İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

GÜVENLİ ÇOĞA GÖNDERİMDE ANAHTAR AĞAÇLARI KULLANILARAK ANAHTAR YÖNETİMİ

YÜKSEK LİSANS TEZİ
Müh. Sibel YÜKSEL
504981153

Tezin Enstitüye Verildiği Tarih : 13 Mayıs 2002
Tezin Savunulduğu Tarih : 30 Mayıs 2002

Tez Danışmanı : Prof.Dr. Bülent ÖRENCİK
Diğer Jüri Üyeleri Prof.Dr. Ufuk ÇAĞLAYAN (B.Ü.)
Doç.Dr. Nadia ERDOĞAN

MAYIS 2002

ÖNSÖZ

Günümüzde Internet'in yaygınlaşması ile bilgisayar ve iletişim teknolojilerinin gelişmesi pek çok ağ uygulamasının önem kazanmasına sebep olmuştur. Ağ uygulamalarının pek çoğu, grup iletişimi modeline dayanmaktadır. Bu nedenle grup iletişimini güvenli hale getirmek kritik bir konu halini almıştır. Grup iletişim çeşitleri içerisinde çoğa gönderim, verinin bir grup katılımcıya en verimli şekilde aktarılmasını sağlaması nedeniyle ayrı bir yere sahiptir. Bu nedenle, literatürde çoğa gönderim grup iletişimini güvenli hale getirmek için yapılmış pek çok çalışma bulunmaktadır.

Bu çalışmada çoğa gönderim iletişimini güvenli hale getirmek için anahtar ağaç graflarından yararlanan yöntem incelenmiş ve diğer yöntemler ile karşılaştırılmıştır. Karşılaştırmaların yapılabilmesi için bir benzetim yazılımı geliştirilmiş, yöntemlerin fiziksel olarak uygulanabilirliğini ifade etmek için Çoğa Gönderimde Anahtar Yönetimi yazılımı geliştirilmiştir.

Tez çalışmam süresince yardımlarını esirgemeyen değerli danışman hocam Sayın Prof..Dr. Bülent ÖRENCİK'e ve her zaman yanımda olan aileme katkılarından dolayı teşekkür ederim.

Mayıs, 2002 Sibel YÜKSEL

İÇİNDEKİLER

KISALTMALAR **vi**
TABLO LİSTESİ **viii**
ŞEKİL LİSTESİ **x**
SEMBOL LİSTESİ **xii**
ÖZET **xiii**
SUMMARY **xv**

1. GİRİŞ **1**

2. BİLGİSAYAR İLETİŞİMİ **4**
2.1. Grup İletişimi 4
2.2. Grup İletişimi Tipleri 5
2.2.1. Teke Gönderim İletişimi 5
2.2.2. Çoğa Gönderim İletişimi 6
2.2.3. Çoktan Teke Gönderim İletişimi 7
2.2.4. Çokeşli Gönderim İletişim 8
2.2.5. Diğer Grup İletişim Tipleri 9
2.3. Çoğa Gönderim ve Teke Gönderim 11
2.4. Grup Karakteristikleri 13
2.2. Grup Adresleme ve Yönetimi 15

3. IP ORTAMINDA ÇOĞA GÖNDERİM **18**
3.1. Çoğa Gönderim Grupları 19
3.2. Grup Üyelik Protokolü 20
3.3. Çoğa Gönderim Yönlendirme Protokolü 20
3.3.1. Çoğa Gönderim Yönlendirmesi ve Teke Gönderim 20
3.4. Çoğa Gönderim Adresleme 21
3.4.1. Sınıf D Adresleri 21
3.4.2. Sınıf D Adresini IEEE-802 MAC Adresleri ile Eşleme 22
3.4.3. Paket Ömrü 24
3.5. Internet Grup Yönetim Protokolü (IGMP) 25
3.6. IPv6 Güvenlik Mekanizmaları 26
3.6.1. Anahtar ve SA Yönetimi 27
3.6.2. Asıllama Başlığı 28
3.6.3. Kozalama Güvenlik Yararlı Yükü 29
3.6.4. IPv6 Anahtar Yönetimi Gereksinimleri 29

4. GÜVENLİ ÇOĞA GÖNDERİM **30**

4.1. Çoğa Gönderim Hizmeti 30
4.2. Çoğa Gönderim Ortamında Güvenlik 33
4.2.1. Asıllama 33
4.2.2. Veri Bütünlüğü 34
4.2.3. Sırdaşlık 34
4.3. Güvenlik Gereksinimleri 35
4.4. Güvenli Çoğa Gönderim İşlemi 37
4.5. Güvenli Çoğa Gönderim Kriterleri 39
4.6. Çoğa Gönderimde Anahtar Yönetimi 41

5. ÇOĞA GÖNDERİMDE ANAHTAR YÖNETİMİ 44
5.1. Anahtar Yönetim Mimarisine İlişkin Hususlar 44
5.2. Anahtar Dağıtım Mimarileri 45
5.2.1. Elle Anahtar Dağıtma 46
5.2.2. N Kök/Yaprak Çiftli (Pairwise) Anahtar Yaklaşımı 47
5.2.3. Tümleyici Değişken Yaklaşımı 50
5.2.4. Hiyerarşik Ağaç Yaklaşımı 51
5.2.4.1. Çıkarma Prensibi 54
5.2.5. Güvenli Kilit 57
5.2.6. Dağıtık Kayıt ve Anahtar Dağıtma (DiRK) 58
5.3. Anahtar Tahsisi Mimarilerinin Karşılaştırılması 60
5.4. Grup Anahtar Yönetimi Konusunda Yapılmış Çalışmalar 61
5.4.1. GKMP 61
5.4.2. SMKD 62
5.4.3. Iolus 62
5.4.4. Anahtar Grafları 62

6. ANAHTAR GRAFLARI KULLANILARAK GÜVENLİ ÇOĞA GÖNDERİM 63
6.1. Güvenli Gruplar 65
6.1.1. Anahtar Grafları 65
6.1.2. Anahtar Graflarının Özel Tipleri 68
6.2. Yeniden Anahtar Tahsisi Stratejileri ve Protokolleri 69
6.2.1. Yıldız Anahtar Grafına Katılma 70
6.2.2. Yıldız Anahtar Grafından Ayrılma 71
6.2.3. Ağaç Anahtar Grafına Katılma 73
6.2.3.1. Kullanıcı-Odaklı Yeniden Anahtar Tahsisi 74
6.2.3.2. Anahtar-Odaklı Yeniden Anahtar Tahsisi 76
6.2.3.3. Grup-Odaklı Yeniden Anahtar Tahsisi 78
6.2.4. Ağaç Anahtar Grafından Ayrılma 80
6.2.4.1. Kullanıcı-Odaklı Yeniden Anahtar Tahsisi 81
6.2.4.2. Anahtar-Odaklı Yeniden Anahtar Tahsisi 82
6.2.4.3. Grup-Odaklı Yeniden Anahtar Tahsisi 83

6.3. Şifreleme ve Şifre Çözme Maliyetleri 85
6.4. Yeniden Anahtar Tahsisi Mesajlarının İmzalanması 86
6.5. Başarım Karşılaştırmaları 88
6.6. Anahtar Yönetimi Konusunda Yapılan Diğer Çalışmalar ile Bu Yaklaşımın Karşılaştırılması 93
6.6.1. Başarım 94
6.6.2. Güven 95
6.6.3. Güvenilirlik 95

7. ANAHTAR GRAFLARI KULLANILARAK GÜVENLİ ÇOĞA GÖNDERİM UYGULAMASI 96
7.1. Benzetim Yazılımı 96
7.2. Çoğa Gönderim Anahtar Yönetimi Yazılımı 117
7.2.1. Anahtar Sunucusu 119
7.2.2. Veri Sunucusu 122
7.2.3. İstemci 122

8. SONUÇLAR VE TARTIŞMA 126

KAYNAKLAR 129

EKLER 132

ÖZGEÇMİŞ 144

KISALTMALAR

MPEG : Motion Pictures Experts Group
CPU : Central Processin Unit
LAN : Local Area Network
TTL : Time-to-Live
IP : Internet Protocol
IANA : Internet Assigned Numbers Authority
MAC : Media Access Control
IEEE : Institue of Electrical and Electronics Engineers
IGMP : Internet Group Management Protocol
DVMRP : Distance Vector Multicast Routing Protocol
OSPF : Open Shortest Path First
MOSPF : Multicast Extentions Open Shortest Path First
AS : Autonomous System
CBT : Core Based Tree
PIM : Protocol Independent Multicast
PIM_DM : Protocol Independent Multicast-Dense Mode
PIM_SM : Protocol Independent Multicast-Spare Mode
MBONE : Multicast Backbone
UDP : User Datagram Protocol
RTP : Real Time Protocol
AH : Authentication Header
DSS : Digital Signature Standard
CA : Cetificate Authority
TCP : Transport Control Protocol
ESP : Encapsulating Security Protocol
ISAKMP : Internet Security Association and Key Management Protocol
SA : Security Association
SPI : Security Parameter Index
SAP : Session Announcement Protocol
SIP : Session Initiation Protocol
SDP : Session Description Protocol
ACL : Access Control List
GTEK : Group Traffic Encryption Key
GKEK : Group Key Encryption Key
KDC : Key Distribution Center
KEK : Key Encryption Key
CRT : Chinese Remainder Theorem
DiRK : Distributed Registration and Key Distribution
PRT : Participant Registration Tree
GKMP : Group Key Management Protocol
SMKD : Scalable Multicast Key Distribution
GSC : Group Security Controller

GSIs	: Group Security Intermediaries
MD5	: Message Digest 5
MKMA	: Multicast Key Management Application
ÇGAY	: Çoğa Gönderim Anahtar Yönetimi Uygulaması
DES	: Data Encryption Standart
DES-CBC	: Data Encryption Standart-Chipher Block Chaining
RSA	: Encryption Algorithm of Rivest, Shamir, and Adleman
DNS	: Domain Name System

TABLO LİSTESİ

		Sayfa No
Tablo 3.1	TTL eşik değerleri ve karşı düşen alanlar ……...	25
Tablo 4.1	Çoğa gönderim yönlendirme protokolleri …….....	31
Tablo 4.2	Güvenli çoğa gönderim mimarileri karşılaştırma kriterleri ……	40
Tablo 5.1	Saklama alanı ve iletim maliyetleri.....	55
Tablo 6.1	Sunucu ve her bir kullanıcı tarafından tutulan anahtar sayıları....	69
Tablo 6.2	Bir katılma/ayrılma isteğinin maliyeti ….....	85
Tablo 6.3	İstek başına ortalama maliyet …….....	86
Tablo 6.4	Ortalama yeniden anahtarlama mesaj uzunluğu ve sunucu işlem zamanı.....	88
Tablo 6.5	Sunucu tarafından yollanan yeniden anahtarlama mesajlarının (şifreleme ve imza ile birlikte) sayısı ve büyüklüğü…….....	90
Tablo 6.6	İstemci tarafından alınan yeniden anahtarlama mesaj (şifreleme ve imza ile birlikte) sayısı ve büyüklükleri …….....	92
Tablo 7.1	Çeşitli kullanıcılar için mesaj sayıları ve şifreleme maliyetleri (Her üç yöntem için).....	111
Tablo 7.2	Yeniden Anahtar Tahsisi mesaj yapısı komutları.....	121
Tablo A.1	Grup-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı=4).....	132
Tablo A.2	Grup-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı=8).....	132
Tablo A.3	Grup-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı=16).....	133
Tablo A.4	Grup-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı=32).....	133
Tablo A.5	Grup-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı=64).....	134
Tablo A.6	Anahtar-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı=4).....	135
Tablo A.7	Anahtar-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı=8)	135
Tablo A.8	Anahtar-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı=16)	136
Tablo A.9	Anahtar-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı=32)	137
Tablo A.10	Anahtar-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı=64)	138
Tablo A.11	Kullanıcı-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı=4)	139
Tablo A.12	Kullanıcı-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı=8)	140

Tablo A.13 Kullanıcı-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı=16) 140
Tablo A.14 Kullanıcı-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı=32) 141
Tablo A.15 Kullanıcı-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı=64) 142

ŞEKİL LİSTESİ

		Sayfa No
Şekil 2.1	**:** Teke gönderim iletişimi.....	5
Şekil 2.2	**:** Çoğa gönderim iletişimi.....	7
Şekil 2.3	**:** Çokeşli gönderim iletişim.....	8
Şekil 2.4	**:** Grup iletişimi tipleri.....	9
Şekil 2.5	**:** Tekten rastgele teke gönderim iletişimi.....	10
Şekil 2.6	**:** Teke gönderim kullanarak yapılan grup iletişimi.....	11
Şekil 2.7	**:** (a) Açık grup (b) Kapalı grup.....	13
Şekil 3.1	**:** Çoğa gönderim IP dağıtım servisi.....	21
Şekil 3.2	**:** Sınıf D tipi IP adres formatı.....	21
Şekil 3.3	**:** Sınıf D adresinin Ethernet adresine karşı düşürülmesi.....	23
Şekil 3.4	**:** Aynı IEEE-802 MAC-katmanı adresine karşı düşen Sınıf D adresleri.....	24
Şekil 4.1	**:** Güvenli grup içerisindeki konaklar.....	36
Şekil 5.1	**:** Mantıksal anahtar dağıtım mimarisi.....	53
Şekil 6.1	**:** Anahtar Grafı.....	66
Şekil 6.2	**:** Katılma öncesi ve sonrasında yıldız anahtar grafı.....	71
Şekil 6.3	**:** Ayrılma öncesi ve sonrasında yıldız anahtar grafı.....	72
Şekil 6.4	**:** Katılma (ayrılma) öncesi ve sonrası bir anahtar ağacı.....	75
Şekil 7.1	**:** Başlangıç Anahtar Tahsisi Mesaj Sayıları (Kullanıcı Sayısı = 32).	100
Şekil 7.2	**:** Başlangıç Anahtar Tahsisi Şifreleme Maliyetleri (Kullanıcı Sayısı = 32).....	100
Şekil 7.3	**:** Başlangıç Anahtar Tahsisi Mesaj Sayıları (Kullanıcı Sayısı = 64).	101
Şekil 7.4	**:** Başlangıç Anahtar Tahsisi Şifreleme Maliyetleri (Kullanıcı Sayısı = 64).....	101
Şekil 7.5	**:** Başlangıç Anahtar Tahsisi Mesaj Sayıları (Kullanıcı Sayısı = 128).....	102
Şekil 7.6	**:** Başlangıç Anahtar Tahsisi Şifreleme Maliyetleri (Kullanıcı Sayısı = 128).....	102
Şekil 7.7	**:** Yeniden Anahtar Tahsisi Ortalama Mesaj Sayıları (Katılma) (Kullanıcı Sayısı = 32).....	103
Şekil 7.8	**:** Yeniden Anahtar Tahsisi Ortalama Şifreleme Maliyetleri (Katılma) (Kullanıcı Sayısı = 32).....	103
Şekil 7.9	**:** Yeniden Anahtar Tahsisi Ortalama Mesaj Sayıları (Ayrılma) (Kullanıcı Sayısı = 32).....	103
Şekil 7.10	**:** Yeniden Anahtar Tahsisi Ortalama Şifreleme Maliyetleri (Ayrılma) (Kullanıcı Sayısı = 32).....	104
Şekil 7.11	**:** Yeniden Anahtar Tahsisi Ortalama Mesaj Sayıları (Katılma) (Kullanıcı Sayısı = 64).....	104
Şekil 7.12	**:** Yeniden Anahtar Tahsisi Ortalama Şifreleme Maliyetleri (Katılma) (Kullanıcı Sayısı = 64).....	105

Şekil 7.13 **:** Yeniden Anahtar Tahsisi Ortalama Mesaj Sayıları (Ayrılma) (Kullanıcı Sayısı = 64)..... 105
Şekil 7.14 **:** Yeniden Anahtar Tahsisi Ortalama Şifreleme Maliyetleri (Ayrılma) (Kullanıcı Sayısı = 64)..... 105
Şekil 7.15 **:** Yeniden Anahtar Tahsisi Ortalama Mesaj Sayıları (Katılma) (Kullanıcı Sayısı = 128)..... 106
Şekil 7.16 **:** Yeniden Anahtar Tahsisi Ortalama Şifreleme Maliyetleri (Katılma) (Kullanıcı Sayısı = 128)..... 106
Şekil 7.17 **:** Yeniden Anahtar Tahsisi Ortalama Mesaj Sayıları (Ayrılma) (Kullanıcı Sayısı = 128)..... 106
Şekil 7.18 **:** Yeniden Anahtar Tahsisi Ortalama Şifreleme Maliyetleri (Ayrılma) (Kullanıcı Sayısı = 128)..... 107
Şekil 7.19 **:** Son Anahtar Tahsisi Mesaj Sayıları (Kullanıcı Sayısı = 32)..... 108
Şekil 7.20 **:** Son Anahtar Tahsisi Şifreleme Maliyetleri (Kullanıcı Sayısı = 32)..... 108
Şekil 7.21 **:** Son Anahtar Tahsisi Mesaj Sayıları (Kullanıcı Sayısı = 64)..... 109
Şekil 7.22 **:** Son Anahtar Tahsisi Şifreleme Maliyetleri (Kullanıcı Sayısı = 64)..... 109
Şekil 7.23 **:** Son Anahtar Tahsisi Mesaj Sayıları (Kullanıcı Sayısı = 128)..... 110
Şekil 7.24 **:** Son Anahtar Tahsisi Şifreleme Maliyetleri (Kullanıcı Sayısı = 128)..... 110
Şekil 7.25 **:** Yeniden Anahtar Tahsisi Mesaj Sayıları (Kullanıcı Sayısı = 1024)..... 111
Şekil 7.26 **:** Yeniden Anahtar Tahsisi Şifreleme Maliyetleri (Kullanıcı Sayısı = 1024)..... 106
Şekil 7.27 **:** Grup-Odaklı Yeniden Anahtar Tahsisi (Kullanıcı Sayısı = 4096).. 113
Şekil 7.28 **:** Anahtar-Odaklı Yeniden Anahtar Tahsisi (Kullanıcı Sayısı = 4096)..... 113
Şekil 7.29 **:** Kullanıcı-Odaklı Yeniden Anahtar Tahsisi (Kullanıcı Sayısı = 4096)..... 114
Şekil 7.30 **:** Grup-Odaklı Yeniden Anahtar Tahsisi (Kullanıcı Sayısı = 8192).. 114
Şekil 7.31 **:** Anahtar-Odaklı Yeniden Anahtar Tahsisi (Kullanıcı Sayısı = 8192)..... 115
Şekil 7.32 **:** Kullanıcı-Odaklı Yeniden Anahtar Tahsisi (Kullanıcı Sayısı = 8192)..... 115
Şekil 7.33 **:** Grup-Odaklı Başlangıç Anahtar Tahsisi (Kullanıcı Sayısı = 16384) 116
Şekil 7.34 **:** Anahtar-Odaklı Başlangıç Anahtar Tahsisi (Kullanıcı Sayısı = 16384)..... 116
Şekil 7.35 **:** Kullanıcı-Odaklı Başlangıç Anahtar Tahsisi (Kullanıcı Sayısı = 16384)..... 116
Şekil 7.36 **:** Yeniden anahtar tahsisi mesaj yapısı..... 121
Şekil 7.37 **:** Dört kullanıcıdan oluşan anahtar ağaç grafı..... 124

SEMBOL LİSTESİ

I1, I2..,In	**:** Ara sistemler
n	**:** Kullanıcı sayısı
k_x , k_x^{-1}	**:** x kullanıcısının kamusal-özel anahtar çifti
U	**:** Kullanıcı kümesi
K	**:** Anahtar kümesi
R	**:** U ve K arasındaki kullanıcı-anahtar ilişkisi
s	**:** Sunucu
u	**:** Kullanıcı

GÜVENLİ ÇOĞA GÖNDERİMDE ANAHTAR AĞAÇLARI KULLANILARAK ANAHTAR YÖNETİMİ

ÖZET

Bilgisayar ve iletişim teknolojilerinin gelişmesi grup iletişimi modeline dayanan ağ uygulamalarının (örneğin; bilgisayar destekli konferans sistemleri, bilgi sistemleri, dağıtık etkileşimli benzetim uygulamaları, haber sitelerindeki çoğul ortam yayınları, açık arttırma uygulamaları vb.) önem kazanmasına sebep olmuştur. Bunun doğal bir sonucu olarak, gizdeşlik ve asıllama servisleri ile birlikte, grup üyeleri arasında yollanan mesajların veri bütünlüğünü sağlayarak, grup iletişimini güvenli hale getirmek önemli bir iletişim konusu olmuştur.

Grup iletişim çeşitleri içerisinde çoğa gönderim, verinin bir grup katılımcıya en verimli şekilde aktarılmasını sağlaması nedeniyle, diğer yöntemlere göre daha kullanışlıdır. Bu nedenle çoğa gönderim iletişimini güvenli hale getirmek yapılan çalışmalar diğer yöntemlerin güvenli hale getirilmesi için yapılan çalışmalardan daha fazladır. Ancak, çoğa gönderim grup iletişimini güvenli hale getirmek için yapılmış çalışmaların pek çoğunda ölçeklenebilirlik problem olmuştur.

Bu tez çalışmasında, çoğa gönderim iletişimini güvenli hale getirmek için anahtar ağaç graflarından yararlanan yeniden anahtar tahsisi yöntemleri incelenmiştir. Güvenli grup; U kullanıcı kümesini, K kullanıcıların sahip oldukları anahtarların kümesini, R ise kullanıcı-anahtar ilişkisini göstermek üzere (U, K, R) üçlüsü ile ifade edilmektedir. Ardından, güvenli grupları açıkça belirtmek için, anahtar grafları tanıtılmaktadır. Anahtar graflarının özel bir çeşidi olan anahtar ağaçları için, ayrılma/katılma sonrasında yeniden anahtar tahsisi mesajlarını güvenli olarak dağıtmak için üç yöntem tanıtılmış ve güvenli gruptan ayrılma ve katılma için protokoller tanımlanmıştır. Bu üç yeniden anahtar tahsisi yöntemi; Kullanıcı-Odaklı Yeniden Anahtar Tahsisi, Anahtar-Odaklı Yeniden Anahtar Tahsisi ve Grup-Odaklı Yeniden Anahtar Tahsisi'dir.

Yeniden anahtar tahsisi yöntemleri ve ayrılma/katılma protokolleri bir benzetim yazılımında ve bir demo sisteminde uygulanmıştır.Yapılan benzetim çalışmaları, bu üç yöntemden Grup-Odaklı Yeniden Anahtar Tahsisi yaklaşımının sunucu tarafında en iyi başarımı sağladığını göstermiştir. İstemci tarafında ise, Kullanıcı-Odaklı Yeniden Anahtar Tahsisi en iyi başarımı sağlamıştır. Yapılan benzetim çalışmaları, yukarıdaki üç yöntemden hangisi kullanılırsa kullanılsın, anahtar ağaç graflarını kullanan yöntemlerin oldukça büyük gruplar için bile ölçeklenebilir olduğunu göstermiştir.

Benzetim yazılımı dışında, yöntemlerin uygulandığı bir demo programı da geliştirilmiştir. "Çoğa Gönderim Anahtar Yönetimi Uygulaması" adı verilen bu yazılım üç parçadan oluşmaktadır: Anahtar Sunucusu, Veri Sunucusu ve İstemci.

Anahtar Sunucusu, çoğa gönderim iletişimini güvenli hale getirmek için kullanılan grup anahtarının ve yeniden anahtar tahsisi mesajlarının şifrelenmesinde kullanılan şifreleme anahtarlarının tutulduğu anahtar grafının oluşturulmasından, yeni anahtarların yaratılmasından ve yeniden anahtar tahsisi mesajlarının yollanmasından sorumludur.

Veri Sunucusu, güvenli grubun oluşturulmasından sonra bu gruba, grup anahtarının son hali ile şifrelenen mesajların yollanmasını sağlar.

İstemci programı, güvenli gruba katılmak isteyen kullanıcıların, güvenli gruba üye olmaları ve gruba yollanan şifrelenmiş mesajları alarak mesajını çözdükten sonra veriyi alabilmelerini sağlayan programdır.

"Çoğa Gönderim Anahtar Yönetimi Uygulaması" anahtar ağaç graflarını kullanan yeniden anahtar tahsisi yöntemlerinin bir prototip olarak uygulandığı yazılımdır. İleriki çalışmalarda, bu yazılımın çoğa gönderim yazılımları ile entegre edilmesi suretiyle, tam bir çoğa gönderim sistemi oluşturulması mümkündür.

SECURE MULTICAST USING KEY TREE GRAPHS FOR KEY MANAGEMENT

SUMMARY

Because of developments in computer and communication technologies, network applications based on the group communication model (e.g., computer supported conference systems, information systems, distributed interactive simulation applications, push technologies) have become important networking issues in recent years.

In Group Communication types, multicast is the most efficient way of sending data to a set of users. As a result, most of the research is done in securing multicast communication as opposed to securing other types of grup communication. However, scalability is the main problem of this research.

In this thesis, rekeying strategies for securing multicast communications using key graphs are examined. The notation of a secure group is formalized as a set of three labels (*U*, *K*, *R*) where *U* denotes a set of users, *K*--a set of keys held by the users, and *R*--a user-key relation. Then, key graphes are introduces to specify secure groups. For a special class of key graphs, called "Key Trees", three strategies for securely distributing rekey messages after a join/leave are presented and protocols for joining and leaving a secure group are specified. These three rekeying strategies are User-Oriented Rekeying, Key-Oriented Rekeying and Group-Oriented Rekeying.

The rekeying strategies and join/leave protocols are implemented both in simulation software and in demo prototype software. Simulation results from experiments that were done by using simulation software are presented. These results shows that on the server site, Group-Oriented Rekeying provides the best performance. On the client side, User-Oriented Rekeying provides the best performance. On the other hand, the method presented in this thesis, using any of these three rekeying strategies, are scalable to large groups with frequent joins and leaves.

In addition to the simulation software, a demo prototype software called "Multicast Key Management Application" was developed with the implementation of these rekeying strategies. This demo prototype software has three parts: Key Server, Data Server and, Multicast Client.

The Key Server is responsible for building a key tree graph which holds key encryption keys and group traffic encryption key, creating new keys and, sending rekeying messages.

The Data Server is responsible for sending messages that are encrypted with the latest group traffic encryption key.

Multicast Client is for users who want to be a member of the secure group and receiving and decrypting messages that are sent to the secure group.

“Multicast Key Management Application” is a prototype program in which rekeying strategies based on key tree graphes are implemented. In future work, a complete multicast system can be developed by integrating this software with some other multicast applications.

1. GİRİŞ

Bilgisayar teknolojisindeki her yeni gelişme, onun kullanımını daha da arttırmış ve ondan beklenen imkanları hızla çeşitlendirmiştir. Bu imkanlardan biri de bilgisayarların birbirleri ile haberleşerek, sahip oldukları kaynakları paylaşmaları ve kullanıcıları arasında mesaj iletiminin sağlanmasıdır. [1]

Internet, çoğulortamlılık ve bilgi otoyolu günümüz teknolojisinin bazı anahtar kelimeleridir. Bilgisayar ve iletişim teknolojilerinin gelişmesi grup iletişimi modeline dayanan ağ uygulamalarının (örneğin; bilgisayar destekli konferans sistemleri, bilgi sistemleri, dağıtık etkileşimli benzetim uygulamaları, haber sitelerindeki çoğul ortam yayınları, açık arttırma uygulamaları vb.) önem kazanmasına sebep olmuştur. Bunun doğal bir sonucu olarak, gizdeşlik ve asıllama servisleri ile birlikte, grup üyeleri arasında yollanan mesajların veri bütünlüğünü sağlayarak, grup iletişimini güvenli hale getirmek önemli bir iletişim konusu olmuştur.

Internet, çoğulortam ve bilgi otoyolu günümüz teknolojisinin bazı anahtar kelimeleridir. Bilgisayar ve iletişim teknolojilerinin gelişmesi grup iletişimi modeline dayanan ağ uygulamalarının (örneğin; bilgisayar destekli konferans sistemleri, bilgi sistemleri, dağıtık etkileşimli benzetim uygulamaları, haber sitelerindeki çoğul ortam yayınları, açık arttırma uygulamaları vb.) önem kazanmasına sebep olmuştur. Bunun doğal bir sonucu olarak, gizdeşlik ve asıllama servisleri ile birlikte, grup üyeleri arasında yollanan mesajların veri bütünlüğünü sağlayarak, grup iletişimini güvenli hale getirmek önemli bir iletişim konusu olmuştur.

Grup iletişimi tipleri arasında çoğa gönderim bir grup kullanıcıya verinin en verimli şekilde yollanmasını sağlaması nedeniyle diğer yöntemlere göre daha kullanışlıdır. Bu nedenle son yıllarda çoğa gönderimin güvenli hale getirilmesi önem kazanmıştır.

Bu çalışmada, Grup İletişimi'nin özel bir şekli olan çoğa gönderimde güvenliğin sağlanabilmesi için uygulanabilecek hiyerarşik bir anahtar yönetimi yöntemi incelenmiştir. Yöntemlerin ölçeklenebilir olduğunu ve birbirleri ile

karşılaştırılabilmelerini sağlamak amacıyla bir benzetim yazılımı geliştirilmiştir. Benzetim programından elde edilen sonuçlar kullanılarak başlangıç, yeniden ve son anahtar dağıtma işlemleri için değişik grup büyüklükleri için yöntemlerin birbirleri ile karşılaştırılması sağlanmıştır.

Benzetim yazılımı dışında, yöntemlerin uygulandığı bir demo programı da geliştirilmiştir. "Çoğa Gönderim Anahtar Yönetimi Uygulaması" adı verilen bu yazılım üç parçadan oluşmaktadır: Anahtar Sunucusu, Veri Sunucusu ve İstemci. Bu modüllerin yapıları verilmiş ve bu uygulama kullanılarak bir çoğa gönderim sisteminin nasıl kurulabileceği anlatılmıştır.

