLERİ

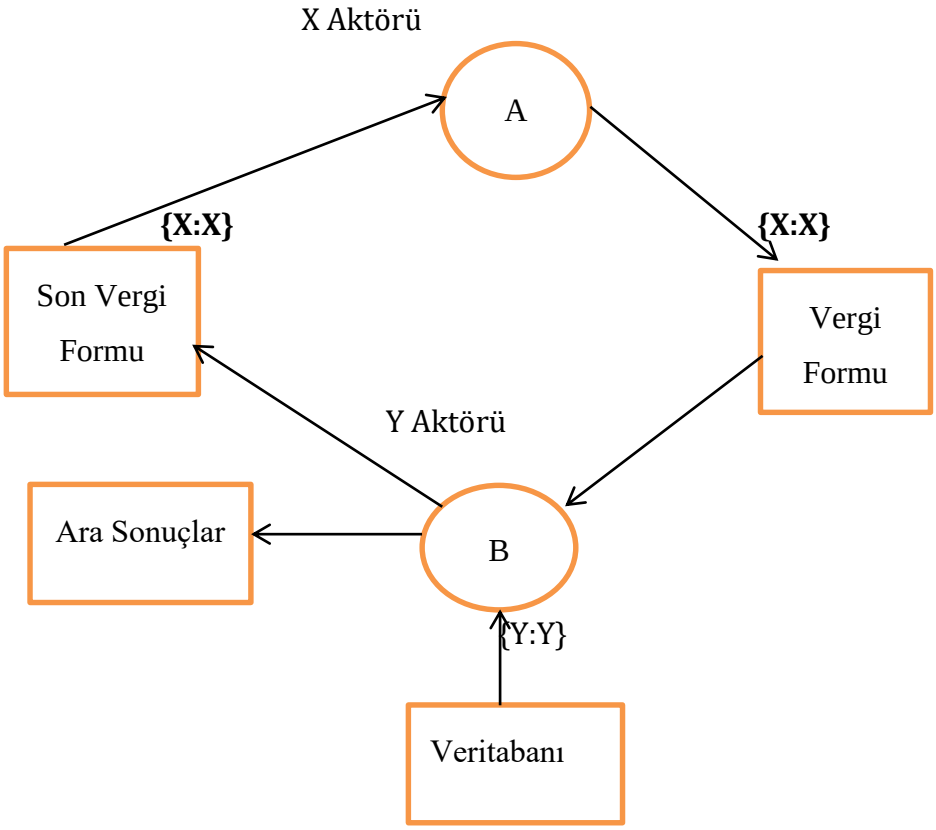
Vergi ve banka olmak üzere iki problemin dağıtık etiket modeli kullanılarak çözümü aşağıda ele alınmıştır:

## 5.1 Vergi Örneği

X ve Y karşılıklı olarak birbirine güvenmeyen iki aktör olsun. X aktörü A programına güveniyor ve tüm yetkilerini bu programa verirken; Y aktörü ise B programının dağıtıcısı olduğundan bu programa güvenir. X aktörü vergi verisini A programından B programına göndererek son vergi formunun B programı ile hesaplanılıp hazırlanmasını istiyor. Aynı zamanda da Y aktörüne güvenmediği için kendi veri gizliliğinin korunmasını istiyor. Bu durumda güvenilir olmayan ortamlarda her iki aktörün de kişisel bilgilerini koruyacak bir yönteme ihtiyaç duyulur. B programı X aktörünün vergi formunu hesaplaması için bir veritabanı kullanır. Bu veritabanında vergi ödemelerini hesaplayan çeşitli veriler ve algoritmalar vardır.

Şekil 5.1'de vergi hazırlama örneğinin etiketleme modeli ile çözümü gösterilmiştir. X ve Y aktörleri verilerini şekilde görüldüğü gibi etiketler. Burada kare olarak gösterilenler verileri ve veritabanını gösterir. Oklar ise aktörler arasındaki ilişkileri ifade eder. X aktörü {X:X} etiketi ile vergi verisini sadece kendisi okur. Bu etiket ile başka aktörlerin verisini okumasına izin vermez. Y aktörü ise {Y:Y} etiketi ile verisini okur. B programı {X:X} etiketini silmeden X aktöründen veriyi alır ve vergisini kendi özel veritabanını kullanarak hesaplar. Çünkü B programı etiketleme modelinde bu politikayı silemez. Y aktörü veritabanını {Y:Y} ile etiketlediği için bu veritabanına kendisi dışında başka aktörlerin ulaşmasına izin vermiyor. X aktörünün vergi hesaplamasına ilişkin verisinin ara sonuçları için X ve Y aktörünün verilerinin etiketleri {X:X; Y:Y} şeklinde birleştirilir. Böylelikle X ve Y'nin vergi verisinin ara sonuçlarını okunması engellenir. X, son vergi formunun sonuçlarını okuyabilmek

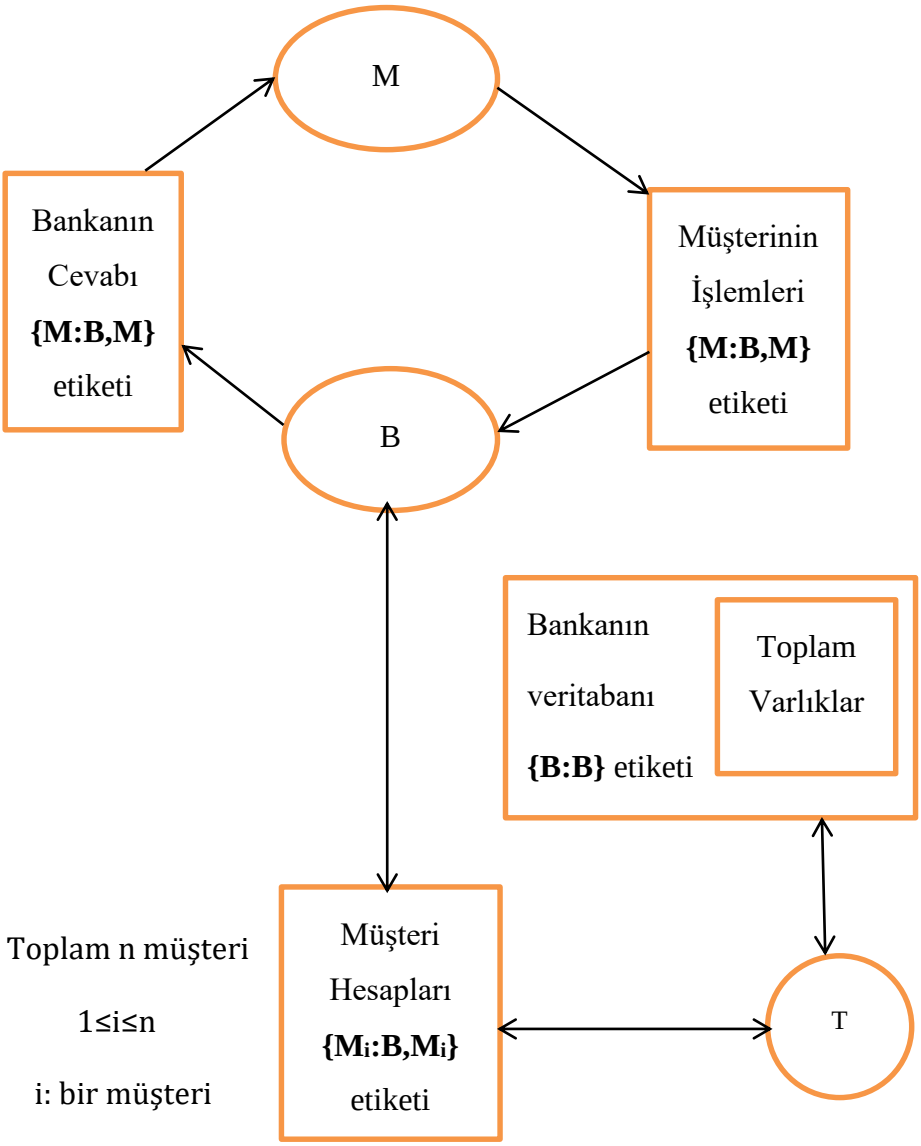
için yeniden etiketlemeye ihtiyaç duyar. Y, B programında {Y:Y} politikasını kaldırarak, declassification yapmış ve X'in bu ihtiyacını karşılamış olur.




**Şekil 5.1** Vergi hesaplatması için etiketler

### 5.2 Banka Örneği

Bir bankanın birçok müşterisi vardır. Her banka müşterisinin mal, para ve yatırım gibi hesap bilgilerini diğer müşterilerden ya da müşteri olmayan aktörlerden saklamak ve korumak zorundadır. Şekil 5.2'de bir bankanın müşteri işlemleri etiketleme modeli kullanılarak gösterilmiştir. Bu şekilde yuvarlak olarak gösterilenler **M** müşteri, **B** banka, **T** ise müşterilerin varlıklarını hesaplayan aktörlerdir. Oklar aktörler arasındaki bilgi akışını, kareler ise veritabanını ve verileri ifade eder.




**Şekil 5.2** Banka müşteri işlemleri için etiketler

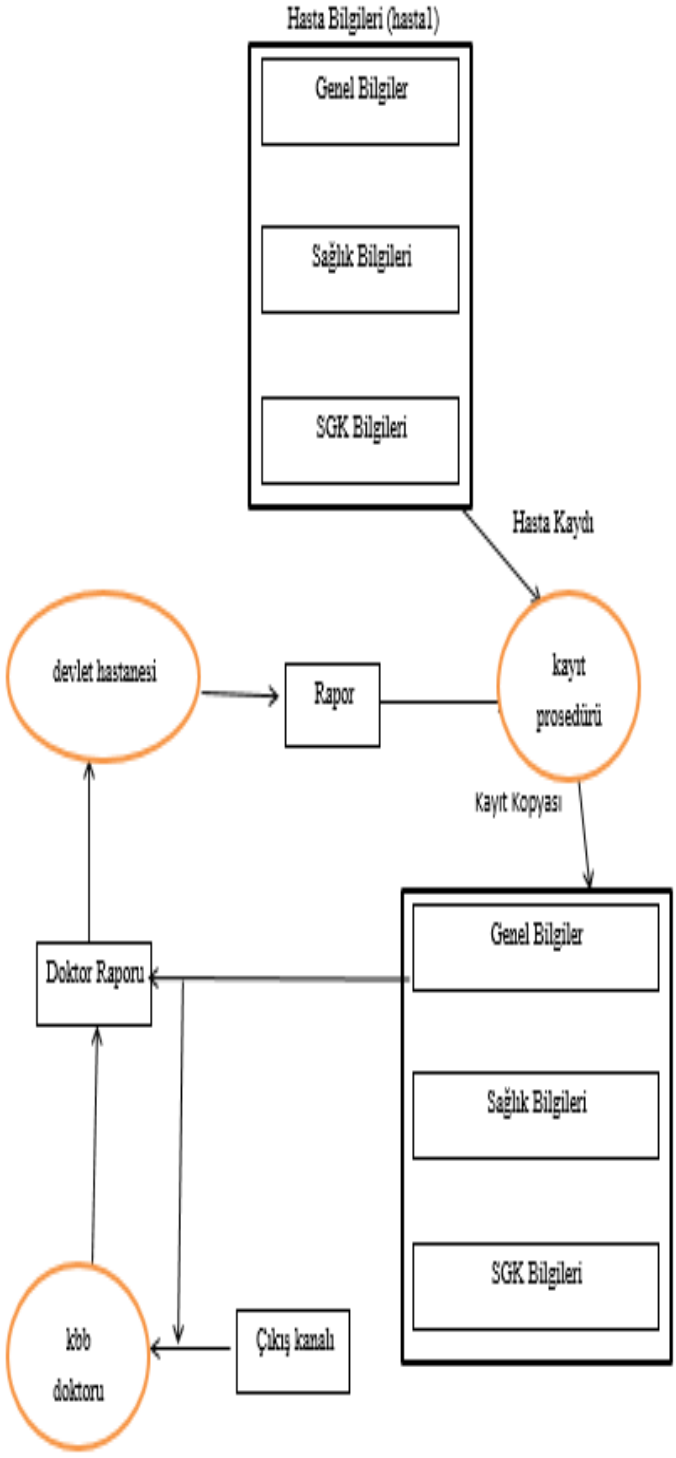
Herhangi bir müşteri **i** (1≤i≤n) varlıklarını **{Mi:B,Mi}** ile etiketleyerek kendi güvenlik politikasını belirler. Ayrıca her müşteri farklı zamanlarda para yatırma, para çekme vs. gibi işlemler yapar. Banka bu işlemleri güvenli bir şekilde gerçekleştirmek zorundadır. Bu sebeple banka bu işlemleri gerçekleştirmek için yapılan tüm müşteri işlemlerini **{M:B,M}** ile etiketler. Bu durumda banka müşterilerin bilgilerini okur. Para çekme, yatırma, havale vs. gibi müşteri işlemleri **T** aktörünce gerçekleştirilir. **T** müşterilerin varlık bilgilerini hesaplayan bir programdır. **T** aktörü her **i** müşterisinin **{Mi:B,Mi}** ile etiketlediği varlık bilgisini "declassify" yapar ve **{B:B}** etiketi ile bankanın veritabanına aktarır. Böylelikle banka bu bilgilerin akışını

kontrol eder ve sistemde bulunan diğer aktörlerin bu verileri okuyamaması için kendi özel veritabanında bu verileri **{B:B}** etiketi ile saklar.

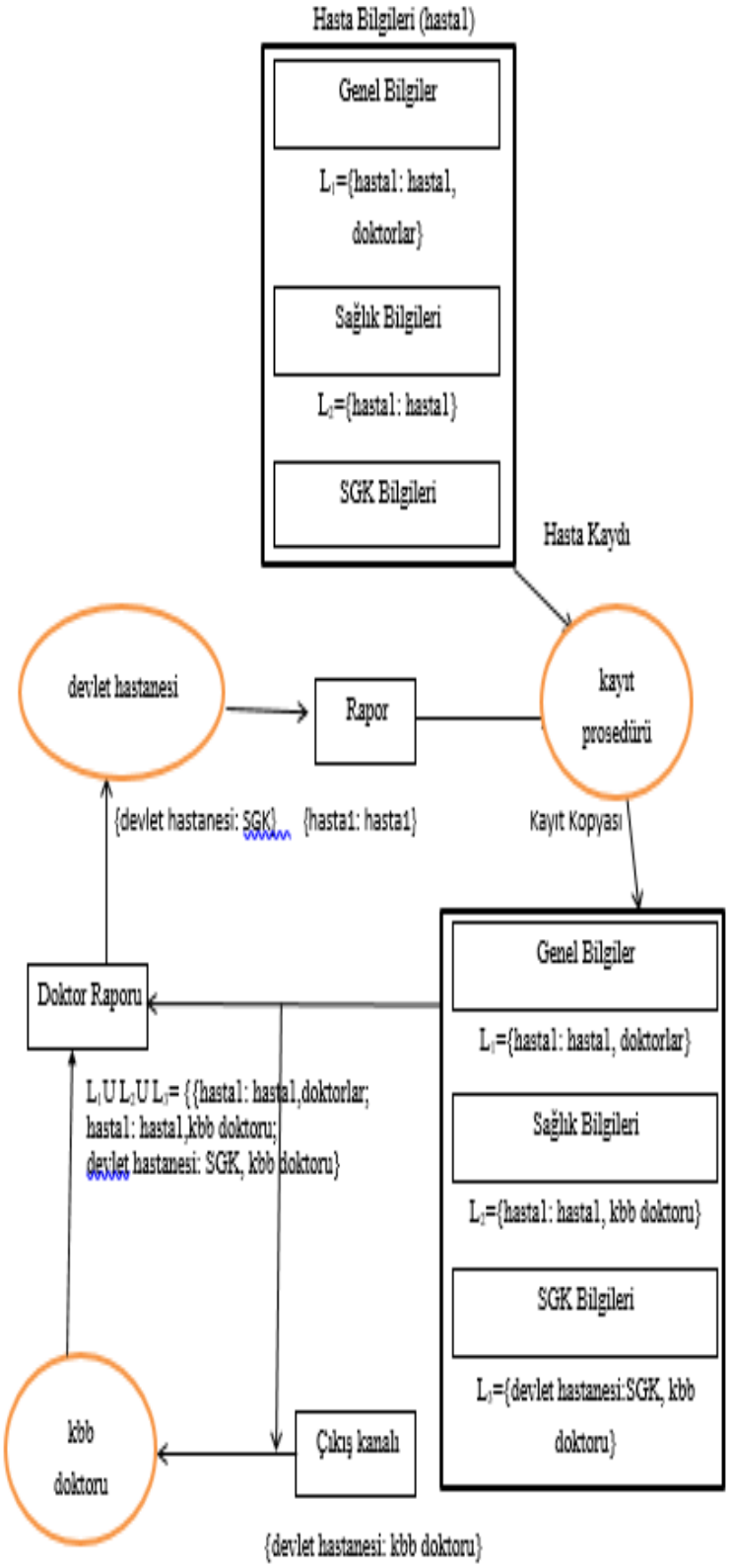
### 5.3 Hastane Örneği

hasta1'in kulak burun boğaz şikayetiyle devlet hastanesine gittiğini varsayalım. Bu hastanın tedavisi için Şekil 5.3'deki bilgi akışları sırasıyla gerçekleşecektir. Hasta bilgilerinin alınması ile ilgili işlemler gerçekleşir. Kayıt prosedürünü takip eden resepsiyondaki görevli hasta1'in tedavi olabilmesi için hasta kaydı açarak hasta1'in bilgilerini sisteme kaydeder. hasta1'in bilgilerinin kbb doktorunun görebilmesi için kayıtların bir kopyası oluşturulur. Böylelikle hasta1'i kbb doktoru muayene etmeye hazırlar. kbb doktoru hasta1'in laboratuvar, kan tahlili, tomografi gibi işlemlerden sonra çıkış kanalında hasta1'in durumunu izler. Kbb doktoru hasta1'in tanı, tedavi, laboratuvar sonuçları ve kullanması gerektiği ilaçları hastanenin veritabanına kaydeder. Ayrıca kbb doktoru hasta1'in muayene sonuçlarını görmesi için de bir rapor hazırlar. hasta1 ile kbb doktoru tarafından yapılan tüm tedavi işlemleri SGK'ya iletilir.