Tez kitapçığının ikinci bölümünde "Bilgisayar İletişimi" ve "Grup İletişimi" konuları ele alınmıştır. Grup İletişimi tipleri açıklanmış ve bu tiplerden teke gönderim ile çoğa gönderim iletişimleri karşılaştırılmıştır. Grup karakteristiklerine değinilmiş, grup adresleme ve yönetiminden bahsedilmiştir.

Üçüncü bölümde "IP Ortamında Çoğa Gönderim" ele alınmıştır. Çoğa gönderim adresleme, IP Sınıf D adresleri ve bu adreslerin IEEE-802 MAC katmanı adreslerine nasıl karşı düşürüleceği anlatılmıştır. "IPv6 için Güvenlik Gereksinimleri" belirtilmiştir.

Dördüncü bölümde "Güvenli Çoğa Gönderim" anlatılmıştır. Çoğa gönderim ortamında güvenliğin sağlanması için verilmesi gereken hizmetlerden (asıllma, veri bütünlüğü ve sırdaşlık) bahsedilmiştir. Bir çoğa gönderim grubunun kurulması için yapılması gereken işlemlerin üzerinden adım adım geçilmiştir.

Beşinci bölüm "Çoğa Gönderimde Anahtar Yönetimi"ni içerir. Bu bölümde iyi ve ölçeklenebilir bir anahtar dağıtma mimarisinin belirlenmesinde kullanılacak kriterler verilmiş ve şimdiye kadar üzerinde çalışılmış olan anahtar dağıtma mimarilerinden bahsedilmiştir.

Altıncı bölümde "Anahtar Grafları Kullanılarak Güvenli Çoğa Gönderim" anlatılmaktadır. Güvenli grup tanımı yapılmış ve anahtar grafları tanımlanmıştır. Anahtar kapsama problemini en kolay şekilde çözebilmek için anahtar graf çeşitleri verilmiştir. Anahtar grafları kullanılarak anahtar dağıtma yöntemi hiyerarşik anahtar dağıtma mimarilerinin özel bir halidir. Anahtar ağaç graflarında katılma/ayrılma

durumları için yeniden anahtar dağıtma için protokolleri tanımlanmıştır. Anahtar ağaç graflarını kullanan yaklaşımın diğer yaklaşımlar ile karşılaştırması yapılmıştır.

Yedinci bölüm anahtar ağaç graflarını kullanan yeniden anahtar dağıtma yöntemlerinin karşılaştırılması için gerçekleştirilen benzetim yazılımının tanıtıldığı bölümdür. Benzetim yazılımından elde edilen sonuçlar ve karşılaştırma grafikleri verilmiş hangi yöntemin diğerlerinden başarım açısından üstün olduğu belirtilmiştir.

Son bölüm "Sonuçlar ve Tartışma" kısmında tez çalışmasında yapılanlar özetlenmiş, elde edilen sonuçlar verilmiştir. İleriki çalışmalarda bu çalışmada anlatılan yöntemlerden nasıl yararlanılabileceği anlatılmıştır.

2. BİLGİSAYAR İLETİŞİMİ

Bilgisayar iletişimini karakterize eden belirleyici nitelik, iki bilgisayar arasındaki veri değiş tokuşudur. Bu konu, telefonun gelişimine paralellik göstermektedir. Telefonun gelişiminde de başlangıçta, iki kişinin birbiri ile konuşabilmesi söz konusu olmuştur. Günümüzde ise konferans çağrıları, birkaç kişinin dahil olduğu telefon görüşmeleri yapılmasına imkan tanımaktadır. [2]

Yakın zamanda bilgisayar iletişimi alanında kaydedilen önemli teknik gelişmeler, bir grup içerisindeki bilgisayarların birbirleri ile iletişim kurmalarına imkan sağlamıştır. Bu şekildeki iletişime "*Grup İletişimi"* adı verilmektedir. Eldeki veriler, Grup İletişimi'nin gelecekte çok önemli bir rol oynayacağını göstermektedir. [2]

2.1. GRUP İLETİŞİMİ

Bilgisayar iletişimi, en az iki bilgisayar arasında veri alışverişini sağlamak üzere, tüm sayısal iletişim konularını içerir; iki bilgisayar ya da bir bilgisayar ile bir yazıcının doğrudan bağlanarak noktadan noktaya doğrudan iletişimde bulunması da bilgisayar iletişimi konusuna girer. [1]

Bir grup içerisindeki bilgisayarların birbirleri ile iletişim kurmalarına "*Grup İletişimi*" adı verilmektedir. Grup içerisindeki bilgisayarların, birbirleri ile haberleşebilmeleri için kullanılabilecek değişik yöntemler bulunmaktadır. Ancak, bilgiyi grup içerisindeki her bir üyeye ayrı ayrı yollamak yerine, tüm gruba aynı anda yayınlamak çok daha verimlidir. Bu durumu daha iyi anlamak için bir örnek ele alalım: Bir konferans sırasında konuşmacı, her bir dinleyiciye ayrı sunum yapmasına göre daha az çaba sarfederek, anlatmak istediği konuyu tüm dinleyicilere aynı anda aktarabilir.

Grup İletişimi için kullanılabilecek yöntemler arasında *"Çoğa Gönderim"* özel bir yere sahiptir. Çoğa gönderim, bazı alanlarda teknik olarak uygulanmış olduğu halde pek çok

alanda hala araştırmaya açık görünmektedir. “Güvenli Çoğa Gönderim” araştırmaya açık alanlardan biridir.

2.2. GRUP İLETİŞİMİ TİPLERİ

İletişime dahil olan gönderici ve alıcıların sayısına göre, Grup İletişimini çeşitli tiplere ayırmak mümkündür. Bunlar:

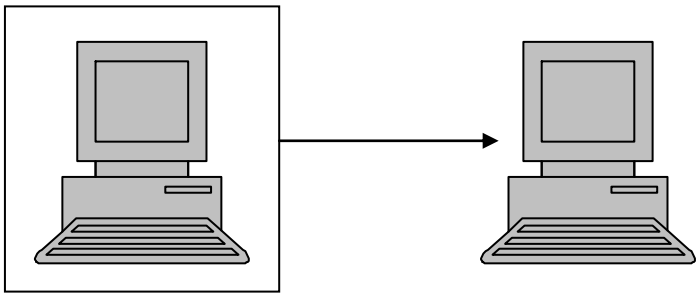
- Teke gönderim (Unicast) (1:1)
- Çoğa gönderim (Multicast) (1:n)
- Çoktan teke gönderim (Concast) (m:1)
- Çokeşli/çoknoktalı gönderim (Multipeer/Multipoint) (m:n)

dir.

Parantez içerisindeki gösterilimde ilk rakam gönderici sayısını, ikinci rakam ise alıcı sayısını ifade etmektedir. Tek gönderici ya da tek alıcı olan özel durumlar “1” rakamı ile gösterilmiştir. [2] Noktadan-noktaya iki kişi arasındaki iletişim, grup iletişiminin özel bir durumu olarak görülebilir.

2.2.1. TEKE GÖNDERİM İLETİŞİMİ

Teke gönderim, bir gönderici ve bir alıcının olduğu geleneksel noktadan noktaya iletişime (1:1) eşdeğerdir. Sonuç olarak, veri tek yönde değiş tokuş edilir (Şekil 2.1).



Şekil 2.1 Teke gönderim iletişimi

Verinin çift taraflı değiş tokuşu, iki teke gönderim iletişimin olmasını gerektirir. Burada, verinin tek taraflı olarak aktığı, ancak, iletişimin teknik olarak kontrol edilmesi için

gerekli olan kontrol verilerinin iletişim yönüne ters olarak iletilebileceği dikkate alınmalıdır. Örneğin alıcı, veriyi doğru ve tam olarak aldığını bildirmek için göndericiye bir alındı yollayabilir.

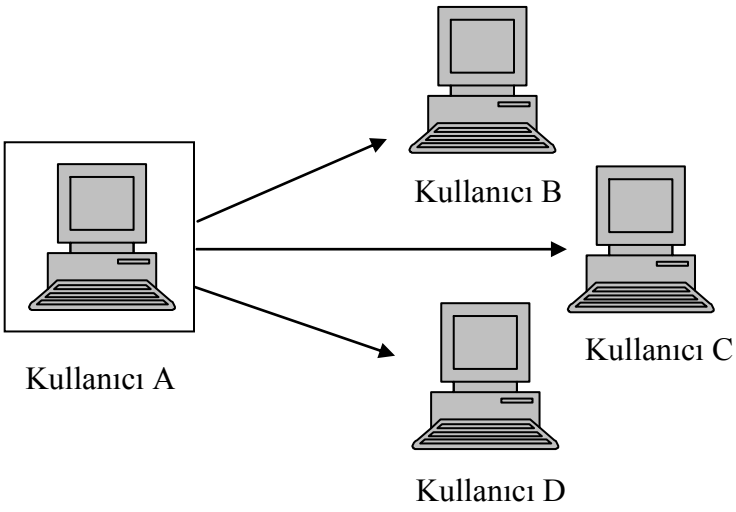
Teke gönderim teriminin asıl anlamı yukarıda açıklandığı şekilde de olsa literatürde, çoğunlukla iki taraf arasında çift yönlü veri değiş tokuşunu ifade etmek için kullanılmaktadır. Teke gönderimde önemli olan, iletişimin iki iletişim tarafı ile sınırlı olmasıdır.

Teke gönderim grup iletişimini desteklemek için kullanılmak istenirse, her iki grup üyesi arasında iki adet teke gönderim iletişim bağlantısının kurulması gerekecektir. Grup içerisinde *n* adet üyenin bulunması durumunda, her bir grup üyesinin gönderici olabildiği çokeşli gönderim yapılması için en çok $n(n-1)$ iletişim bağlantısı, çoğa gönderim yapılması için ise $\frac{n(n-1)}{2}$ iletişim bağlantısı kurulması gerekecektir. Bu, çok büyük gruplar için olası bir seçenek olarak gözükmemektedir. Başka bir değişle, grup büyüklüğüne göre ölçeklenirlik yoktur.

Teke gönderim iletişimine örnek olarak, bir göndericiden bir alıcıya mektup yollanması verilebilir. Eğer alıcı göndericiye cevap yollar ise, bu taktirde, ters yönde başka yeni bir iletişim ilişkisi kurulmuş olacaktır. Bu açıdan bakacak olursak, telefon iletişimi bir teke gönderim iletişimi olarak kabul edilemez. Çünkü, bir telefon çağrısı ters yönlerde iki teke gönderim iletişiminden oluşur.

2.2.2. ÇOĞA GÖNDERİM İLETİŞİMİ

Çoğa gönderim iletişiminde tek bir gönderici, veriyi, bir veya birden fazla alıcıya yollar. Tek bir alıcının bulunduğu durum, çoğa gönderimin özel bir halidir. Bu nedenle, çoğa gönderim, teke gönderimin bir uzantısını oluşturur ve *1:n* olarak ifade edilir. Şekil 2.2'de görüldüğü gibi Kullanıcı A; Kullanıcı B, Kullanıcı C ve Kullanıcı D'ye veri yollayabilmektedir. Kullanıcı A haricindeki diğer grup üyeleri, Kullanıcı A'nın yolladığı verileri alabilmelerine rağmen diğer üyelere veri yollayamamaktadırlar. Bu durum, iletişimin tek yönlü olduğu başka bir şekildir. Teke gönderimde olduğu gibi, iletişimin doğruluğunu güvence altına almak için, kontrol verileri iletişimin tersi yönde yollanabilir.



Şekil 2.2 Çoğa gönderim iletişimi

Farklı çoğa gönderim uygulamalarını aşağıdaki şekilde gruplamak mümkündür [7]:

- Tekten-Çoğa: Bir kaynağın birden fazla alıcıya veri yolladığı durumdur. Örnek: Haber sitelerindeki çoğul ortam yayını.
- Çoktan-Teke: Birden fazla göndericinin tek bir alıcıya veri yolladığı durumdur. Örnek: Kaynak bulma, veri toplama ve açık arttırma uygulamaları.
- Çoktan-Çoğa: Birden fazla göndericinin birden fazla alıcıya veri yolladığı durumdur. Örnek: Dağıtık oyun oynama ve konferans uygulamaları.

Bu çalışma boyunca çoğa gönderim terimi ile, tek bir göndericinin ve birden fazla alıcının olduğu (tekten-çoğa) çoğa gönderim durumları ifade edilecektir.

Çoğa gönderim iletişiminin en tipik örneği, bilgisayar-destekli konferans sistemi kullanılarak yapılan, bir konferans sunumudur. Sunum, her biri farklı fiziksel bölgelerde olabilen konferans katılımcılarına yollanmaktadır. Alıcıların bu ortamda, diğer konferans katılımcılarına ve de sunumu yapan kişiye bir şey yollamaları söz konusu değildir.

2.2.3. ÇOKTAN TEKE GÖNDERİM İLETİŞİMİ

Çoktan teke gönderim iletişiminde, birden çok gönderici tek bir alıcıya veri gönderebilmektedir.

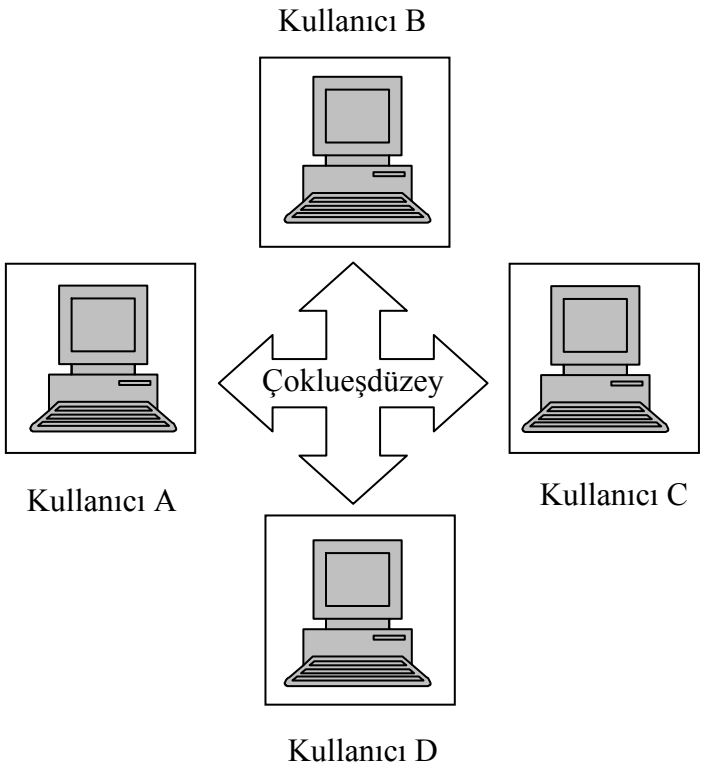
Çoktan teke gönderim iletişimine örnek olarak, benzetim sonuçlarının birden fazla bilgisayardan tek bir alıcı bilgisayara yollandığı ve burada işlenerek çeşitli raporların üretildiği durumu verebiliriz. Burada, yollanan verilerin eşzamanlı olması gerekmemektedir.

Çoktan teke gönderim iletişimi, yaygın olarak uzaktan öğrenme alanında, öğrencilerin ev ödevlerini öğretmenlerine yollamalarında kullanılmaktadır.

2.2.4. ÇOKEŞLİ GÖNDERİM İLETİŞİMİ

Çokeşli gönderim iletişimi, aynı alıcı kümesine birden fazla göndericinin veri yolladığı durumdur (Şekil 2.3). Bu durum, *m:n* tipi iletişime karşılık gelir ve çoğunlukla *"Çokeşli/Çoknoktalı Gönderim İletişimi"* olarak adlandırılır.

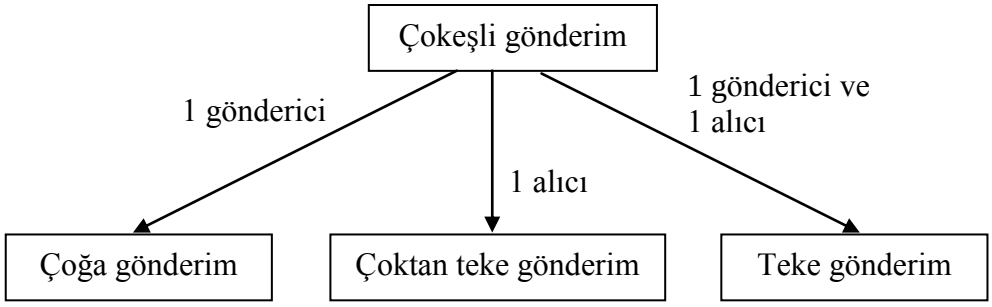
Örneğin, bir proje ekibini ele alalım. Her biri farklı yerlerde bulunan proje ekibi üyelerinin, bir konuyu tartışmak istemeleri durumunda, çokeşli iletişim gerekli olacaktır. Bu durumda, her bir ekip üyesinin kendi fikrini bildirebilmesi gerekecektir. Bu nedenle, her bir proje ekibi üyesi potansiyel bir göndericidir. Aynı zamanda her bir üye, diğer üyelerin yolladığı verilerin alıcısı durumundadırlar.



Şekil 2.3 Çokeşli gönderim iletişimi

Çokeşli iletişim, uygulanması en zor olan iletişim çeşididir. [2]

Şekil 2.4'te, bu noktaya kadar anlatılan iletişim tipleri ile ilgili bir gözden geçirme verilmektedir. Çokeşli, en genel iletişim tipi olarak, en tepede yer almaktadır. Diğer iletişim şekillerinin, gönderici ve alıcı sayıları cinsinden kısıtları, şekil üzerindeki okların yanına not edilmiştir. Bu kısıtlar ile, çokeşli iletişimden çoğa gönderim, çoktan teke gönderim ve teke gönderim türetilmiştir. [2]



Şekil 2.4 Grup iletişimi tipleri

2.2.5. DİĞER GRUP İLETİŞİM TİPLERİ

Yukarıda anlatılan tüm iletişim tipleri, teke gönderim iletişimi iki iletişim ortağından daha fazlasını içermemesi nedeniyle bir istisnai durum olarak görülmesine rağmen, grup iletişimi şemsiyesi altında yer alır.

Günümüzde, yukarıda anlatılanların dışında, başka grup iletişimi şekilleri de kullanılmaktadır. Bunlar:

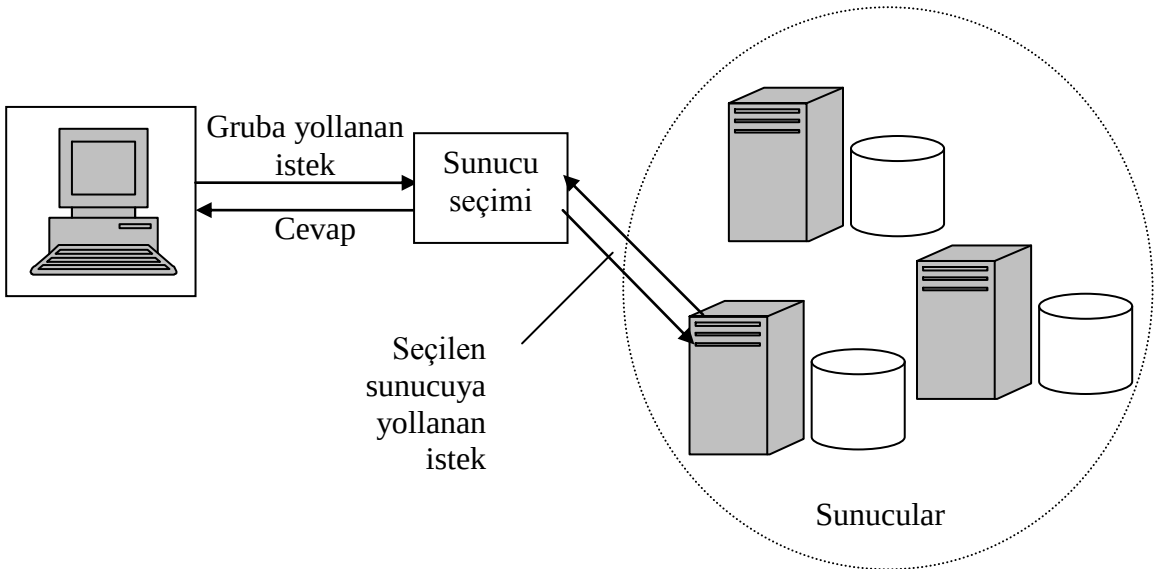
- Tekten rastgele teke gönderim (Anycast)
- Yayın (Broadcast)

olarak iki başlıkta ifade edilebilir.

Tekten rastgele teke gönderim de grup kavramını kullanır. Ancak burada grup, veri değiş tokuşu için kullanılmaz. Veri aktarımı, mevcut teke gönderim mekanizmaları ya da yeni tekten rastgele teke gönderim mekanizmaları ile sağlanır. Fakat alıcı, bir grup potansiyel aday arasından seçilir.

Tipik bir tekten rastgele teke gönderim uygulaması Şekil 2.5'te verilmiştir. Burada da görüldüğü gibi, sistemde bir grup sunucu yer almaktadır. Kullanıcı bir sorgulama yapmak istediğinde, sorgu, tüm grup yerine grup içerisindeki tek bir sunucuya, teke gönderim ya da tekten rastgele teke gönderim yönlendirme mekanizmaları yardımıyla gönderilir. Seçilen sunucu uygun cevabı verir. Kullanıcı, gerçekte sorguyu bir grup sunucuya yollamış olduğu halde, ağ içerisindeki seçim mekanizması tek bir sunucuyu seçerek, sorguyu bu sunucuya teke gönderim aracılığı ile yönlendirmiştir. Grup içerisindeki diğer sunucular sorgulama sürecine dahil olmamaktadır.

Tekten rastgele teke gönderimde, iletişim sistemi, uygun olası alıcılar grubu içerisinden, bir alıcıyı seçmektedir. Kullanıcının bu seçim işlemi sırasında hiçbir etkisi olmamakta ve hatta uygun sunucu grubunu bilmemektedir. Kullanıcı sadece, her bir başarılı tekten rastgele teke gönderim çağrısı için, farklı sunucuların grup içerisinden hedef sunucu olarak seçilebildiğinin farkına varmaktadır.

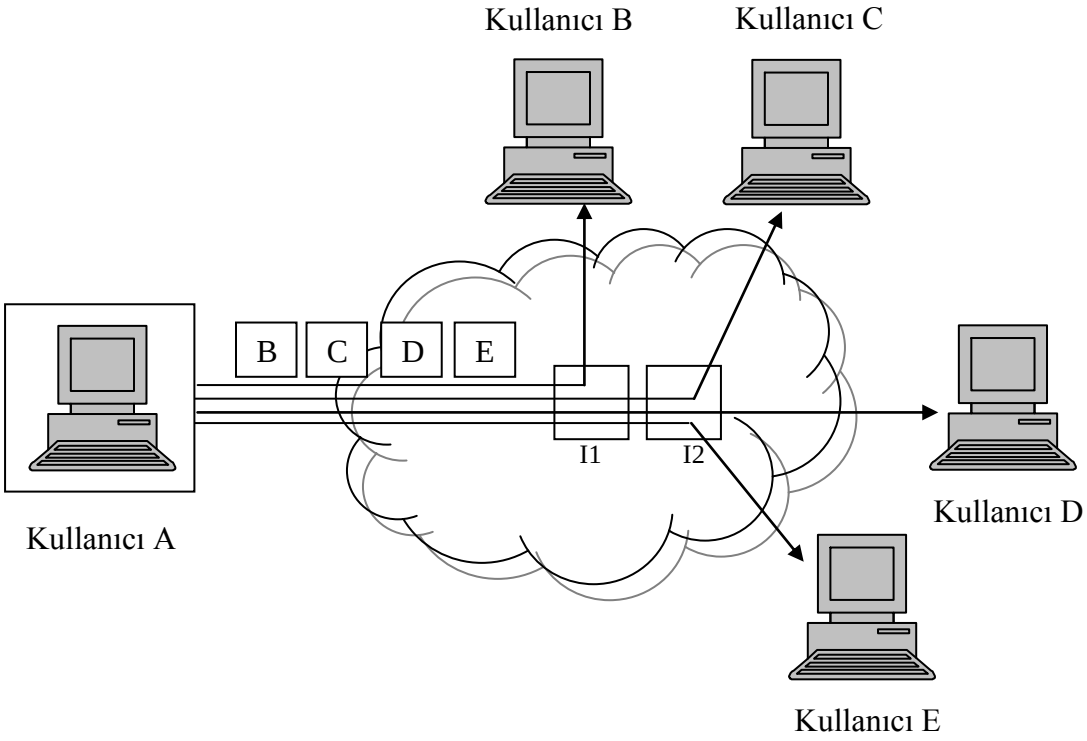


Şekil 2.5 Tekten rastgele teke gönderim iletişimi

Yayın, başka bir iletişim şeklidir ve tek bir gönderici bulunduğu için, çoğa gönderim ile karşılaştırılabilir. Ancak, yayın iletişiminde, alıcı grubu için bir kısıtlama söz konusu değildir. Veri, potansiyel alıcılara yollanır. Gerekli olan aletler ile donanmış herkes veriyi alabilir. Bu açıdan, yayın, çoğa gönderimin basitleştirilmiş halidir. Çünkü yayın; kurulma, adresleme ve grup yönetimi gerektirmez. Yayın iletişiminin günlük hayatta kullanılan iki örneği televizyon ve radyodur.

2.3. ÇOĞA GÖNDERİM VE TEKE GÖNDERİM

Neden çoğa gönderim, bilgisayar-destekli grup iletişiminde önemli bir rol oynamaktadır? Bunu daha iyi anlamak için Şekil 2.6'da verilen örneği inceleyelim. Bir grup kullanıcının bir proje ekibini oluşturduklarını düşünelim. Bu ekibin 5 üyesi bir konuşmanın içerisinde yer alsınlar. Kullanıcı A, ekibe video imgeleri yollasın. Eğer teke gönderim kullanılıyor ise, Kullanıcı A'nın; Kullanıcı B, Kullanıcı C, Kullanıcı D ve Kullanıcı E ile 4 farklı teke gönderim iletişim ilişkisi kurması gerekecektir. Daha sonra, video imgeleri her bir iletişim bağlantısından ayrı ayrı olarak yollanacaktır. Şekil 2.6'da, kare içerisinde verilen harfler her bir alıcıyı ifade etmektedir. I1 ve I2 ara sistemleri, iletişim sürecine, ağ içi sistemleri oldukları için dahil olmaktadırlar.



Şekil 2.6 Teke gönderim kullanarak yapılan grup iletişimi

Video imgeleri, Kullanıcı A ve birinci ara sistem I1 arasındaki iletim bağı üzerinde 4 kez ve sonra I1 ve I2 arasındaki bağ üzerinde ise 3 kez yollanır. Bu, yalnızca bant genişliği açısından değil, gönderici ve ara sistem arasındaki işleme başarımı açısından da ağ kaynaklarının israf edilmesine sebep olur. Burada problem, verinin birden fazla kere alınması ve yollanmasının gerekmesidir. Ara sistem I1, her seferinde, 4 veri birimi

almakta ve iletmektedir. Üstelik, göndericinin eşzamanlı olarak birden fazla iletişim bağlantısını sürdürmesiyle, kaynaklarının kullanımında (örneğin, bant genişliği, yastık belleği, işleme süresi) bir artış söz konusu olmaktadır.

Bu çeşit bir yaklaşım, birkaç yüz ya da birkaç bin kullanıcının bulunduğu çok büyük gruplar için hiç de uygulanabilir değildir.

Eğer teke gönderim iletişimi yerine çoğa gönderim iletişimi kullanılır ise, Kullanıcı A, veriyi ağ içerisine sadece 1 kez yollayacaktır. Dolayısıyla Kullanıcı A ve I1 arasındaki bağda yalnızca 1 veri birimi iletilecektir. Aynı şey I1 ve I2 arasındaki bağ için de geçerlidir. Ara sistemler, grup üyelerinin bağlı olduğu çıkış uçlarına veri birimleri kopyalayacaklardır. Bu suretle ağ trafiği, biraz önceki örneğe göre, büyük ölçüde azalacaktır. Dolayısıyla, I1 sadece 1 veri birimi alacak ve 2 veri birimi yollayacaktır. Fakat bu durumda, ara sistem veriyi kopyalayabilmelidir. Çünkü, bir giriş ucundan alınan verinin birden fazla çıkış ucuna kopyalanması gerekebilir.

Üstelik, çoğa gönderim yerine teke gönderim kullanılması durumunda, veri değişik zaman gecikmeleri ile her bir grup üyesine ulaşacaktır. Çünkü, verinin Kullanıcı A tarafından *n* kez yollanması gerekir. Yollanan bir veri biriminin büyüklüğünün 2 Kbyte olduğunu ve ağ yapısının veriyi 100 Mbit/s ile yolladığını düşünelim. Bu durumda göndericinin bir veri birimini yollaması için 16 μs gerekecektir. 200.000 km/s yayılım gecikmesi olduğu kabul edilirse, bu süre içerisinde ağda 3,2 km'lik bir alan kapsanabilir. Bu, verinin son alıcıya henüz ulaşamadan birinci alıcıya ulaşabileceği anlamına gelir. Bu durum, özellikle çok büyük gruplarda önemli bir problem olarak karşımıza çıkacaktır.

Bu çeşit zaman farkları, özellikle eşzamanlı çalışmanın önemli olduğu etkileşimli grup uygulamalarında, problemlere neden olabilir.

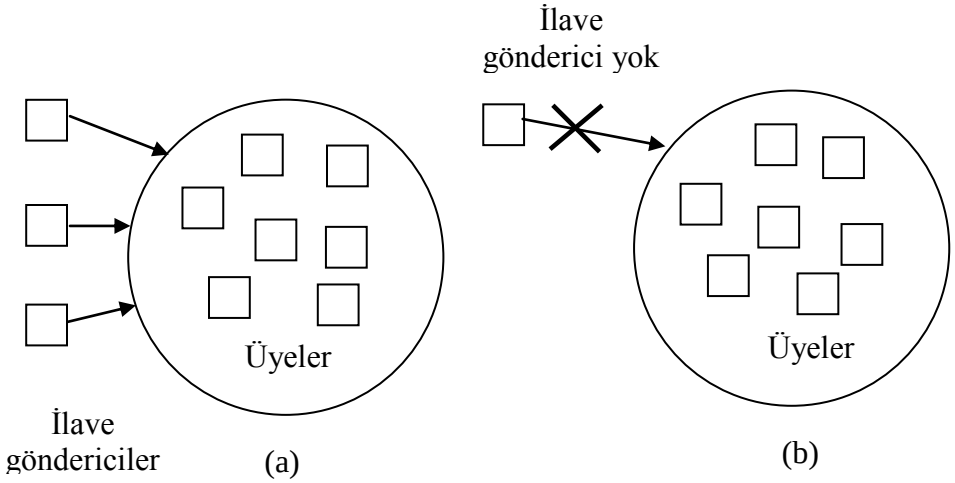
Özet olarak, çoğa gönderim iletişimi, teke gönderime kıyasla daha fazla avantaja sahiptir. İletişime dahil olan sistemler daha az veri hacimleri ile başa çıkmak zorundadır ve daha az işleme kapasitesi gerektirir. Bu avantaj, çok büyük gruplarda öne çıkmaktadır. Diğer bir avantaj ise, adreslemenin teke gönderim ile sağlanan çoğa gönderime (örneğin; alıcı adresleri listesi) göre, çoğa gönderimde (örneğin; grup adresleri) daha kolay olmasıdır. Üstelik, çoğa gönderimde göndericinin tüm alıcıları bilmesi gerekmemektedir.

2.4. GRUP KARAKTERİSTİKLERİ

İletişim gruplarının tipik karakteristikleri arasında aşağıdakiler yer alır:

- Açıklık
- Dinamiklik
- Yaşam süresi (Ömür)
- Güvenlik
- Haberdar olma
- Çoktürellik

İlk grup karakteristiği *açıklık*tır. Açıklık teriminin anlaşılması için, *açık* ve *kapalı grup* ayrımının yapılması gerekir (Şekil 2.7). Herhangi bir göndericiden gelen veriler *açık gruplara* yollanabilir. Başka bir deyişle, göndericinin o grubun üyesi olması gerekli değildir. Tabiki her zaman alıcıların grup üyesi olması zorunluluğu vardır. *Kapalı gruplarda,* veri, sadece o grubun üyeleri arasında değiş tokuş edilebilir. Bu, alıcı gibi göndericinin de, o grubun üyesi olması gerektiği anlamına gelmektedir. Grup dışındaki hiçbir gönderici gönderme işlemi yapamaz.



Şekil 2.7 (a) Açık grup (b) Kapalı grup

İkinci karakteristik, *grup dinamiği*dir. Bu karakteristik, *dinamik* ve *statik grup* arasındaki ayırımı yapar. *Statik gruplarda* grup üyelikleri önceden belirlenir ve iletişim

sağlandıktan sonra değişiklik göstermez. *Dinamik gruplarda* iletişim sırasında grup üyelikleri değişebilir. Statik gruba bir örnek olarak, projenin bazı yönlerini açığa çıkarmak için, proje ekibinin yapmış olduğu kısa toplantılardaki kişilerin oluşturduğu bir grup verilebilir. Bu tür bir toplantıda, herhangi bir konu hakkında daha fazla detaya inilmek istendiğinde, başka uzmanların da toplantıya dahil olması sağlanacak ve grup dinamik gruba dönüşecektir. Açık gruplar ve kapalı gruplar dinamik olabilir; yani, zaman içerisinde bu grupların bileşimi değişebilir.

Yaşam süresi karakteristiği *kalıcı gruplar* ile *geçici gruplar* arasındaki ayrımın yapılmasını sağlar. *Geçici grup*, grup en az bir üye içerdiği sürece var olur. Diğer taraftan *kalıcı grup*, güncel olarak aktif bir üyesi olmasa dahi var olacaktır. Alt ağdaki tüm yönlendiricilerin dahil olduğu grup kalıcı grubun tipik bir örneğidir. Videokonferans ise, geçici grubun bir örneği olarak verilebilir.

Güvenlik açısından bakıldığında, her bir grup iletişimi örneğinde, öncelikle güvenlik gereksinimleri belirlenmelidir. Bu gereksinimler tüm iletişim boyunca değişmeyeceği gibi, kurulmuş olan bir iletişim sırasında dinamik olarak da değişebilir. Üstelik, her bir farklı veri katarı için de değişiklik gösterebilir. Noktadan noktaya iletişimde kullanılan tekniklerin biraz daha genişletilerek grup iletişiminde de kullanılması söz konusu olmuştur; ancak, daha sonraları bunun yeterli olmadığı ve yeni mekanizmaların gerekli olduğu ortaya çıkmıştır: Örneğin, özel anahtarların dağıtılması. [3]

Haberdar olma karakteristiği, bir grup üyesinin, grup içerisindeki diğer üyelerin farkında olmasını ifade eder. Bir *anonim grupta,* grup üyeleri, tüm iletişim boyunca birbirlerinden haberdar olmayabilir. Örneğin, ağ üzerinde bir sunumun çoğa gönderiminde, bazı dinleyiciler kendi kimliklerini açıklamayabilirler. Dolayısıyla, diğer grup üyeleri tarafından tanınmazlar. Bir proje ekibi örneğinde ise, bir üyenin kimliği diğer tüm üyeler tarafından bilinmektedir. Bu tip gruplara *bilinen grup* adı verilir. Gruptan haberdar olma, verilebilecek grup hizmetlerini oldukça fazla etkilemektedir. Örneğin, güvenilirlik anonim gruplarda sağlanamaz.

Başka bir grup karakteristiği ise *çoktürelliktir.* Bu karakteristik ile *çoktürel grup* ve *tektürel grup* olarak grup ayrımı yapılabilir. *Çoktürel gruplar* içerisinde grup üyeleri, örneğin; ağ bağlantıları (örneğin; veri hızları), video kodlama için alternatif

mekanizmalar (örneğin; H261, MPEG) vb. açılarından değişik yeteneklere sahip olabilirler. Bunun için bir iletişim sistemi, bu çoktürellik ile başa çıkabilecek mekanizmalara sahip olmalıdır. Bu açıdan tektürel bir grup içerisinde grup iletişimini sağlamak çok daha kolaydır. Daha önce verdiğimiz proje ekibi örneğini ele alalım: Proje ekibinden bir kişi, bir iş gezisi için ofis dışına çıksın. Bu kişi ofisden ayrı olduğu süre içerisinde, yapılan bir toplantıya dahil olmak istediğinde, bunu ancak daha düşük hızlı bir telsiz (wireless) bağlantı üzerinden gerçekleştirebilecektir. Bu durumda, grubun çoktürel grup olarak yeniden sınıflandırılmasına ihtiyaç duyulacaktır. Grup içerisindeki diğer üyelere verilen hizmet niteliğinde bir değişiklik olmadan, grup içerisindeki bazı üyelere daha düşük bir hizmet niteliği verilebilmelidir. Bu tür bir hizmet verilebiliyor ise, çoktürellik grup üyelerine saydam olmalıdır. Eğer bu sağlanamıyor ise, hizmet niteliği tüm grup içinde uygun olan en alt seviyeye indirilmek durumunda kalınmakta, bu da pek çok grup üyesinin memuniyetsiz olmasına sebep olmaktadır.

2.5. GRUP ADRESLEME VE YÖNETİMİ

Tüm alıcılar (yayın) ya da tek bir alıcı (teke gönderim) yerine özel bir alıcı grubu adreslenmesi gerektirdiğinden, çoğa gönderim iletişimi için yeni bir adresleme yöntemi gereklidir. Olası yöntemlerden ikisi şunlardır:

- Alıcı listesi aracılığıyla adresleme,
- Özel grup adresi aracılığıyla adresleme

Alıcı listesi kullanılması durumunda, göndericinin, grup içerisindeki tüm alıcıların teke gönderim adreslerini içeren listeye erişim yetkisi olması gerekir. Gönderici, (örneğin bir e-posta sunucusu olabilir) grup içerisindeki tüm üyeleri ayırt edebilir ve o üyeye özel veriler yollayabilir. Buna örnek olarak, e-posta için e-posta adreslerinin nasıl ele alındığı verilebilir. Alıcı listesi, her zaman grup içerisindeki güncel üyeleri göstermelidir. Liste, grup içerisinden bir üye ayrıldığında ya da grup içerisine yeni bir üye dahil olduğunda güncellenmelidir.

Özel grup adresi kullanıldığında, grup, özel bir adres ile ayırt edilir. Gönderici veriyi bu grup adresine yollar. Bu nedenle, grup içerisindeki her bir alıcının ayrı ayrı tanınmasına gerek yoktur. Grup adresi kullanmanın avantajı, göndericinin veriyi birden fazla alıcıya

ayrı ayrı göndermesinin gerekli olmamasıdır. Gönderici veriyi, sadece bir kere grup adresine yollar. Alıcılar, belirli grup adresindeki verinin kendileri ile ilgili olduğuna karar verip, bu veriyi almalıdır. Grup adresi, ağ içerisindeki ara sistemlerin verinin nasıl yollanacağına karar vermelerine izin verir. Eğer gerekirse, bu sistemler veriyi kopyalayarak, birden fazla çıkışa yollayabilirler.

Grup adresi atama iki şekilde olabilir:

- Merkezi,
- Dağıtık

Merkezi adres atama yetkili kurumlar aracılığı ile yapılır. Adres, grup kurulmadan önce bu kurumlardan talep edilmelidir. Adresin geçerliliği, grup içerisindeki aktif iletişim süresinden bile daha uzun olmalıdır. Adresin talep edildiği kuruma adresin iptal edilmesi için başvurulması sonucunda adres geçersiz hale gelecektir. Bu şekildeki adres atama telefon numaralarının atanmasına benzetilebilir. Merkezi adres atama özellikle uzun ömürlü gruplar için çeşitli avantajlara sahip olduğu halde kısa ömürlü gruplar için avantajlı değildir.

*Dağıtık adres atama*da, grup kurulduğunda, grup adresi yerel olarak seçilir. Dinamik olarak atanan adresin tek olduğu, başka bir deyişle başka bir grup tarafından kullanılmadığı, güvence altına alınmalıdır. Özel bir kontrol veri birimi, grup adresinin o an kullanılıp kullanılmadığını belirlemek için kullanılabilir. Ancak, bu tür kontrol veri birimlerinin kaybolabileceği düşünülürse ya da adres sahiplerinin bu veri birimine cevap verme zorunluluğu bulunmadığı için, bu şekilde seçilen adresin tek olduğunun açık bir garantisi hiçbir zaman yoktur.

Alternatif olarak, ağ içerisindeki çoğa gönderim adreslerini sağlamak için sunucular kullanılabilir. Gerektiğinde çoğa gönderim adresi bu sunucudan istenir. Merkezi adres atamadan farklı olarak, iletişim tamamlandığında grup adresi serbest bırakılır. Bu sayede aynı adres başka bir grup tarafından kullanılabilir. Merkezi adres atamaya kıyasla dağıtık adres atama yöntemi kısa ömürlü gruplar için daha avantajlıdır.

Grup yönetimi, grup içerisindeki üyelerin yönetilmesi ile ilgili tüm işlerden sorumludur. Bu işler, grup kurulmadan önce yapılması gereken işler ve grup iletişimi sırasında

ortaya çıkan işler şeklinde ikiye ayrılabilir. Üstelik, grup yönetiminin merkezi mi yoksa dağıtık olarak mı kurulacağına karar verilmesi gerekir. Ölçeklenebilirlik açısından yerel grup yönetimi daha avantajlıdır. Ancak, tüm grup için tutarlı durum bilgisi sağlamak gerektiği için -ki bu dağıtık sistemlerde merkezi sistemlere oranla daha karmaşıktır- bazı dezavantajlar da içerir.

Bilgisayar destekli grup iletişimi yapılmadan önce uygun bir duyurunun yapılması gerekir. Örneğin, bir proje ekibi yapacağı bir toplantı öncesi, toplantı zamanı ve kimlerin katılacağı gibi bazı detayları önceden belirlemiş olmalıdır. Kullanılacak araçlar (örneğin; video ve ses) da duyuru anında belli olmalıdır.

Bu çeşit bir toplantı aktif olduğunda, grubun dinamik mi statik mi (üyelerin çıkıp girmelerine izin verilip verilmeyeceği) olacağı konusunda bir kararın olması gerekir. Bu nedenle, grup yönetiminin grup karakteristiklerini çok iyi bilmesi gerekmektedir.

3. IP ORTAMINDA ÇOĞA GÖNDERİM

Günümüzde, Internet uygulamalarının büyük bir çoğunluğu noktadan-noktaya iletim üzerine kuruludur. Noktadan-çoknoktaya iletimden yararlanma, yerel alan ağları uygulamaları ile sınırlıdır. Geçen birkaç yıl içerisinde, çoğa gönderim iletişimi üzerine kurulu yeni Internet uygulamalarında artış görülmüştür. Çoğa gönderim, ağı sadece gerekli olduğu durumlarda paketlerin kopyasını çıkarmaya zorlayarak, bant genişliğinin korunmasını sağlar ve pek çok yeni ve teke gönderime göre daha çekici uygulamaların kullanılmasına imkan sağlar. [4] Bu uygulamalar arasında çok partili videokonferans, paylaşılan beyaz tahta uygulamaları, canlı borsa fiyatları vb. yer alır. Çoğa gönderim IP uygulamalarının sadece Internet ile sınırlı olmadığı, büyük ticari ağlararası iletişimde de önemli bir rol oynayabileceği göz ardı edilmemelidir. [4]

Çoğa gönderimi çekici kılan pek çok özellik bulunmaktadır. Bunlardan bazıları şöyle sıralanabilir:

Ağ Trafiğini Azaltma: Bir borsa teleksi programının, bir organizasyonun ağ yapısı içerisindeki 100 adet durağa paketler göndermesi gerektiğini düşünelim. Bu gruba teke gönderim kullanılarak yollama yapılmak istendiğinde, 100 ayrı paketin periyodik olarak bu duraklara yollanması gerekecektir. Aslında, yollanan 100 paketin pek çoğu hedeflerine ulaşıncaya kadar aynı bağlar üzerinden geçerler. Bu nedenle çoğa gönderim, gönderici tarafından tek bir paketin yollanmasını ve sadece gerekli olduğu durumlarda paketin çoğullanmasını (çoğa gönderim dağıtım ağacındaki çatallarda) gerektirdiğinden bu tür uygulamalar için ideal çözümdür. Aynı zamanda, bu şekilde iletim yapıldığında, ağ bant genişliği boşa harcanmamış olur. [4]

Yayın iletimi bu tip uygulamalar için uygun değildir. Çünkü, paketi gören her bir durakta CPU zamanının harcanmasına, dolayısıyla CPU başarımının etkilenmesine sebep olur. Bunun yanı sıra bant genişliğini de boşa harcar.

Kaynak Bulma: Bazı uygulamalar, aynı alt ağ içerisindeki grup üyelerine paket göndermek için, yayın gönderimi yerine çoğa gönderimi kullanırlar. Ancak, çoğa gönderimi tek bir yerel alan ağı (LAN) içerisinde gönderim ile sınırlandırmak için hiçbir neden yoktur. Çoğa gönderim paketlerinin faaliyet bölgesi ömür (TTL) alanı kullanılarak sınırlandırılabilir.

Çoğa gönderim mesajlarını kullanarak, hizmet sorgulama mesajını bu hizmeti verebilecek sunuculara yollamanın, kısa sürede ve daha az kaynak kullanarak hizmeti veren sunucuyu bulmakta çok büyük yardımı olacaktır.

Veri Gönderimi Uygulamaları İçin Destek: Geçtiğimiz bir iki yıl içerisinde, çoğa gönderim iletişimi çoğul ortam verilerinin gönderilmesinde oldukça popüler olmuştur. Ses ve video sinyalleri yakalanmakta, sıkıştırılmakta ve bir grup alıcıya yollanmaktadır. Gönderici ve alıcılar arasında bir grup noktadan-noktaya bağlantının kurulması yerine, çoğa gönderim bu tür çoğul ortam verilerinin taşınması için kullanılabilir. Ancak günümüzde, çoğa gönderim ses ve video ile sınırlı değildir.

3.1. ÇOĞA GÖNDERİM GRUPLARI

Teke gönderim, yayın ve çoğa gönderim olmak üzere 3 temel IPv4 adres tipi bulunmaktadır. Bir teke gönderim adresi bir paketi tek bir hedefe, bir yayın adresi bir datagramı tüm alt ağa göndermek için kullanılır. Bir çoğa gönderim adresi ise, datagramların çeşitli alt ağlar içerisinde yer alan bir grup alıcının üyesi oldukları bir çoğa gönderim grubuna dağıtımına imkan sağlamaktadır. [4]

Çoğa gönderim bağlantılı değildir. Bir çoğa gönderim datagramı hedef grup üyelerine, standart teke gönderim IP datagramı ile aynı güveninirlikte yollanır. Başka bir deyişle, çoğa gönderim datagramlarının tüm grup üyelerine ulaşması, ya da datagramların yollandıkları sırada alıcı grup üyelerine ulaşması garanti edilmez.

Çoğa gönderim IP paketi ile teke gönderim IP paketi arasındaki tek fark, IP başlığı içerisinde *"Varış Adresi"* alanında grup adresinin bulunmasıdır. Çoğa gönderim Sınıf A, Sınıf B ve Sınıf C IP varış adreslerinin yerine 224.0.0.0 ile 239.255.255.255 arasında değişen Sınıf D adres formatını kullanır.

Her bir konak, bir çoğa gönderim grubuna istediği zaman girmekte ve çıkmakta serbesttir. Çoğa gönderim grubundaki üye sayısı ya da üyelerin bulundukları fiziksel yer için de bir kısıt bulunmamaktadır. Bir konak aynı anda birden fazla çoğa gönderim grubuna üye olabilir. Çoğa gönderim grup üyelerine paket yollamak için konağın o gruba üye olması gerekli değildir.

3.2. GRUP ÜYELİK PROTOKOLÜ

Grup üyelik protokolü, yönlendiriciler tarafından kendilerine direk bağlı olan ağ içerisinde yer alan grup üyelerinin varlığını belirlemek için kullanılır. Bir konak çoğa gönderim grubuna katıldığında, paketlerini almak istediği grup ya da gruplara bir grup üyelik protokol mesajı yollar ve IP sürecini ve ağ arayüzü kartını çoğa gönderim grubuna yollanan çerceveleri alacak şekilde ayarlar.

3.3. ÇOĞA GÖNDERİM YÖNLENDİRME PROTOKOLÜ

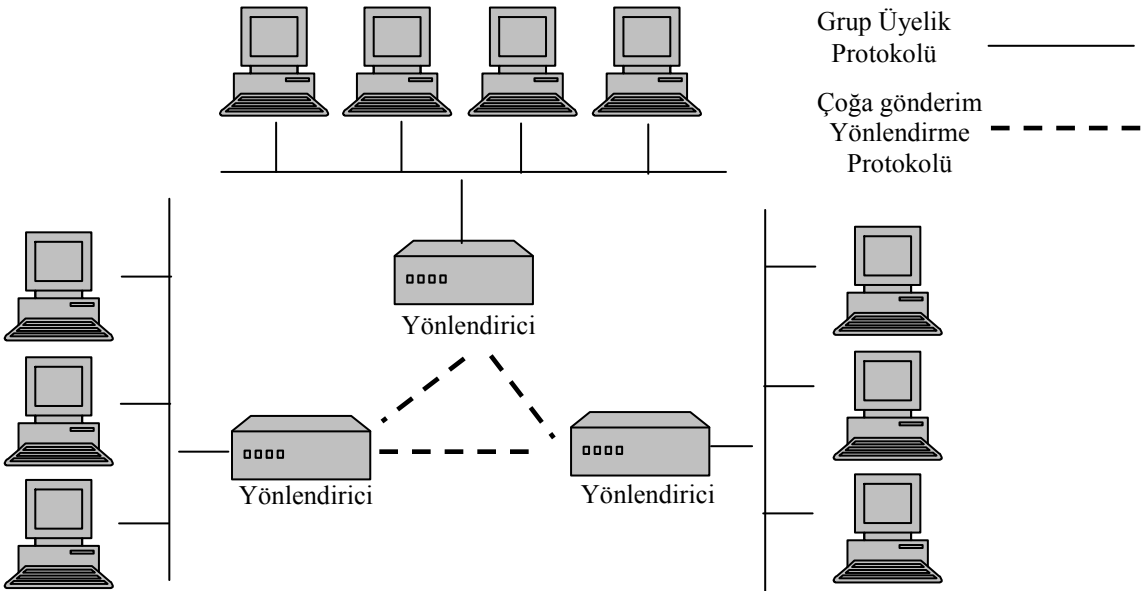
Çoğa gönderim yönlendiricileri, çoğa gönderim datagramlarının ağlararasında gönderilmesini sağlayan dağıtım yollarını belirlemek için, bir çoğa gönderim yönlendirme protokolü kullanırlar.

3.3.1. ÇOĞA GÖNDERİM YÖNLENDİRMESI VE TEKE GÖNDERİM

Çoğa gönderim yönlendirme protokolleri, gruba yollanan paketlerin gönderilmesi için, dağıtım ağacının oluşturulmasına yardımcı olurlar. Teke gönderimde ise, gönderme tablosunun (genellikle yönlendirme tablosu olarak bilinir) oluşturulması için yönlendirme protokolleri kullanılır. Teke gönderim varış adresi ile birlikte, bir metrik ve varış adresine ulaşmak için bir sonraki yönlendiricinin adresi yönlendirme tablosuna girilir. Teke gönderim iletimi ile çoğa gönderim iletimi arasındaki en önemli fark, çoğa gönderim paketlerinin kaynaktan uzaklaşacak şekilde yollanmasının gerekli olmasıdır. Eğer, paket kaynağına geri dönecek şekilde gönderilir ise bir döngü oluşacaktır. [4]

Her bir yönlendirme protokolü kendi yöntemi ile bir gönderme tablosu oluşturur. Bu tablo, her bir yönlendiriciye belirli bir kaynak için ya da belirli bir kaynaktan belirli bir

gruba yollama için ((kaynak, grup) çiftleri) ya da herhangi bir kaynaktan bazı gruplara yollama için paketlerin nasıl gönderileceğini belirtir.



Şekil 3.1 Çoğa gönderim IP dağıtım servisi

3.4. ÇOĞA GÖNDERİM ADRESLEME

Çoğa gönderim grubunu oluşturan Internet konakları için bir çoğa gönderim adresi atanır. Göndericiler, tüm grup üyelerine gönderilecek paketlerin varış IP adresi olarak çoğa gönderim adresini kullanırlar.

3.4.1. SINIF D ADRESLERİ

Bir IP çoğa gönderim grubu bir Sınıf D adresi ile ayırt edilir. Sınıf D tipi adreslerin en yüksek anlamlı 4 biti "1110" değerini taşır. Takip eden 28 bit çoğa gönderim grubunun grup ID'sini gösterir. Standart noktalı-desimal gösterilim ile ifade edilecek olunursa, çoğa gönderim grup adresi 224.0.0.0 ile 239.255.255.255 arasında değişir.

Şekil 3.2 32-bit Sınıf D tipi adres formatını göstermektedir.

28 bit

1	1	1	0	Çoğa gönderim Grup ID

Şekil 3.2 Sınıf D tipi IP adres formatı

IANA tarafından, kayıtlı IP çoğa gönderim gruplarının bir listesi tutulmaktadır. 224.0.0.0 adresi ayırtılmıştır ve hiçbir gruba atanamaz. 224.0.0.1'den 224.0.0.255'e kadar olan çoğa gönderim adresleri, yönlendirme protokolleri ya da herkesçe bilinen kalıcı adrese ihtiyaç duyan diğer protokoller vb. kullanımlar için ayrılmıştır. Çoğa gönderim yönlendiricileri, varış adresi bu aralık içerisinde kalan datagramları, paket ömür değerine bakmaksızın ileri göndermemelidir.

Bazı herkesçe bilinen gruplar şunlardır:

"Bu alt ağ içerisindeki tüm sistemler" : 224.0.0.1

"Bu alt ağ içerisindeki tüm yönlendiriciler" : 224.0.0.2

"Tüm DVMRP yönlendiricileri" : 224.0.0.4

"Tüm OSPF yönlendiricileri" : 224.0.0.5

"Tüm atanmış OSPF yönlendiricileri" : 224.0.0.6

"Tüm RIP2 yönlendiricileri" : 224.0.0.9

"Tüm PIM yönlendiricileri" : 224.0.0.13

"Tüm CBT yönlendiricileri" : 224.0.0.15

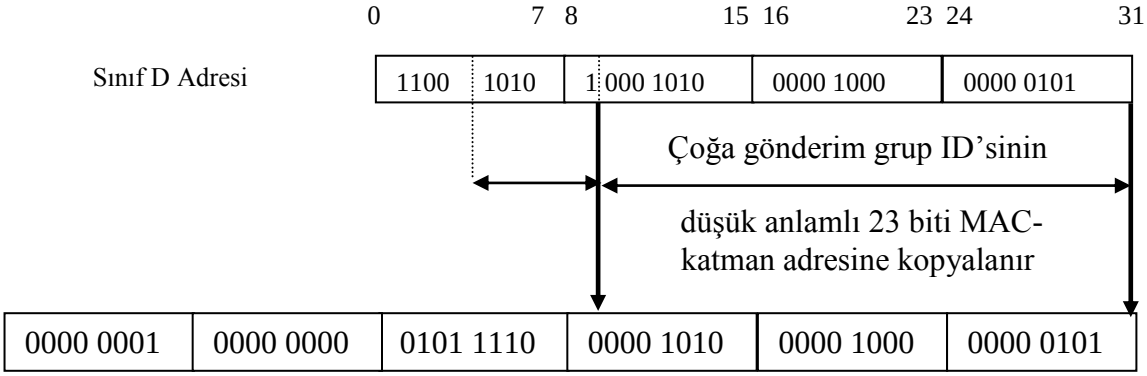
224.0.1.0 ile 239.255.255.255 arasında kalan grup ise, çeşitli çoğa gönderim uygulamalarına kalıcı olarak atanabileceği gibi dinamik atamalar için de kullanılabilir. Bu aralık içerisinde, 239.0.0.0 ile 239.255.255.255 arasında kalan adresler özel ağ içerisindeki çeşitli "yönetimsel uygulamalar" için ayrılmıştır.

3.4.2. SINIF D ADRESİNİ IEEE-802 MAC ADRESI ILE EŞLEME

IANA tarafından 01-00-5E-00-00-00 ile 01-00-5E-FF-FF-FF IEEE-802 MAC-katmanı adres aralığı IP çoğa gönderim grupları için ayırtılmıştır. Sınıf D tipi adresler ile IEEE-802 (örneğin; FDDI, Ethernet) MAC-katmanı çoğa gönderim adresleri ile eşlenmeleri şu şekilde sağlanır: Sınıf D tipi adresin en düşük anlamlı 23 biti, IANA tarafından ayrılmış MAC-katmanı çoğa gönderim adres bloğunun en düşük anlamlı 23 biti olarak yerleştirilir. Bu basit prosedür, yerel alan ağlarında çoğa gönderim adreslerinin çözümlenmesi için bir protokol gereksinimini ortadan kaldırır. Tüm yerel alan ağ

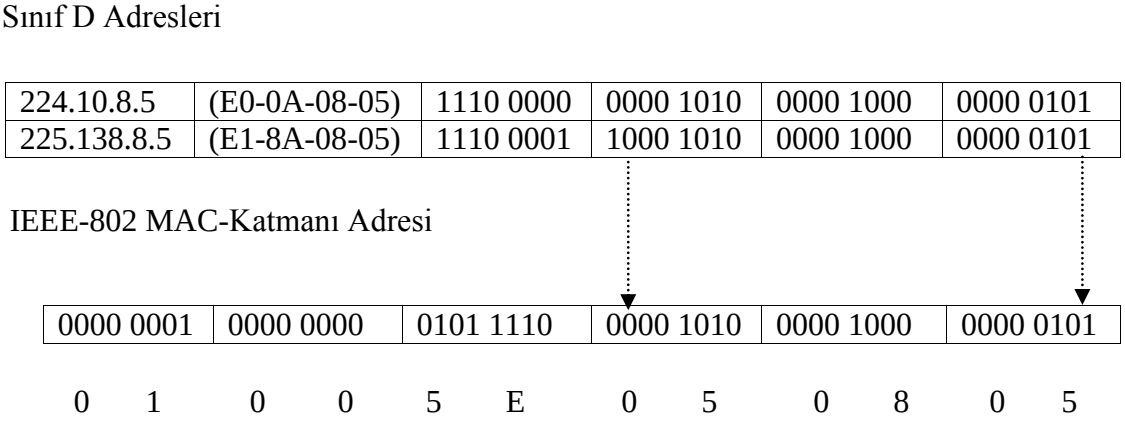
istasyonları bu basit dönüşümü bilir ve herhangi bir IEEE-802 tabanlı yerel alan ağı üzerinde IP çoğa gönderimi yapabilir.

Şekil 3.3'te 234.138.8.5 çoğa gönderim grup adresinin, bir IEEE-802 çoğa gönderim adresine nasıl karşı düşürüldüğü gösterilmektedir. IP adresinin yüksek anlamlı 9 bitinin MAC-katmanı çoğa gönderim adresine karşı düşürülmediği unutulmamalıdır. Bu örnekte, IP çoğa gönderim grup ID'sinin en düşük anlamlı 23 biti, IEEE-802 çoğa gönderim adresinin en düşük anlamlı 23 biti olarak yerleştirilmiştir.



Şekil 3.3 Sınıf D adresinin Ethernet adresine karşı düşürülmesi

Sınıf D (2^28), MAC-katmanında yer alabilecek (2^23) grup sayısından daha fazla grup içerebileceğinden, bazı grup adresleri aynı IEEE-802 adreslerine karşı gelebilir. Eğer yerel alan ağındaki iki ya da daha fazla grup aynı MAC-katmanı çoğa gönderim adresine karşılık gelen Sınıf D adreslerini seçerlerse, üye konaklar aynı adrese karşılık gelen tüm grupların mesaj trafiğine dahil olurlar. Bundan sonra hangi paketin gerçekten konağın ait olduğu grutan geldiğini tespit etmek IP katmanındaki yazılıma düşmektedir. Aynı IEEE-802 MAC-katmanı çoğa gönderim adresine (01-00-5E-0A-08-05) karşı düşen Sınıf D adresleri 224.10.8.5 ve 225.138.8.5 Şekil 3.4'te gösterilmiştir.