Örnek olarak; hasta kayıtlarının tutulması ile ilgili Şekil 5.4'deki bir senaryoyu ele alalım: Bu senaryo hastanelerde gizliliğin etiketleme modeli ile olması gereken durumunu gösterir. Amaç kayıtların hızlı ve güvenli bir şekilde paylaşılmasıdır. Bu örnekteki aktör hiyerarşileri Şekil 5.5 ve Şekil 5.6'da gösterilmiştir. Şekil 5.3'de yuvarlak olarak gösterilenler aktörleri, kare olarak gösterilenler ise veri ya da veritabanını ifade eder. Verilen bu örnek, ortak bir bilgi nesnesi ve buna katkı yapan aktörleri göstermektedir. Başlıca aktörler hasta, kayıt prosedürü danışanı, doktorlar, SGK ve SGK'ya bağlı olan devlet, özel, üniversite ve şehir hastaneleridir. Yani, bahsedilen aktörler bu bilgi nesnesine ortaklaşa sahiptir diyebiliriz. Bir aktör diğer aktörden hasta ile ilgili bilgilere erişmek istediğinde erişmek istediği bilgileri yerel güvenlik politikalarına göre alır ve gönderir.

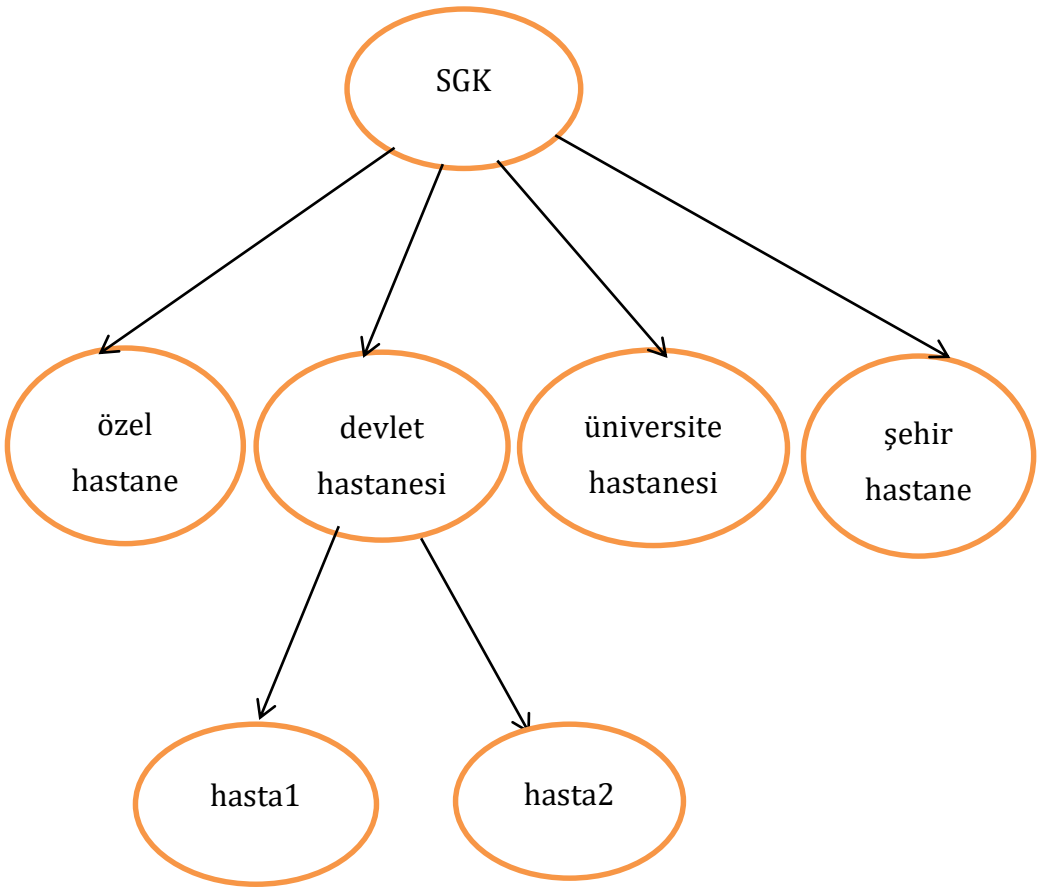



**Şekil 5.3** Bir hastanede hastanın geçtiği sağlık döngüsü




**Şekil 5.4** Bir hastanede aktörlerin kullandığı etiketler

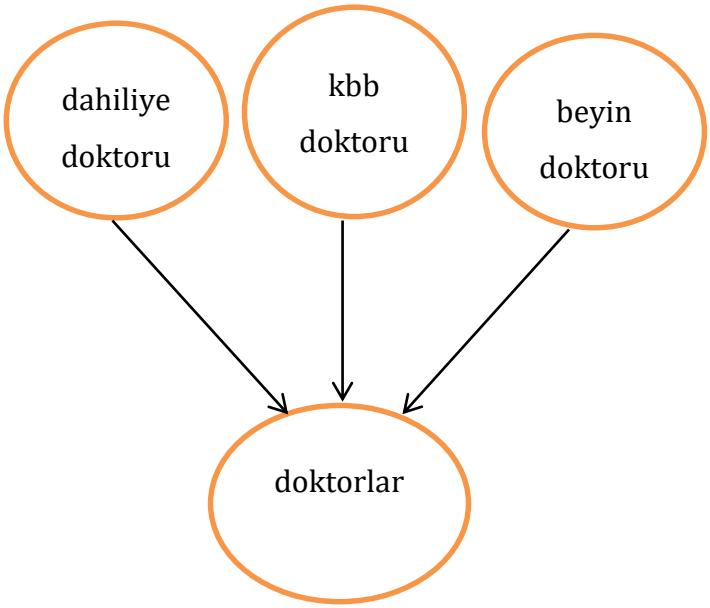
Şekil 5.3'de hasta verileri genel, sağlık ve SGK bilgileri olmak üzere üç kısma ayrılmıştır. Genel bilgiler hastaların ad, soyad, TC Kimlik numarası, doğum tarihi, doğum yeri, kan grubu, cinsiyet, tel, adres gibi kimlik bilgilerini; sağlık bilgileri tanı, tedavi süreci, geçmiş sağlık bulguları, kullandığı ilaçlar, laboratuvar raporları, radyoloji raporları, geçirdiği ameliyatlar, kronik rahatsızlıklar, bulaşıcı hastalıklar, gebelik durumu vs. gibi sağlıkla ilgili kişiye özel kalması gereken bilgilerini; SGK bilgileri ise ödeme bilgileri, hastaların sigortalılık durumları, sigortalı ise hangi tür sigorta olduğu, sigortanın kapsadığı hastalıkları içerir. SGK bünyesinde ise hastaların tanı ve tedavi için gittikleri devlet, özel, üniversite ve şehir hastaneleri vardır.




**Şekil 5.5** Hastane örneği için aktör hiyerarşisi

Şekil 5.5'de hastane örneği için geliştirilen aktör hiyerarşisi gösterilmiştir. SGK bünyesinde hastaların tanı, tedavi için gittikleri özel, devlet, üniversite ve şehir hastaneleri vardır. hasta1 ve hasta2 ise muayene için devlet hastanesinden

yararlanan hastaları göstersin. SGK en büyük otoriteye sahiptir ve kendi bünyesinde bulunan hastanelere vekalet verir.




**Şekil 5.6** Hastane örneği için doktorlar aktörü hiyerarşisinin bir kısmı

Hastane örneği için görev yapan doktorlar arasındaki aktör hiyerarşisinin bir kısmı Şekil 5.6'da gösterilmiştir. Ortopedi, dahiliye, kbb, beyin ve göz doktorlar grubunun aktörlerini gösterir. Bu aktörler doktorlar grubunun üyeleridir ve bu grup adına hareket ederler.

Bu modelde hasta1'in genel, sağlık ve SGK'ya ait verileri Şekil 5.4'deki görüldüğü gibi sırasıyla $L_1$ , $L_2$ ve $L_3$ ile etiketlendirilmiştir. hasta1 kopyalanmış verisinde $L_2$ ve $L_3$ ile etiketlenen verilerine kbb doktoru okuyucu olarak eklenerek bu doktorun kendi verilerini görmesine izin verir. Böylelikle kbb doktoru kopyalanmış veride hasta1'in verisine ulaşarak tanı ve tedavisini sağlar. Çıkış kanalında ise sadece kbb doktoru hasta1'in kayıtlarını görebilmesi için $L_3$ etiketinde okuyucu kaldırılarak yeniden etiketleme yapılır. {devlet hastanesi: kbb doktoru} yeni etiket ile çıkış kanalına kbb doktoru dışında hiçbir doktor ve çalışan erişemez ve değişiklik yapamaz.

Hastanın genel, gizli ve SGK'ya ait bilgileri gösteren üç etiket kbb doktorunun raporunda birleştirilir ($L_1UL_2UL_3$). Hastanın genel bilgileri bu etiket ile kbb doktoruna açıkça gösterilmez. SGK'nın kullandığı veritabanı ise {devlet hastanesi: SGK} ile etiketlenerek hasta1'in sağlık verileri SGK'ya kaydedilir. Böylelikle bu örnek

için hasta kayıtları ile ilgili veritabanına SGK ve devlet hastanesi dışında hiçbir aktör erişemez. Çünkü hasta1 dışında diğer hastaların verileri de bu veritabanında tutulur. Ayrıca hasta1'in doktoru olan kbb doktorunun tanı ve tedavisi için hazırladığı raporu görebilmesi için hasta verisi {hasta1: hasta1} ile etiketlenir.

### 5.4 Eğitim Örneği

L ile etiketlenen bir verinin sahipleri çoklu politika belirlemişse, bu politikaları ancak tüm okuyucu kümelerinin kesişim kümesindeki okuyucular okur.

K: toplam politika sayısı

i: herhangi bir politika ($1 \leq i \leq K$ olmak üzere)

$oK_i$ : i politikasının veri sahipleri kümesini

$rK_i$ : i politikasının okuyucu kümesini

oK: tüm politikaların veri sahipleri kümesini

rK: tüm politikaların okuyucu kümesini ifade etsin.

Bu örnekte akademik görevle ilgili bir bilgi hocalar kümesindeki aktörler arasında iletilmek istensin.

Aktörler Kümesi={ Yrd.Doç_X, Doç.Dr_Y, Prof.Dr_Z, Arş.Gör_A, Doç.Dr_B, Prof.Dr_C, Arş.Gör_D, Prof.Dr_E, Doç.Dr_F, Yrd.Doç_H} bölümdeki tüm hocalar kümesini göstersin.

Bu bilgi L ile etiketlensin.

L= { Prof.Dr_Z: Yrd.Doç_X , Doç.Dr_Y, Yrd.Doç_H, Prof.Dr_E, Doç.Dr_B;

Prof.Dr_C: Yrd.Doç_X, Arş.Gör_A, Doç.Dr_Y, Doç.Dr_B, Prof.Dr_E;

Doç.Dr_F: Doç.Dr_Y, Prof.Dr_E, Arş.Gör_D, Doç.Dr_B} olsun.

L etiketinin tüm veri sahipleri kümesi eşitlik 5.1'de gösterilmiştir.

$$oK = \coprod_{i=1}^{K} oK_i = \{\text{Prof. Dr\_Z, Prof. Dr\_C, Doç. Dr\_F}\} \quad (5.1)$$

$K_1$ politikası akış kümesi ($X_1$)

- Prof.Dr_Z eğitim verisi Prof.Dr_Z
- Prof.Dr_Z eğitim verisi Yrd.Doç_X
- Prof.Dr_Z eğitim verisi Yrd.Doç_Y
- Prof.Dr_Z eğitim verisi Yrd.Doç_H
- Prof.Dr_Z eğitim verisi Prof.Dr_E
- Prof.Dr_Z eğitim verisi Doç.Dr_B
- $rK_1$={ Prof.Dr_Z , Yrd.Doç_X , Doç.Dr_Y, Yrd.Doç_H, Prof.Dr_E, Doç.Dr_B}

$K_2$ politikası akış kümesi ($X_2$)

- Prof.Dr_C eğitim verisi Prof.Dr_C
- Prof.Dr_C eğitim verisi Yrd.Doç_X
- Prof.Dr_C eğitim verisi Arş.Gör_A
- Prof.Dr_C eğitim verisi Doç.Dr_Y
- Prof.Dr_C eğitim verisi Doç.Dr_B
- Prof.Dr_C eğitim verisi Prof.Dr_E
- $rK_2$={ Prof.Dr_C, Yrd.Doç_X, Arş.Gör_A, Doç.Dr_Y, Doç.Dr_B, Prof.Dr_E }

$K_3$ politikası akış kümesi ($X_3$)

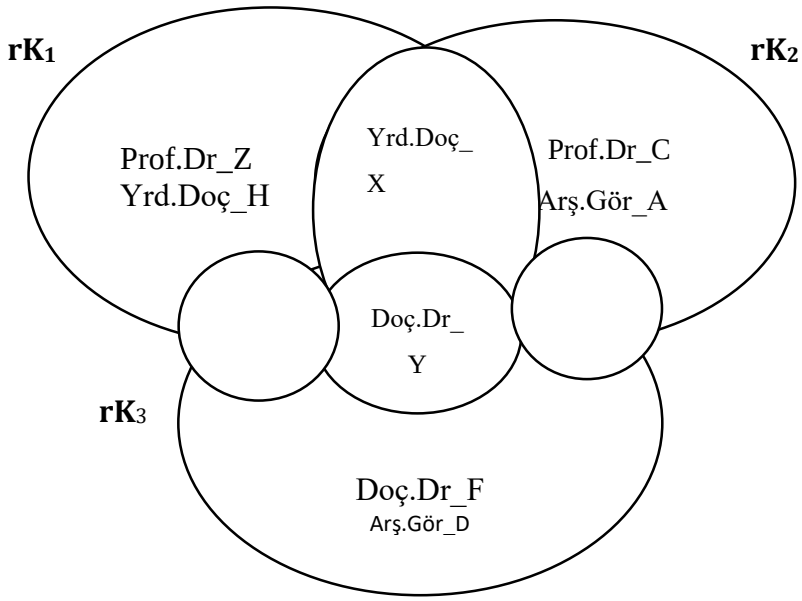
- Doç.Dr_F eğitim verisi Doç.Dr_F
- Doç.Dr_F eğitim verisi Doç.Dr_Y
- Doç.Dr_F eğitim verisi Prof.Dr_E
- Doç.Dr_F eğitim verisi Arş.Gör_D
- Doç.Dr_F eğitim verisi Doç.Dr_B
- $rK_3$={ Doç.Dr_F, Doç.Dr_Y, Prof.Dr_E, Arş.Gör_D, Doç.Dr_B }

L etiketinin tüm okuyucu kümesi eşitlik 5.2'de gösterilmiştir.

$$rK = \coprod_{i=1}^{K} rK_i$$

= {Prof. Dr_Z , Yrd. Doç_X , Doç. Dr_Y, Yrd. Doç_H, Prof. Dr_E, Doç. Dr_B,

Prof. Dr_C , Arş. Gör_A, Doç. Dr_F, Arş. Gör_D} (5.2)




**Şekil 5.7** $rK_1$, $rK_2$ ve $rK_3$ okuyucu kümelerinin kesişimi

Şekil 5.7'de L etiketi ile etiketlenen verinin $rK_1$, $rK_2$ ve $rK_3$ okuyucu kümeleri verilmiştir. Bu kümeye göre $rK_1 \eta rK_2 \eta rK_3$={ Doç.Dr_Y, Prof.Dr_E, Doç.Dr_B} olan tüm okuyucu kümelerinin kesişim kümesindeki okuyucular bu eğitim verisini okur.