Şekil 3.4 Aynı IEEE-802 MAC-katmanı adresine karşı düşen Sınıf D adresleri

3.4.3. PAKET ÖMRÜ

IP başlığındaki ömür (TTL) alanı çoğa gönderimde iki kat önem taşımaktadır. Daha önce olduğu gibi, burada da bir datagramın yönlendirme hataları nedeniyle sonsuza kadar ağ içerisinde dönüp durmasını engellemek için, yaşam süresini kontrol eder. Yönlendiriciler, bir ağdan diğerine geçerken her datagramın TTL değerini bir azaltırlar. TTL 0 değerine ulaştığında paket atılır. [5]

IPv4'te TTL aynı zamanda *"eşik"* anlamı da taşımaktadır. Bunu bir örnek ile açıklayalım: Örneğin, bölümünüz içerisindeki konaklar arasında, uzun ve bant genişliği harcayan bir videokonferansı kurmak istiyorsunuz. Böyle bir durumda, yaratılacak trafiğin oldukça büyük bir kısmının yerel alan ağınız içerisinde kalmasını istersiniz. Bölümünüz pek çok yerel alan ağını içerecek kadar büyük olabilir. Her bir yerel alan ağı içerisindeki konakların konferansa katılmalarını istemeniz durumunda, bu, tüm Internet'i kendi çoğa gönderim trafiğiniz ile çökertmek istemeniz anlamına gelecektir. Bu durumda, çoğa gönderim trafiğinizin hangi yönlendiriciye kadar ilerlemesini istediğinizi belirtmeniz gerekecektir. İşte bu noktada, yönlendiricilerinizin arayüzlerine bir eşik değeri atarsınız ve bu eşik değerinden daha büyük TTL'e sahip olan datagramların yönlendirilmelerine izin verilmemesini sağlarsınız. Datagram, belirli bir eşik değeri atanmış olan yönlendiriciler üzerinden geçerken, TTL değeri azaltılmayacak sadece eşik değeri ile karşılaştırma yapılacaktır. Bu TTL'in başka bir kullanım şeklidir.

TTL eşik değerleri ve karşı düşen alanların bir listesi şöyledir:

Tablo 3.1 TTL eşik değerleri ve karşı düşen alanlar

TTL	*Alan*
0	Aynı konak ile sınırlı. Hiçbir arayüz tarafından iletilmeyecektir.
1	Aynı Alt ağ ile sınırlı. Hiçbir yönlendirici tarafından iletilmeyecektir.
< 32	Aynı site, bölüm ya da organizasyon ile sınırlı.
< 64	Aynı bölge ile sınırlı.
< 128	Aynı içerik ile sınırlı.
< 256	Sınırsız, global.

Burada geçen "site" ve "bölge" sınırları, sistem yöneticinin limitleri nasıl uyguladığına bağlı olarak değişir.

3.5. INTERNET GRUP YÖNETİM PROTOKOLÜ (IGMP)

Internet Grup Yönetim Protokolü (Internet Group Management Protokol-IGMP), konak ile hemen yanında komşusu olan yönlendirici arasında çalışır. Bu protokol aracılığı ile, konağın yerel yönlendiriciye, belirli bir çoğa gönderim grubuna gönderilenleri almak istediğini bildirmesi sağlanır. Aynı zamanda yönlendiriciler, periyodik olarak yerel ağı, herhangi bir grup üyesinin hala aktif olup olmadığını sınamak için sorgularlar. Eğer, yerel alan ağında birden fazla IP çoğa gönderim yönlendiricisi varsa, yönlendiricilerden birisi "sorgucu" olarak seçilir ve grup üyelerinin varlıklarını sorgulama sorumluluğunu taşır.

IGMP'den öğrendiği grup üyelik bilgilerine dayanarak, bir yönlendirici, hangi çoğa gönderim trafiğinin hangi "yaprak" alt ağlara yönlendirileceğine karar verme yeteneğine sahip olur. "Yaprak" alt ağlar, aşağı yönde başka yönlendiricisi olmayan ağlardır. Bu ağlar, bazı çoğa gönderim gruplarına dahil olan alıcılar içerebildikleri gibi içermeye de bilirler. IGMP'den elde edilen bilgilere göre, belirli bir gruptan bir çoğa gönderim paketi alındığında, yönlendirici alt ağı içerisinde bu gruba dahil olan aktif bir üye olup

olmadığına bakar. Eğer varsa, yönlendirici paketi bu alt ağa yollar. Eğer aktif bir üye yoksa paket yönlendirici tarafından atılacaktır.

IGMP'nin üç farklı versiyonu [23], [24], [25] bulunmaktadır. Ancak, bu çalışmanın konusu içerisinde yer almadığından, versiyonların detaylarına burada yer verilmeyecektir.

3.6. IPv6 GÜVENLİK MEKANİZMALARI

IPv6 güvenlik mimarisi ve ilişkili mekanizmalar, güvenlik isteyen Internet kullanıcılarının güvenlik mekanizmalarına sahip olacakları, bu mekanizmalara ihtiyaç duymayan kullanıcıların ise etkilenmeyeceği şekilde tasarlanmıştır. Bu mekanizmalar herhangi bir belirli kriptografik algoritmaya bağlı değildir. Bu nedenle uygulamanın diğer kısımlarını etkilemeden değiştirilebilir. Birlikte işlerlik standartlarını güvence altına almak için varsayılan algoritmalar da tanımlanmıştır.[22]

IP güvenlik mimarisi güvenlik değişkenleri, mekanizmaları, kontrol ve yönetimden oluşur.

Asıllama Başlığı (Authentication Header-AH) ve Kozalama Güvenlik Yararlı Yükü (Encapsulating Security Payload-ESP) kullanımı, alıcı ve göndericinin anahtar, asıllama ya da şifreleme algoritması ve algoritmanın kullanılması sırasında gereken ek parametre kümesi üzerinde anlaşılmasını gerektirir. Bu anlaşmalar kümesi alıcılar ve göndericiler arasında bir güvenlik işbirliği oluşturur. Uygulama bakış açısından AH ve ESP hesaplamaları ve kontrolü için gerekli tüm değişkenleri içerir.

Güvenlik işbirliği (Security Association-SA) anahtar yönetimi ve güvenlik mekanizmalarını birbirinden ayırır. Anahtar yönetimi güvenlik işbirliği içerisindeki değişkenleri günceller, mekanizma ise bu değerleri okur ve kullanır. Anahtar Yönetimi Protokolü ve Güvenlik Protokolünün bir araya getirilmesi Güvenlik Parametre İndeksi (Security Parameter Index-SPI) ile olur.

Bir güvenlik işbirliği tek yönlüdür. Bir iletişim oturumu iki konak arasında her bir yön için bir tane olmak üzere iki güvenlik işbirliğinin kullanımını gerektirecektir. Gönderici konak göndericinin KullanıcıID'si ve varış adresini kullanarak uygun güvenlik

işbirliğini (ve SPI) seçer. Alıcı konak SPI değeri ve varış adresinin bileşimini kullanarak doğru işbirliğini ayırt eder. Alıcı-odaklı güvenlik işbirliği alıcının SPI'yı seçmesi anlamına gelir.

Güvenlik işbirliği sunucu-odaklı ya da kullanıcı-odaklı olabilir. Sunucu-odaklı güvenlik işbirliği aynı sunucudaki tüm kullanıcılar tarafından aynı oturum anahtarının kullanılmasını destekler. Eğer kullanıcı-odaklı güvenlik işbirliği destekleniyorsa, tüm kullanıcılar ayrı oturum anahtarları kullanırlar. Sunucu-odaklı ve kullanıcı-odaklı güvenlik işbirliği güvenlik yönetimi tarafından seçilir. Uygulama açısından, güvenlik yönetimi doğru güvenlik işbirliğini ve datagramlar için SPI seçerken güvenlik yönetimini uygulatmak için gerekli tüm değişkenleri tanımlar.

İki temel IPv6 güvenlik mekanizması Asıllama Başlığı (Authentication Header-AH) ve Kozalama Güvenlik Yararlı Yükü (Encapsulating Security Payload-ESP) bazı durumlarda istenilen güvenlik özelliklerinin sağlanması için bir araya getirilebilir.[22]

3.6.1 ANAHTAR VE SA YÖNETİMİ

Anahtar ve SA yönetimi IP güvenlik mekanizmalarından güvenlik işbirliği arayüzü tarafından ayırt edilir. Yönetim ve mekanizmalar yanlızca SPI değerleri aracılığı ile birleştirilebilir.

Anahtar yönetimi güvenlik işbirliği içerisindeki değişkenlerin kurulmasından ve güncellenmesinden sorumludur.

RFC1825 tüm IPv6 uygulamaları için anahtar yönetim gereksinimlerini tanımlar.

- Uygulama güvenlik işbirliklerinin elle konfigüre edilmesini desteklemek zorundadır.
- Uygulamalar bir Internet standart anahtar yönetimi protokolü yayınlandığında bu protokolü desteklemelidir.
- İki uç arasında birden fazla eşzamanlı güvenlik işbirliği olması mümkün olmak zorundadır.

- Çok-kullanıcılı sunucular üzerindeki uygulamalar kullanıcı-odaklı güvenlik işbirliklerini desteklemelidir. Bu şekildeki uygulamalar sunucu-odaklı anahtar tahsisine de izin vermelidir.
- Bir IP sunucusu anahtarları ve güvenlik işbirliği parametrelerini korumak için bazı adımlara sahip olmalıdır.

Elle yapılan anahtar yönetiminde, sistemin kendi anahtarları ve aynı zamanda sistemin iletişim anahtarları elle konfigüre edilir. Bu küçük ve statik ortamlarda çalışabilir ancak ölçeklenebilir değildir.

IP güvenliğinin yaygın kullanımı bir Internet standart ölçeklenebilir anahtar yönetimi protokolünü gerektirecektir. Önerilen bir grup aday anahtar yönetimi protokolü şunlardır: ISAKMP (Internet Security Association and Key Management Protocol) [26], SKIP [27], Oakley [28], Photuris [29] ve SKEME [30]'dir. Ancak bugüne kadar hiç biri standart olarak adapte edilmemiştir.

Anahtar yönetimi protokollerinin pek çoğu diğer birimin asıllanmış kamusal anahtarının kullanılabilir olduğunu kabul eder. Bu kamusal anahtarın iki taraf arasında güçlü bir asıllama mekanizması ile sağlanması için değişik yöntemler ileri sürülmüştür. Bunlardan birisi de KERBEROS'dur.

KERBEROS ağ asıllama protokolüdür. İstemci/Sunucu uygulamalarına gizli-anahtar kriptografisi kullanarak güçlü bir asıllama sağlamak için tasarlanmıştır. Protokol MIT tarafından geliştirilmiştir. KERBEROS pek çok ticari üründe de kullanılmaktadır. KERBEROS protokolü güçlü bir kriptografi kullanmaktadır. Böylelikle bir istemci sunucuya kendi kimliğini güvenli olmayan bir ağ bağlantısı üzerinden kanıtlayabilir. İstemci ve sunucu kimliklerini kanıtlamak için KERBEROS'u kullanmalarından başka, veri bütünlüğü ve gizliliği sağlamak için tüm iletişimlerini şifreleyebilirler.

3.6.2 ASILLAMA BAŞLIĞI

Asıllama başlığının amacı asıllama bilgilerini IP datagramı içerisinde taşımaktır. Asıllama bilgisi datagramın aktarım sırasında değişmeyen tüm alanları kullanılarak hesaplanır. IPv6 Asıllama Başlığı (AH) IPv6 datagramlarına veri bütünlüğü ve asıllama sağlar. Gönderici IPv6 paketinin yollanmasından hemen önce statik alanlar için asıllama

verisini hesaplar ve alıcı asıllama verisinin doğruluğunu veri alınır alınmaz kanıtlar. Yalanlama (Non-repudiation), Asıllama Başlığı ile kullanılan asıllama algoritmaları (örneğin: asimetrik) aracılığı ile engellenebilir. Varsayılan asıllama algoritması MD5 algoritmasını kullanır. Ancak bu algoritma tüm diğer simetrik algoritmalar gibi yalanlamayı engelleyemez. Ayrıca sırdaşlık ve trafik analizi koruması da AH tarafından sağlanamaz.[22]

3.6.2 KOZALAMA GÜVENLİK YARARLI YÜKÜ

Kozalama Güvenlik Yararlı Yükünün amacı IP datagramının şifrelenmiş verisini taşımaktır. Şifrelenmiş veri, korunacak olan veriye belirlenmiş olan bir şifreleme algoritmasının uygulanması ile elde edilir.

ESP, IPv6 datagramının tamamını ya da sadece ESP içerisindeki üst-katman verisini kozalayarak ESP içeriğinin büyük bir kısmını şifrelemiş olur ve ESP'ye bu sayede eklenen yeni bir IPv6 başlığı ile IPv6 datagramları için veri bütünlüğü, asıllama ve sırdaşlık sağlanmış olur. Datagramın alıcısı yeni eklenen IPv6 başlığını siler, ESP'nin şifresini çözer ve veriyi normal olarak işler. [22]

3.6.3 IPv6 ANAHTAR YÖNETİMİ GEREKSİNİMLERİ

IPv6 Anahtar Yönetimi Gereksinimleri şu şekilde tanımlanabilir: Tüm IPv6 uygulamaları, elle yapılan anahtar yönetimini desteklemeli ve onaylanan Internet standart anahtar yönetimi protokollerini desteklemelidir. Tüm IPv6 uygulamaları sistemde trafik yaratmak için kullanıcıdan-kullanıcıya anahtar tahsisini konfigüre etmeye ve kullanmaya izin vermeli ve sistem yöneticisine esneklik sağlamak ve elle yapılan anahtar dağıtmayı kolaylaştırmak için ek olarak o sistemde trafik yaratmak için sunucudan-sunucuya anahtar tahsisinin konfigüre edilmesine izin vermelidir. [22]

4. GÜVENLİ ÇOĞA GÖNDERİM

Çoğa gönderim iletişimi, bir grup katılımcıya veri dağıtmanın en verimli ve etkin yoludur. Teke gönderimin tersine, çoğa gönderim yönlendirmesi tek bir IP datagramının birden fazla konağa eş zamanlı olarak yönlendirilmesine imkan sağlar. Bir çoğa gönderim grubu içerisindeki üyelik dinamiktir. Böylece bir konak diğer konakların haberi olmaksızın çoğa gönderim oturumuna dahil olabilir ya da ayrılabilir. Çoğa gönderimin bazı doğal faydaları, güvenliğin sağlanmaması durumunda, grup iletişimini saldırıya açık hale getirebilir. Güvenli çoğa gönderimin başlıca amacı, çoğa gönderimin faydalarından yararlanırken, bu güvenlik açıklarını ortadan kaldırmaktır. [5]

4.1. ÇOĞA GÖNDERİM HİZMETİ

IP ortamında çoğa gönderim, IP datagramlarının bir grup konağa gönderilmesidir. [6] Konak grubu, belirli bir Sınıf D adresinden datagram alabilen ve yollayabilen, çoğa gönderim yeteneğine sahip bir grup konaktan oluşmaktadır. Sınıf D adresleri özel olarak çoğa gönderim için ayrılmış adreslerdir ve dinamik olarak çoğa gönderim gruplarına atanabilirler. Güvenli olmayan ya da herkese açık olan gruplarda, sadece çoğa gönderim adresinin bilinmesi üyelik için yeterlidir. [5]

IP datagramlarının bir çoğa gönderim grubuna yollanması için birkaç protokole ihtiyaç vardır. Konaklar çoğa gönderim grubunun bir üyesi olma isteklerini, Internet Grup Yönetim Protokolü (IGMP) aracılığı ile, kendi yerel yönlendiricilerine bildirirler. [6] IP datagramlarının grup üyelerine dağıtılması için, yönlendiriciler ağ yönlendirme topolojisinde tanımlanan protokollerden birini kullanabilirler. Bu protokollerden bazıları Tablo 4.1'de verilmiştir.

Tablo 4.1 Çoğa gönderim yönlendirme protokolleri

Çoğa gönderim Uzaklık Vektör Yönlendirme Protokolü (Distance Vector Multicast Routing Protocol-DVMRP)
Önce En Kısa Açık Yol Çoğa gönderim Eklentileri (Multicast Extensions to OSPF-MOSPF) • Bölge-İçi Yönlendirme (Intra-Area Routing) • Bölgeler-Arası Yönlendirme (Inter-Area Routing) • Özerk Sistemler-Arası Yönlendirme (Inter-AS Routing)
Çekirdek Tabanlı Ağaç Çoğa gönderim Yönlendirme Protokolü (Core Base Tree Multicast Routing Protocol-CBT)
Protokol Bağımsız Çoğa gönderim (Protocol Independent Multicast-PIM) • PIM-Yoğun Kipi (PIM-Dense Mode-PIM_DM) • PIM-Seyrek Kipi (PIM-Sparse Mode-PIM_SM)

Çoğa Gönderim Omurgası (Multicast Backbone-MBONE), geleneksel teke gönderim Internet yapısının tepesinde yer alan ve DVMRP protokolünü kullanan bir çoğa gönderim ağı örneğidir. Bu yönlendirme topolojilerinin bazı özellikleri çoğa gönderim anahtar dağıtımında yararlı olabilir. Örneğin, [8]'de Ballardie, Çekirdek Tabanlı Ağaç (Core Based Tree-CBT) çoğa gönderim protokolü içerisinde tanımlanmış olan bir çekirdek yönlendirici çevresine dayanan bir anahtar dağıtım mekanizması tanımlamaktadır. [9]

Çoğa gönderim grubunun alanını, IP datagramlarının yönlendirilmelerini kısıtlamak sureti ile sınırlandırabiliriz. Her bir IP datagramı içerisindeki ömür alanını değiştirmek sureti ile, konak kendi yarattığı trafiğin alanını, yönlendirici tarafından atılmadan önce, IP datagramlarının kaç adet sekme üzerinden geçeceğini kontrol ederek sınırlandırabilir. Bu şekildeki bir kontrol, çok basit olarak, verinin potansiyel dinleyicilerini sınırlandırmak yoluyla güveninirlik sağlamanın en basit yöntemidir. Ancak, daha fazla

güveninirlik sağlanmak isteniyor ise, daha karmaşık mekanizmalara ihtiyaç duyulacağı açıktır.

Çoğa gönderim uygulama katmanı verisi çoğunlukla ulaşım katmanı UDP protokolü ile çevrelenir. UDP ve IP protokollerinin birleşimi, hata giderme yeteneklerine sahip olmayan ve güvenilir olmayan bir datagram servisi ortaya çıkarır. Gerçek-zamanlı protokol (Real Tine Protocol-RTP) gibi bazı protokoller bu eksiklikleri gidermek için tasarlanmıştır. [10] RTP, UDP ve alttaki ağ protokolünün üzerinde çalışır ve çoğa gönderim ses ve video konferansları ya da oturumları için uçtan-uca ağ ulaşım fonksiyonları sağlar. Bir oturum, bir çoğa gönderim grubu içerisinde çoğul ortam verilerinin değiş tokuşu olarak tanımlanabilir. [11] Bir çoğa gönderim grubu içerisinde birden fazla aktif oturum (örneğin; bir tane ses ve bir tane de video konferans için) bulunabilir.

Çoktan-çoğa tip uygulamalarda, birden fazla grup üyesi oturum süresince aynı anda veri alıp yollayabilirler. Tersine tekten-çoğa tip uygulamalarda, tek bir gönderici ve oturum içerisinde birden fazla alıcı yer alır. Uygulamanın tipi, güvenlik yapısının tasarımı üzerinde etkili olmaktadır. Örneğin, tekten-çoğa tip bir uygulama doğal olarak merkezidir ve tek bir güvenilir konak üzerine kurulu bir güvenlik mimarisinden yararlanabilir. Dağıtık çoktan-çoğa uygulamalar ise, kayıt ve anahtar dağıtımı işlemlerini gerçekleştiren, birden çok güvenilir konak içeren, dağıtık bir güvenlik mimarisinden yararlanabilir.

Çoğa gönderim oturumu, üyelik açısından da ifade edilebilir. Genelde bir oturum, genel ya da özel olarak tanımlanabilir. Her iki tip de, çoğa gönderim grubuna katılım için gereken erişim kontrol seviyesiyle tanımlanabilir. Genel oturumlar genellikle MBONE üzerindedir ve çoğa gönderim iletişiminin dinamik yapısı tarafından desteklenir (örneğin; üyelik için gereken tek şey çoğa gönderim adresinin bilinmesidir). Genel oturumları, kullanıcıların kayıt olmasını ve katılımcı olabilmeleri için asıllama kontrolü yapılması gerekliliğini getirerek kısıtlayabiliriz. Özel çoğa gönderim oturumunun alanını kısıtlamak için, katılım için hem kayıt hem de asıllama gerekmektedir. Güvenli oturumların görünürlüğünü kısıtlamak için, oturum trafiği genellikle şifrelenir.

4.2. ÇOĞA GÖNDERİM ORTAMINDA GÜVENLİK

IP çoğa gönderim iletişimi için geçerli olan tehditler, IP teke gönderim iletişimi için geçerli olan tehditler ile benzerlik gösterir. Genellikle bu tehditlerden bazıları; gizlice dinleme, yetkilendirilmemiş veri üretimi, yetkilendirilmemiş veri değiştirme, yetkilendirilmemiş veri yapısı değişiklikleri, hizmetin yalanlanması ve verinin kurallara aykırı olarak kullanılmasıdır. [5]

Bir sistem, asıllama, veri bütünlüğü, sırdaşlık vb. gibi güvenlik servisleri ile bu tehditlere karşı güvenli hale getirilebilir. Sistemin güvenlik yönetimi bu güvenlik servislerinin nasıl uygulanacağını belirler. Bir çoğa gönderim grubunu kapsayan güvenlik servislerinin örtüşmesi durumunda uygulama çatışmaları görülebilir. Bu durumda, çakışan güvenlik yönetimleri, farklı anahtar uzunluklarının ve şifreleme algoritmalarının kullanılmasına zorlayabilir. Bu nedenle, çoğa gönderim uygulamalarında kullanılan güvenlik protokollerinin çok çeşitli güvenlik mekanizmalarını destekleyecek şekilde esnek olması gerekir.

4.2.1. ASILLAMA

Bu servis, konağın kimliğinin güvence altına alınmasını sağlar. Asıllama mekanizmazları, çoğa gönderim iletişiminin pek çok tarafına uygulanabilir. Asıllama, güvenli çoğa gönderim grubuna erişim kontrolü sağlamak için temel kısımdır. Kayıt sürecinde asıllama mekanizmalarının uygulanması, güvenli gruba sadece yetkili konakların girmesini güvence altına alır. Eğer grup güvenliği için şifreleme gibi kriptografik teknikler kullanıyor ise, asıllama yöntemleri, grup iletişiminin güvenliğinde kullanılan anahtarlara sadece yetkili grup üyelerinin erişimini sağlamak ve diğer erişimleri kısıtlamak için kullanılabilir. [5]

Çoğa gönderim trafiğinin kaynağını belirlemek için, asıllama mekanizması, doğrudan her bir IP datagramına uygulanabilir. Asıllama Başlığı (Authentication Header-AH) gibi protokoller, IP datagramları için asıllamayı sağlarlar ve konak asıllama için de kullanılabilirler. Asıllama, aynı zamanda anahtar dağıtım algoritmalarının da en temel kısmını oluşturur. Anahtarın hassas bir materyal olması nedeni ile, asıllama

mekanizmaları anahtar materyalinin kaynağını belirleme, kılık değiştirme ve yeniden yollama ataklarını önlemek için kullanılabilir.

Güvenli çoğa gönderim uygulamaları için sadece güçlü asıllama algoritmaları tavsiye edilir. Dijital İmza Standartları (Digital Signature Standard-DSS) gibi dijital imza düzenleri, kamusal anahtar mekanizmaları üzerine kurulu, güçlü asıllama mekanizmalarının bir örneği olarak verilebilir. Konağın kimliğini, konağın kamusal anahtarı ile ilişkilendirmek için, güvenilir sertifika birimi (Certificate Authority-CA) tarafından dijital olarak imzalanan sertifikalar oluşturulur.

4.2.2. VERİ BÜTÜNLÜĞÜ

Bu servis, çoğa gönderim trafiğinin aktarım sırasında değiştirilmediğini güvence altına alır. Veri bütünlüğü IP datagram trafiği için doğal değildir ve genellikle ulaşım katmanı protokolleri (örneğin; TCP) için ayrılmıştır. IP'de veri bütünlüğü servislerinin eksik olması, aldatma saldırılarının olmasına sebep olabilir. Veri bütünlüğü ağ katmanında, dolaylı olarak, Kozalama Güvenlik Protokolü (Encapsulating Security Protocol-ESP) ve Asıllama Başlığı (Authentication Header-AH) gibi güvenlik protokolleri aracılığı ile sağlanabilir. Bozulmuş olan verinin kolaylıkla belirlenebildiği uygulamalarda (örneğin; ses uygulamaları), bu servis çok önemli değildir. Ancak, anahtar yönetim protokolleri gibi uygulamalar açısından, veri bütünlüğü servisinin, aldatma ataklarını önlemek için önemli bir yeri vardır.

4.2.3. SIRDAŞLIK

Bu servis, özel çoğa gönderim oturumu yaratmak için temel teşkil eder. Tipik olarak, şifreleme bu servisin sağlanması için kullanılsa da, oturuma ait IP datagramlarının yönlendirilmelerini kısıtlayarak, daha zayıf bir sırdaşlık sağlanabilir. İstediğimiz uçtan-uca servisin sağlanması için, protokolün birkaç katmanında şifreleme uygulanabilir. RTP gibi aktarım protokolleri, protokol tanımları içerisinde şifrelemeyi desteklemektedirler. Ağ katmanında, ESP IP datagramları için sırdaşlık servislerini sağlamaktadır. Sırdaşlık servisleri, anahtar değiş tokuşu sırasında anahtar yönetim hareketlerine uygulanmalıdır. Internet Güvenlik Birliği ve Anahtar Yönetimi Protokolü (Internet Security Association and Key Management Protocol-ISAKMP) gibi anahtar yönetim protokolleri, anahtar

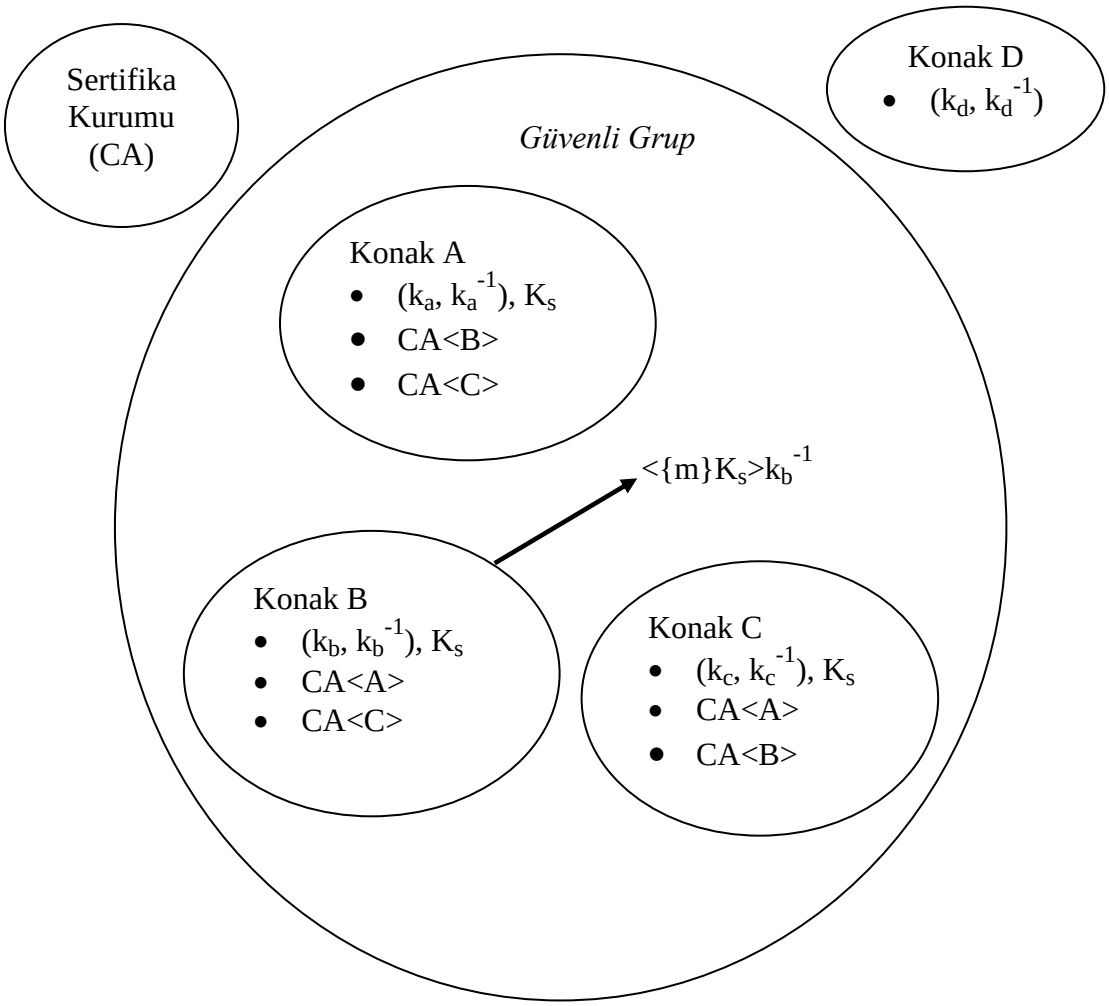
değiş tokuşu için sırdaşlığı desteklemektedir. Sırdaşlık, oturum duyurularına da uygulanabilir. Böylece, oturumun detayları saklanırken, duyuru kamuya açık olarak yapılmış olur.

Şekil 4.1, yukarıda anlatılan güvenlik kavramlarının bir uygulamasını göstermektedir. Örnekte, her bir konak bir kamusal-özel anahtar çifti (k_X, k_X^{-1}) yaratır. Özel anahtar (k_X^{-1}) konak tarafından saklanır ve mesajların ve anahtar materyalinin imzalanması için kullanılır. Bu imza, konağın, diğer grup üyeleri tarafından ayırt edilmesini sağlar. Kamusal anahtar (k_X) CA tarafından imzalanır ve sertifika (CA<X>) şeklinde diğer grup üyelerine dağıtılır. Konaklar, diğer grup üyelerinin dijital imzalarını, sertifikalarında yer alan kamusal anahtarları ile doğrularlar. Örnekte, konak A bir grup anahtarı (K_S) üretmiş ve çoğa gönderim grubuna dağıtmıştır. Konakların, Konak A tarafından asıllanmalarından sonra, imzalanmış grup anahtarı geçerli grup üyelerine (yani; Konak B, Konak C) güvenli olarak dağıtılır. K_S, güvenli grubu tanımlamaktadır. Konak D, güvenli grubun bir üyesi değildir ve K_S'nin bir kopyasını almayacaktır. Bundan sonra K_S ile şifrelenmiş çoğa gönderim grup mesajları, grup üyelerinin özel anahtarları kullanılarak imzalanacaktır. Şekil 4.1 K_S ile şifrelenmiş ve mesajı yollayan Konak B (örnek; k_b^{-1}) tarafından imzalanmış *m* mesajını göstermektedir. Grup üyeleri A ve C, K_S ve CA<B>'yi kullanarak mesajın şifresini çözebilecek ve mesajın kaynağını doğrulayabileceklerdir.

4.3. GÜVENLİK GEREKSİNİMLERİ

Bir çoğa gönderim oturumu için gerekli olan güvenlik servisleri tanımlandıktan sonra, bu servislerin uygulamaları ile ilgili detayların açıklanması önem kazanır. [5] Kriptolama algoritmasının tipi, kripto-periyodu, anahtar uzunluğu, kullanılan asıllama algoritmasının tipi ve diğer güvenliğe ilişkin bilgiler bu uygulama detayları içerisinde yer alır. Bazı teke gönderim ortamlarında, bu parametreler konaklar arasında belirlenebilir. Bu özellik, güvenlik parametrelerinin çakıştığı durumlarda faydalı olabilir. Ancak, çoğa gönderim grubunda çok fazla katılımcı olabileceği için, güvenlik

parametrelerin konaklar ile belirlenmesi etkin değildir. Bu yüzden, oturum gereksinimleri genellikle oturumu başlatıcı tarafından belirlenir ve potansiyel katılımcılara duyurulur. Başlatıcı, güvenli bir oturum için gerekli olan güvenlik parametrelerini bir Güvenlik Birimi (Security Association-SA) şeklinde saklayabilir.



Şekil 4.1 Güvenli grup içerisindeki konaklar

ile şifrelenirken, video trafiği başka bir anahtar ile şifrelenebilir ve her bir anahtar ayrı bir SA içerisinde yer alabilir. ESP kullanılırken, her bir IP datagramı içerisinde yer alan bir *güvenlik parametre indeksi* (Security Parameter Index-SPI), trafiğin şifresini çözmek için gerekli olan SA'yı tanımlar. Her durumda, SA'nın detaylarının güvenli oturum başlamadan önce tüm grup üyeleri tarafından biliniyor olması gereklidir.

Başlatıcı, oturumun başlamasından önce güvenli oturuma ilişkin SA'yı, güvenlik parametreleri haricindeki diğer parametrelerin (örneğin; oturum başlangıç zamanı, kullanılan oturum protokolü vb.) duyurulmasında kullanılan tekniklere benzer teknikler ile dağıtabilir. Bir oturumu duyurmak için genellikle iki metod kullanılır: İlan ve davet.

Başlatıcı bir oturum başlatmak isteğini, Oturum İlan Protokolü (Session Announcement Protocol-SAP) aracılığı ile duyurabilir. Duyuru, oturum duyurularının alınması için kullanılan belirli bir çoğa gönderim adresine yollanmak suretiyle, potansiyel üyelere duyurulabilir. Alternatif olarak, duyurular daha seçici bir gruba, potansiyel katılımcılar ile direkt olarak iletişim kuran Oturum Başlatma Protokolü (Session Initiation Protocol-SIP) gibi bir davet protokolü aracılığı ile dağıtılabilir. SAP ve SIP protokollerinin her ikisi de çoğa gönderim oturumu için gerekenleri tanımlamak için Oturum Tanımlama Protokolü'nü (Session Description Protocol-SDP) kullanırlar. Çoğa gönderim oturumu için gerekenler, güvenlik ve güvenlik harici parametreleri içerebilir. Bu duyuru teknikleri, güvenli oturum başlamadan önce potansiyel katılımcılara, güvenlik harici konferans bilgilerinin ve SA'nın dağıtılması için etkin bir yöntemdir.

4.4. GÜVENLİ ÇOĞA GÖNDERİM İŞLEMİ

Kullanılacak anahtar tahsisi yönteminin uygulama detayları ne olursa olsun, bir güvenli çoğa gönderim grubunun kurulması ve bakımı için ortak bir işlem kullanılabilir. [12] Özel çoğa gönderim oturumları için gereken kayıt ve asıllama işlemlerinden ortak bir işlem türetilebilir. Aşağıdaki örnekte, asıllama servislerinin, kamusal anahtar sertifika hiyerarşisi üzerine kurulu olduğu kabul edilmiştir. Bir sertifika birimi (CA), katılımcıların teşhis edilmesi için güvenilir ve merkezi bir otorite olarak hizmet vermektedir. Tüm katılımcıların dijital imzaların doğrulanması için CA'ya erişim yetkileri bulunmaktadır. Aşağıdaki adımlar, güvenli çoğa gönderim oturumları için gereken başlatma, kayıt ve bakım işlemlerini gözden geçirmektedir:

1. Başlatıcı güvenli bir oturum ihtiyacı belirtir.
2. Güvenli oturum isteği belirlendikten sonra, başlatıcı SDP protokolü gibi bir protokol kullanarak, oturuma katılmak için gerekli olan parametreleri belirler. Parametreler arasında, bir güvenlik birimi (SA) içerisinde tanımlanan güvenliğe

ilişkin detaylar olabileceği gibi güvenlik harici parametreler de (örneğin; oturum başlama zamanı, oturum adresi vb.) yer alabilir.

3. Güvenli oturumun başlamasından önce, başlatıcı, katılımcıların kayıt işlemleri ve anahtar dağıtım fonksiyonlarının yerine getirilmesi için yardım gerekli olup olmadığına karar verir. Bu fonksiyonlara yardımcı olacak güvenilir kaynakların yerleri ve kimlikleri oturum tanımı içerisine kaydedilebilir. Bu, katılımcıların oturum başlangıcında, ilgili güvenilir kaynak ile doğrudan iletişim kurmasına izin verir. Bazı anahtar tahsisi mimarilerinde, örneğin CBT, çoğa gönderim yönlendirme ağları için tanımlanan anahtar tahsisi mimarisi, anahtar dağıtma ve kayıt fonksiyonları, hemen başlangıçta, güvenilir bir ağ çekirdeğine aktarılır. Diğer mimariler, sadece gerekli olduğu durumlarda, diğer güvenilir kaynaklara görev dağıtımı yaparlar.

4. Güvenli oturuma üyelik sağlamak için, oturum tanımı potansiyel katılımcılara duyurulur. Duyuru, postalanmış bir davet şeklinde gelebileceği gibi, her bir grup üyesine ayrı bir özel davet şeklinde de ulaşabilir. Başlatıcı, davet edilen katılımcıları oturum erişim kontrol listesine (Access Control List-ACL) ekler. ACL yetkili kaynaklar tarafından tutulmalıdır. ACL üzerinde yapılacak olan değişiklikler bu güvenilir kaynaklar arasında paylaştırılmalıdır. Sırdaşlık oturum anonslarına güvenli oturumun varlığını gizlemek için uygulanabilir. Pek çok durumda, başlatıcı, hızlı ve verimli olarak oturumu tüm potansiyel katılımcılara oturumun başlamasından önce duyurmak için, çoğa gönderim tekniklerini kullanır.

5. Oturum duyurusunun alınmasından sonra, potansiyel katılımcılar, gereksinimlerini karşılayabileceklerse güvenli oturuma kayıt olabilirler. Kayıt, başlatıcı ya da diğer güvenilir kayanklar tarafından yapılır. Kayıt işleminin bir parçası olarak, katılımcılar asıllanır ve haklarının doğruluğu kanıtlanır. Sadece geçerli kullanıcılara güvenli grubun üyesi olmaları için davetiye sunulur ve gerekli şifreleme materyali dağıtılır. Kayıt sırasında değiş tokuş edilen anahtarlar, temel güvenlik servisleri tarafından sağlanmalıdır. Oturum güvenlik yönetimine bağlı olarak, gruba kayıt olan katılımcıların üyelik listesinin

sağlanması gerekebilir. Üyelik listesinin gözden geçirilmesinden sonra, bazı katılımcılar oturum içerisinde daha fazla kalmak istemeyebilir. Bu durumda, oturum güvenlik yönetimi, katılımcıları, oturumdan çıkarken oturum anahtarını silmeye zorlayabilir.

6. Güvenli oturumun gidişatı sırasında, anahtar tahsisi ve kayıt gibi birkaç bakım işinin yapılması gerekebilir. Çoğa gönderim iletişiminin dinamik özelliğini desteklemek için, oturumun başlamasından sonra, katılımcıları eklemek ya da silmek gerekebilir. Devam eden bir oturum sırasında, eklenen katılımcılar grup anahtar materyalini almadan önce, kanıtlanmış kayıt işlemini tamamlamaları gerekir. Ferdi ya da bir grup katılımcı gönüllü ya da zorunlu olarak güvenli oturumdan çıkarılabilir. Oturumdan gönüllü olarak çıkışta, katılımcı gönüllü olarak grubu terk eder ve tüm oturum anahtar materyallerini silmesi istenir. Oturumu yöneten güvenlik yöntemine bağlı olarak, katılımcıların elindeki anahtarlar güvenlikleri kaybolmuş olarak değerlendirilebilir. Anahtarlarının güvenliklerini yitirmeleri durumunda, zorunlu atma, bu anahtarlara sahip olan katılımcının güvenli oturumdan atılması için gereklidir. Katılımcıyı silmek için, potansiyel olarak güvenliği kaybolmuş olan tüm anahtarlar yenisi ile değiştirilmelidir. Yeniden anahtar tahsisi, etkilenen tüm katılımcılara yeni bir oturum anahtarının tahsis edilmesini içerir.

4.5. GÜVENLİ ÇOĞA GÖNDERİM KRİTERLERİ

Güvenli çoğa gönderim uygulaması geliştirirken, ele alınması gereken pek çok sorun vardır. [13] geliştiricilerin ele alması gereken güvenlik harici sorunlara genel bir bakış sunmaktadır. Bu makale, özel çoğa gönderim oturumları gerçekleştirilirken düşünülmesi gereken, güvenlik ile ilişkili sorunları da ele almaktadır. Bu sorunlardan bazıları çoğa gönderim trafiğini güvenli hale getirmek için kullanılan anahtar materyali ile ilgilidir. İlk olarak, kayıt işlemi anahtar materyaline erişim için, bir erişim kontrol seviyesi sağlamaktadır. Özel bir oturum için, kayıt işlemi sırasında tüm katılımcıların asıllanmaları gereklidir. Başarılı bir kayıt işlemi sonucunda, anahtar materyali ve grup üyeliği sunulabilir. Güvenli bir oturum için, grup üyeliği, oturumun IP çoğa gönderim

adresi ile tanımlanır ve anahtar materyali, diğer grup üyeleri ile güvenli olarak iletişim kurulması için gereklidir.

Çoğa gönderim güvenlik problemi için tam bir çözüm, oturum ilanı, kayıt vb. gibi güvenlikle ilişkili sorunları adresler. Daha önce bahsedildiği gibi, pek çok güvenlik sorununu anahtar yönetimi ve dağıtımı problemine indirgemek mümkündür. Bu, güvenli çoğa gönderim problemi için geliştirilen pek çok çözümün etkin bir şekilde karşılaştırılması ve değerlendirilmesine imkan sağlar. Adil bir karşılaştırma yapmak için, tüm uygulanabilir çözümler için bir grup kriter tanımlanmıştır. Bu kriterler, anahtar dağıtım verimliliği ve mimarinin genelde ölçeklenebilirliğine odaklanmak sureti ile, her bir anahtar tahsisi mimarisinin avantaj ve dezavantajlarının belirlenmesine yardımcı olur. Diğer sorunlar; katılımcı kayıt işlemleri, asıllama vb. burada ele alınmamıştır ancak bunlar kadar önem taşıyan konulardır.

Tablo 4.2 Güvenli çoğa gönderim mimarileri karşılaştırma kriterleri

Kriter	
Verimlilik: Başlangıç anahtar tahsisi	Oturumun başlangıç anahtar dağıtım değiş tokuş işlemlerinin verimliliği nedir?
Verimlilik: Yeniden anahtar tahsisi	Yeniden anahtar tahsisi (örneğin: anahtarın güvenliği kaybolduğunda ya da kripto-periyod değişimlerinde) işlemlerinin verimliliği nedir?
İşlem Gereksinimleri	Anahtar tahsis mesajlarını işlemek için, anahtar dağıtıcı ve üyeler tarafından gereksinim duyulan işlem gereksinimleri nelerdir?
Saklama Alanı Gereksinimi	Anahtar saklamak için katılımcılar tarafından gereksinim duyulan saklama alanı ne kadardır?

Ölçeklenebilirlik	Çözüm, küçük ve büyük gruplar için (örneğin; büyük grup; üye sayısı 100'den daha fazla olan grup) ölçeklenebilir midir?

Çoğa gönderim anahtar dağıtım problemi için pek çok makale yazılmıştır ve çözümler sunulmuştur. Ortaya sürülen çoğa gönderim anahtar tahsisi mimarilerinin değerlendirilmesi için kullanılan ana kriterler Tablo 4.2'de verilmiştir. Bu kriterler; anahtar dağıtım verimliliği, kurulu olan güvenli oturum içerisinden dinamik olarak kullanıcı girişi ve çıkışını destekleme yeteneği, anahtar tahsisi işlemlerini yerine getirmek için gereksinim duyulan işlem gereksinimleri, anahtar saklama alanı gereksinimleri ve ölçeklenebilirlik üzerine yoğunlaşmıştır. Verimlilik, "O" notasyonu ile gösterilecektir ve işlem başına yollanan anahtara ilişkin mesaj (örneğin; yeniden anahtar tahsisi, başlangıç anahtar tahsisi) sayısını ifade edecektir. Her bir mesajın büyüklüğü de ele alınacaktır. Ancak, bazı ağların iş çıkarma yeteneğinin zaman içerisinde artarak mesaj büyüklüğünü önemsiz hale getirdiği kabul edilmektedir. Kısıtlı bant genişliği olan ağlar için verimlilik birincil konu olmakla beraber, sistem başarımı için de uygun bir ölçüm sunmaktadır. [5]

4.6. ÇOĞA GÖNDERİMDE ANAHTAR YÖNETİMİ

Çoğa gönderim uygulamalarının güvenlik gereksinimleri gerçek uygulama içeriğine göre değişiklik gösterir. Ancak, çoğa gönderim iletişimini güvenli hale getirmek, teke gönderim iletişimini güvenli hale getirmekten farklıdır. Teke gönderimde, iki kullanıcı, paylaşılmış anahtar ile veriyi şifreleyip, özel anahtarları ile de veriyi imzalayarak sırdaşlık ve asıllamayı sağlayabilirler. Çoğa gönderimde, iletişimi güvenli hale getirmek için veriyi şifrelemek için kullanılan anahtarların tüm alıcılar tarafından biliniyor olması gereklidir. Bu durum grup üyeliğinin dinamik olması durumunda daha da karmaşık hale gelir. Çoğa gönderim iletişimini güvenli hale getirmek için, gruptan ayrılan üyelerin daha sonra yapılacak grup iletişimine erişememeleri gerekir. Bu, grup tarafında yeni bir şifreleme anahtarının kullanılması anlamına gelir. Aynı zamanda, yeni bir üye gruba katıldığında, bu yeni üyenin gruba dahil olduğu andan önceki grup iletişimine

erişememesi gereklidir. Bu, yeni bir üye gruba dahil olduğunda yeni bir grup anahtarının kullanılmasını gerektirir. [2]

Grup anahtarının kullanılması, alınan verinin grup içerisindeki bir üye tarafından yollanılmış olduğunu güvence altına alır. Verinin hangi üye tarafından yollanmış olduğunu göstermez. Alınan paketlerin kaynak asıllamalarının yapılması için başka bir metoda ihtiyaç vardır.

Şifreleme ve dijital imzaların kullanımı ile, istenilen seviyede sırdaşlık, veri bütünlüğü ve asıllama sağlanabilir. Düzgün olarak gerçeklendiği zaman, önemsiz kriptoanalitik ataklar tarafından kırılamayan güçlü bir güvenlik mekanizması kullanıldığını kabul edelim. Bu taktirde, güvenliğe ilişkin bakış açısı anahtar materyalinin korunmasına çevrilebilir. Genellikle kriptografik algoritmaların kırılamayacağı kabul edilir ve bu yüzden tüm güvenliğin anahtarda yattığı düşünülür. Onun için, anahtar yönetimi, anahtar dağıtımı ve anahtar materyali için erişim kontrolü güvenlik denilince ele alınması gereken konulardır. Bütün bunlar göz önüne alındığında, güvenli bir çoğa gönderim oturumu, Sınıf D tipi bir IP adres ve oturum için gerekli olan anahtar materyali ile tanımlanabilir. [5]

Büyüklük, tip (örneğin; asimetrik, simetrik) ve bir çoğa gönderim ortamını güvenli hale getirmek için gerekli olan anahtar sayısı, anahtarların yenilenme periyodu (kripto-periyod) kullanılan şifreleme mekanizması ve anahtar tahsisi mimarisi tarafından belirlenir. Genelde, oturum katılımcıları, oturum verisini şifrelemek için ortak bir *grup trafik şifreleme anahtarı* (Group Traffic Encryption Key-GTEK) kullanabilirler. Oturum başlatıcısı, grup üyelerine dağıtılan oturum anahtarlarının şifrelenmesi için, *grup anahtar şifreleme anahtarı* (Group Key Encryption Key-GKEK) kullanabilir. Özel bir çoğa gönderim oturumunda, tüm oturumun güvenliğini sağlamak için, bu anahtarlara erişim kontrollü olmalıdır. Bu yüzden, kayıt işlemleri sırasında, anahtar materyalinin dağıtımını yapmadan önce, potansiyel katılımcının asıllanması için güçlü asıllama mekanizmalarının kullanılması gerekmektedir. Erişim kontrol mekanizması uygulamaya özel olabilir. Örneğin, kimlik tabanlı erişim kontrol mekanizmaları ticari uygulamalar için uygun olabilirken, askeri uygulamalarda, katılımcının bilgi erişim yetki seviyesini tanımlayan yetki tabanlı yöntemler kullanılabilir. Bu kişisel öznitelikler imzalanmış bir

sertifikaya bağlandığında, sertifikanın dijital imzası ve sertifika hiyerarşisi ile ilişkisi kullanılarak, katılımcının kimliği ve atanmış olan yetkileri doğrulanabilir. [5]

Sistemin güvenlik yönetimi ve belirli bir anahtar ile şifrelenen trafik miktarına bağlı olarak, çoğa gönderim oturumuna periyodik olarak yeniden anahtar tahsis etmek gerekebilir. Yeniden anahtar tahsisi, anahtarlardan birinin güvenliğinin kaybolması durumunda da gerekebilir. Bu durumda, güvenliği kaybolmuş olan alanın, ileriki iletişimden çıkarılması gereklidir. Bu yüzden, bir yeniden anahtar tahsisi, güvenliği kaybolmuş olan alanın güvenli gruptan çıkarılmasını sağlarken, aynı zamanda, geri kalan üyelerin grup üyeliklerinin devam etmesini de sağlamalıdır. Kullanılan güvenlik yönetimine bağlı olarak, güvenliğin kaybolması, katılımcının güvenli oturumdan gönüllü olarak çıkmasını da içerebilir. Böyle bir durumda, çıkarılan katılımcının ileriki bir zamanda oturum için yeniden kayıt olmadan gruba yeniden katılmasını engellemek için tüm gruba yeniden anahtar tahsisi gerekebilir. Buna ek olarak, anahtar tahsisi mekanizması, bir grup üyenin grup anahtarını yeniden yaratmak ya da elde etmek için bir araya gelmesini de engellemelidir.

Çoğa gönderimde anahtar yönetiminin detayları ve kullanılan mimariler bir sonraki bölümde ayrıntılı olarak ele alınacaktır.

5. ÇOĞA GÖNDERİMDE ANAHTAR YÖNETİMİ

Anahtar yönetimi alt yapısı, kilobit/saniye ile gigabit/saniye arasında değişen bant genişliğini desteklemeli, değişik çoğa gönderim grup büyüklükleri ile başa çıkabilmeli ve mobil ya da bağlantısız teknolojiler ile oluşturulmuş iletişim ortamlarını ele alabilecek kadar esnek olmalıdır. Bu başarım ve iletişim gereksinimlerine ek olarak, farklı senaryolar için güvenlik gereksinimleri de çok çeşitli olabilir. Kullanıcıların tek tek ya da grup halinde, güvenli ve verimli olarak eklenip çıkarılabilmeleri gereklidir. Sistem, çıkarılmış olan kullanıcıların kendi özel bilgilerini paylaşmış bile olsalar, daha sonraki trafiği okuyamamaları için yeteri kadar dirençli olmalıdır. Burada, minimize edilmeye çalışan maliyetler; kurulum için gereken zaman, her bir son kullanıcı için gereken saklama alanı ve kurulum, yeniden anahtar tahsisi ve bakım için gerekli olan toplam iletim sayısıdır. Önerilecek çoğa gönderim güvenlik mekanizmalarının, ağ katmanından daha alt katmanlarda gerçeklenmeyeceği düşünülmelidir. Herhangi bir anahtar yönetim sistemi için, bant genişliği verimliliği de göz önüne alınması gereken bir konudur. [12]

5.1. ANAHTAR YÖNETİM MİMARİSİNE İLİŞKİN HUSUSLAR

İstenilen anahtar yönetim mimarisinin geliştirilmesi için göz önünde bulundurulması gereken pek çok faktör bulunmaktadır. Bunlar arasında; güvenlik seviyesi, maliyet, sistemin ilk kullanıma hazırlanması, yönetim meseleleri, erişim kontrol prosedürleri, başarım gereksinimleri ve destek mekanizmaları sayılabilir. Çoğa gönderim gruplarına özel diğer konular arasında şunlar da yer alır:

- Grup üyelerinin güvenlik gereksinimleri nelerdir? Muhtemelen bir grup kontrolörü ya da kontrolörleri olacaktır. Diğer grup elemanları kontrolör\kontrolörler ile aynı güvenlik gereksinimlerine sahip olacaklar mıdır?

- Ara yönetim bölgesi meseleleri - Bir “grup yönetim bölgesi”nden, potansiyel olarak güvenlik yönetimi farkı olan başka bir yönetim bölgesine geçiş yapılırken hangi yönetim uygulanacaktır? Birbiri ile iletişime geçmek isteyen ancak farklı kripto-periyoduna ve/veya anahtar uzunluğu yönetimine sahip olan iki kullanıcı örnek olarak verilebilir.
- Çoğa gönderim grubu nasıl oluşturulmalıdır? Kullanıcının gruba katılma işlemini kontrolör mü yoksa çoğa gönderim grubuna katılmak isteyen kullanıcı mı başlatacaktır?
- Belirli bazı grup üyelerinin, ağ anahtarının oluşturulması işlemini geciktirebilecek düzeyde, alt işleme kapasitesi olması durumu ile nasıl başa çıkılacaktır? Bu kullanıcılar, çoğa gönderim grubunun oluşmasını mı geciktirecekler yoksa diğer katılımcıların daha çabuk çalışır duruma gelmelerine izin verecek şekilde, daha sonra mı çevrimiçi olacaklardır?
- Çoğa gönderim grup anahtarı dağıtımı için gerekli bit sayısını minimize etmelidir. Bu durumun, özellikle bant genişliği kısıtlı olan donanımlar üzerinde önemli bir etkisi vardır.

Bütün bu meselelerin, çoğa gönderim için seçilecek olan protokol ile birlikte göz önünde bulundurulması gereklidir.[12]

5.2. ANAHTAR DAĞITIM MİMARİLERİ

Güvenli çoğa gönderim uygulamaları için anahtar tahsisi çözümünün verimliliğini iyileştirmek üzere, çoğunlukla çoğa gönderim iletişiminin özelliklerinden faydalanılır. Bir çoğa gönderim ortamında ideal anahtar dağıtım verimliliği O(1)’dir. Böyle bir senaryoda, merkezi bir sunucu, gruba yeniden anahtar dağıtma işlemi için tek bir anahtar dağıtma mesajını tüm gruba yollayabilir. Her bir grup üyesi, bu tek mesaj içerisinden gerekli anahtar materyalini çıkarabilir. Tersine, teke gönderim teknikleri kullanılarak tüm grup üyelerine ayrı ayrı olarak grup anahtarının dağıtılmasının verimliliği O(n)’dir. Ancak, pek çok durumda, katılımcıların başlangıç anahtar tahsisinin kayıt sırasında teke gönderim şeklinde yapılmasının çok daha verimli olduğu unutulmamalıdır. Kayıt

fonksiyonları, doğal olarak, tek bir katılımcı ve başlatıcı ya da diğer güvenilir kayıt birimleri arasında bire-bir olarak yapılır. Anahtar dağıtımı ile kayıt işlemlerini birleştirmek, her iki fonksiyonu da yerine getirmek için gerekli olan aktarım sayısını azaltabilir.[5]

Anahtar tahsisi fonksiyonları merkezi ya da mimarinin başından sonuna kadar dağıtılmış olabilir. Merkezi mimaride, anahtar tahsisi fonksiyonları tek bir güvenilir birim ile sınırlandırılmıştır. Bazı durumlarda, bu birim oturum başlatıcısı ya da bu çok önemli fonksiyonların yerine getirilmesi için başlatıcı tarafından atanmış başka bir kaynak olabilir. Ölçeklenebilirlik açısından, anahtar tahsisi ve kayıt fonksiyonları diğer güvenilir birimlere dağıtılabilir. Tekten-çoğa tip uygulamalar merkezi mimarilerin avantajlarından yararlanabilir. Alternatif olarak, dağıtık mimariler, işlem ve saklama alanı gereksinimleri tüm ağ boyunca dağıtılmış olduğu için daha ölçeklenebilir bulunabilir.

Aşağıda, birkaç anahtar tahsisi mimarisinden özet olarak bahsedilecektir. Mimarilerin özet olarak yapılacak tanımından sonra başarım analizleri verilecektir. Mimarilerin başarımları daha önceki başlıkta verilen kriterlere göre ölçülecektir. Oturumun tanımlanması, duyurulması ve kayıt işlemleri göz ardı edilecek, sadece anahtara ilişkin kriterler ele alınacaktır.

5.2.1. ELLE ANAHTAR DAĞITMA

Elle anahtar dağıtma yönteminde, simetrik anahtar, kamusal anahtar değiş tokuşuna gerek kalmadan dağıtılır. Bu mimaride, tüm anahtar yaratma ve dağıtma fonksiyonları, merkezi anahtar dağıtma merkezi (Key Distribution Center-KDC) içerisinde yer alır. Anahtar gereksinimleri başlatıcı tarafından belirlenerek KDC'ye önceden bildirilmelidir. Başlatıcı grup anahtarı ile birlikte bir alternatif anahtar yaratılması ve grup anahtarının katılımcılara dağıtılması için KDC'ye istekte bulunur. Önceden belirlenmiş bir sürenin geçmesinden sonra, tüm katılımcılar yeni grup anahtarına geçiş yaparlar. Katılımcılar, bir sonraki kripto-periyoduna kadar, yollanmış olan bu anahtarı kullanır. Bu süre içerisinde, anahtarın güvenliğinin kaybolması durumunda alternatif anahtar katılımcılara dağıtılır. Katılımcılar alternatif anahtarı alır almaz ya da başlatıcıdan alternatif anahtara geçildiği mesajını alır almaz bu yeni anahtarı kullanmaya geçerler. Bu işlemler her bir kripto-periyodunda tekrarlanır.

Bu tür bir mimari, halihazırda kullanılıyor olması ve kullanıcılar tarafından kolaylıkla anlaşılır olması nedeniyle çekici olarak görünebilir. Ancak, KDC ile önemli bir çevrim-dışı koordinasyon gerektirdiği için, ölçeklenebilir değildir. Aynı zamanda, güvenli çoğa gönderim grubuna dinamik kullanıcı giriş ve çıkışlarına cevap vermek için oldukça yavaş kalmaktadır. Elle anahtar tahsisi teknikleri, kullanıcıların zorunlu çıkarılmalarına cevap vermek için de yavaş kalmaktadır. Grup anahtarının güvenliğinin kaybolması durumunda, yeni anahtarın, geçerli katılımcılara elle dağıtılması gerekir.