# 6
# DAĞITIK ORTAM, AKTÖR VE NESNELER

Veriler nesne olarak ifade edilir ve bu nesnelere çeşitli isimler verilir. Bir nesne oid ile gösterilir ve bu nesnenin kimliğini belirtir. Oid; nesnenin saklandığı yeri gösteren host ve host üzerinde kaçıncı nesne olduğunu, yani nesne numarasını gösteren kısımlardan oluşur.

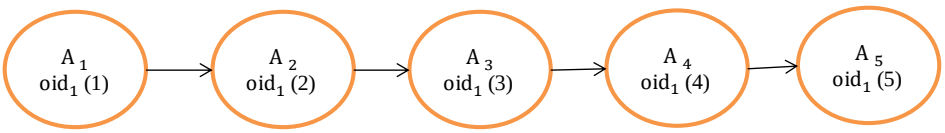
## 6.1 Yol Kısaltma

Yol, aktörler (düğümler) tarafından veri nesnelerinin iletilmesini gösteren bir graf yapısıdır. Düğümler zincir şeklinde birbirine bağlanır. Bir nesne aktörler arasında hareket ettikçe nesnenin kimliğini gösteren nesne belirteci (oid), nesnenin bir sonraki adresini içerecek şekilde güncellenir. Örneğin; Şekil 6.1'de bir nesne ($oid_1$) sırasıyla $A_1$, $A_2$, $A_3$, $A_4$ ve $A_5$ saklama düğümlerine aktarılmıştır. Nesne aktarılırken bir önceki düğümde yalnızca adres bilgisi kalır. Nesne yeni düğüme taşınır. $A_4$ saklama düğümünden hareket ettirildiğinde nesne, $A_5$ düğümüne aktarılır ve yeni adres $oid_1(4)$ olarak $A_4$ düğümüne kaydedilir. Nesnenin geçtiği diğer düğümlerde nesnenin adresi (referans) kalır($oid_1(1)$- $oid_1(2)$- $oid_1(3)$- $oid_1(4)$). $oid_1(5)$ ile gösterilen nesne gerçekte A5 nolu düğümde bulunur. $A_5$'teki $oid_1(5)$ ise nesnenin yeni oid değerini gösterir. Nesne hareketleri, bu biçimde düğümler arasında uzun zincirler oluşturması nedeniyle, nesneye erişim maliyeti artar. Uzun zincirleri önlemek için Şekil 6.2'de gösterilen sonucu doğuran yol kısaltma yöntemi kullanılır. Yol kısaltma, kök düğümden başlayarak nesnenin şu anda bulunduğu düğüme kadar yol üzerinde her düğümdeki referansın şimdiki konum adresi ile güncellenmesi işlemidir (Bkz.Algoritma1).

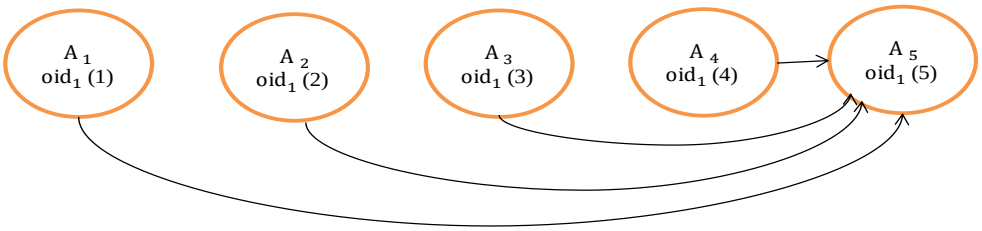
## 6.2 Nesneye Erişim İşlemi Örneği

Yol kısaltma kullanılmadan bir nesnenin yeri bulunmak istendiğinde, o düğüme ulaşıncaya kadar nesnenin geçtiği tüm düğümlere bakmak gerekir. Şekil 6.1'de $A_5$

düğümündeki bir nesnenin konumu $A_1$ düğümüne sorulduğunda, nesne adresinin saklı olduğu $A_1$, $A_2$, $A_3$, $A_4$ ve $A_5$ düğümlerine sırasıyla erişilmesi gerekir. Ancak Şekil 6.2'deki yol kısaltma yöntemi ile doğrudan $A_5$ düğümüne gidilir ($A_1$ $A_5$'deki nesne adresini doğrudan verir). Böylelikle nesneye hızlı erişim mümkün olmakta, erişim maliyeti azalmaktadır.




**Şekil 6.1** Düğüm zinciri oluşumu




**Şekil 6.2** Yol kısaltma sonucu oluşan yeni durum

Yol kısaltma algoritması (Algoritma1:Algorithm YolKısalt), çalışma süresi doğrusal olan etkin bir algoritmadır (Şekil 6.3). Bu, kolayca kanıtlanabilir.

**Hipotez:** Algoritma1'in çalışma zamanı Θ(n)'dir.

**Kanıt:** YolKısalt algoritmasının adımları analiz edildiğinde, çalışma süresi, T(n),

$$T(n) = 2n + c \qquad (6.1)$$

fonksiyonu ile gösterilebilir[6]. Formülde,

T(n) : çalışma süresi

n : zincirde bulunan düğüm sayısı

c : herhangi bir sabit sayı

değerini göstermektedir. Çünkü, adım 1-4 $\mathbf{c_1}$ ile, adım 5-8 **n** ile, adım 9-10 $\mathbf{c_2}$ ile, adım 11-14 **n** ile, adım 15 $\mathbf{c_3}$ ile ifade edilir ve

$$c = c_1 + c_2 + c_3$$

seçilirse bu fonksiyon elde edilir. Bu fonksiyon çözüldüğünde,

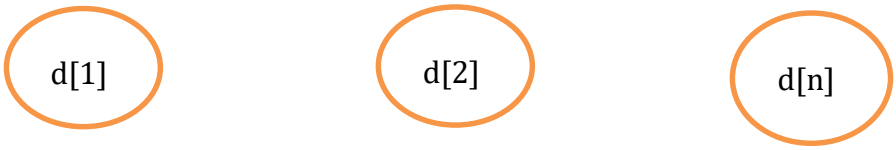
$T(n) = \Theta(n)$ bulunur.

```
Algorithm YolKısalt (Start:Düğüm) //Start başlangıç düğümü
1: X ←Start;
2: Y ←Start;
3: if (X=nill veya next[X]==nill) return;
4: // sondan bir önceki düğümü belirle(Y olarak)
5: while (next[X]!=nill) do
6:       Y←X;
7:       X←next[X];
8: end while
9: // Göstergeleri güncelle
10: Z←Start;
11: while (Z!=X) do
12:     next[Z]←next[Y];
13:     Z←next[Z];
14: end while
15: return;
```

**Şekil 6.3** Yol kısaltma algoritması

# 7
# DAĞITIK ORTAM SİMÜLASYONU VE SONUÇLARI

Dağıtık ortamda nesnelerle ilgili birtakım işlemler yapılır. Örneğin, nesnelere erişim yapılabilir ya da nesneler hareket ettirilebilir. Çalışmamızda, nesnelerle ilgili gerçekleşecek olaylar (object-access, object-move) bağımsız olaylar olarak oluşturulmuş, dağıtık ortam simülasyonu yapılmıştır. Amacımız, yol kısaltma algoritmasının etkinliğini ve avantajlarını göstermektir[55, 56].




d[1] düğümü için

| Nesneler | Adres |
|---|---|
| $o_1$ | 7 |
| $o_2$ | -1 |
| ... | 4 |
| ... | 3 |
| $o_k$ | 2 |

a)

d[2] düğümü için

| Nesneler | Adres |
|---|---|
| $o_1$ | -1 |
| $o_2$ | 3 |
| ... | -1 |
| ... | 5 |
| $o_k$ | 4 |

b)

d[n] düğümü için

| Nesneler | Adres |
|---|---|
| $o_1$ | 2 |
| $o_2$ | 1 |
| ... | 2 |
| ... | 4 |
| $o_k$ | -1 |

c)

**Şekil 7.1** a) a) d[1]'in yerel nesne tablosu, b) d[2]'in yerel nesne tablosu, c) d[n]'in yerel nesne tablosu

Dağıtık bir hesaplama ortamında, **n** tane düğüm, **k** tane nesne olsun. Bu **n** düğüm, örneğin d[1], d[2], …, d[n] şeklinde gösterilsin. Her bir düğümün bir yerel nesne tablosu vardır. **k** tane nesne örneğin $o_1$, $o_2$, …, $o_k$ şeklinde nesneleri göstersin. Her düğümdeki nesne tablosunda, her bir nesnenin, eğer nesne o düğümdeyse bu düğümde bulunur bilgisi ya da başka düğümdeyse nesnenin adresi bulunacaktır. Her bir düğüm için Şekil 7.1'dekine benzer bir gösterim vardır.

Başlangıçta, nesneler düğümlere rastgele atanır. Bu, $1 \leq i \leq k$ olmak üzere F: $o_i \rightarrow d[j]$ fonksiyonu ile gerçekleştirilir (random (1,n)). Her bir oi nesnesi herhangi bir d[j] düğümüne atanır. Nesneler ile ilgili beş tane fonksiyon kullanmayı öngörüyoruz:

**Fonksiyon 1: object_access (i:nesne, j:düğüm):** Bu fonksiyon erişilecek nesne **i** için **j** düğümünün yerel nesne tablosuna bakar. Eğer bu düğümün gösterdiği adres -1 ise nesne bu düğümde bulunuyor demektir. Değilse **j** düğümü nesne tablosunda düğüm adresi alınır ve **j**'ye atanır. Nesne bulunana kadar bu işlem devam eder. Bu fonksiyon zincir uzunluğunu döndürür.

**Fonksiyon 2: object_move (i:nesne , j:kaynak düğüm, x:hedef düğüm):** Bu fonksiyonla nesne **i**, şu an bulunduğu **j** düğümünden **x** düğümüne taşınır. Tüm nesnelerin bulunduğu düğüm genel nesne tablosunda tutulur.

**Tablo 7.1** Genel nesne tablosu

| **Tüm Nesneler** | |
|---|---|
| **Nesneler** | **Bulunduğu Adres** |
| $o_1$ | |
| $o_2$ | |
| … | |
| … | |
| $o_k$ | |

Dağıtık sistemlerde tutulan genel nesne tablosu Tablo 7.1'de gösterilmektedir. Genel nesne tablosu, var olan her bir nesnenin hangi düğümde olduğunu gösterir. Sürekli günceldir. object_access ( i, j ) ve object_move (i, j, x) fonksiyonları birbirinden bağımsız bir şekilde çalışır.

*Bağımlı olay:* Bir olayın gerçekleşmesi başka bir olayın gerçekleşme olasılığını etkiliyorsa diğer bir ifadeyle bir olayın sonucu diğer bir olayın olup olmamasıyla ilgiliyse bu iki olaya bağımlı olay denir.

*Bağımsız olay:* Bir olayın gerçekleşmesi başka bir olayın gerçekleşme olasılığını etkilemiyorsa diğer bir ifadeyle bir olayın sonucu diğer bir olayın sonucuyla hiçbir biçimde ilgili değilse bu iki olaya bağımsız olay denir.

**Fonksiyon 3: break_chain ( i:nesne, j:kaynak düğüm, x:hedef düğüm):** object_access ( i, j) fonksiyonuyla erişilen **i** nesnesinin zincir uzunluğuna bakılır. Zincir uzunluğu, **i** nesnesine erişimde gidilen düğüm sayısı kadardır. Bir eşik değer (T) belirlenir. Zincir uzunluğu eşik değere eşit ya da eşik değerden büyükse **i** nesnesi **j** düğümünden başlayarak nesnenin bulunduğu **x** düğümüne kadar olan zinciri bu fonksiyonla kırar. Eşik değerden küçükse bu fonksiyon çalıştırılmayacaktır. Eşik değer değiştirilerek uygun değerlerin bulunması sağlanacaktır.

**Fonksiyon 4: Book_keeping( i:nesne, L:zincir uzunluğu):** Erişilecek her **i** nesnesi için **L** zincir uzunluğu hesaplanır. Bu fonksiyon ile kayıtların tutulması Tablo 7.2'de gösterilmektedir. Bu simülasyonda, DriverForObjAcess ( ) ve DriverForObjMove( ) fonksiyonları belli oranlarda çalıştırılarak ortalama ve maksimum zincir uzunluğu hesaplanır. Eşik değer değiştirilerek maksimum ve ortalama zincir uzunluğu tekrar bulunur.