5.2.2. N KÖK/YAPRAK ÇİFTLİ (PAIRWISE) ANAHTAR YAKLAŞIMI

Bu mimaride, başlatıcı, belirli bir oturum için güvenlik özniteliklerini belirler; istenilen grup katılımcılarının bir listesini üretir ve bu listeyi bütün grup katılımcılarına dağıtır. Arkasından yaprak, bu katılım çağrısına kabul ya da red cevabı döner. Bu listenin önceden yollanması, oturuma katılımı reddetmiş olan katılımcılar ile başlatıcı arasında anahtar değişimlerinin yapılmaması nedeniyle zaman kazanımı sağlayacaktır. Başlatıcı (kök), bazı standart kamusal anahtar değiş tokuş tekniklerini (örneğin; Diffie-Hellman kamusal anahtar değiş tokuş yöntemi) kullanarak çoğa gönderim grubundaki katılımcılardan (yapraklardan) biri ile bir çiftli anahtar üretir. Kök, ardından, çoğa gönderim güvenlik işbirliği parametrelerini bu ilk yaprağa verir. Parametreler, sınıflandırma ve yönetim gibi maddeleri içerebilir. Bazı parametrelerin karşılıklı olarak üzerinde anlaşılması gerekebilir. Yaprağın, yukarıdaki parametrelere ve çoğa gönderim listesine dayanarak, çoğa gönderim grubuna katılmayı reddetme ihtimali mevcuttur. Eğer yaprak bu oturumu red ederse, kök bu işlemi başka bir yaprak ile tekrar eder.[12]

Yapraklardan biri çoğa gönderim oturumunu kabul ettiğinde, bu yaprak ve kök, çoğa gönderim grubu için kullanılacak olan bir grup anahtarı seçerler. Grup anahtarı, bu yaprak ve kök arasında başka bir kamusal anahtar değiş tokuşu sayesinde üretilebileceği gibi, çoğa gönderim grubu için geçerli olan yönetime (yönetim seçimi gerçek zamanlı bir seçim olmayacaktır) dayanılarak kök tarafından da üretilebilir.

Ardından, kök, başka bir yaprak ile bir çiftli anahtar değiş tokuşu yapar ve bazı durumlarda, yukarıda bahsedilen parametreler üzerinde anlaşılması işlemini bu yaprakla da yapar. Yaprak, çoğa gönderim oturumuna katılmayı kabul ettikten sonra kök, grup anahtarını yaprağa gönderir.

Bu çiftli anahtar değiş tokuşu ve grup anahtarının dağıtımı, çoğa gönderim grubunda yer alan tüm n adet kullanıcı ile gerçekleştirilir.

Kök ve yapraklar, çiftli anahtarları daha sonraki kullanımlar için önbelleklerler. Bu anahtarlar, Anahtar Şifreleme Anahtarı (Key Encryption Key-KEK) olarak, ağdaki yaprakların yeniden anahtarlanmasında kullanılmak üzere hizmet verirler. [12]

Yukarıda da anlatıldığı gibi, böyle bir mimaride güvenli oturum için gerekli olan anahtar materyali başlatıcı tarafından dağıtılır. Başlatıcı, aynı zamanda kayıt ve asıllama işlemlerini de yapabileceği gibi bu işlemleri başka bir güvenilir kaynağa da aktarabilir. Anahtar tahsisi fonksiyonları da güvenilir kaynaklar arasında dağıtılabilir. Kayıt işlemi sırasında başlatıcı, kendisi ve her bir katılımcı arasında, bir çiftli anahtar tahsisi ilişkisi yaratarak bir tekil oturum anahtarını yaratır ve önbellekler. Bu katılımcı tekil oturum anahtarları, her bir kimliği kanıtlanmış katılımcı için grup anahtarının şifrelenmesinde kullanılır. Şifrelenmiş anahtarlar her bir katılımcıya ayrı ayrı olarak ya da her bir şifrelenmiş anahtarı içeren tek bir mesajın çoğa gönderilmesi ile dağıtılabilir.

[8]'de, Ballardie Çekirdek Tabanlı Ağaç (Core Based Tree-CBT) Çoğa Gönderim Yönlendirme protokolüne dayanarak, bir çiftli çoğa gönderim anahtar dağıtım çözümü ileri sürmektedir. CBT mimarisinde, bir yönlendirme ağacı bir çekirdek yönlendirici etrafında merkezlendirilmiştir. Çekirdek yönlendirici, asıllama ve anahtar dağıtma işlevlerini yerine getirmek amacıyla çoğa gönderim grubunun merkezinde yer almaktadır. Diğer yönlendiriciler ağaca girmek istediğinde, çekirdek ile bir asıllama işlemine maruz kalabilirler. Ağaca diğer güvenilir yönlendiricilerin eklenmesi sayesinde, anahtar tahsisi fonksiyonları çekirdek yönlendirici dışında dağıtılabilir. Bu sayede çözüm, çok büyük gruplar için de ölçeklenebilir hale gelir. [5]

CBT mimarisinde, güvenli oturumun başlatıcısı, oturum için bir erişim kontrol listesi (ACL) ve güvenlik birimi (SA) yaratır. ACL ve SA daha sonra oturum için GTEK ve GKEK'i yaratacak olan çekirdeğe yollanır. Çekirdek ağaca asıllanmış ikincil yönlendiriciler eklendiğinde, ACL, GTEK ve GKEK'i bu yönlendiricilere de dağıtır. Ballardie, grup üyeleri ve yönlendiriciler arasında anahtarların dağıtılması için anahtar tahsisi protokolü olarak ISAKMP (Internet Security Association and Key Management Protocol-ISAKMP) yi kullanmayı tavsiye etmiştir.

Bu şekilde bir yapı kullanıcılar için çekici olabilir; çünkü kamusal anahtar değiş tokuş tekniklerinin doğruluğu kanıtlanabilir basit bir uzantısıdır ve üçüncü partileri gerektirmez. Çoğa gönderim oturumunun katılımcıları, sadece anahtar tahsisi mekanizmasına katılabilirler. Bu yapının tercih edilmemesinin en büyük sebebi, kullanıcı sayısını belli bir orana göre arttırdığımızda, işlem açısından yeterli katılımcılar için bile iletim açısından hassas hale geliyor olmasıdır. Yetkisiz bir katılımcının düşürülmesi gerektiği her durumda yeni ağ anahtarının dağıtılması gerekmektedir. Bu dağıtım, kökten her bir katılımcıya bir iletim yapılmasını gerektirir. Çünkü, yeni ağ anahtarı, her bir kullanıcının tekil KEK'i ile şifrelenmektedir.

Her bir katılımcının ayrı ayrı anahtarlanması gerektiği için, çiftli yaklaşımın başlangıç anahtar tahsisi iletim verimliliği O(n)'dir. Gruba yeni bir grup anahtarı dağıtmak, güvenilir olmayan katılımcıların çıkarıldığı başka bir güvenli grubun kurulmasını gerektirir. Bu nedenle, başlangıç anahtar tahsisi fonksiyonlarında olduğu gibi, yeniden anahtar tahsisi fonksiyonu O(n) verimliliğine sahiptir.

Bu mimarinin birkaç değişik şekli mevcuttur. Örneğin, kökün tüm N anahtar değişimini yapması yerine, kök bu fonksiyonalitenin bir kısmını altında yer alan bazı yapraklara geçirebilir. Örneğin, çoğa gönderim listesi ikiye ayrılabilir ve kök, listenin diğer yarısı ile ilgilenirken, bir yaprağa kullanıcıların yarısını almasını ve anahtar değişimlerini yapmasını söyleyebilir. Böyle bir yaprağa *"alt kök"* adı verilir. Bu yaklaşım N büyük sayılara giderken daha iyi sonuçlar vemektedir ve daha iyi ölçeklenebilir. Özellikle, çoğa gönderim grubunun daha kısa sürede kurulmasına sebep olur. Çünkü, çiftli anahtar değiş tokuşunun yapılması ve grup anahtarının dağıtılması birden çok grup katılımcısı tarafından paylaşılır ve paralel olarak gerçekleştirilebilir. Bu da bu işlemlerin sadece kök tarafından yerine getirilmesine göre daha az zaman almasına sebep olur.

Bu yöntemde kök, her bir alt kökün güvenilir olduğunu kabul etmektedir. Her bir alt kök çoğa gönderim boyunca sistemde kalmalıdır. Eğer sistemden ayrılmak isteyen bir alt kök var ise bu durumda grup anahtarının yeniden gönderilmesinin yanısıra kökün bu alt kök yerine geçecek olan yaprağa bu görevi vermesi gerekmektedir. Yeni alt kök bu görevi aldıktan sonra listesinde yer alan yapraklar ile anahtar değiş tokuşu yapar. Arkasından,

kök, grup anahtarını dağıtır. Eğer sistemden ayrılan bir yaprak ise, bu taktirde, sadece son basamağın yapılması gerekecektir.

5.2.3. TÜMLEYİCİ DEĞİŞKEN YAKLAŞIMI

N adet yaprağımız olduğunu düşünelim. Kök tüm yaprak i (i=1,2,....,N) ile kamusal anahtar değiş tokuşu işlemini gerçekleştirir. Kök, her bir çiftli KEK'i önbellekler. Her bir yaprak kendi KEK'ini saklar. Kök, çoğa gönderim grup listesini ve özniteliklerini tüm katılımcılara yollar. Katılımcılar, daha önceki bölümde anlatıldığı gibi, çoğa gönderim oturumuna katılmayı kabul ya da reddederler. Kök, grup anahtarını, çoğa gönderim grubunda yer alan her bir yaprağın kendi tekil KEK(i)'si ile şifreleyerek gönderir. Şifrelenmiş grup anahtarının yanı sıra kök, *tamamlayıcı değişkenler* adı verilen bazı bilgileri de şifreleyerek yapraklara yollar.

Bir yaprak, kendine özel tamamlayıcı değişken haricinde, diğer N-1 yaprağa ilişkin tamamlayıcı değişkenleri alır. Kök, grup anahtarını ve tamamlayıcı değişken j'yi, j=1,2,...,N ve j<>i , KEK(i) ile şifreleyerek her bir yaprağa yollar. Dolayısı ile, her bir yaprak N adet değişken alır ve saklar. Bunlar ağ anahtarı ve N-1 adet tamamlayıcı değişkendir.

Böyle bir mimaride, bir kullanıcının çoğa gönderim grubundan çıkması ve geride kalan katılımcıların yeni grup anahtarı ile çalışır duruma gelmesi için şunlar yapılır: Örneğin, yaprak 20'nin çıkmak istediğini düşünelim. Bu durumda "yaprak 20'yi ağdan çıkar" mesajı tüm gruba yollanır. Diğer bütün yapraklar ve kök o anki grup anahtarı ve yaprak 20'nin tamamlayıcı değişkenini kullanarak yeni grup anahtarını üretir. Yaprak 20 haricinde tüm yapraklar bu yeni grup anahtarını üretebilir. Çünkü yaprak 20 kendi tamamlayıcı değişkenini tutmamaktadır.

Bu şekilde bir yapı, iletim sayısı kazancı bakımından çok çekici görülebilir. Çünkü, bir kullanıcının ağdan çıkması durumunda, yeniden anahtar tahsisi mesajının her bir i yaprağının KEK(i) ile şifrelenip teker teker yollanmasını gerektirmez. Başka bir değişle, bu yapıda, çoğa gönderim grubuna gönderilen tek bir düz metin mesaj vardır. Bunun yanında, diğer yöntemlerde, N adet şifrelenmiş mesaj yollanması gerekmektedir.

Bu yöntemin iki önemli dezavantajı vardır. Birincisi, (N-1) tamamlayıcı değişken için gerekli olan saklama alanıdır. İkincisi ise, birden fazla kullanıcının ağdan çıkarılması durumunda bu kullanıcıların bir araya gelerek, tamamlayıcı değişkenlerini birleştirip yeni grup anahtarını üretmeleri ve çoğa gönderim oturumuna ulaşabilmeleridir.

5.2.4. HİYERARŞİK AĞAÇ YAKLAŞIMI

Ağacın kökünde yer alan ve çoğa gönderim trafiğini şifrelemek için kullanılan GTEK ile anahtar şifreleme anahtarlarından (KEKs) oluşan bir hiyerarşik ağaç yaratılır. Katılımcılar, kendi tekil KEK'leri ile, bu ağacın birer yaprağı olurlar. Bir katılımcının üzerindeki her bir kat, farklı bir KEK'e karşılık düşer. Katılımcılar, kendileri ile kök arasındaki yol boyunca, ağaç üzerinde yer alan tüm anahtarları saklarlar. Anahtar güvenliğinin kaybolması durumunda başlatıcı, tek bir katılımcıyı ya da bir grup katılımcıyı gruptan çıkarmak için, yeniden anahtar tahsisi sırasında aşamalı KEK'leri kullanabilir. Yeniden anahtar tahsisi işlemi sırasında tüm güvenliği kaybolmuş KEK'ler ve GTEK değiştirilir.

Güvenliğin kaybolması durumunda, ağaçta bu durumdan etkilenmiş olan katılımcıların yeniden anahtarlanması için gerekli iletim sayısı, O(logn) derecesindedir. k-lı (k-ary), yüksekliği d olan[1] bir ağacta yeniden anahtar tahsisi için gerekli mesaj sayısı (k-1)d'dir. Her bir katılımcı için gereken saklama alanı gereksinimi d+1 anahtar derecesindedir. Başlatıcı, ağaçtaki tüm anahtarları saklamalıdır.

Bu yöntemde, verimlilik, k-lı ağaçlarda böl-ve-yönet algoritması kullanılarak sağlanır. Ağacın güvenliği kaybolmuş olan kısmı için birden fazla yeniden anahtar tahsisi mesajı gerekirken, ağacın diğer az etkilenen kısımları daha küçük ve daha az sayıda mesaj ile yeniden anahtarlanabilir. Bu özellik, bu yöntemin çok büyük gruplar için ölçeklenebilir olmasını sağlamaktadır.

Hiyerarşik ağaç yönteminde, anahtar tahsisi işlem gereksinimi başlatıcının tarafında oldukça fazladır. Ağacın başlangıç anahtarlanması sırasında, başlatıcı, çiftli yaklaşımda olduğu gibi her bir katılımcı ile tekil bir KEK üretmelidir. Buna ek olarak, başlatıcı

[1] Her bir katılımcıya yollanan yeniden anahtar tahsisi mesajı birden fazla anahtar içerebilir

ağaçtaki her bir kat için anahtarlar üretir. Yeniden anahtar tahsisi sırasında başlatıcı, yeni grup anahtarını ağaç hiyerarşisindeki KEK sıralaması ile şifreler.

Bu mimarinin diğer mimarilere göre üstünlükleri şunlardır:

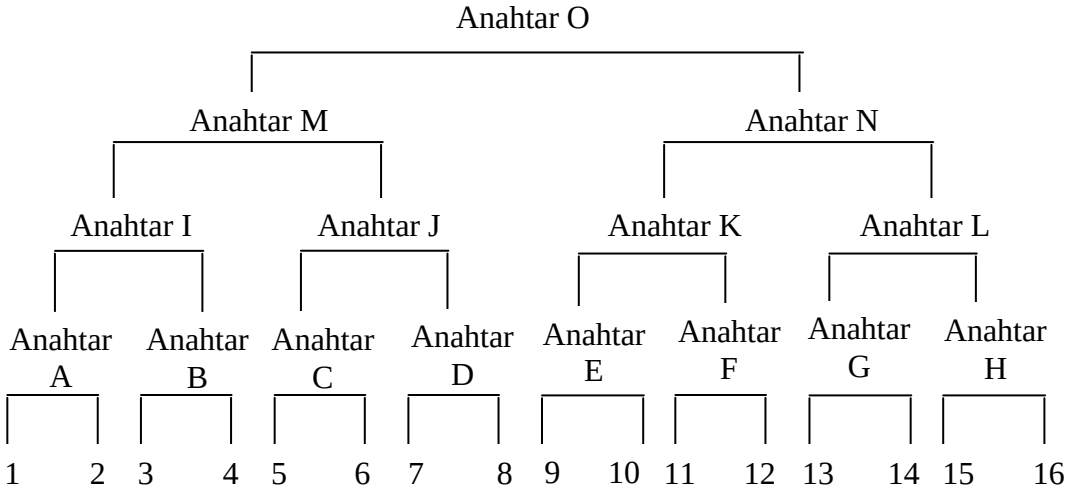
1. Anahtarlarının güvenliği kaybolmuş olan kullanıcının çoğa gönderim grubundan güvenli olarak çıkarılmasını sağlar.
2. İletim verimliliği sağlar.
3. Saklama alanı verimliliği sağlar.

Bu açılardan bakılacak olursa bu mimari, diğer mimarilere göre tercih edilecek bir yaklaşımdır. Yeni grup anahtarının dağıtımını kolaylaştırmak için hiyerarşik yardımcı anahtarlar sistemi kullanarak zaman maliyeti, saklama alanı ve gerekli mesaj iletim sayısını dengeler. Her bir kullanıcı için gereken saklama alanı gereksinimi ve anahtar değişimi için gerekli olan iletim sayısı kullanıcı sayısının logaritmik fonksiyonudur. Bu yaklaşım, gruptan çıkarılmış olan kullanıcıların gizlice anlaşmalarına karşı sağlamdır. Üstelik, yöntem doğal olarak hiyerarşik olduğundan, sunucunun dışında bir altyapı sistemine gerek duyulmamaktadır.

Bu yaklaşımın, avantajları ve dezavantajları aşağıda daha detaylı olarak açıklanacaktır. Şekil 5.1'i ele alalım. Bu şekil, anahtarların sunucu ve kullanıcılarda tutulduğu, mantıksal anahtar dağıtım mimarisini göstermektedir. Bu mimaride sunucu, A'dan O'ya kadar tüm anahtarları ve her bir kullanıcının şahsi KEK'ini bulundurur. Kullanıcı 11 ise, kendi şahsi KEK'ine ve F, K, N, O anahtarlarına sahiptir.

Bu noktada, anahtar hiyerarşinin organizasyonundan ve kurulum işlemlerinden bahsetmek yararlı olacaktır. Hiyerarşi kurulduktan sonra kullanıcıların nasıl ekleneceği açıkça belli olur. Burada, bir kullanıcının nasıl çıkarılacağı da tarif edilecektir.

Şekil 5.1'de verildiği gibi, her bir yaprağın bir kullanıcıya karşılık geldiği, tek kökü olan bir ağaç oluşturalım. Burada, inceleme kolaylığı açısından, örnek olarak dengeli bir ikili ağaç verilmiştir. Ağacın ikili ya da dengeli olması zorunluluğu yoktur. Her bir kullanıcı sunucu ile bir anahtar çifti oluşturur. İletim kapasitesi olan kullanıcılar ile bu işlem, kamusal anahtar değiş tokuş protokolü aracılığı ile sağlanabilir.



Şekil 5.1 Mantıksal anahtar dağıtım mimarisi

Tüm kullanıcılar, sunucu tarafından bilinen bir çiftli anahtara sahip olduğunda, sunucu oturum için tanımlı olan güvenlik yöntemine göre, ağaç üzerindeki diğer düğümler için anahtarlar üretir. Bu düğümler mantıksal düğümlerdir, kullanıcı içermezler. Bu düğümler için anahtarlar, yalnızca sunucu tarafından üretilebilir. Ardından sunucu, çocukları ağacın yaprakları olan düğümlerden başlayıp, köke kadar, bu düğümlerde yer alan anahtarları, o düğümün çocuklarının anahtarları ile şifreleyerek yollar. Bu işlemin sonunda her bir kullanıcı, kendi yaprağı üzerinde yer alan düğümlere karşı düşen anahtarları elde etmiş olur. Ayrıca, her bir kullanıcı grup anahtarı olarak kullanılan kök anahtarını da tutacaktır. d yüksekliğinde yer alan bir kullanıcı için $d+1$ adet anahtar saklama alanı gerekecektir. Bu anahtarlardan bir tanesi, sunucu ile birlikte üretilen kullanıcının şahsi KEK anahtarı, bir tanesi grup anahtarı, diğerleri ise ara düğümlerdeki anahtarlardır.

Tüm ara düğüm anahtarlarını, kullanıcıların şahsi anahtarları ile şifreleyerek, tüm ara düğüm anahtarlarını ve grup anahtarını tek bir mesaj içerisinde iletmek mümkündür. Bu durumda tüm anahtarların alınması için, her bir kullanıcı başına, yalnızca bir iletim yeterlidir. Ancak, yaprakların anahtarları ayırt edebilmesi için bir yöntem gerekli olacaktır.

5.2.4.1. ÇIKARMA PRENSİBİ

Çoğa gönderim grubundan Kullanıcı 11'in çıkarılmak istendiğini düşünelim. Bu durumda, Kullanıcı 11 tarafından tutulan tüm anahtarların değiştirilmesi ve yeni anahtarların, Kullanıcı 11 tarafından elde edilmesi engellenerek, bu anahtarlara ihtiyaç duyan diğer kullanıcılara aktarılmaları gerekir. Bunun için, Kullanıcı 11'in elinde bulundurduğu anahtarlar, aşağıdan yukarıya olarak değiştirilir. Sunucu, en alttaki düğüm için yeni bir anahtar üretir, çocuklarının anahtarları ile şifreler ve yollar. Dolayısı ile bizim örneğimiz için, değiştirilecek ilk anahtar F anahtarıdır ve yeni anahtar Kullanıcı 12'nin şahsi anahtarı ile şifrelenerek yollanacaktır.[12]

Aşağıdan yukarıya doğru işlem yaptığımız için, her bir anahtarın değiştirilmesi işlemi, o anahtarın başka bir anahtarı şifrelemek için kullanılmasından önce yapılmış olmalıdır. Dolayısı ile K anahtarının değiştirilmesi işleminde, bu yeni anahtar daha önce değiştirilmiş olan F anahtarı ile şifrelenir ve Kullanıcı 12'ye yollanır. Ayrıca tek bir mesaj olarak E anahtarı ile şifrelenerek Kullanıcı 9 ve 10'a çoğa gönderim mesajı olarak yollanabilir. Anahtar N'nin değiştirilmesi için, Kullanıcı 9, 10 ve 12'ye yollanmak üzere yeni anahtar, daha önce değiştirilmiş olan K anahtarı ile şifrelenecek, ayrıca Kullanıcı 13, 14, 15 ve 16 tarafından paylaşılan L anahtarı ile şifrelenerek bu kullanıcılara yollanacaktır. O anahtarının değiştirilmesi için, yeni anahtar daha önce değiştirilmiş olan N anahtarı ile şifrelenerek Kullanıcı 9, 10, 12, 13, 14, 15 ve 16'ya yollanacak, Kullanıcı 1, 2, 3, 4, 5, 6, 7 ve 8'e yollanmak üzere de M anahtarı ile şifrelenecektir. Gerekli olan iletim sayısı, değiştirilen düğümlerin derecelerinin toplamına eşittir. k-lı bir ağaç için gerekli iletim sayısı, en kötü durumda d ağacın yüksekliği olmak üzere, kd-1 olacaktır. Burada verilen örnekte, Kullanıcı 11'in çoğa gönderim grubundan çıkarılarak diğer 15 kullanıcı ile yeni bir güvenli grubun kurulması ve Kullanıcı 11'in sahip olmadığı yeni grup anahtarının bu kullanıcılara dağıtılması için, 7 adet iletim gereklidir.[12]

Bir önceki kısımda anlatıldığı gibi, birden fazla anahtarın tek mesaj içerisinde yollanması sağlanırsa, iletim sayısı 4'e kadar inecektir. Bu durumda, iletilen bit sayısı bakımından mesajın daha büyük olacağı göz önüne alınmalıdır. Buna ek olarak, her bir düğüm tarafından hangi anahtarın hangi düğüme karşılık geldiğinin belirlenmesi için bir mekanizmanın olması gerekmektedir. Dolayısı ile buradaki örnekte, değişen F, K, N ve

O anahtarları için üretilen yeni anahtarlar tek bir mesaj olarak Kullanıcı 12'nin şahsi anahtarı ile şifrelenip Kullanıcı 12'ye yollanabilir. Kullanıcı 9 ve 10'a yollanmak üzere değişen K, N ve O anahtarları tek bir mesaj olarak Kullanıcı 9 ve 10 tarafından paylaşılan düğüm anahtarı E ile şifrelenebilir. Kullanıcı 13, 14, 15 ve 16 için, değişen N ve O anahtarları tek bir mesaj olarak bu kullanıcılar tarafından paylaşılan düğüm anahtarı L ile şifrelenebilir. Son olarak Kullanıcı 1, 2, 3, 4, 5, 6, 7 ve 8 için, değişen O anahtarı tek bir mesaj olarak bu kullanıcılar tarafından paylaşılan M anahtarı ile şifrelenerek yollanabilir. Bu durumda gerekli olan iletim sayısı en kötü durumda $(k\text{-}1)d$ adet olacaktır.

Tablo 5.1'de bir kullanıcının çoğa gönderim grubundan çıkarılması için gerekli olan iletim ve saklama alanı gereksinimlerinin kullanıcı sayısı ile nasıl arttığı gösterilmektedir.

Tablo 5.1 Saklama alanı ve iletim maliyetleri

Kullanıcı sayısı	Derece (k)	Saklama alanı (d+1)	Yeniden anahtar tahsisi için gerekli iletim sayısı -mesaj başına tek anahtar (kd-1)	Yeniden anahtar tahsisi için gerekli iletim sayısı -mesaj başına birkaç anahtar (k-1)d
8	2	4	5	3
9	3	3	5	4
16	2	5	7	4
2048	2	12	21	11
2187	3	8	20	14
131072	2	18	33	17
177147	3	12	32	22

Bu yöntemin faydalarını şu şekilde özetlemek mümkündür:

1. Kullanıcı sayısı artarken kullanıcı saklama alanı maliyeti ve yeniden anahtar tahsisi mesaj sayısı maliyeti dengelidir ve ölçeklenebilir. Bu, diğer mimarilere göre hiyerarşik ağaç yaklaşımının bir üstünlüğüdür.
2. Yardımcı anahtarlar yalnızca anahtarların gönderilmesinde değil, mesajların yollanmasında da kullanılabilir. Dolayısı ile hiyerarşi, tüm grubun haricinde, birbirleri ile güvenli olarak iletişime geçmek isteyen alt gruplar halinde oluşturulabilir. Böylelikle, bu alt grupların, ayrıca bir güvenli grup oluşturarak grup anahtarı üretmeleri gerekli olmayacaktır. Bu durum, kullanıcıların da bir hiyerarşisinin bulunduğu ve bu hiyerarşinin anahtar hiyerarşisi ile ifade edilebileceği ortamlar için (örneğin; askeri uygulamalar) oldukça uygun olacaktır.
3. Bu yaklaşım, kullanıcıların gizlice anlaşmalarına karşı diğer yaklaşımlara oranla daha dayanıklıdır.
4. Hiyerarşi, ağ mimarisine verimlilik artışı sağlayacak şekilde belirlenebilir. Ayrıca, sunucunun sorumlulukları alt köklere paylaştırılabilir.

Daha önce bahsedildiği gibi, kullanıcılardan birinin çıkması durumunda, her bir anahtar için tek bir mesaj yollanıyor ise, bir sonraki anahtarın değiştirilmesinden önce her bir değiştirilen anahtarın başarılı olarak şifresinin çözülmüş olması gerekir (kullanıcının yeniden anahtar tahsisi mesajlarını önbellekleyemediği durumlarda). Bu darboğaz, gürültülü ya da yavaş bir ağ içinde problem oluşturabilir. Eğer birden fazla kullanıcı çıkarılıyor ise bu işlem paralel hale getirilebilir. Böylece, yeniden anahtar tahsisi için gereken süre, çıkarılan kullanıcı sayısından kabaca bağımsız hale getirilebilir.

Ağacın derinliğini azaltmak sureti ile, iletim sayısını arttırmak pahasına, kullanıcılarda gerekli olan saklama alanı gereksinimi azaltılabilir. Örneğin, mesaj başına tek anahtar yollandığı durumlarda, eğer n kullanıcı k-lı bir ağaç içerisine yerleştirilirse, her bir kullanıcı saklama alanına ihtiyaç duyacaktır. Çıkarılan bir kullanıcıdan sonra yeniden anahtar tahsisi için iletim gerekecektir. k, n'ye yaklaştığında bu yöntem daha önce anlatılmış olan çiftli anahtar yaklaşımı haline gelir.

5.2.5. GÜVENLİ KİLİT

Chiou ve Chen tarafından [14]'de tanımlanan güvenli kilit, oturum katılımcılarına anahtar tahsisi ve yeniden anahtar tahsisi için çoğa gönderim iletişiminin verimliliğini kullanmaktadır. Çinli Kalanlar Teoremi (Chinese Remainder Theorem-CRT) kullanılarak bir güvenli kilit oluşturulur ve grup oturum anahtarının şifresini çözmeyi "kilitlemek" için kullanılır. Şifrelenmiş her bir mesaj ile bu güvenli kilit yollanır. Sadece güvenli grup içerisindeki kullanıcılar oturum anahtarının kilidini çözebilir.

Güvenli kilitin arkasındaki prensip CRT'nin matematiğinde yatmaktadır. CRT şöyledir: N_1, N_2**,.....** ,N_n pozitif, ikişer ikişer birbirlerine asal tamsayılar ve R_1, R_2**,.....** ,R_n pozitif tamsayılar olmak üzere,

$X = R_1 \bmod N_1$

$X = R_2 \bmod N_2$

.....

$X = R_n \bmod N_n$

$L = N_1 * N_2 * .... * N_n$ ve n grup içerisindeki katılımcı sayısı

olmak üzere [0, L-1] aralığında X vardır ve tektir. [14] Chiou ve Chen, CRT özelliklerini kullanarak, $R_i = E_{eki}(d)$ ve R_i katılımcı u_i'nin kamusal şifreleme anahtarı *eki* (kamusal anahtar çiftinin parçası) kullanılarak E fonksiyonu tarafından şifrelenmiş, *d* oturum anahtarını göstermek üzere, X'i üretmişlerdir. Ortak kilit X, her bir katılımcının kamusal şifreleme anahtarı kullanılarak, başlatıcı tarafından üretilir. Her bir katılımcı, kilitlenmiş oturum anahtarı *d*'yi CRT'yi uygulayarak elde edebilir. Katılımcı, *d*'yi gizli şifre çözme anahtarı *dki*'yi kullanarak hesaplamaktadır. Sadece X'in hesaplanmasında şifreleme anahtarları (eki) kullanılan katılımcılar *d*'nin kilidini açabilirler.