**Tablo 7.2** Z= Book_keeping (i: nesne, L: zincir uzunluğu) ile erişilen her i nesnesi için L zincir uzunluğu

| **Nesne(i)** | **7** | **2** | **4** | **1** | **4** | **6** | **...** | **Zmax** |
|---|---|---|---|---|---|---|---|---|
| **Zincir Uzunluğu (L)** | | | | | | | | |

**Fonksiyon 5: compute_ statistics ( ) :** Erişilen **Zmax** tane nesnenin zincir uzunluğunu kaydeder. **Zmax** tane nesneye erişim yapıldığında, yeterli istatistik toplandığı kabul edilerek ortalama zincir uzunluğu hesaplanır. Ortalama zincir uzunluğu, erişilen tüm nesnelerin toplam zincir uzunluğunun erişilen nesne sayısına oranıyla bulunur. Ortalama zincir uzunluğu aşağıdaki eşitlik 7.1'deki gibi hesaplanır.

$$ortalama\ zincir\ uzunluğu = \sum_{i=1}^{Zmax} L[i]\ /Zmax \qquad (7.1)$$

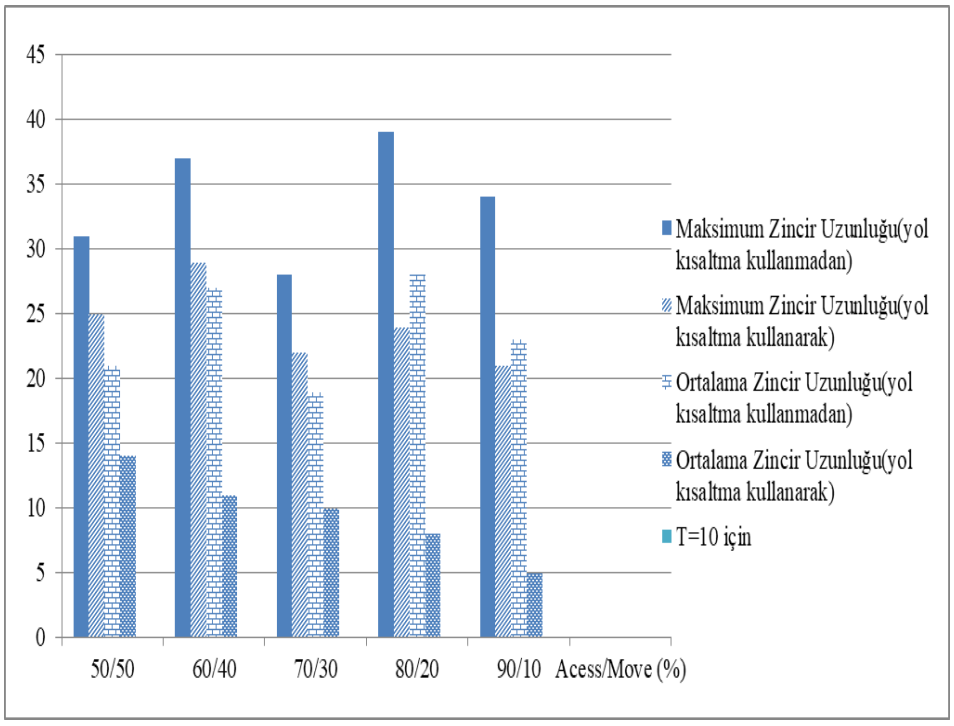
## 7.1 Dağıtık Ortam Simülasyonu Uygulama Sonuçları

Dağıtık ortam simülasyonu binom ve normal dağılım olmak üzere iki dağılım kullanılarak gerçekleştirilmiştir:

### 7.1.1 Binom Dağılımı

Dağıtık ortamda nesnelerle ilgili birtakım işlemler yapılır. Örneğin, nesnelere erişim yapılabilir ya da nesneler hareket ettirilebilir. Çalışmamızda, nesnelerle ilgili gerçekleşecek olaylar (object-access, object-move) bağımsız olaylar olarak oluşturulmuş, dağıtık ortam simülasyonu yapılmıştır. Amacımız, yol kısaltma algoritmasının etkinliğini ve avantajlarını göstermektir. Çalışmamızda nesnelere erişim Access, nesnelerin taşınması Move operasyonuyla gösterilmiştir.

Dağıtık ortamda sistem modellemesi statik ve dinamik olmak üzere iki şekilde gerçekleştirilir. Statik ortamda nesnelere sıklıkla erişim yapılır, nesnelerin taşınması ise daha az yapılır. Dinamik ortamda nesnelerin taşınması işlemi daha fazla, nesnelere erişim ise daha az yapılır. Çalışmamızda statik ve dinamik ortamda nesnelerle ile ilgili işlemler için modelleme yapılmıştır.

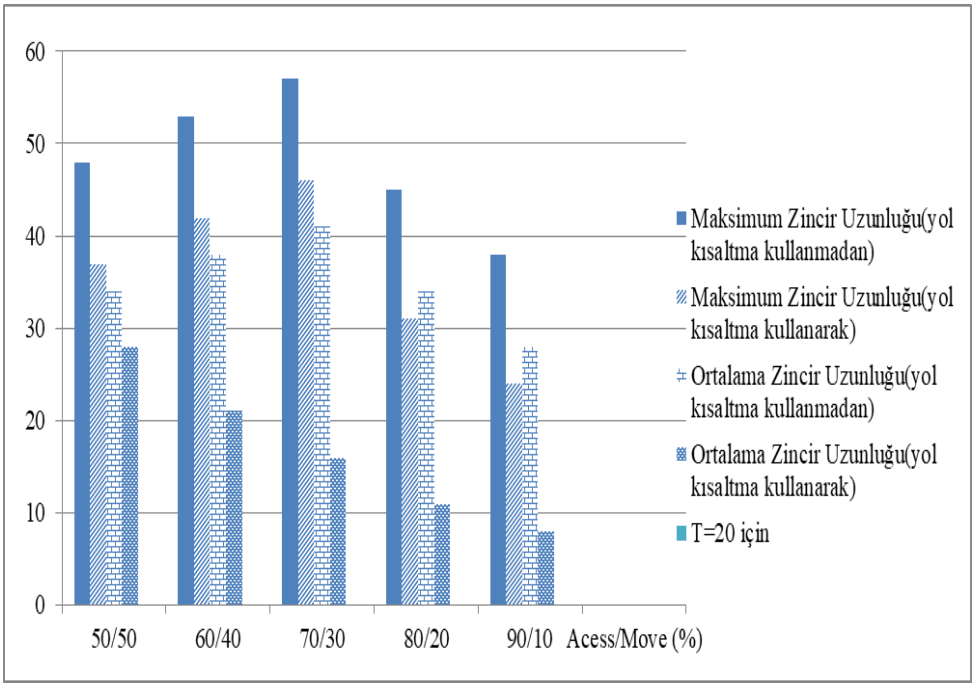



**Şekil 7.2** Binom dağılımı kullanarak T=10 için maksimum ve ortalama zincir uzunlukları

Şekil 7.2' de T=10 için binom dağılımı kullanarak çeşitli Access/Move (%) oranlarına göre dağıtık ortam simülasyonun sonuçları verilmiştir. Grafikte yol kısaltma algoritması kullanmadığımız ve kullandığımız durumlarda maksimum zincir uzunluğu ve ortalama zincir uzunluğu karşılaştırılmalı olarak gösterilmiştir. Bu grafikteki T eşik değerdir ve nesne erişim işleminde (Access) zincir uzunluğu eşik değere eşit ya da eşik değerden büyükse zincir uzunluğu kırılır. Çalışmamızda T değeri değiştirilerek maksimum ve ortalama zincir uzunluğu nasıl/ne yönde değiştiğini gözlemlemek amaçlanmıştır. Grafikteki sütunlar sırasıyla yol kısaltma algoritması kullanmadığımız ve kullandığımız durumlardaki maksimum ve ortalama zincir uzunluklarını gösterir. Grafikte 50/50, 60/40, 70/30, 80/20 ve 90/10 gibi Acess/Move(%) oranları ile nesnelere erişim (Access) miktarı arttırılıp, taşınan (Move) nesne sayısı azaltarak maksimum ve ortalama zincir uzunlukları hesaplanır. Nesnelere erişim miktarını arttırıp yol kısaltma algoritması kullandığımızda ortalama zincir uzunluğu dramatik biçimde kısalmıştır.

Ayrıca binom dağılımı kullandığımızda bazı nesnelere erişim yapılır. Tüm nesnelere erişim yapılmadığı için her nesnenin içinde bulunduğu zincir kırılmaz. Bazı

nesnelerin içinde bulunduğu zincir olduğu gibi kalır. Ortalama zincir uzunluğu ise maksimum zincir uzunluğuna göre daha dramatik olarak azalır. Bu sonuç göstermektedir ki, yol kısaltma algoritması nesnelerin oluşturduğu uzun düğüm zincirini kırarak nesnelere ortalama erişim maliyetini azaltır ve nesnelere hızlı erişim yapılır.

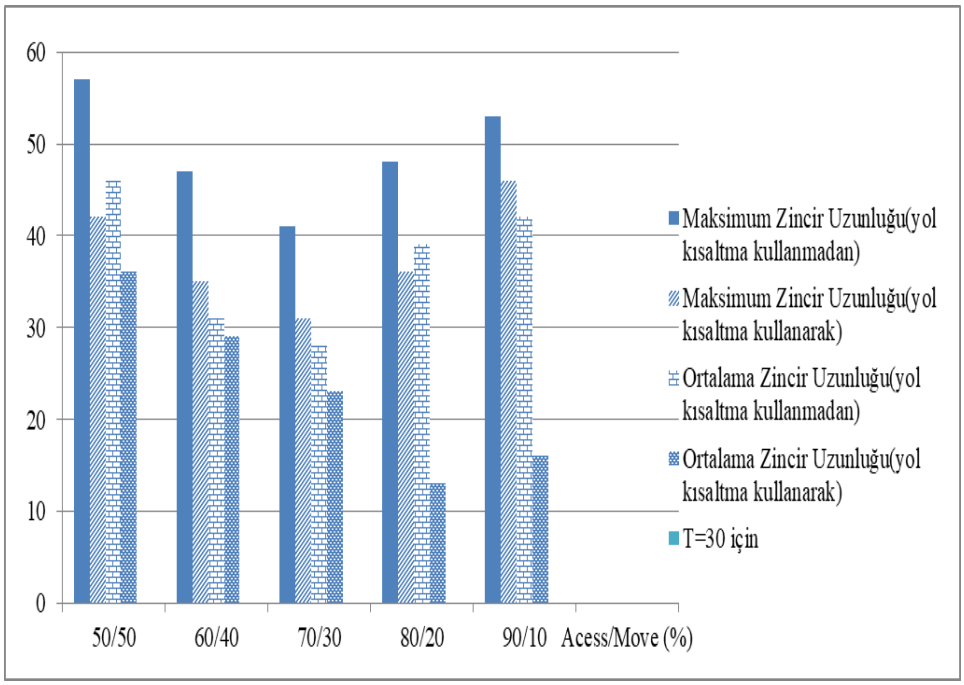



**Şekil 7.3** Binom dağılımı kullanarak T=20 için maksimum ve ortalama zincir uzunlukları

Şekil 7.3' de T=20 için binom dağılımı kullanarak çeşitli Access/Move (%) oranlarına göre dağıtık ortam simülasyonun sonuçları verilmiştir. Grafikte yol kısaltma algoritması kullanmadığımız ve kullandığımız durumlarda maksimum zincir uzunluğu ve ortalama zincir uzunluğu karşılaştırılmalı olarak gösterilmiştir. Nesnelere erişim miktarını arttırıp yol kısaltma algoritması kullandığımızda ortalama zincir uzunluğu dramatik biçimde kısalmıştır.

Ayrıca binom dağılımı kullandığımızda bazı nesnelere erişim yapılır. Tüm nesnelere erişim yapılmadığı için her nesnenin içinde bulunduğu zincir kırılmaz. Bazı nesnelerin içinde bulunduğu zincir olduğu gibi kalır. Ortalama zincir uzunluğu ise maksimum zincir uzunluğuna göre daha dramatik olarak azalır. Bu sonuç göstermektedir ki, yol kısaltma algoritması nesnelerin oluşturduğu uzun düğüm

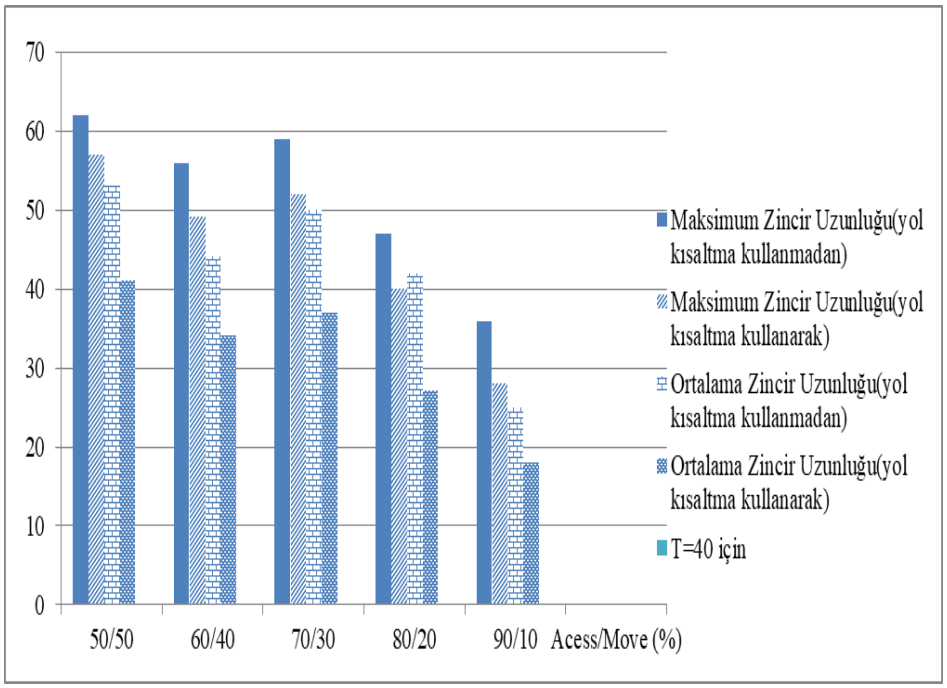
zincirini kırarak nesnelere ortalama erişim maliyetini azaltır ve nesnelere hızlı erişim yapılır.




**Şekil 7.4** Binom dağılımı kullanarak T=30 için maksimum ve ortalama zincir uzunlukları

Şekil 7.4' de T=30 için binom dağılımı kullanarak çeşitli Access/Move (%) oranlarına göre dağıtık ortam simülasyonun sonuçları verilmiştir. Grafikte yol kısaltma algoritması kullanmadığımız ve kullandığımız durumlarda maksimum zincir uzunluğu ve ortalama zincir uzunluğu karşılaştırılmalı olarak gösterilmiştir. Nesnelere erişim miktarını arttırıp yol kısaltma algoritması kullandığımızda ortalama zincir uzunluğu dramatik biçimde kısalmıştır.

Ayrıca binom dağılımı kullandığımızda bazı nesnelere erişim yapılır. Tüm nesnelere erişim yapılmadığı için her nesnenin içinde bulunduğu zincir kırılmaz. Bazı nesnelerin içinde bulunduğu zincir olduğu gibi kalır. Ortalama zincir uzunluğu ise maksimum zincir uzunluğuna göre daha dramatik olarak azalır. Bu sonuç göstermektedir ki, yol kısaltma algoritması nesnelerin oluşturduğu uzun düğüm zincirini kırarak nesnelere ortalama erişim maliyetini azaltır ve nesnelere hızlı erişim yapılır.




**Şekil 7.5** Binom dağılımı kullanarak T=40 için maksimum ve ortalama zincir uzunlukları

Şekil 7.5' de T=40 için binom dağılımı kullanarak çeşitli Access/Move (%) oranlarına göre dağıtık ortam simülasyonun sonuçları verilmiştir. Grafikte yol kısaltma algoritması kullanmadığımız ve kullandığımız durumlarda maksimum zincir uzunluğu ve ortalama zincir uzunluğu karşılaştırılmalı olarak gösterilmiştir. Nesnelere erişim miktarını arttırıp yol kısaltma algoritması kullandığımızda ortalama zincir uzunluğu dramatik biçimde kısalmıştır.

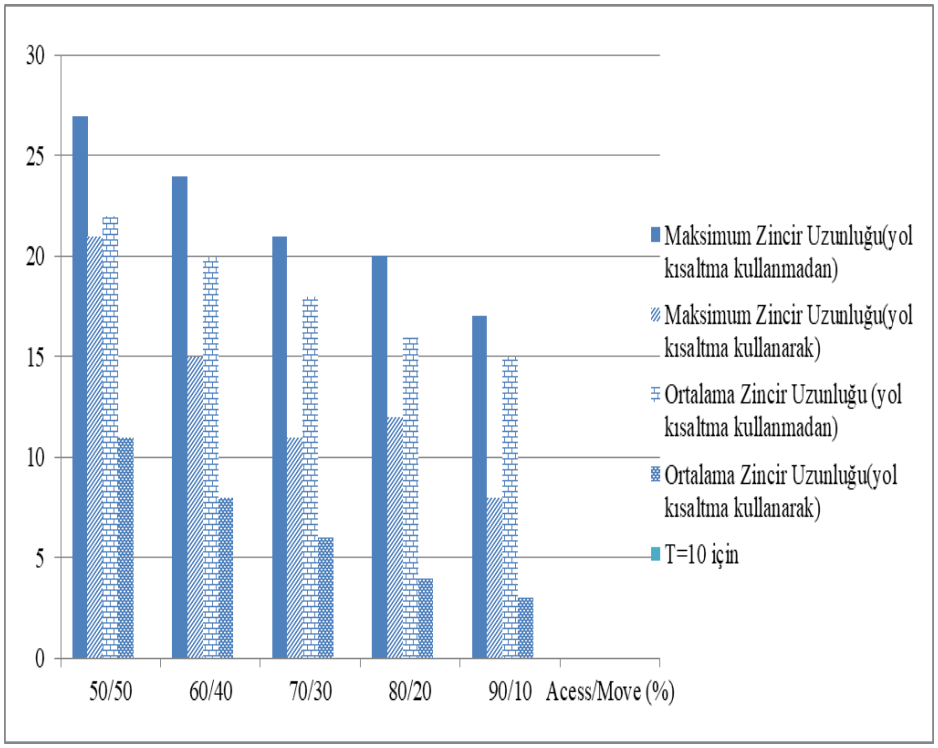
Ayrıca binom dağılımı kullandığımızda bazı nesnelere erişim yapılır. Tüm nesnelere erişim yapılmadığı için her nesnenin içinde bulunduğu zincir kırılmaz. Bazı nesnelerin içinde bulunduğu zincir olduğu gibi kalır. Ortalama zincir uzunluğu ise maksimum zincir uzunluğuna göre daha dramatik olarak azalır. Bu sonuç göstermektedir ki, yol kısaltma algoritması nesnelerin oluşturduğu uzun düğüm zincirini kırarak nesnelere ortalama erişim maliyetini azaltır ve nesnelere hızlı erişim yapılır.

### 7.1.2 Normal Dağılım

Dağıtık ortamda nesnelerle ilgili birtakım işlemler yapılır. Örneğin, nesnelere erişim yapılabilir ya da nesneler hareket ettirilebilir. Çalışmamızda, nesnelerle ilgili

gerçekleşecek olaylar (object-access, object-move) bağımsız olaylar olarak oluşturulmuş, dağıtık ortam simülasyonu yapılmıştır. Amacımız, yol kısaltma algoritmasının etkinliğini ve avantajlarını göstermektir. Çalışmamızda nesnelere erişim Access, nesnelerin taşınması Move operasyonuyla gösterilmiştir.

Dağıtık ortamda sistem modellemesi statik ve dinamik olmak üzere iki şekilde gerçekleştirilir. Statik ortamda nesnelere sıklıkla erişim yapılır,nesnelerin taşınması ise daha az yapılır. Dinamik ortamda nesnelerin taşınması işlemi daha fazla, nesnelere erişim ise daha az yapılır. Çalışmamızda statik ve dinamik ortamda nesnelerle ile ilgili işlemler için modelleme yapılmıştır.

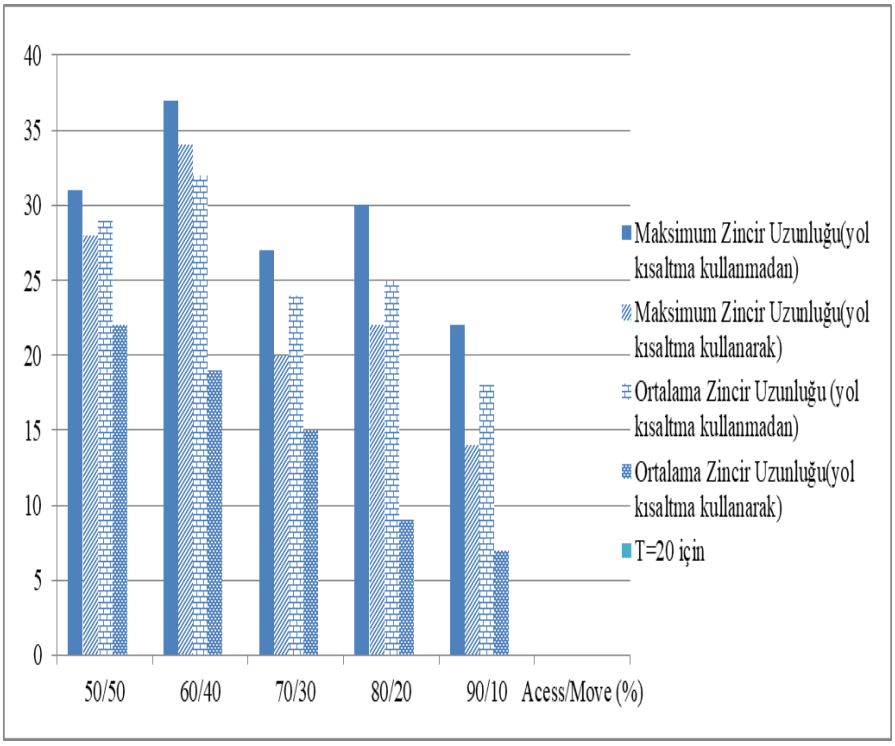



**Şekil 7.6** Normal dağılım kullanarak T=10 için maksimum ve ortalama zincir uzunlukları

Şekil 7.6' da T=10 için normal dağılım kullanarak çeşitli Access/Move (%) oranlarına göre dağıtık ortam simülasyonun sonuçları verilmiştir. Grafikte yol kısaltma algoritması kullanmadığımız ve kullandığımız durumlarda maksimum zincir uzunluğu ve ortalama zincir uzunluğu karşılaştırılmalı olarak gösterilmiştir. Bu grafikteki T eşik değerdir ve nesne erişim işleminde (Access) zincir uzunluğu eşik değere eşit ya da eşik değerden büyükse zincir uzunluğu kırılır. Çalışmamızda T değeri değiştirilerek maksimum ve ortalama zincir uzunluğu nasıl/ne yönde

değiştiğini gözlemlemek amaçlanmıştır. Grafikteki sütunlar sırasıyla yol kısaltma algoritması kullanmadığımız ve kullandığımız durumlardaki maksimum ve ortalama zincir uzunluklarını gösterir. Grafikte 50/50, 60/40, 70/30, 80/20 ve 90/10 gibi Acess/Move(%) oranları ile nesnelere erişim (Access) miktarı arttırılıp, taşınan (Move) nesne sayısı azaltarak maksimum ve ortalama zincir uzunlukları hesaplanır. Nesnelere erişim miktarını arttırıp yol kısaltma algoritması kullandığımızda ortalama zincir uzunluğu dramatik biçimde kısalmıştır.

Ayrıca normal dağılım kullandığımızda tüm nesnelere erişim yapılır. Tüm nesnelere erişim yapıldığı için her nesnenin içinde bulunduğu zincir kırılır. Ortalama zincir uzunluğu ise maksimum zincir uzunluğuna göre daha dramatik olarak azalır. Bu sonuç göstermektedir ki, yol kısaltma algoritması nesnelerin oluşturduğu uzun düğüm zincirini kırarak nesnelere ortalama erişim maliyetini azaltır ve nesnelere hızlı erişim yapılır.

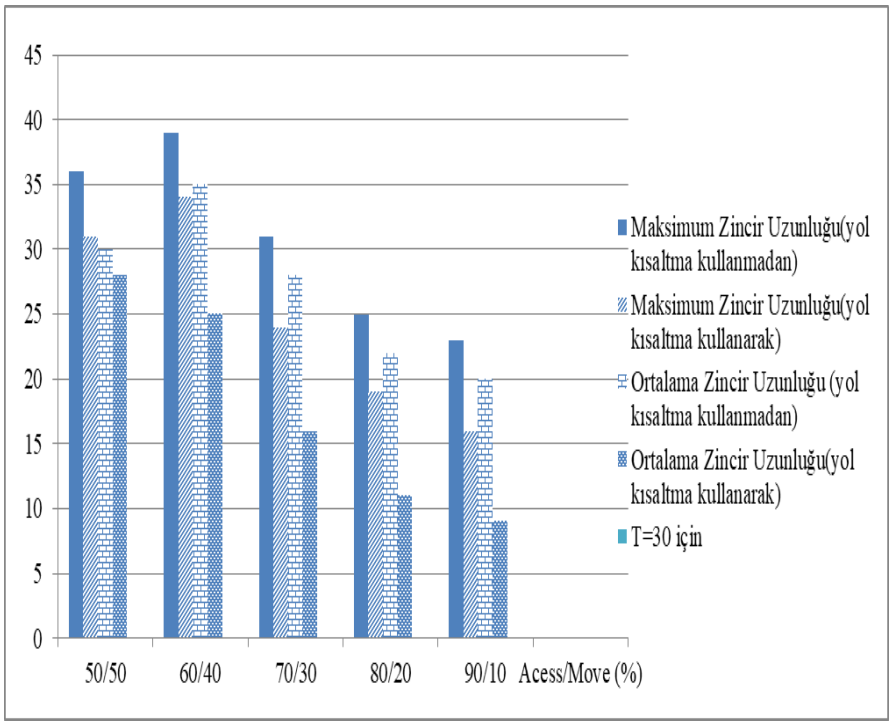



**Şekil 7.7** Normal dağılım kullanarak T=20 için maksimum ve ortalama zincir uzunlukları

Şekil 7.7' de T=20 için normal dağılım kullanarak çeşitli Access/Move (%) oranlarına göre dağıtık ortam simülasyonun sonuçları verilmiştir. Grafikte yol

kısaltma algoritması kullanmadığımız ve kullandığımız durumlarda maksimum zincir uzunluğu ve ortalama zincir uzunluğu karşılaştırılmalı olarak gösterilmiştir. Nesnelere erişim miktarını arttırıp yol kısaltma algoritması kullandığımızda ortalama zincir uzunluğu dramatik biçimde kısalmıştır.

Ayrıca normal dağılım kullandığımızda tüm nesnelere erişim yapılır. Tüm nesnelere erişim yapıldığı için her nesnenin içinde bulunduğu zincir kırılır. Ortalama zincir uzunluğu ise maksimum zincir uzunluğuna göre daha dramatik olarak azalır. Bu sonuç göstermektedir ki, yol kısaltma algoritması nesnelerin oluşturduğu uzun düğüm zincirini kırarak nesnelere ortalama erişim maliyetini azaltır ve nesnelere hızlı erişim yapılır.

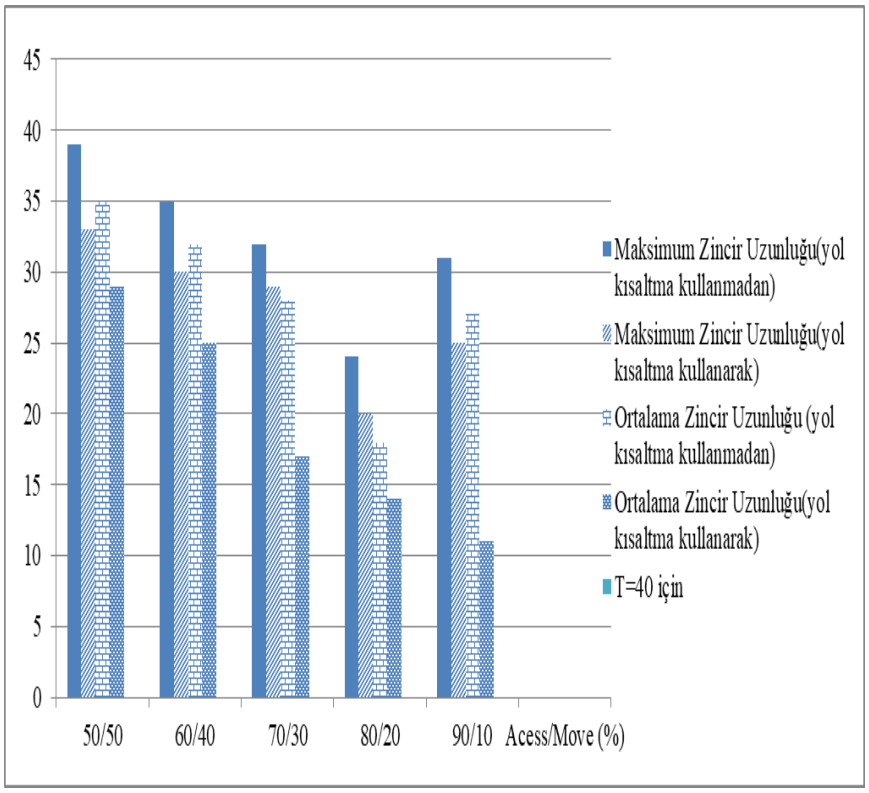



**Şekil 7.8** Normal dağılım kullanarak T=30 için maksimum ve ortalama zincir uzunlukları

Şekil 7.8' de T=30 için normal dağılım kullanarak çeşitli Access/Move (%) oranlarına göre dağıtık ortam simülasyonun sonuçları verilmiştir. Grafikte yol kısaltma algoritması kullanmadığımız ve kullandığımız durumlarda maksimum zincir uzunluğu ve ortalama zincir uzunluğu karşılaştırılmalı olarak gösterilmiştir.

Nesnelere erişim miktarını arttırıp yol kısaltma algoritması kullandığımızda ortalama zincir uzunluğu dramatik biçimde kısalmıştır.

Ayrıca normal dağılım kullandığımızda tüm nesnelere erişim yapılır. Tüm nesnelere erişim yapıldığı için her nesnenin içinde bulunduğu zincir kırılır. Ortalama zincir uzunluğu ise maksimum zincir uzunluğuna göre daha dramatik olarak azalır. Bu sonuç göstermektedir ki, yol kısaltma algoritması nesnelerin oluşturduğu uzun düğüm zincirini kırarak nesnelere ortalama erişim maliyetini azaltır ve nesnelere hızlı erişim yapılır.




**Şekil 7.9** Normal dağılım kullanarak T=40 için maksimum ve ortalama zincir uzunlukları

Şekil 7.9' da T=40 için normal dağılım kullanarak çeşitli Access/Move (%) oranlarına göre dağıtık ortam simülasyonun sonuçları verilmiştir. Grafikte yol kısaltma algoritması kullanmadığımız ve kullandığımız durumlarda maksimum zincir uzunluğu ve ortalama zincir uzunluğu karşılaştırılmalı olarak gösterilmiştir. Nesnelere erişim miktarını arttırıp yol kısaltma algoritması kullandığımızda ortalama zincir uzunluğu dramatik biçimde kısalmıştır.

Ayrıca normal dağılım kullandığımızda tüm nesnelere erişim yapılır. Tüm nesnelere erişim yapıldığı için her nesnenin içinde bulunduğu zincir kırılır. Ortalama zincir uzunluğu ise maksimum zincir uzunluğuna göre daha dramatik olarak azalır. Bu sonuç göstermektedir ki, yol kısaltma algoritması nesnelerin oluşturduğu uzun düğüm zincirini kırarak nesnelere ortalama erişim maliyetini azaltır ve nesnelere hızlı erişim yapılır.