Güvenli kilit metodu, grup katılımcılarının eklenmesi ve silinmesi açısından esnek bir yapı sunmaktadır. CRT kullanılarak, başlatıcı ortak X'i üretebilir ve yeni bir kullanıcıyı dahil etmek ya da var olan bir kullanıcıyı gelecekteki iletişimden çıkarmak için yeniden anahtar tahsisi yapabilir. *Eki*'leri X'in hesaplanmasında kullanılan katılımcılar, oturum

anahtarını yeniden elde edebilirler. X, tüm geçerli katılımcılar arasında ortak olduğu için, kilidin iletilmesinin verimliliği O(1)'dir. Her bir katılımcıdaki saklama alanı gereksinimi kamusal anahtar çifti ile sınırlıdır. Başlatıcının kilidi yaratması için, her bir katılımcının kamusal anahtarlarını saklaması gerekir.

CRT kullanılarak ortak X'in hesaplanma zamanı sınırlayıcıdır ve sadece küçük gruplar için verimli olabilir. Fakat, her bir katılımcı tarafından *d*'nin şifresinin çözülmesi oldukça verimlidir. Güvenli kilitin dağıtımının genel olarak verimliliği, büyük gruplar için gerekli işlem zamanı ile gölgelenir. Buna ek olarak, X'in hesaplanması dağıtılabilir olmadığından, yöntem doğal olarak merkezidir. Bu nedenle, güvenli kilit yöntemi X'in üretilmesi için dağıtık bir yöntem bulunmadığı sürece büyük gruplara ölçeklenebilir değildir. Çünkü X'in hesaplanması işlemi ile sadece başlatıcı ilgilendiğinden gerekli olan işlem zamanı oldukça fazla olacaktır.

5.2.6. DAĞITIK KAYIT VE ANAHTAR DAĞITMA (DiRK)

Opliger ve Albanies tarafından [15]'de tanıtılan *"Dağıtık Kayıt ve Anahtar Dağıtma (Distributed Registration and Key Distribution-DiRK)"*, .MBONE üzerindeki uygulamalar için tasarlanmış anahtar dağıtma protokolüdür. Bir çoğa gönderim oturumundaki aktif ve pasif katılımcıları ayırt eder. Tamamiyle dağıtık bir ortamda, herhangi bir aktif katılımcı, başlatıcıya kayıt ve anahtar dağıtma işlemleri için yardımcı olabilir.[5]

Başlangıç evresinde başlatıcı, bir oturum kamusal anahtar çifti (k_a, k_a^{-1}) üretir. Başlatıcı oturumu, kamusal anahtar k_a'nın imzalanmış bir kopyası ile duyurur. Özel anahtar k_a^{-1} sonraki mesajları imzalamak için başlatıcı tarafından kullanılır. Başlatıcı aynı zamanda oturum için grup anahtarı K_s'yi yaratır.

Oturum duyurusu alındıktan sonra, çoğa gönderim grubuna (grubun Sınıf D IP adresine) bir *kayıt istek mesajı* yollayarak, konaklar gruba katılmak istediklerini belirtebilirler. Bu istek mesajı, katılımcı tarafından bu oturum için yaratılmış kamusal anahtar çiftinin kamusal kısmını (yani; k_x) da içerir. K_s anahtarına sahip herhangi bir aktif katılımcı, geçerli kullanıcılara k_x ile şifrelenmiş K_s ile cevap verebilir. Zamanlayıcılar ve IP ömür (TTL) alanı *kayıt isteklerine* cevapların yalnızca lokal aktif katılımcılar ile sınırlı

kalması ve mesajların gruptan taşmasının engellenmesi için kullanılabilir. k_x ile şifrelenmiş grup anahtarınının şifresini sadece X kullanıcısı çözebilir. Cevap, geri dönen anahtarı doğrulayan kayıt istek mesajına cevap veren katılımcı tarafından imzalanmış bir sertifika da içerir. Geri dönen sertifikalar, başlatıcının aslını araştırıp bulmak için kullanılabilen bir katılımcı kayıt ağacı (Participant Registration Tree-PRT) oluşturur.

DiRK, o an geçerli olan grup üyeliği konusunda tüm katılımcıları güncel tutmak için bir *kayıt doğrulama mesajı* içerir. Katılımcılar, gruba periyodik olarak katılımcının kamusal oturum anahtarınının (yani; k_x) bir kopyasını ve oturuma kayıt olduklarında aldıkları sertifikayı içeren doğrulama mesajları yollarlar.

Birkaç yeniden anahtar tahsisi seçeneği DiRK tarafından desteklenmektedir. *Tüm oturuma yeniden anahtar tahsisi* başlatıcı tarafından, tüm katılımcılara yeni bir anahtar tahsisi için gerçekleştirilebilir. Bu yeniden anahtar tahsisi protokolü, bir önceki oturum anahtarı ile yeni anahtar şifreler. Anahtar güvenliğinin kaybolması durumunda, *seçici yeniden anahtar tahsisi*, başlatıcı tarafından sadece seçilen katılımcılara yeniden anahtar tahsisi için gerçekleştirilebilir. Başlatıcı, her bir geçerli katılımcı için yeni oturum anahtarını ayrı ayrı şifrelemek için, kayıt doğrulama mesajları ile alınmış kamusal oturum anahtarlarını kullanır. Ardından ayrı ayrı şifrelenmiş anahtarları içeren tek bir mesaj gruba yollanır. Aktif katılımcılara, gruba yeniden anahtar tahsisi izni veren *dağıtık oturum yeniden anahtar tahsisi* mesajı DiRK'e özgüdür. Başlangıç dağıtık oturum yeniden anahtar tahsisi mesajı katılımları yasaklanmış katılımcıların bir listesini içerir. Liste aktif katılımcılar tarafından yeni grup anahtarı potnsiyel katılımcılara yayımlanmadan önce incelenir.

DiRK genel mimarisi, dağıtık doğası nedeniyle, oldukça verimlidir. *n* katılımcıya anahtar tahsisi ve yeniden anahtar tahsisi için gerekli süreç (yani; O(n)) ağ boyunca tüm aktif katılımcılara dağıtılır. Seçici anahtar tahsisi fonksiyonu O(1) olduğu halde, yeniden anahtar tahsisi mesajının büyüklüğü n'dir.

DiRK dağıtık özellikleri nedeniyle son derece ölçeklenebilirdir. Ancak, aynı özellikler, güvenilir kayıt ve asıllama işlemlerini de tüm ağ boyunca dağıtır. Mesajların doğrulanması için, kullanıcıların, tüm yerel aktif katılımcıların sertifikalarını ellerinde bulundurmaları gerekir.

5.3. ANAHTAR TAHSİSİ MİMARİLERİNİN KARŞILAŞTIRILMASI

Yukarıda anlatılan her bir mimari, çoğa gönderim anahtar dağıtım problemini verimli bir şekilde çözmek için değişik çözümler sunmaktadır. Burada şu nokta unutulmamalıdır ki, bir uygulama için en iyi çözüm başka bir uygulama için uygun olmayabilir. Örneğin, çoğa gönderim video sunucusu gibi merkezi uygulamalar merkezi anahtar tahsisi mimarilerinden yararlanırken, dağıtık komut ve kontrol uygulamaları eşit şekilde dağıtılmış anahtar dağıtma yöntemlerinden yararlanabilir.[5]

Elle anahtar tahsisi metotları, oturumun başlamasından önce üyeliklerin belirlenmiş olmadığı dinamik çoğa gönderim oturumları için çok yavaş bulunmuştur. Ancak, çok iyi planlamış elle anahtar dağıtma mimarilerinin hali hazırda kullanıldığı bazı askeri uygulamalarda, bu çözüm gerçeklenmesi en kolay çözümdür.

N kök/yaprak çiftli anahtar tahsisi teknikleri başlangıç anahtar tahsisi ve yeniden anahtar tahsisi için doğrusal bir verimlilik sağlamaktadır. Tüm yeniden anahtar tahsisi mesajlarını tek bir mesaj içerisinde birleştirerek, yeniden anahtar tahsisinin verimliliği oldukça fazla bir şekilde iyileştirilebilir. Ancak, bu teknik yeniden anahtar tahsisi mesajının büyüklüğünü *n*'e çıkarmaktadır. Çiftli teknikler için anahtar saklama alanı gereksinimleri katılımcı tarafında en azdır; ancak, anahtar dağıtıcıda *n* adet anahtarın tutulmasını gerektirir. Bu metod, anahtar tahsisi ve kayıt fonksiyonları diğer güvenilir kaynaklara dağıtılabilir ise ölçeklenebilirdir.

Hiyerarşik Ağaç Yaklaşımı doğrusal başlangıç anahtar tahsisi başarımı ve iyileştirilmiş logaritmik yeniden anahtar tahsisi başarımı sağlar. Herhangi bir yeniden anahtar tahsisi mesajının büyüklüğü[2] *(k-1)*d'den fazla değildir. Her bir katılımcının anahtar saklama alanı gereksinimi *d+1* anahtardır. Bu arada, başlatıcı, tüm KEK ve GTEK'leri tutmalıdır. Logaritmik yeniden anahtar tahsisi başarımı nedeniyle bu çözüm ölçeklenebilirdir.

Tümleyici Değişken Yaklaşımı iletim sayısı kazancı sağlar. Bir kullanıcının ayrılması durumunda tüm gruba tek bir düz metin mesaj gönderilir. Ancak bu yöntemin iki önemli dezavantajı vardır. Birincisi, (N-1) tümleyici değişken için saklama alanı gereksinimi, ikincisi ise birden fazla kullanıcının ağdan çıkarılması durumunda bu kullanıcıların

[2] d derinliğinde k-lı ağacı için

tümleyici değişkenlerini birleştirip yeni grup anahtarını üretmeleri ve çoğa gönderim oturumuna ulaşabilmeleridir.

Güvenli kilit metodu doğrusal başlangıç anahtar tahsisi başarımına sahiptir ve etkileyici bir sabit yeniden anahtar tahsisi başarımı vardır. Yeniden anahtar tahsisi mesajının büyüklüğü sabittir ve incelenen tüm metodlardan daha iyi bir yeniden anahtar tahsisi başarımı sağlar. Bu metodun dezavantajları arasında, kilidin hesaplanması için geçen zaman ve tekniğin doğal olarak merkezi olması nedeniyle büyük gruplara ölçeklenebilir olmaması yer alır.

Tüm sistemin verimliliğini iyileştirmek için *DiRK,* doğrusal başlangıç anahtar tahsisi ve yeniden anahtar tahsisi fonksiyonlarını aktif grup üyeleri arasında paylaştırır. Ancak, kayıt ve anahtar dağıtma fonksiyonları dağıtıldığı için, güvenliğe ilişkin bir soru akla gelebilir. Aksi taktirde çözüm çok büyük ağlar için ölçeklenebilirdir.

Belirli bir uygulama için en iyi olan çözüm diğer bir uygulama için kullanışlı olmayabilir. Bu nedenle, güvenlik yönetimi seçilmeden önce, uygulamanın güvenlik yönetimi ve gereksinimlerini tamamıyla anlamak önem kazanmaktadır. Güvenlik çözümü, çoğa gönderim uygulamasının gerçeklenmesini yönlendirmek yerine tamamlayıcı olmalıdır. Güvenliğin uygulanması kullanıcıya saydam olmalıdır ve diğer gerekli protokoller ile verimli bir şekilde çalışmalıdır.

5.4. GRUP ANAHTAR YÖNETİMİ KONUSUNDA YAPILMIŞ ÇALIŞMALAR

5.4.1. GKMP

Grup Anahtar Yönetimi Protokolünde (Group Key Management Protokol-GKMP), her bir çoğa gönderim grubu için anahtarları yönetmek üzere bir grup kontrolörü (Group Controller-GC) atanır. Kontrolör, her bir üye için simetrik anahtarlar yaratır ve tutar. GC, grup üyelerinden birini, birlikte anahtar yaratmak için seçer. Ardından, her bir grup üyesinin haklarını doğrular ve özel anahtarları ile şifrelenmiş olarak grup anahtarını yollar. Bu teknik, her bir grup kendi anahtarlarını yönettiği için, tüm ağ açısından ölçeklenebilirdir. Bu işi yapmak için merkezi bir birimin olmasına gerek yoktur. Ancak, GC, grubun tüm anahtarlarından sorumlu olduğu için, grup büyüklüğü açısından ölçeklenebilir değildir.

5.4.2. SMKD

Ölçeklenebilir Çoğa Gönderim Anahtar Dağıtma (Scalable Multicast Key Distribution-SMKD), Çekirdek Tabanlı Ağaç Yönlendirmesi olarak adlandırılan belirli bir yönlendirme protokolü üzerine kurulmuştur. Yöntem, CBT grup ağacına güvenli olarak katılmayı sağlamak için çekirdek tabanlı ağaçların durum değişmezliklerinden yararlanır. Her bir yönlendirici, komşusunu bildiği ve konfigürasyon hiçbir zaman değişmediği için CBT değişmez durumludur. Ağacın çekirdeği grup kontrolörünün işini yapar ve grup oturum anahtarlarını ve anahtar dağıtma anahtarlarını üretir. Ağaç genişledikçe, yeni yönlendiriciler eklenir ve anahtar dağıtma işlemleri bu yönlendiricilere dağıtılır. Bu yönlendiricilere, yeni eklenen üyelerin asıllanması ve grup üyeliğinin verilmesi işlemlerini yapma yetkisi verilir. Bu ölçeklenebilir bir yönetim protokolü olmasını sağlar; ancak, tüm ara yönlendiricilerin güvenilir olmasını gerektirir.

5.4.3. IOLUS

Iolus, çoğa gönderim grubunu hiyerarşik alt gruplara bölen bir anahtar yönetim yapısıdır. Merkezi ya da en üst grup, güvenlik kontrolörü (Group Security Controller-GSC) tarafından yönetilir ve grup güvenlik arabirimleri (Group Security Intermediaries-GSIs) diğer alt grupları yönetmek için kullanılır. Her bir alt grup ayrı bir alt grup anahtarı kullanır. Her bir alt grubun GSI'sı komşu alt grupların anahtarlarını bilir. Mesaj göndermek için, üye kendi şahsi anahtarını kullanarak, mesajı yerel GSI'ya yollar. Ardından GSI, alt grubun anahtarını kullanarak mesajı alt gruba yollar. Eğer komşu gruplar varsa, mesajı, o komşu grubun alt anahtarını kullanarak şifreler ve komşu grubun GSI'sına yollar. Diğer alt gruplara yollamak için, mesajın gerçekten şifrelenmesi yerine, mesajın şifrelenmesi için, rastgele bir anahtar seçilir ve GSI, mesajı şifreleyen anahtarı şifreler ve komşu GSI'ya mesaj ile birlikte yollar. Bu, birden fazla GSI aracılığı ile iletilen mesajın işlem maliyetini azaltır.

5.4.4. ANAHTAR GRAFLARI

Anahtar grafları, grupların hiyerarşisi yerine anahtarların hiyerarşisinin kullanıldığı bir yöntemdir. Tüm anahtarları sağlayan tek bir sunucu vardır. Kullanıcılar alt gruplara bölünmüştür ve her bir kullanıcıya bir şahsi anahtar, bir alt grup anahtarı ve bir grup anahtarı verilir.

6. ANAHTAR GRAFLARI KULLANILARAK GÜVENLİ ÇOĞA GÖNDERİM

Pek çok ağ uygulaması istemci-sunucu modeli üzerine kurulmuştur ve teke gönderim paket iletişimini (noktadan-noktaya) kullanır. Ancak günümüzde, teknolojinin gelişmesi ile birlikte ortaya çıkan pek çok uygulama grup iletişimine dayanmaktadır. Internet'te, çoğa gönderim, çok büyük gruplara verimli olarak en iyi şekilde hizmet verilmesi için kullanılmaktadır. Gelecekte, grup iletişimini gerektiren ağ uygulamalarının geliştirilmesinin hız kazanacağı tahmin edilmektedir. Sonuç olarak, grup iletişiminin güvenli hale getirilmesinin, yakın gelecekte, kritik iletişim konusu olması beklenmektedir. [15]

İstemci-sunucu arasındaki teke gönderim iletişimini güvenli hale getirmek sırasında ele alınması gereken konular oldukça iyi anlaşılır olmasına rağmen, grup iletişimini güvenli hale getirmek için ele alınması gereken konular henüz açıklık kazanmamıştır. Kavramsal olarak, her bir noktadan-çok noktaya iletişim bir noktadan-noktaya iletişim kümesi olarak ifade edilebileceğinden, teke gönderim iletişimini güvenli hale getirmek için kullanılan halihazırdaki teknolojiler, kolaylıkla grup iletişimini güvenli hale getirmek için kullanılabilir. Ancak, böyle bir çözümün büyük gruplar için ölçeklenebilirliği söz konusu olamaz.

Bir örnek ile yukarıda ifade ettiklerimizi açıklayalım: İstemci-sunucu arasındaki teke gönderim iletişimini güvenli hale getirmek için gerekli basamakları şu şekilde ifade edebiliriz. İlk olarak, istemci ve sunucu bir asıllama protokolü ya da hizmeti aracılığı ile karşılıklı olarak birbirlerini asıllarlar. Ardından simetrik bir anahtar yaratılır ve karşılıklı olarak sırdaş iletişimde kullanulmak üzere iki taraf arasında paylaşılır. Bu işlem, bir gruba şu şekilde genişletilebilir. Grup erişim kontrolünü yapmak üzere, grup üyelik bilgilerinin paylaşıldığı, güvenilir bir sunucu olduğunu düşünelim. İstemci gruba katılmak istediğinde, sunucu ve istemci karşılıklı olarak bir asıllama protokolü aracılığı ile birbirlerini asıllarlar. Asıllandıktan ve gruba katılımları onaylandıktan sonra, her bir üye sunucu ile, ilgili üyenin *şahsi anahtarı* olarak adlandırılacak bir anahtar paylaşır.

Grup iletişimi için ise, sunucu, her bir grup üyesine diğer grup üyeleri tarafından paylaşılan bir grup anahtarı dağıtır.

n üyeli bir grup için, tüm üyelere bu grup anahtarının güvenli olarak iletilmesi, ilgili üyenin şahsi anahtarı ile şifrelenmiş *n* adet mesajın yollanmasını gerektirir. Bu mesajlardan her biri teke gönderim aracılığı ile yollanabilir. Alternatif olarak, bu *n* adet mesaj birleştirilerek, tek bir mesaj olarak tüm gruba çoğa gönderilebilir. Her iki şekilde de, grup büyüklüğü ile orantılı bir iletişim maliyeti (mesaj sayısı ya da birleştirilmiş mesajının büyüklüğü cinsinden ölçüldüğünde) vardır.

Noktadan-noktaya oturumda, oturumun kurulması ve anahtar dağıtımı için gerekli maliyetlere sadece bir kez maruz kalınır. Bu da, oturumun başındadır. Diğer taraftan bir grup oturumu, ayrılan ve yeni katılan üyelerin varlığı nedeniyle daha uzun süre devam edebilir. Sonuç olarak, grup anahtarı sıklıkla değiştirilmelidir. Yüksek seviyede güvenlik sağlamak için, her bir katılma ve ayrılma sonrasında, yeni katılan üyenin önceki iletişime erişimini, ayrılan üyenin de ayrılma zamanından sonraki iletişime erişimini engellemek için grup anahtarının değiştirilmesi gerekir. [15]

Güvenilir sunucunun, her bir ayrılma ve katılma sonrasında yeni bir grup anahtarı yarattığını kabul edelim. Bir katılma sonrasında, yeni grup anahtarı yeni katılan grup üyesine teke gönderim olarak (üyenin kendi şahsi anahtarı ile şifrelenerek) ve mevcut üyelere de çoğa gönderim olarak (eski grup anahtarı ile şifrelenerek) yollanabilir. Dolayısıyla, katılım sonrasında, grup anahtarını değiştirmek için yapılacak işlem sayısı çok azdır. Ancak, bir ayrılma sonrasında, eski grup anahtarının artık kullanılmaması ve yeni grup anahtarının her bir üyenin şahsi anahtarı ile şifrelenerek yollanması gerekecektir. Dolayısıyla, bir ayrılma sonrasında, grup anahtarının güvenli olarak değiştirilmesi, *i* ile orantılı bir hesaplama ve iletişim maliyeti gerektirecektir. Bu durum, çok sık olarak ayrılma ve katılmaların yaşandığı büyük gruplarda ölçeklenebilirlik problemlerine sebep olabilir. Bu çalışmada anlatılan yaklaşımlarda ölçeklenebilirlik iyileştirilmeye çalışılmaktadır. Üyelerin hiyerarşisi yerine anahtarların hiyerarşisi kullanılmaktadır.

Burada anahtar olarak simetrik kripto sistemleri ile üretilen anahtarlar kasdedilmektedir. Bu çalışma boyunca ele alınan anahtarlar simetrik anahtarlar olacaktır. Grup anahtarı,

grup içerisinde gizli iletişimin yapılmasına imkan sağlar. Buna ek olarak, dijital imza, mesaj özü gibi teknikler kullanılarak asıllama, veri bütünlüğü ve reddedilmenin önlenmesi sağlanabilir. Ancak bu konular bu çalışmanın kapsamı içerisinde yer almayacaktır.

6.1. GÜVENLİ GRUPLAR

Bir *güvenli grup (U, K, R)* üçlüsü şeklinde ifade edildiğinde:

- *U* sonlu ve boş olmayan kullanıcılar kümesi,
- *K* sonlu ve boş olmayan anahtarlar kümesi,
- *R, U* ve *K* arasındaki kullanıcı-anahtar ikili ilişkisi ve $R \subset U \times K$'dır. Başka bir deyişle kullanıcı *u*, anahtar *k*'ya yalnız ve yalnız *(u, k)* ilişkisi *R* kümesi içerisinde yer alıyor ise sahiptir.

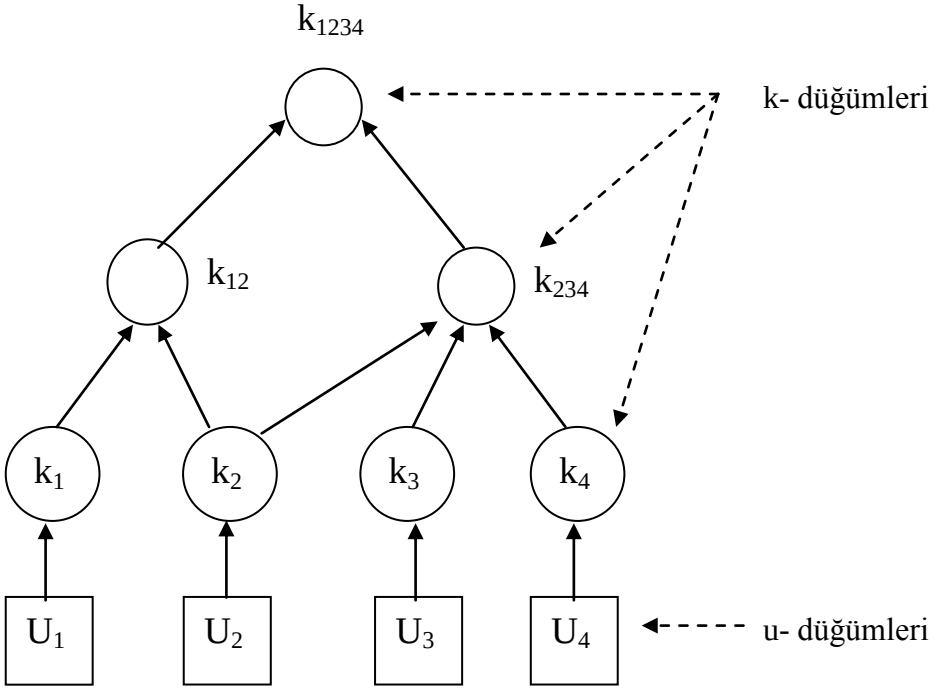
Her bir güvenli grup, *K* kümesi içerisindeki anahtarların üretilmesi ve güvenli olarak dağıtılmasından sorumlu, güvenilir bir *anahtar sunucusu'*na sahiptir. Yani anahtar sunucusu, *U* kullanıcı kümesini ve *K* anahtar kümesini bilir ve kullanıcı-anahtar ilişkisi *R*'yi sağlar. *U* kümesi içerisindeki her bir kullanıcı, *K* kümesi içerisinde bir anahtara sahiptir. Bu anahtar *şahsi anahtar* olarak adlandırılır, yalnızca anahtar sunucusu ile paylaşılır ve anahtar sunucusu ile gizli olarak haberleşilmesini sağlar. *K* kümesi içerisinde anahtar sunucusu ve *U* içerisindeki tüm kullanıcılar tarafından paylaşılan bir *grup anahtarı* yer alır. Grup anahtarı, her bir kullanıcı tarafından grup içerisindeki diğer üyelere gizli olarak mesaj yollanmasını sağlar.[16]

6.1.1. ANAHTAR GRAFLARI

Bir anahtar grafı, biri kullanıcıları gösteren *u-düğümleri* ve diğeri anahtarları gösteren *k-düğümleri* olmak üzere iki tip düğüm ile, çevrim içermeyen, yönlü bir *G* grafıdır. *u-düğümleri'*nin bir ya da birden fazla hiç çıkan kenarı vardır; fakat giren kenarı yoktur. *k-düğümleri'*nin bir ya da birden fazla giren ve çıkan kenarı vardır. Giren kenarı olup çıkan kenarı olmayan *k-düğümü'*ne *kök* adı verilir. Bir anahtar grafında birden fazla kök bulunabilir.

Bir *G* grafı ele alındığında, bu graf, aşağıdaki şekilde bir *(U, K, R)* güvenli grubunu ifade eder:

1. *U* kümesi ile *G*'deki *u-düğümleri* kümesi arasında bire-bir bir uyuşma vardır.
2. *K* kümesi ile *G*'deki *k-düğümleri* kümesi arasında bire-bir bir uyuşma vardır.
3. *(u, k) R* kümesi içerisinde yer alır; ancak ve ancak *G*'de *u'*ya karşı düşen *u-düğümü* ile *k*'ya karşı düşen *k-düğümü* arasında yönlü bir yol olmalıdır.



Şekil 6.1 Anahtar Grafı

Anahtar grafından bir u-düğümünün doğrudan bağlı olduğu k-düğümü, bu u-düğümüne karşı düşen kullanıcının şahsi anahtarına karşı düşen k-düğümüdür. Dolayısı ile bu k-düğümüne başka u-düğümlerinden gelen kenar olamaz. k-düğümleri arasındaki kenarlar özel bir anlam taşımamakla birlikte bir R kümesi içerisinde yer alan ilişkilerin graf üzerinde gösterilmesi sırasında kullanıcıyı ifade eden u-düğümünden anahtarı ifade eden k-düğümüne yönlü yolun gösterilmesi için kullanılır.

Şekil 6.1'deki anahtar grafını örnek olarak alalım. Bu şekilde verilen anahtar grafı aşağıdaki güvenli grubu tanımlamaktadır:

U = $\{u_1, u_2, u_3, u_4\}$,

$K = \{k_1, k_2, k_3, k_4, k_{12}, k_{234}, k_{1234}\}$,

$R = \{(u_1, k_1), (u_1, k_{12}), (u_1, k_{1234}), (u_2, k_2), (u_2, k_{12}), (u_2, k_{234}), (u_2, k_{1234}),$

$(u_3, k_3), (u_3, k_{234}), (u_3, k_{1234}), (u_4, k_4), (u_4, k_{234}), (u_4, k_{1234})\}$

Her bir güvenli grup *(U, K, R)* ile ilişkili olarak aşağıdaki iki fonksiyon tanımlanabilir:

$$\text{keyset}(u) = \{ k \mid (u, k) \in R \}$$

$$\text{userset}(k) = \{ u \mid (u, k) \in R \}$$

keyset(u), U kümesi içerisinde yer alan bir u kullanıcısının elinde bulundurduğu anahtarlar kümesi, userset(k) ise K kümesi içerisinde yer alan bir k anahtarına sahip olan kullanıcılar kümesidir. Şekil 6.1'de yer alan anahtar grafından örnek verecek olursak userset(k_{234}) = { u_2 ,u_3 , u_4 } ve keyset(u_4) = { k_4 , k_{234} , k_{1234} } olacaktır.

Kullanıcı u güvenli grup *(U, K, R)*'den ayrılmak istediğinde, u tarafından tutulan ve U kümesi içerisindeki diğer kullanıcılar ile paylaşılan tüm anahtarların değiştirilmesi gerekir. K'nın değiştirilecek anahtarlardan birisi k olsun. k'nın değiştirilmesi için, sunucu rastgele bir k_{yeni} anahtarı yaratır ve u haricinde userset(k) içerisinde yer alan tüm kullanıcılara yollar. Bunu çok güvenli olarak yapabilmek için, sunucunun userset(K') = userset(k) – {u} olan bir K' anahtar alt kümesi bulması ve k_{yeni} anahtarını şifreleyerek yollaması gerekir. Yeniden anahtar tahsisi işini azaltmak için, sunucunun en az sayıda elemana sahip K' kümesini bulması gerekir. Bu durum, verilen bir (U, K, R) güvenli grubu ve U'nun altkümesi S için, userset(K') = S olan K'nın en az elemanlı K' alt kümesinin bulunması problemini ortaya çıkarır. Bu problem *anahtar-kapsama problemi* olarak adlandırılır. Anahtar-kapsama problemi genelde NP-zorluk dereceli bir problemdir.[16]

6.1.2. ANAHTAR GRAFLARININ ÖZEL TİPLERİ

Bu bölümde, anahtar-kapsama probleminin daha kolay çözülebilmesi için kullanılabilecek özel anahtar grafı tipleri anlatılmaktadır. Buna göre üç çeşit anahtar grafı vardır:

Yıldız: *U* içerisindeki her bir kullanıcının, birisi kendi şahsi anahtarı, diğeri ise *U* içerisindeki her bir kullanıcı tarafından paylaşılan *grup anahtarı* olan iki anahtara sahip olduğu, (*U, K, R*) güvenli grubunun özel bir çeşididir. Bu grup, anahtar yönetiminin ölçeklenebilirliğini iyileştirmek için ek anahtarların olmadığı temel durumdur.