## 7.2 Önerilen Dağıtık Etiket Modelinin Uygulama Sonuçları

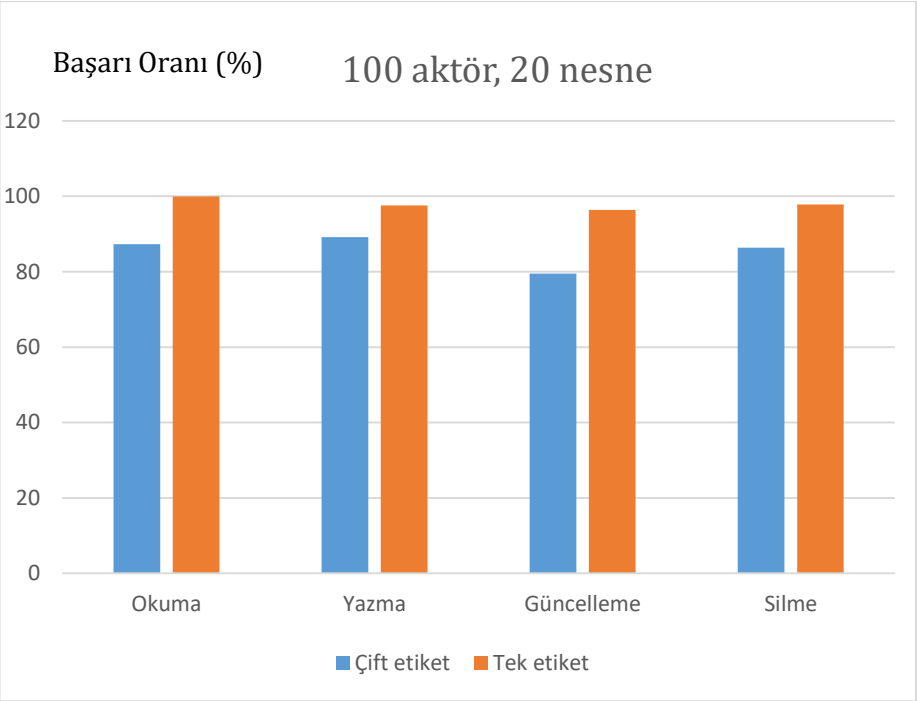
Önerilen tek etiket modelinin literatürde olan çift etiket modeli ile doğruluk ve süre açısından parformans sonuçları aşağıda verilmiştir.

### 7.2.1 Doğruluk

Tablo 7.3'de önerdiğimiz model (tek etiket) ile literatürde olan modelin başarısı bir hastaneden alınan sınıfları belli olan gerçek veri seti üzerinde karşılaştırılmalı olarak verilmiştir. Yaklaşık 100 aktör ve 20 nesne için doğruluk oranları hesaplanmıştır. Gerçek sınıflarına göre doğruluk oranları hesaplanmıştır. Önerdiğimiz modelin başarısı Şekil 7.10'da açıkça gösterilmiştir. Nesneler üzerinde gerçekleştirilen tüm işlemlerde her iki yöntemin doğruluk oranları açısından performansları karşılaştırıldığında önerdiğimiz modelin başarısı belirgin bir şekilde görülmektedir. Özellikle okuma ve silme işlemlerinde daha başarılı sonuçlar vermektedir. Bunun sebebi de yazma ve güncelleme işlemlerinin diğer işlemlere göre daha zor olmasıdır.

**Tablo 7.3** 100 aktör ve 20 nesne için doğruluk oranları

| Doğruluk Oranı (%) | Çift Etiket | Tek Etiket |
|---|---|---|
| Okuma | 87.27 | 99.94 |
| Yazma | 89.17 | 97.61 |
| Güncelleme | 79.50 | 96.37 |
| Silme | 83.34 | 97.81 |

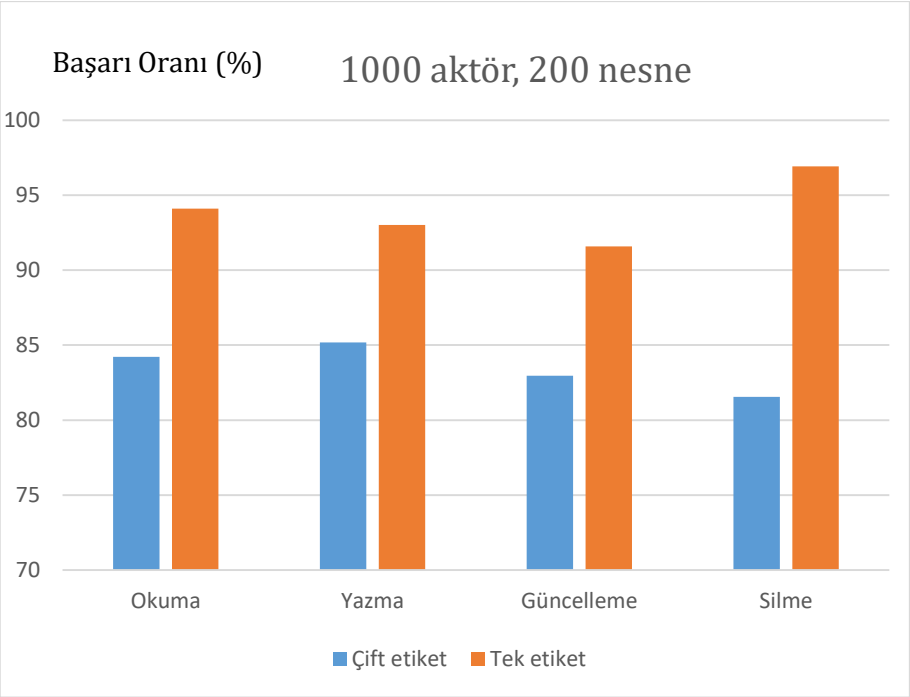



**Şekil 7.10** 100 aktör ve 20 nesne için doğruluk oranları

Tablo 7.4'de önerdiğimiz model (tek etiket) ile literatürde olan modelin başarısı (doğruluk açısından) karşılaştırılmalı olarak verilmiştir. Yaklaşık 1000 aktör ve 200 nesne için doğruluk oranları hesaplanmıştır. Önerdiğimiz modelin başarısı Şekil 7.11'de açıkça gösterilmiştir. Nesneler üzerinde gerçekleştirilen tüm işlemlerde her iki yöntemin doğruluk oranları açısından performansları karşılaştırıldığında önerdiğimiz modelin başarısı belirgin bir şekilde görülmektedir.

**Tablo 7.4** 1000 aktör ve 200 nesne için doğruluk oranları

| Doğruluk Oranı (%) | Çift Etiket | Tek Etiket |
|---|---|---|
| Okuma | 84.21 | 94.10 |
| Yazma | 85.17 | 93.02 |
| Güncelleme | 82.96 | 91.58 |
| Silme | 81.55 | 96.93 |

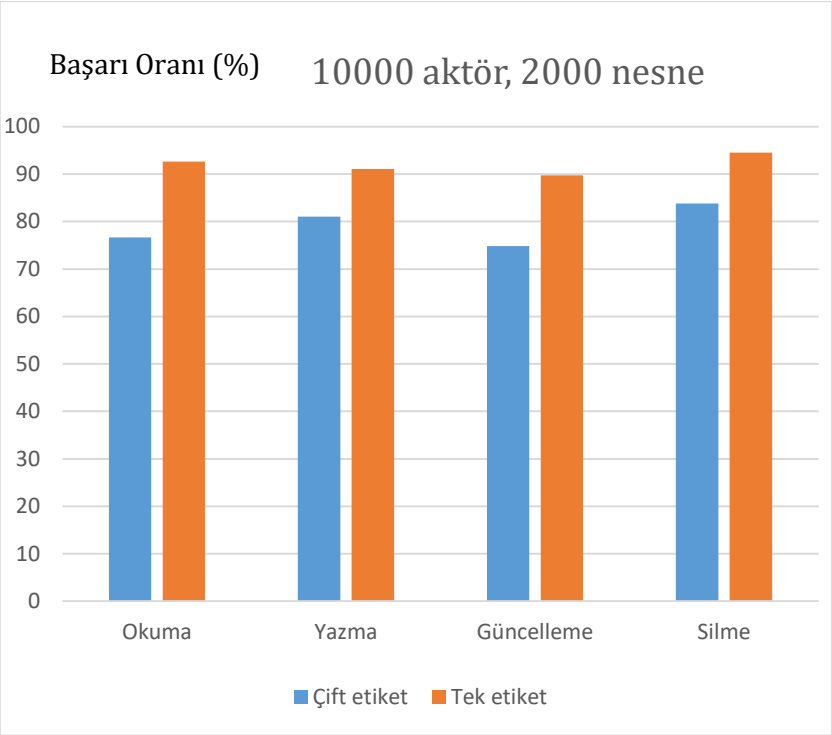



**Şekil 7.11** 1000 aktör ve 200 nesne için doğruluk oranları

Tablo 7.5'de önerdiğimiz model (tek etiket) ile literatürde olan modelin başarısı (doğruluk açısından) karşılaştırılmalı olarak verilmiştir. Yaklaşık 10000 aktör ve 2000 nesne için doğruluk oranları hesaplanmıştır. Önerdiğimiz modelin başarısı Şekil 7.12'de açıkça gösterilmiştir. Nesneler üzerinde gerçekleştirilen tüm işlemlerde her iki yöntemin doğruluk oranları açısından performansları karşılaştırıldığında önerdiğimiz modelin başarısı belirgin bir şekilde görülmektedir.

**Tablo 7.5** 10000 aktör ve 2000 nesne için doğruluk oranları

| Doğruluk Oranı (%) | Çift Etiket | Tek Etiket |
|---|---|---|
| Okuma | 75.63 | 92.64 |
| Yazma | 81.05 | 91.09 |
| Güncelleme | 74.87 | 89.75 |
| Silme | 83.78 | 94.56 |

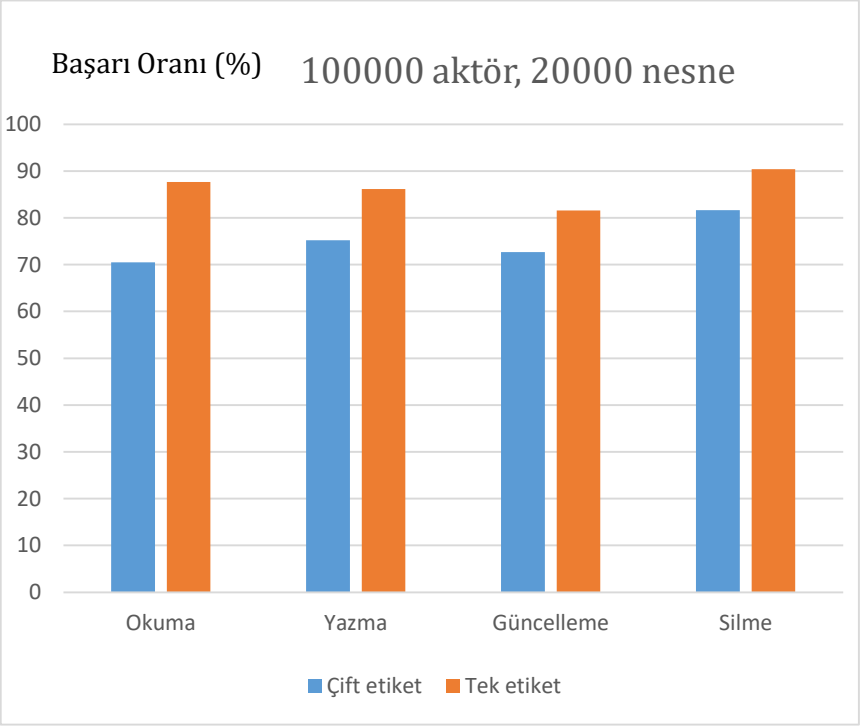



**Şekil 7.12** 10000 aktör ve 2000 nesne için doğruluk oranları

Tablo 7.6'da önerdiğimiz model (tek etiket) ile literatürde olan modelin başarısı (doğruluk açısından) karşılaştırılmalı olarak verilmiştir. Yaklaşık 100000 aktör ve 20000 nesne için doğruluk oranları hesaplanmıştır. Önerdiğimiz modelin başarısı Şekil 7.13'de açıkça gösterilmiştir. Nesneler üzerinde gerçekleştirilen tüm işlemlerde her iki yöntemin doğruluk oranları açısından performansları karşılaştırıldığında önerdiğimiz modelin başarısı belirgin bir şekilde görülmektedir. Özellikle okuma ve silme işlemleri tüm tablolarda diğer işlemlere oranla daha başarılı sonuçlar vermektedir. Bunun sebebi yazma ile güncelleme işlemleri okuma ve silme işlemlerine göre daha zordur. Ayrıca aktör ve nesne sayısı azaldıkça başarı oranları her iki yöntemde de artmaktadır. Çünkü daha az aktör ve nesneyle daha doğru sonuçlar üretilmektedir.

**Tablo 7.6** 100000 aktör ve 20000 nesne için doğruluk oranları

| Doğruluk Oranı (%) | Çift Etiket | Tek Etiket |
|---|---|---|
| Okuma | 70.50 | 87.64 |
| Yazma | 75.19 | 86.17 |
| Güncelleme | 72.65 | 81.60 |
| Silme | 81.64 | 90.38 |




**Şekil 7.13** 100000 aktör ve 20000 nesne için doğruluk oranları

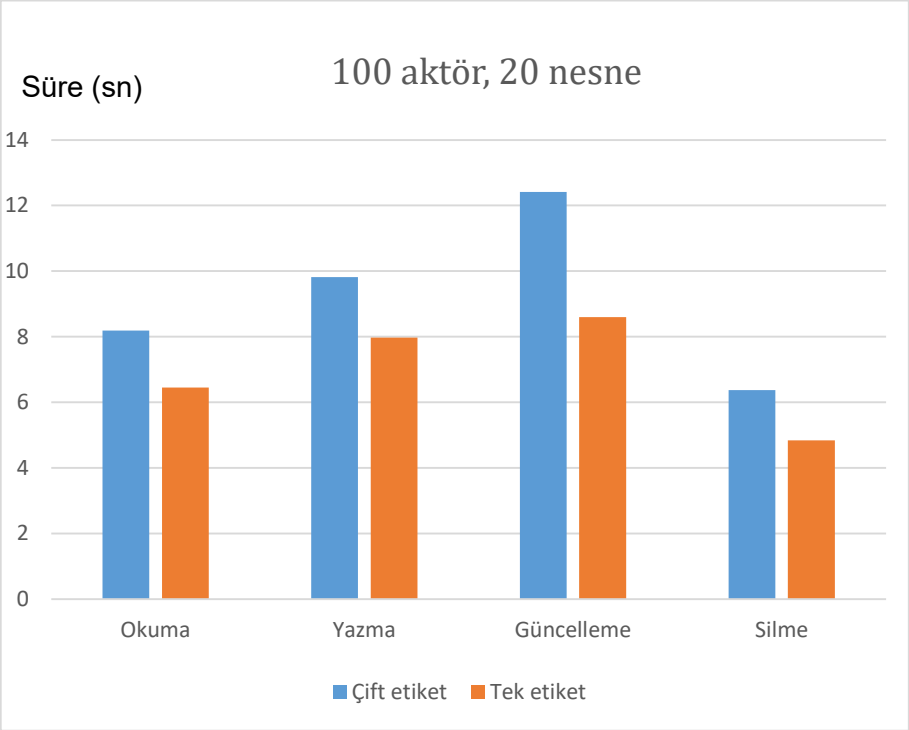
### 7.2.2 Süre

Tablo 7.7'de önerdiğimiz model (tek etiket) , literatürde olan modelin süre hastaneden alınan gerçek veri seti üzerinde karşılaştırılmalı olarak verilmiştir. Yaklaşık 100 aktör ve 20 nesne için verilmiştir. Önerdiğimiz modelin başarısı Şekil

7.14'de açıkça gösterilmiştir. Süre açısından baktığımızda önerdiğimiz modelle daha az bir zamanda veri üzerinde işlemler yapıldığı görülmektedir. Yazma ve güncelleme işlemleri her iki yöntemde de süre açısından diğer işlemlere göre daha fazla sürmektedir. Bunun sebebi nesne üzerinde yazma ve güncelleme yapmak daha fazla zaman almaktadır. Ayrıca önerdiğimiz model ile süre açısından karşılaştırdığımızda nesne üzerinde yapılan tüm işlemlerde oldukça başarılı sonuçlar vermektedir.

**Tablo 7.7** 100 aktör ve 200 nesne için süreler

| Süre (sn) | Çift Etiket | Tek Etiket |
|---|---|---|
| Okuma | 8.19 | 6.45 |
| Yazma | 9.82 | 7.97 |
| Güncelleme | 12.41 | 8.60 |
| Silme | 6.37 | 4.84 |

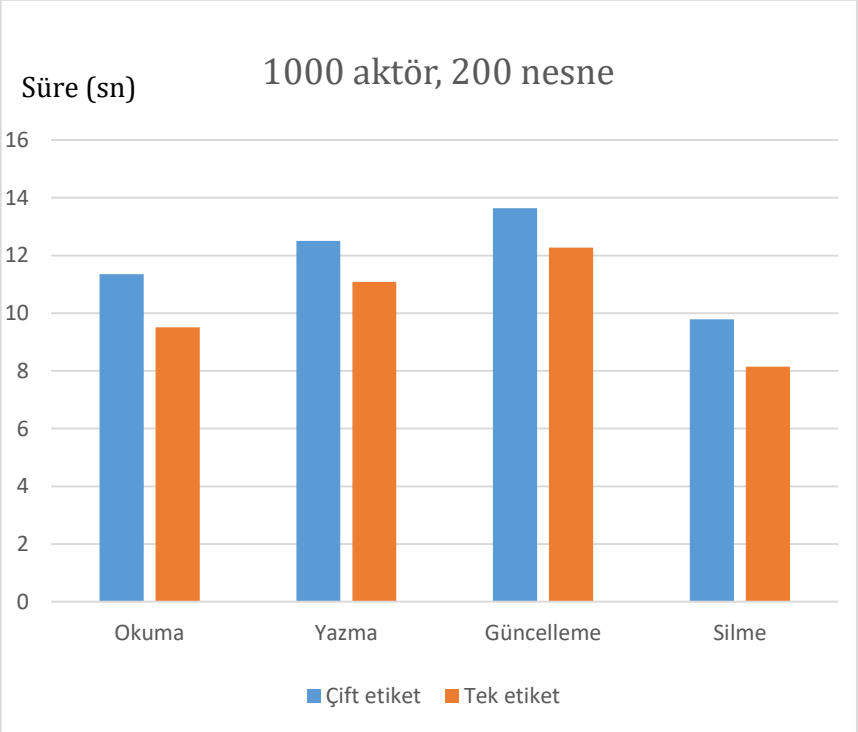



**Şekil 7.14** 100 aktör ve 20 nesne için süreler

Tablo 7.8'de önerdiğimiz model (tek etiket) , literatürde olan modelin süre hastaneden alınan gerçek veri seti üzerinde karşılaştırılmalı olarak verilmiştir. Yaklaşık 1000 aktör ve 200 nesne için verilmiştir. Önerdiğimiz modelin başarısı Şekil 7.15'de açıkça gösterilmiştir. Süre açısından baktığımızda önerdiğimiz modelle daha az bir zamanda veri üzerinde işlemler yapıldığı görülmektedir.

**Tablo 7.8** 1000 aktör ve 2000 nesne için süreler

| Süre (sn) | Çift Etiket | Tek Etiket |
|---|---|---|
| Okuma | 11.35 | 9.51 |
| Yazma | 12.51 | 11.09 |
| Güncelleme | 13.64 | 12.27 |
| Silme | 9.79 | 8.15 |



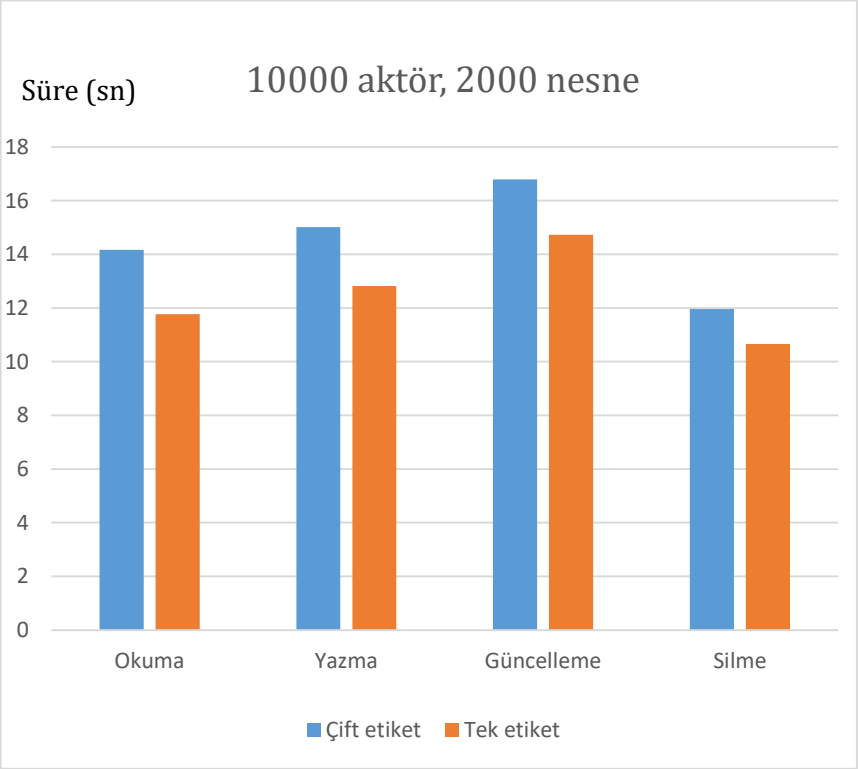

**Şekil 7.15** 1000 aktör ve 200 nesne için süreler

Tablo 7.9'da önerdiğimiz model (tek etiket), literatürde olan modelin süre hastaneden alınan gerçek veri seti üzerinde karşılaştırılmalı olarak verilmiştir.

Yaklaşık 10000 aktör ve 2000 nesne için verilmiştir. Önerdiğimiz modelin başarısı Şekil 7.16'da açıkça gösterilmiştir. Süre açısından baktığımızda önerdiğimiz modelle daha az bir zamanda veri üzerinde işlemler yapıldığı görülmektedir.

**Tablo 7.9** 10000 aktör ve 20000 nesne için süreler

| Süre (sn) | Çift Etiket | Tek Etiket |
|---|---|---|
| Okuma | 14.17 | 11.77 |
| Yazma | 15.02 | 12.82 |
| Güncelleme | 16.79 | 14.73 |
| Silme | 11.96 | 10.66 |



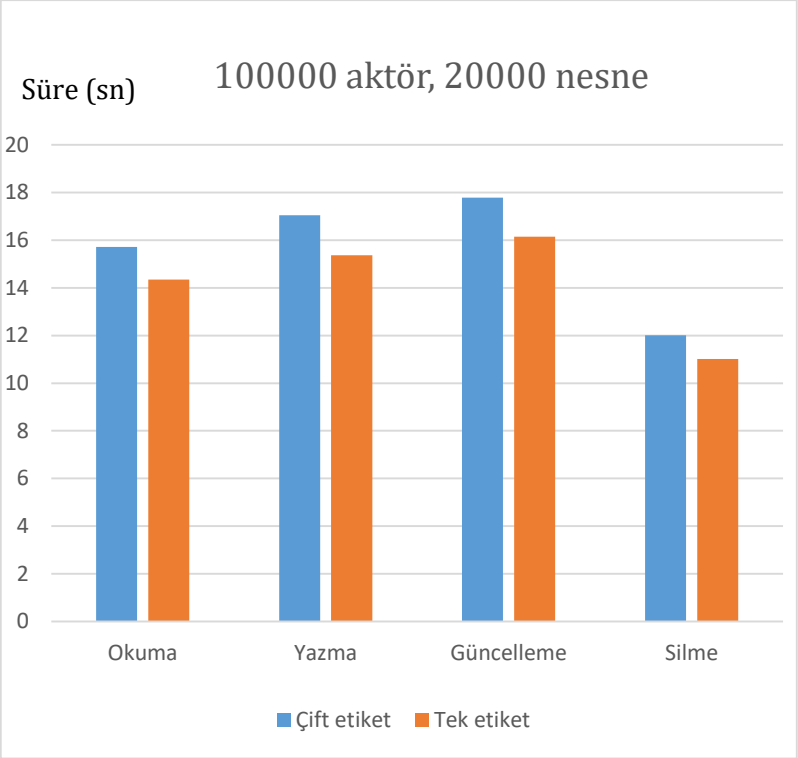

**Şekil 7.16** 10000 aktör ve 2000 nesne için süreler

Tablo 7.10'da önerdiğimiz model (tek etiket), literatürde olan modelin süre hastaneden alınan gerçek veri seti üzerinde karşılaştırılmalı olarak verilmiştir. Yaklaşık 10000 aktör ve 2000 nesne için verilmiştir. Önerdiğimiz modelin başarısı

Şekil 7.17'de açıkça gösterilmiştir. Süre açısından baktığımızda önerdiğimiz modelle daha az bir zamanda veri üzerinde işlemler yapıldığı görülmektedir.

**Tablo 7.10** 100000 aktör ve 200000 nesne için süreler

| Süre (sn) | Çift Etiket | Tek Etiket |
|---|---|---|
| Okuma | 15.72 | 14.35 |
| Yazma | 17.04 | 15.37 |
| Güncelleme | 17.78 | 16.14 |
| Silme | 12.01 | 11.02 |




**Şekil 7.17** 100000 aktör ve 20000 nesne için süreler

# 8
# SONUÇ VE ÖNERİLER

Bu tezde, akış denetimi ile veri gizliliği (confidentiality) sağlanmasına yönelik etiketleme modeli tanıtılmıştır. Her aktör ortak veriyi etiketleyerek güvenlik politikasını belirtir.

Bizim çalışmamızda aktörler arasında yetki verme ve alma işlemlerinin her ikisi de yapılır. Çalışmamızda okuma, yazma, güncelleme, silme gibi her bir işlem için ayrı bir yetkilendirme ya da erişim denetimi yapılmaz. Erişim denetimi ve yetkilendirme işlemleri etiketler aracılığıyla yapılır. Önceki çalışmalardan farklı olarak dağıtık veritabanında yapılan tüm işlemler için veri güvenliği sağlanır. Aktörler istedikleri zaman verdiği yetkiyi geri alabilir ya da istediği aktöre yetki verebilir. Dağıtık veritabanlarına güvenlik politikalarının uygulanması sırasında oluşan zorluklar aşılmış olur.

Önceki çalışmalarda nesne üzerinde gerçekleştirilen herbir işlem (okuma, yazma) için ayrı bir etiket kullanılmıştır ve sadece okuma ile yazma işlemi yapılmıştır. Bizim tarafımızdan önerilen çalışmada ise nesne üzerinde gerçekleştirilen tüm işlemler (okuma, yazma, güncelleme, silme) tek etiket kullanarak gerçekleştirilir. Bu da önerdiğimiz modelin esnek olduğunu gösterir. Kötü niyetli aktörlerin veriye erişimlerinin takipleri ile bilgi ifşasının önüne geçilmeye çalışılmıştır. Önerilen tek etiket modelinin veriler üzerinde gerçekleştirilen tüm işlemler için sonuçları deneysel çalışma ile de gösterilmiştir. Özellikle okuma ve silme işlemlerinde daha başarılı sonuçlar vermektedir. Ayrıca önceki çalışmalarda kullanılan yöntemle süre açısından da karşılaştırılmıştır ve daha kısa sürede işlemleri gerçekleştirmektedir.

Ancak, bu modelde yeniden etiketleme diye adlandırılan aktör politikalarındaki bir değişiklik, daha kapsamlı bir yorumlamayı gerektirmektedir. Herşeyden önce, yeni etiketin koruma kapsamından “emin” olmak zorunluluğu vardır. Burada kastedilen,

etikette yapılan değişiklik, ait olduğu veriyi ya daha kısıtlayıcı hale getirmeli (örn. önceki okuyucu kitlesinin bir alt kümesi olmalı) veya eşit ölçüde kısıtlayıcı olmalıdır. Örnek olarak, bir politikadan bir okuyucu çıkarma işlemi, yeni etiket oluşturur. Bu, verinin daha az okuyucuya ulaşmasını netice verdiği için, 'emin' yeniden etiketleme işlemi, olduğu kolayca görülür. Buna göre, 'emin' yeniden etiketleme kurallarının belirlenmesi ve tanımlanması gerekir.

Diğer taraftan, bir etiket veri akışlarının hepsini açıkça göstermediği için, belirsizliklerin yorumlanması gerekmektedir. Örneğin, bir veri sahibi kendi güvenlik politikasını etikete koymadığında veya ihmal ettiğinde etiketin anlamı, farklı bir yoruma açık olmaktadır. Aynı şekilde, bir aktör hiyerarşisi tanımlandığında, yeniden etiketleme işlemlerinin, hiyerarşi dikkate alınarak tekrar yorumlanması gerekir. Örneğin, bir okuyucu hiyerarşide kendinden aşağıda birine 'vekalet' vermesinin yeniden etiketleme işleminin 'emin' olmasına etkisi ne olacaktır? Özellikle, askeri veya yüksek güvenlikle ilgili bir uygulamada, aktör hiyerarşisine ihtiyaç olduğu açıktır. İleride, aktör hiyerarşisi gözönüne alınarak, bu modelde yeniden etiketlemedeki etkileri çalışılacaktır.

Son olarak, yol kısaltma algoritması'nda dikkate almamız gereken önemli bir nokta, nesne erişim sıklığının yüksek, yol kısaltma işleminin seyrek yapılacağı gerçeğidir. Bunu garanti etmek için, zincir uzunluğu ancak belli bir eşik değeri aştığı zaman yol kısaltma algoritması çalıştırılmalıdır. Bu eşik değer, dikkatli değerlendirmeler veya deneysel çalışmalar sonucu belirlenmesi gereken bir parametredir. Çünkü eğer eşik değer büyük seçilirse, uzun zincir oluşması nedeniyle nesne erişimi yavaşlayacak; ya da çok küçük seçilirse YolKısalt algoritması, sık çalışacaktır.

Bu tez kapsamında dağıtık veritabanlarında veri güvenliği problemi ele alınmakta, özellikle veri akış denetimi ile ilgili dağıtık etiket modelinin tanıtımı yapılmakta ve kullanılmasına örnek uygulamalar gösterilmektedir. Ayrıca, dağıtık ortamda veri nesnesi akışı, graf ile modellenmektedir. Nesne erişimi sırasında düğümler arasında uzun zincir oluşabilir. Yol kısaltma yöntemi ile bu zincir kırılarak nesneye erişim maliyeti azaltılır ve performans kazancı elde edilir. Deneysel çalışma yapılarak, dağıtık ortam benzetimi yapılmış, algoritmanın değerlendirilmesine ek olarak etkinliği deneysel çalışma ile de gösterilmiştir. Binom ve normal dağılım kullanarak

çeşitli Access/Move (%) oranlarına göre dağıtık ortam simülasyonun sonuçları karşılaştırmalı olarak verilmiştir. Normal dağılım kullandığımızda önerdiğimiz algoritmanın başarısı daha net bir şekilde gösterilmiştir.

Gelecek çalışması olarak, etiket modelinin çalışmasını gösteren prototip bir uygulama oluşturulacak ve aktörler hiyerarşisini de dikkate alan yeniden etiketleme ile model zenginleştirilecektir.

# KAYNAKÇA


[1] Lin J., Yu W., Zhang N., "A survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy and Applications", *IEEE Internet of Things*, vol. 4, no. 15, pp. 1125-1142, 2017.

[2] Clifton C.W., "Privacy Beyond Confidentiality", *In Procedings of the ACM SIGSAC Conference and Communications Security (CCS'14),* 2014, pp. 1156-1156.

[3] M. Dağdeviren, N. Dönemez, M. Kurt, "Bir İşletmede Tedarikçi Değerlendirme Süreci için Yeni bir Model Tasarımı ve Uygulaması", *Gazi Üniversitesi Mühendislik ve Mimarlık Fakültesi Dergisi,* vol. 21, no. 2, pp. 247-255, 2006.

[4] S.Vimercati, S.Foresti, G.Livraga at al, "Privacy in Pervasive Systems: Social and Legal Aspects and Technical Solutions", *Data Management in Pervasive Systems*, 2015, pp.43-65.

[5] S.Bajaj, R.Sion, "TrustedDB:A Trusted Hardware-Based Database with Privacy and Data*", IEEE Transactions on Knowledge and Data Engineering*", vol. 26, no.3, pp. 752-765, 2014.

[6] F.Rocha, S.Abreu, M.Correia at all., "The Final Frontier: Confidentiality and Privacy in the Cloud", *IEEE Computer Society*, vol. 44, no. 9, pp. 44-50, 2011.

[7] E. Olca, Ö. Can, "Türkiye'de Elektronik Sağlık Kaydı Bağlamında Gizlilik ve Güvenlik Üzerine Teknolojiler.", *3rd Intl. Symposium on Digital Forensics and Security,* 2015, pp. 259-263.

[8] Y. Vural, Ş. Sağıroğlu, "Kurumsal Bilgi Güvenliği ve Standartları üzerine Bir İnceleme", *Gazi Üniversitesi Mühendislik ve Mimarlık Fakültesi Dergisi*, vol. 23, no. 2, pp. 507-522, 2008.

[9] K.Bogaert, "Confidentiality and Privacy: What is the difference?", *South African Family Practice,* vol. 51, no. 3, pp. 194-195, 2009.

[10] J.E.Barett, W.B.Johnson, "Confidentiality and Privacy", *Wiley Online Library*, 2015.

[11] R. T. Mercuri, "The HIPAA-Potamus in Health Care Data Security", *Communications of the ACM Security Watch*, 47(2), 25-28, 2004.

[12] S.A.Buckovich, H.E.Ripper, "Driving toward guiding principles:a goal for privacy, confidentiality and security of health information", *Journal of the American Medical Informatics*, vol. 6, no.2, pp. 122-133, 1999.

[13] Shen N., Bernier T., Squeria L., "Understanding the patient privacy perspective on health information exchange: A systematic review", Elseiver International Journal of Medical Informatics, vol. 125, no. pp.1-12, 2019.

[14] Vorakulpipat C., Sirapaisan S., Rattanalendrunson E., Savargsuk V., "A Policy-Based Framework for Preserving Confidentiality in BYOD Environments. A

review of Information Security Perspectives", *Hindawi Security and Communication Networks,* pp. 1-11, 2017.

[15] Faria P.L., Cordeiro J.V, "Health data privacy and confidentility rights: Crisis or redemption?", *Springer Revista Portuguesa de Saute Publica*, vol. 2, no.2, pp.123-133, 2014.

[16] Gupta B.B., Shingo Y., "Agrawal D.H, Advances in Security and Privacy of Multimedia Big Data in Mobile and Cloud Computing", *Multimedia Tools and Applications*, vol. 77, no. 7, pp. 9203-9208, 2018.

[17] Bakır Ç., Hakkoymaz V., "Classifiying Database Users Intrusion Prediction and Detection in Data Security", *Technical Gazette*, vol. 27, no. 6, 2020.

[18] Bakır Ç., Hakkoymaz V., "Veritabanı Güvenliğinde Saldırı Tahmini ve Tespiti için Kullanıcıların Sınıflandırılması", *8.th International Conference on Information Security and Cryptology*, 2015, pp. 28-33.

[19] Ramasubramanian P., Kannan A., "Multi- Agent based Quickprop Neural Network Short-term Forecasting Framework for Database Intrusion Prediction System", *CiteSeerX*, 2014.

[20] Alcaraz L., " Security and Privacy Trends in the Industrial Internet of Things", *Springer Advanced Sciences and Technologies for Security Applications,* pp.123-133, 2019.

[21] Vimercati S., Foresti S., Livraga G., "Privacy in Pervasive Systems: Social and Legal Aspects and Technical Solutions", *Data Management in Pervasive Systems*, pp. 43-65, 2015.

[22] Debelva F., Mosquera I., "Privacy and Confidentiality in Exchange of Information Procedures: Some Uncertainties, Many Issues, But Few Solutions", *SSRN Peer Rewieved Article*, vol. 45, no. 5, pp. 362-381, 2018.

[23] Esfandiari H., Hajigohayi M., Liaghat V., Monemizadeh M., "Streaming Algorithms for Estimating the Matching Size in Planar Graphs and Beyond", *ACM Transactions on Algorithms,* vol. 14, no. 8, 2018.

[24] T. H. Cormen, C. E. Leiserson, R. Rivest, C. Stein, "Introduction to Algorithms", *MIT Press Cambridge, McGraw Hill Book Company, Third Edition*, 2009.

[25] Liu J., George M. D., "Fabric: A Platform for Secure Distributed Computation and Storage", *ACM Symposium on Operating Systems Principles and Implementation (SOSP)*, 2009, pp. 321-334.

[26] Liu J., Arden O., George M., Myers A.C, "Fabric: Building Open Distributed Systems Securely by Construction", *Journal of Computer Security*, vol. 25 no. 4-5, pp. 367-426, 2017.

[27] N.Zeldovich, S.Wickizer and E.Kohler, "Making Information Flow Explicit in HiStar", *OSDI '06 Proceedings of the 7th symposium on Operating systems design and implementation*, 2006, pp.263-278.

[28] Kim N.Y, Ryu J.H., Kwon B.W, Pan Y., "CF- CloudOrch:Container fog node-based cloud orchestration for IoT networks", *The Journal of Supercomputing*, vol. 74, no.12, pp. 7024-7045, 2018.

[29] R. Elmasri, S.B. Navathi, "Fundamentals of Database Systems", Sixth Edition, *Addision Wesley-Pearson*, 2011.

[30] Cai F., Zhu N., He J., Mu P., Li W., "Survey of Access Control Models and Technologies for Cloud Computing", *Springer Cluster Computing*, pp. 1-12, 2018.

[31] Servos D., Osborn S.L, "Current Research and Open Problems in Attribute-Based Access Control", *ACM Computing Surveys,* vol. 49, no. 4, 2017.

[32] Li Q., Snadhu R., Zhang X., Xu M., "Mandatory Content Access Control for Privacy Protectionin Information Centric Networks", *IEEE Transactions on Dependable and Secure Computing,* vol. 14, no. 5, pp. 494-506, 2017.

[33] S. Osborn, R. Sandhu, "Configuring Role-based Access Control to Enforce Mandatory and Discretionary Access Control Policies", *ACM Transactions on Information and System Security*, vol. 3, no. 2, pp. 85-106, 2000.

[34] Almutairi A., Sarfraz I., Ghafoo A., "Risk-Aware Management of Virtual Resources in Access Controlled Service-Oriented Cloud Datacenters", *IEEE Transactions on Cloud Computing,* vol. 6, no. 1, pp. 168-181, 2018.

[35] Aafaf Q., Hajar M., Anas A.E., Abdellah A.Q., "Access Control in the Internet of Things: Big challenges and new opportunities", *Elsevier Computer Networks*, vol. 12, pp. 237-262, 2017.

[36] B. D. Joshi, E. Bentino at al. , "A Generalized Temporal Role Based Access Control Model", *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no.1, pp. 4-23, 2005.

[37] Elhoseny M., Gustavo O., Showkat S., "Secure Medical Data Transmission Model for IoT-Based Healtcare Systems", *IEEE Access Special Section on Information Security Solutions,* vol. 6, pp. 20596-20608, 2018.

[38] Alizadeh M., Abolfazli S., Zamari M., Baharun S., "Authentication in Mobile cloud computing: A survey", *Journal of Network and Computer Applications*, vol. 61, pp. 59-80, 2016.

[39] Rana M., Kubbo M., Jayabalan M., "Privacy and Security Challenges towards Cloud Based Access Control in Electronic Health Records", *Asian Journal of Information Technologu*, vol. 16, no.2-5, pp. 274-281, 2017.

[40] Burow N., Carr S.A, Nash J., Larsen P., Franz M., "Control-Flow Integrity; Preccision, Security and Performance", *ACM Computing Surveys*, vol. 50, no. 1, pp. 1-16, 2017.

[41] Bakir Ç., Hakkoymaz V., "Dağıtık Veritabanında Veri Etiketleme ile Bilgi Akış Denetimi*, 5.Ulusal Yüksek Başarımlı Hesaplama Konferans"ı*, Esenler İstanbul, 2017.

[42] Cecchetti, E., Myers, A.C., "Nonmalleable Information Flow Control", *ACM Conference on Computer and Communication Security*, 2017.

[43] Arden, O., Myers., A.C., "A calculus for flow-limited authorization", *IEEE Computer Security Foundations*, pp. 135-149, 2016.

[44] Cheng, W., Ports, R.K, Schultz, D., "Abstractions for Usable Flow Control in Aelous", *USENIX ATC'12 Proc. USENIX Conference on Annual Technical Conference*, 2012, pp. 1-12.

[45] Ulaştırma Denizcilik ve Haberleşme Bakanlığı, *Ulusal Siber Güvenlik Stratejisi*, 2016-2019, Referans.

[46] Cheng W., Ports R.K, Schultz D., "Abstractions for Usable Flow Control in Aelous", *USENIX ATC'12 Proc. USENIX Conference on Annual Technical Conference*, 2012, pp. 1-12.

[47] Myers A. C., Liskov B., "Protecting Privacy using the Decentralized Label Model", *ACM Transactions on Software Engineering and Methodology*, vol. 9, no.4, pp. 410-442, 2000.

[48] Myers A. C., Liskov B., "Complete, Safe Information Flow with Decentralized Labels", In Proc. *IEEE Symposium on Security and Privacy*, 1998.

[49] Dennis J. B., VanHorn E. C., "Programming Semantics for Multiprogrammed Computations", *Comm. of the ACM*, vol. 9, no. 3, pp. 143–155, 1966.

[50] Singh J., Pasquier T.F, Bacon J., Eyers D., "Integrating Messaging Middleware and Information Flow Control", *IEEE International Conference on Cloud Engineering*, 2015, pp. 54-59.

[51] Myers A. C., Liskov B., "A Decentralized Model for Information Flow Control", *In Proc. 17th ACM Symp. on Operating System Principles (SOSP),* 129–142, 1997.

[52] J. B. Dennis, E. C. VanHorn, "Programming Semantics for Multiprogrammed Computations", *Comm. of the ACM*, vol. 9, no. 3, pp. 143–155, 1966.

[53] M. Jed Liu, "Towards A Secure Federated Information Systems", *Cornell University,* Ph.D.thesis, 2012.

[54] D. Gollmann, "Computer Security", *Wiley Interdisciplinary Reviews: Computational Statistics,* vol. 2, no.5, pp. 544-554, 2010.

[55] Hakkoymaz V., Bakir C., "Information Flow Control with Decentralized Labeling Model in Information Security", Journal of Web Engineering , *Special Issue: Advanced in Al and Nature-inspired Approaches for Web Data Security*, vol.19, no.7-8, 1-23, 2020.

[56] Bakir C., Hakkoymaz V., "Object based Distributed Enviroment Modelling and Simulation", *9th International Conference on Advanced Technologies*, 2020.

# A
# DAĞITIK ORTAM SİMÜLASYONU SÖZDE KODLARI

```
#define Zmax 1000
 //Genel değişkenler
   T:=10; // T:eşik değer (zincir uzunluğu)
   x:=0;  //break_chain (i, j, x) x:hedef düğüm
   slot:=1; //book_keeping(i,L) fonksiyonu için kayıt tutulması
define DriverForObjAccess ( )  {
     i=random(1,k);  //k nesneden rastgele bir nesne
     j=random(1,n);  //n düğümden rastgele bir düğüm
     L= object_access (i, j ) //L:i'ye erişimde adım sayısı
     Book_keeping ( i: nesne, L : adım sayısı);
       if (L>=T)  //eşik değere ulaştı
       {
          //i nesnesi için j'den x'e olan zinciri kır.
          if (bcless=1)
          {
             break_chain (i, j, x);
          }
       }
}
```

```
define object_access (i:nesne, j:düğüm) //zincir uzunluğunu bulur.
{
    int adimsayisi=0; //adım sayısını ilk değeri.
    boolean bulundu=false;
     while (true)
     {
        //j düğümün NT'de i nesnesi için -1 ise, nesne burada.
         if(NT[ j, i ] ==-1)
         {
              x=j;
              return adimsayisi;
         }
         else
         {
               adimsayisi++;
               j=NT[j,i];
          }
     }
}
```

```
define DriverForObjMove ( )  {
    i=random(1,k);  //k nesneden rastgele bir nesne
    j=GNT[i];  //GNT'de nesne i  hangi düğümde
    x=random(1,n) //x: taşınacağı düğüm
     while  (x==j)
     {
        x=random (1,n); //aynı düğüme taşınmaz.
     }
   object_move (i, j, x) // i'yi j'den x'e taşı.
}
define object_move ( i:nesne , j: kaynak düğüm, x: hedef düğüm)
{
       NT[ x, i ] ←-1;
       NT[ j, i ] ← x;
       GNT [ i ] ← x;
}
define break_chain ( i:nesne , j: kaynak düğümü, x: hedef düğüm)
{
      while ( NT [ j, i ]!= -1 )  do
          next ← NT [ j, i ]
            NT [ j, i ]  ← x;
            j ←next;
      end while
}
```

```
define Book_keeping (i:nesne, L:adım sayısı)
{
    Z[slot].nesne ← i;
    Z[slot].adim ← L;
    slot=slot+1;
     if (slot>Zmax)
     {
        compute_statistics ( );
        slot ← 1;
     }
}
```

```
define compute_ statistics ( )
{
   toplam=0;
   max=Z[1].adim;
     for (i=1;i<= Zmax;i++)
     {
        toplam=toplam+Z[i].adim; //kümülatif toplam
     }
       if (max<Z[i].adim)
       {
           max←Z[i].adim;
       }
    ortalama=toplam/Zmax;  //ortalama zincir uzunluğu
    print (“Access/Move”,max, ortalama);
    print(“*******”);
    for (i=1;i<=Zmax;i++) {
      print(“nesne”, i, “erişim sayısı”, Z[i].nesne);
   }
}
```

```
main ( )  {
    for (T=10; T<=50;T←T+10) {
      for (Y=10; Y<=50;Y←Y+10) {  // Y Access yüzdesi
        for (bcless←0;bcless<=1;bcless++) {
          slot:=0;
          EventCount:=0; //olay sayısı
          Af:=0; //Access fonksiyonu
          Mf:=0; //move fonksiyonu
          DriverForObjAcess ( );
          Af++;
            while (EventCount<=10000) {
              if ((Af/(Af+Mf)*100)<Y) {
                DriverForObjAcess ( );
                Af++;
              }
              else
              {
                DriverForObjMove ( );
                Mf++;
              }
              EventCount++;  } //EventCount
        } //bcless
      } //AccessCount Y
  } //T için
```

```
} //main ()
```

# TEZDEN ÜRETİLMİŞ YAYINLAR

**İletişim Bilgisi:** cigdem.bakr@gmail.com

**Makaleler**

1. Bakir C., Hakkoymaz V., "Classifying Database Users Intrusion Prediction and Detection in Data Security", *Technical Gazette*, vol.27, no.6, 2020.(SCI-E)

2. Hakkoymaz V., Bakir C., "Information Flow Control with Decentralized Labeling Model in Information Security", Journal of Web Engineering , *Special Issue: Advanced in Al and Nature-inspired Approaches for Web Data Security*, vol.19, no.7-8, 1-23, 2020. ( SCI-E)

3. Bakir C., Hakkoymaz V., Guclu M., "Dağıtık Etiket Modeli ile Bilgi Akış Denetimi", *Bilecik Şeyh Edebali University Journal of Science*, vol.6, no.2, 231-242, 2019.

**Konferans Bildirileri**

**1**. Bakir C., Hakkoymaz V., "Veritabanı Güvenliğinde Saldırı Tahmini ve Tespiti için Kullanıcıların Sınıflandırılması", *VIII.th International Conference on Information Security and Cryptology,* 2015, pp.28-33.

**2**. Bakir C., Hakkoymaz V., "Dağıtık Veritabanında Veri Etiketleme ile Bilgi Akış Denetimi", *5.Ulusal Yüksek Hesaplamalı Başarım Konferansı*, 2017, pp. 1-6.

**3.** Bakir C., Hakkoymaz V., "Object based Distributed Enviroment Modelling and Simulation", *9th International Conference on Advanced Technologies*, 2020.