Ağaç: G anahtar grafının tek köklü bir ağaç olduğu bir (*U, K, R*) güvenli grubunun özel bir çeşididir. Bir anahtar ağaç grafı ya da anahtar ağacı iki parametre ile tanımlanır:

- *Yükseklik (h)*: Ağaçtaki en uzun yönlü yolun kenar sayısı cinsinden uzunluğudur.
- *Derece (d)*: Ağaçtaki bir düğümün maksimum giren kenar sayısıdır.

Bu tanımlar göz önüne alındığında, her bir yolun yaprak düğümü bir *u-düğümü* olduğundan, *U* içerisindeki her bir kullanıcı, en fazla *h* adet anahtara sahip olabilir. Aynı zamanda, kökte yer alan anahtar, *U* içerisindeki tüm kullanıcılar tarafından paylaşılır ve *grup anahtarı* olarak kullanılır. Bu şekilde bakıldığında, yıldız anahtar graflarının ağaç anahtar graflarının özel bir hali olduğu söylenebilir.

Tam: U'nun her bir boş olmayan *S* alt kümesi için, *K* kümesi içerisinde userset(*k*) = *S* olan bir *k* anahtarının bulunduğu bir (*U, K, R*) güvenli grubunun özel bir çeşididir. *n, U* kümesi içerisindeki kullanıcıların sayısı olsun. *K* içerisinde 2^n-1 anahtar yer alır. Bu anahtarlardan her biri *U*'nun boş olmayan 2^n-1 alt kümesi içindir. *U* kümesi içerisindeki bir *u* kullanıcısının 2^{n-1} adet anahtarı vardır ve bu anahtarlardan her biri *u*'yu içeren *U*'nun her bir 2^{n-1} alt kümesi içindir. *U, U* kümesinin bir alt kümesi olduğundan, *U* kümesi içerisinde yer alan tüm kullanıcılar tarafından paylaşılan bir anahtar vardır ve bu anahtar *grup anahtarı* görevini görür. [16]

Tablo 6.1 Sunucu ve her bir kullanıcı tarafından tutulan anahtar sayıları

	Yıldız	**Ağaç**	**Tam**

Toplam anahtar sayısı	$n + 1$	$\frac{d}{d-1}n$	2^n - 1
Her bir kullanıcının anahtar sayısı	2	h	2^{n-1}

n, *U* kümesinin eleman sayısını göstermek üzere, Tablo 6.1'de, sunucu ve her bir kullanıcı tarafından tutulan anahtar sayıları verilmiştir. Burada anahtar ağacındaki anahtarların sayısı, ağaç dolu ve dengeli ise (yani $n = d^{h-1}$) $\left(d^h - 1\right) / \ (d-1) \approx (d / \ (d-1))n$ olacaktır.

6.2. YENİDEN ANAHTAR TAHSİSİ STRATEJİLERİ VE PROTOKOLLERİ

Güvenli gruba katılmak (ayrılmak) isteyen bir *u* kullanıcısı, bu isteğini, *s* ile ifade edilen anahtar sunucusuna yollar. Kullanıcı *u*'dan gelen katılma isteği için, güvenli grubun başlatıcısı tarafından sağlanan erişim kontrol listesi ile *s* sunucusu tarafından grup erişim kontrolünün yapıldığını kabul edelim. Bir katılma isteği, *u* ile *s* arasında bir asıllama yapılmasını gerektirecektir. Eğer *u* kullanıcısı gruba katılmak için yetkilendirilmez ise, *s* sunucusu *u* kullanıcısına bir katılma-yasaklama cevabı yollar. Eğer katılma isteği kabul edilirse, asıllama işlemleri sırasında dağıtılan oturum anahtarının u kullanıcısı tarafından şahsi anahtar olarak kullanılacak *ku* anahtarı olduğunu kabul edelim. Protokol tanımlarını basitleştirmek için aşağıdaki notasyon kullanılacaktır:

$s \Leftrightarrow u$: *u*'yu asılla ve k_u'yu yolla

Bu gösterilim, *s* sunucusu ile *u* kullanıcısı arasında yapılan asıllama işlemlerini ve *u* ve *s* arasında paylaşılan k_u anahtarının güvenli olarak dağıtılmasını ifade etmektedir.

Her bir katılma ve ayrılmadan sonra, yeni bir güvenli grup oluşturulur. *s* sunucusu, grubun anahtar grafını; var olan bazı *k-düğümleri*ndeki anahtarları değiştirerek, bazı *k-düğümleri*ni (ayrılma durumunda) silerek ve bazı yeni *k-düğümleri* (katılma durumunda) ekleyerek güncellemek durumundadır. Ardından *s* sunucusu, yeni güvenli grubun yeni

grup/alt grup anahtarlarını içeren *yeniden anahtar tahsisi* mesajlarını güvenli olarak yollayacaktır. [16]

$$x \rightarrow y : z$$

İleriki bölümlerde, yukarıda verilen notasyon, protokol tanımları yapılırken kullanılacaktır. Bu notasyon:

- Eğer *y* tek bir kullanıcı ise, *z* mesajının *x*'den *y'*ye yollanmasını;
- Eğer *y* bir kullanıcı kümesi ise, *z* mesajının *x*'den *y* içerisindeki her bir kullanıcıya çoğa gönderim ya da teke gönderim aracılığı ile yollanmasını ifade edecektir.

$\{x\}_k$ notasyonu ise x mesajının k anahtarı ile şifrelendiğini ifade edecektir. $\{x, y\}_k$ ise, x ve y mesajlarının beraberce k anahtarı tarafından şifrelendiğini ifade edecektir.

6.2.1. YILDIZ ANAHTAR GRAFINA KATILMA

Kullanıcı *u'*dan gelen katılma isteği onaylandıktan sonra, *s* sunucusu *u* için yeni bir *u-düğümü* ve k_u için yeni bir *k-düğümü* yaratıp köke bağlayarak anahtar grafını günceller. *s* sunucusu, aynı zamanda, kök düğümü için yeni bir $k_{U'}$ grup anahtarı yaratır; kullanıcı *u*'nun şahsi anahtarı k_u ile şifreler ve şifrelenen yeni grup anahtarını *u* kullanıcısına yollar. Diğer kullanıcıların yeni grup anahtarından haberdar olmaları için, *s* sunucusu yeni grup anahtarı $k_{U'}$ yü eski grup anahtarı k_U ile şifreler ve şifrelenmiş olan yeni grup anahtarını, çoğa gönderim ile, grup içerisindeki tüm kullanıcılara yollar.[16]

Bu tip anahtar grafına katılma işlemleri için gerekli protokol adımları aşağıdaki şekildedir:

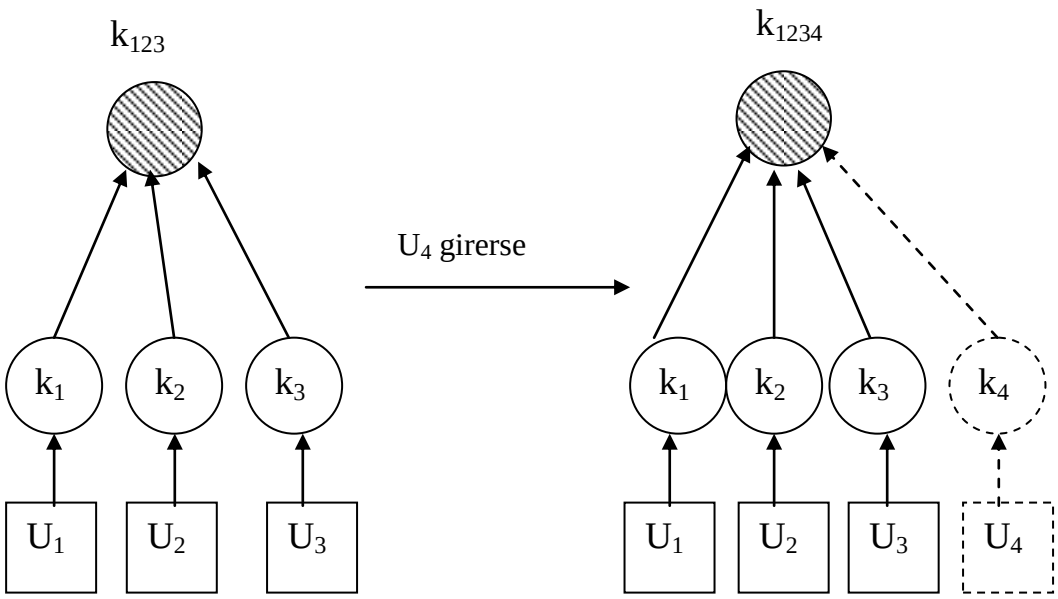
(1) $u \rightarrow s$: katılma isteği

(2) $s \Leftrightarrow u$: *u*'yu asılla ve k_u'yu yolla

(3) s : rastgele olarak yeni bir grup anahtarı $k_{U'}$ yü yarat

(4) $s \rightarrow u$: $\{k_{U'}\}k_u$

(5) $s \rightarrow U : \{ k_{U'} \} k_U$



Şekil 6.2 Katılma öncesi ve sonrasında yıldız anahtar grafı

Şekil 6.2 ele alındığında, kullanıcı u_4 ün, şeklin solunda yer alan güvenli gruba katılmak istediğini düşünelim ve bu isteği kabul edilsin. *s* sunucusu grup anahtarını k_{123}'den k_{1234}'e değiştirdikten sonra aşağıdaki iki yeniden anahtar tahsisi mesajını yollaması gerekir:

$$s \rightarrow \{u_1, u_2, u_3\} \quad : \{k_{1234}\} k_{123}$$

$$s \rightarrow u_4 \quad : \{k_{1234}\} k_4$$

6.2.2. YILDIZ ANAHTAR GRAFINDAN AYRILMA

Kullanıcı *u*'dan gelen ayrılma isteği onaylandıktan sonra, sunucu *s* kullanıcı *u* için yaratılmış *u-düğümü'*nü ve o kullanıcıya ait şahsi anahtar k_u için yaratılmış *k-düğümü*'nü anahtar grafından silerek grafı günceller. Sunucu *s, u* haricindeki kullanıcılardan oluşan yeni güvenli grup için yeni bir $k_{U'}$ grup anahtarı yaratır; geride kalan tüm kullanıcıların şahsi anahtarları ile şifreler ve kullanıcılara yeni grup anahtarını teke gönderim mesajları ile yollar.[16]

Bu tip anahtar grafından ayrılma işlemleri için gerekli protokol adımları aşağıdaki şekildedir:

(1) $u \rightarrow s$: {ayrılma isteği}k_u

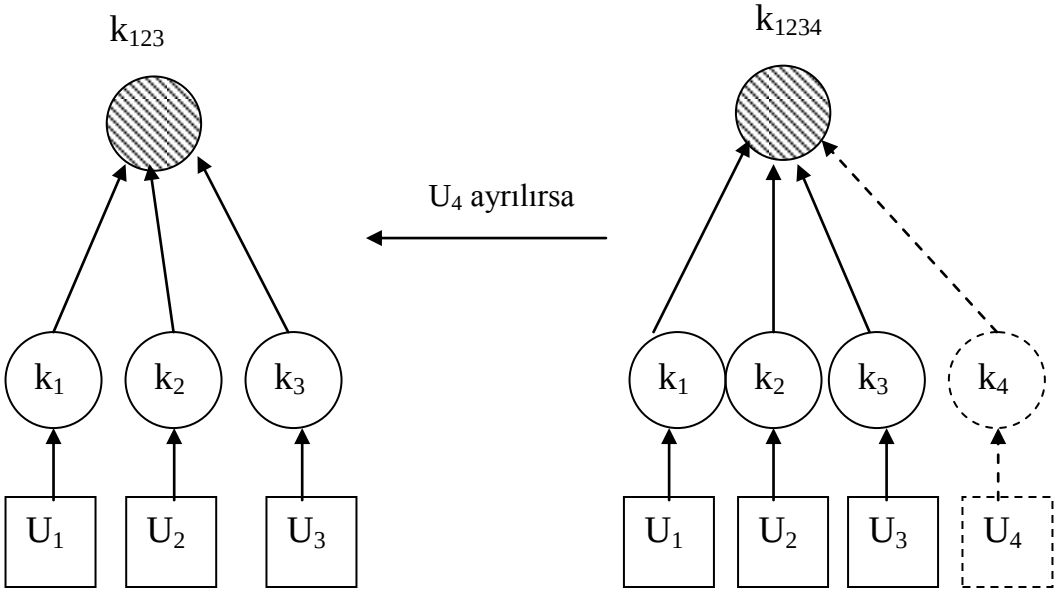
(2) $s \rightarrow u$: {ayrılma isteği onaylandı}k_u

(3) s : rastgele olarak yeni bir grup anahtarı $k_{U'}$ yü yarat

(4) for each kullanıcı $v(v \neq u)$ in U

$s \rightarrow v$: { $k_{U'}$}k_v

Şekil 6.3 ele alındığında, kullanıcı u_4 ün, şeklin sağında yer alan güvenli gruptan ayrılmak istediğini düşünelim ve bu isteği kabul edilsin. *s* sunucusunun grup anahtarını



Şekil 6.3 Ayrılma öncesi ve sonrasında yıldız anahtar grafı

k_{1234}'den k_{123}'e değiştirdikten sonra aşağıdaki yeniden anahtar tahsisi mesajlarını yollaması gerekir:

$s \rightarrow \{u_1\}$: $\{k_{123}\}k_1$

$s \rightarrow \{u_2\}$: $\{k_{123}\}k_2$

$s \rightarrow \{u_3\}$: $\{k_{123}\}k_3$

6.2.3. AĞAÇ ANAHTAR GRAFINA KATILMA

Kullanıcı *u'*dan gelen katılma isteği onaylandıktan sonra, *s* sunucusu, *u* için yeni bir *u-düğümü* ve k_u için yeni bir *k-düğümü* yaratır. *s* sunucusu anahtar ağacında yer alan bir *k-düğümü'*nü bulur ve k_u için yaratılmış yeni *k-düğümü'*nü bu düğümün çocuğu olacak şekilde bağlar. Bu noktaya *katılma noktası* adı verilir. Her bir katılma işleminde farklı bir katılma noktası belirlenebilir.

Yeni katılan kullanıcının geçmişte yapılan iletişime erişimini engellemek için, katılma noktasından köke kadar uzanan yol üzerindeki tüm anahtarların değiştirilmesi gerekir. Bu düğümler için yeni anahtar yaratıldıktan sonra, *s* sunucusunun bu anahtarları, hem daha önce grup içerisinde yer alan kullanıcılara hem de yeni katılan kullanıcıya güvenli olarak dağıtması gerekir. Bunu bir örnek üzerinde açıklayalım. Şekil 6.4'te verilen anahtar ağacını ele alalım. Kullanıcı u_9 un, şeklin üst kısmında yer alan güvenli gruba katılmak istediğini düşünelim. Bu durumda, üst kısımdaki anahtar grafında katılma noktası k_{78} *k-düğümü* olacaktır. Bu düğümde yer alan anahtar, alt kısımdaki anahtar grafından da görülebileceği gibi, k_{789} olarak değiştirilecektir. Kökte yer alan grup anahtarı ise $k_{1\text{-}8}$ den $k_{1\text{-}9}$ a değiştirilecektir. Kullanıcılar u_1,...., u_6 yalnızca yeni grup anahtarı $k_{1\text{-}9}$ a ihtiyaç duyarken, kullanıcı u_7, u_8 ve u_9 hem yeni alt grup anahtarı k_{789}'a hem de yeni grup anahtarı $k_{1\text{-}9}$ a ihtiyaç duyacaklardır. [16]

Yeni anahtarları kullanıcılara güvenli olarak dağıtmak için, sunucu yeniden anahtar tahsisi mesajlarını oluşturur ve kullanıcılara yollar. Bir *yeniden anahtar tahsisi mesajı,* bir ya da birden fazla şifrelenmiş anahtar içerir ve kullanıcıların yeni anahtarları elde edebilmeleri için bu şifreli mesajı uygun anahtar ile çözmeleri gerekir. Bu yeniden anahtar tahsisi mesajlarının oluşturulması ve kullanıcılara yollanması için üç farklı yaklaşım bulunmaktadır. Bundan sonraki bölümde bu yaklaşımlar ele alınacaktır.

6.2.3.1. KULLANICI-ODAKLI YENİDEN ANAHTAR TAHSİSİ

Kullanıcı-Odaklı Yeniden Anahtar Tahsisi yaklaşımının temeli, her bir kullanıcı için, sunucu tarafından, o kullanıcı tarafından ihtiyaç duyulan tüm anahtarları içeren bir yeniden anahtar tahsisi mesajının oluşturulması ve bu mesajın o kullanıcının şahsi anahtarı tarafından şifrelenmesine dayanır.

Kullanıcı-Odaklı Yeniden Anahtar Tahsisi mesajları şu şekilde oluşturulur. Anahtarı *k* yerine *k′* olarak değiştirilecek olan her bir *x k-düğümü* için, anahtar sunucusu bu *x k-düğümü'*nün yeni anahtarını ve bu düğümden köke kadar olan tüm ebeveyn düğümlerde yer alan yeni anahtarları, bu *x k-düğümü'*nün yeni anahtarı ile şifreleyerek bir yeniden anahtar tahsisi mesajı oluşturur. Ardından, bu yeniden anahtar tahsisi mesajı, bu mesaj içerisinde yer alan yeni anahtarların tamamına ihtiyaç duyan kullanıcı alt kümesine yollanır. Gönderme işlemi için, teke gönderim ya da alt grup çoğa gönderimi kullanılabilir. Bunun yanı sıra, bir yeniden anahtar tahsisi mesajı da yeni katılan kullanıcıya yollanır. Bu mesaj, tüm yeni anahtarları içeririr ve katılan kullanıcının şahsi anahtarı kullanılarak şifrelenir.[16]

Şekil 6.4'te verilen anahtar ağaç grafını ele alalım. Kullanıcı u_9 un şeklin üst kısmında yer alan güvenli gruba katılması için, *s* sunucusunun aşağıdaki üç yeniden anahtar tahsisi mesajını oluşturması ve yollaması gerekir.

$$s \rightarrow \{u_1, ..., u_6\} \quad : \{k_{1\text{-}9}\}k_{1\text{-}8}$$

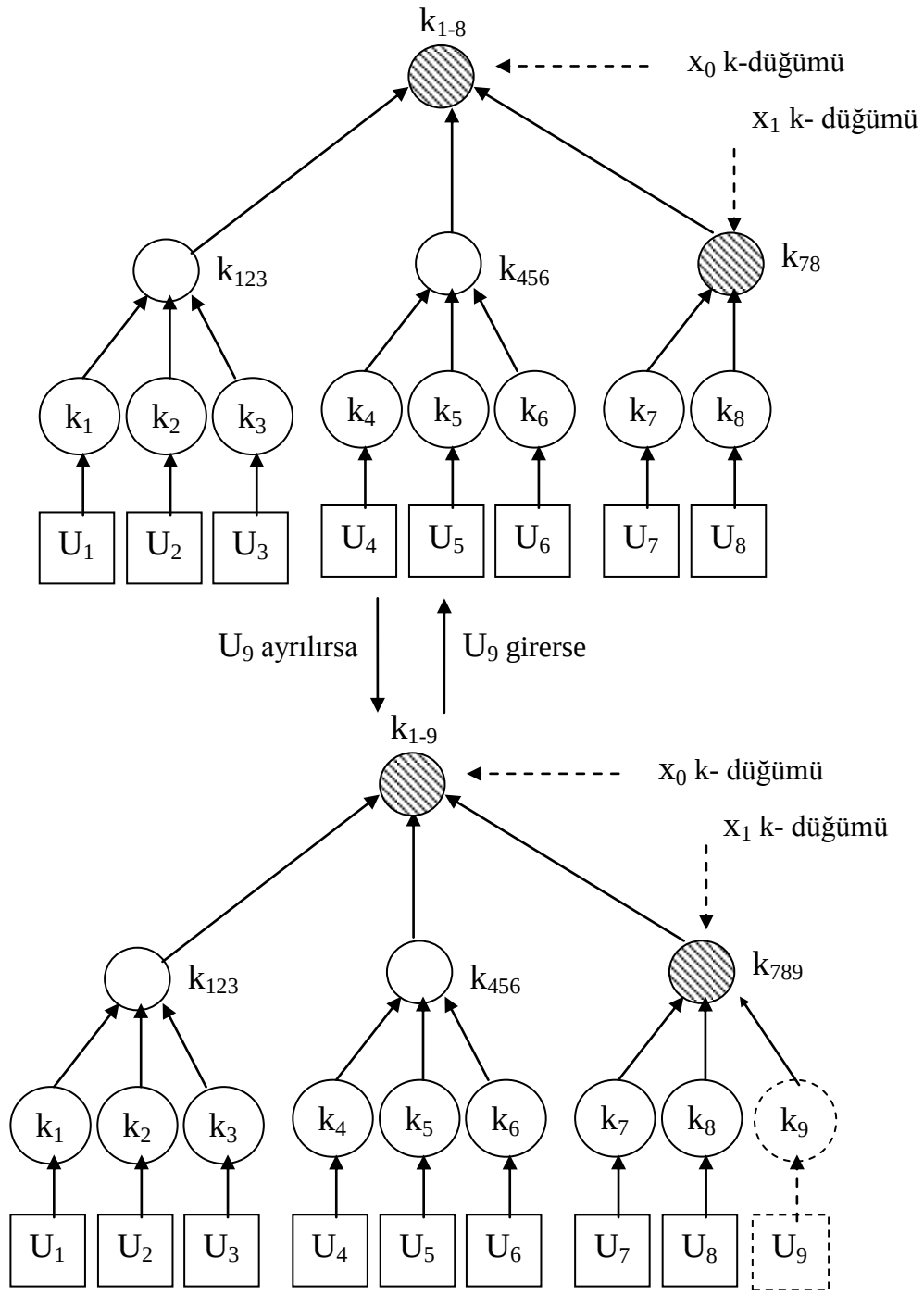
$$s \rightarrow \{u_7, u_8\} \quad : \{k_{1\text{-}9}, k_{789}\}k_{78}$$

$$s \rightarrow \{u_9\} \quad : \{k_{1\text{-}9}, k_{789}\}k_9$$

Bu yaklaşım, *h* adet yeniden anahtar tahsisi mesajı gerektirir. Burada *h*, anahtar ağaç grafının yüksekliğidir. Şifreleme maliyeti olarak şifrelenen anahtar sayısı ele alınacak olursa, sunucu için şifreleme maliyeti;

$$1 + 2 + ... + h - 1 + h - 1 = \frac{h(h+1)}{2} - 1$$

olacaktır.



Şekil 6.4 Katılma (ayrılma) öncesi ve sonrası bir anahtar ağacı

Yukarıdaki örneğe bakacak olursak, burada anahtar ağacının yüksekliğinin 3 olduğunu görürüz. Bu durumda, yeniden anahtar tahsisi mesaj sayısı, $h = 3$ olduğundan, 3 olacaktır. Şifreleme maliyeti ise;

$$1+2+3-1+3-1=\frac{3(3+1)}{2}-1=5$$

olacaktır. Yukarıda verilen mesaj sayısı ve mesajlar içerisinde şifrelenen anahtar sayısı, burada elde ettiğimiz sonuçların doğruluğunu göstermektedir.

6.2.3.2. ANAHTAR-ODAKLI YENİDEN ANAHTAR TAHSİSİ

Bu yaklaşımda, her bir yeni anahtar ayrı ayrı şifrelenir. Ancak yeni katılan kullanıcıya yollanan anahtarlar için bu işlem uygulanmaz. Anahtarı k yerine k' olarak değiştirilecek olan her bir x *k-düğümü* için, anahtar sunucusu iki yeniden anahtar tahsisi mesajı oluşturur. İlk olarak sunucu, yeni k' anahtarını k anahtarı ile şifreler ve userset(k) kullanıcı alt kümesine yollar. Yeni k' anahtarına ihtiyaç duyan tüm eski kullanıcılar, bu yeniden anahtar tahsisi mesajından yeni anahtarı elde edebilirler. Diğer yeniden anahtar tahsisi mesajı ise, katılan kullanıcının şahsi anahtarı ile şifrelenen k' yeni anahtarını içerir ve yeni katılan kullanıcıya yollanır. [16]

Bu yaklaşım kullanılarak, anahtar ağaç grafına katılma işlemleri için gerekli protokol adımları aşağıdaki şekildedir:

(1) $u \rightarrow s$: katılma isteği

(2) $s \Leftrightarrow u$: u'yu asılla ve k_u'yu yolla

(3) s : bir katılma noktası bul ve k_u yu ekle;

x_j katılma noktasını göstermek ve x_0 kök ve i = 1,..... , j olmak üzere x_{i-1} x_i nin ebeveyni olsun.

K_{j+1} k_u yu göstermek ve K_0 ,..... ,K_j x_0 ,.... ,x_j nin eski anahtarları olmak üzere, rastgele olarak K_0' , ..., K_j' yeni anahtarları yarat.

(4) for i = 0 upto j do

$s \rightarrow$ (userset(K_i)) : { K_i' }K_i

$s \rightarrow u$: { K_i' }k_u

Şekil 6.4'te verilen anahtar ağaç grafını ele alalım. Kullanıcı u_9 un şeklin üst kısmında yer alan güvenli gruba katılması için, s sunucusunun, aşağıdaki dört yeniden anahtar tahsisi mesajını oluşturması ve yollaması gerekir. Burada kullanıcı u_7 , u_8 ve u_9 un her birinin iki adet yeniden anahtar tahsisi mesajı aldığı göz önünde bulundurulmalıdır.

$s \rightarrow \{u_1 , ..., u_8 \}$: $\{k_{1\text{-}9}\}k_{1\text{-}8}$

$s \rightarrow \{u_9 \}$: $\{k_{1\text{-}9} \}k_9$

$s \rightarrow \{u_7 , u_8\}$: $\{k_{789} \}k_{78}$

$s \rightarrow \{u_9 \}$: $\{k_{789} \}k_9$

Kullanıcı-Odaklı Yeniden Anahtar Tahsisi ile karşılaştırıldığında yukarıda anlatılan yaklaşım, sunucunun şifreleme maliyetini $\frac{h(h+1)}{2}-1$'den $2(h\text{-}1)$'e indirir. Ancak h yerine $2(h\text{-}1)$ adet yeniden anahtar tahsisi mesajı gerektirir. Şekil 6.4'te, u_9 kullanıcısının şeklin üst kısmında bulunan güvenli gruba dahil olması için gereken yeniden anahtar tahsisi maliyeti, h = 3 olduğundan, 2(3-1) = 4 ve yeniden anahtar tahsisi mesaj sayısı 2(3-1) = 4'tür. Gerçekten de, yukarıdaki paragrafta verilen mesaj sayısı ve mesajların içerisinde yer alan şifrelenen anahtar sayısı bu sonuçları doğrulamaktadır.

Yeniden anahtar tahsisi mesaj sayısını azaltmak için, belirli bir kullanıcıya yollanan yeniden anahtar tahsisi mesajları bir araya getirilip tek bir mesaj olarak yollanabilir. Bu durumda, anahtar ağaç grafına katılma işlemleri için gerekli protokol adımları aşağıdaki şekildedir:

(1) $u \rightarrow s$: katılma isteği

(2) $s \Leftrightarrow u$: u'yu asılla ve k_u'yu yolla

(3) s : bir katılma noktası bul ve k_u yu ekle;

x_j katılma noktasını göstermek ve x_0 kök ve i = 1,..... , j olmak üzere $x_{i\text{-}1}$ x_i nin ebeveyni olsun.

K_{j+1} k_u yu göstermek ve K_0 ,..... ,K_j x_0 ,.... ,x_j nin eski anahtarları olmak üzere, rastgele olarak K_0' , ..., K_j' yeni anahtarları yarat.

(4) for i = 0 upto j do

$s \rightarrow$ (userset(K_i) - userset(K_{i+1})) : { K_0' }K_0 , ... , { K_i' }K_i

(5) $s \rightarrow u$: { K_0' , ... , K_j' }k_u

Şekil 6.4'te verilen anahtar ağaç grafını ele alalım. Kullanıcı u_9 un şeklin üst kısmında yer alan güvenli gruba katılması için, s sunucusunun, dört yeniden anahtar tahsisi mesajı yerine aşağıdaki üç yeniden anahtar tahsisi mesajını göndermesi yeterli olacaktır.

$s \rightarrow \{u_1$,....., $u_6\}$: $\{k_{1\text{-}9}\}k_{1\text{-}8}$

$s \rightarrow \{u_7$, $u_8\}$: $\{k_{1\text{-}9}\}k_{1\text{-}8}$, $\{k_{789}\}k_{78}$

$s \rightarrow u_9$: { $k_{1\text{-}9}$, k_{789} }k_9

Birleştirilmiş mesajların kullanılması ile Anahtar-Odaklı Yeniden Anahtar Tahsisi kullanılması durumunda, yeniden anahtar tahsisi mesaj sayısı h (Kullanıcı-Odaklı yaklaşım ile aynı), şifreleme maliyeti ise 2(h-1) olacaktır. Bu analiz sonucunda, sunucu için Anahtar-Odaklı yaklaşımın Kullanıcı-Odaklı yaklaşıma göre daha iyi olduğu ortaya çıkmaktadır.

Şekil 6.4'te verilen anahtar ağacının yüksekliği h = 3'tür. Buradan yola çıkarak, elde ettiğimiz sonuçları yukarıda verilen örneğe uygulayacak olursak, yeniden anahtar tahsisi mesaj sayısı 3 olacaktır. Şifreleme maliyeti ise 2(3-1) = 4'tür. Yukarıdaki paragrafta verilen mesajlar, bu sonuçların doğru olduğunu göstermektedir.

6.2.3.3. GRUP-ODAKLI YENİDEN ANAHTAR TAHSİSİ

Anahtar-Odaklı yaklaşımda, her bir yeni anahtar ayrı ayrı olarak şifrelenir (yeni katılan kullanıcıya yollanan anahtarlar için bu işlem uygulanmaz). Sunucu, her bir alt grubun ihtiyacına uygun olarak, yeniden anahtar tahsisi mesajları oluşturur. Bir alt grup içerisinde yer alan herhangi bir kullanıcı, yalnızca kendi ihtiyaç duyduğu anahtarları içeren yeniden anahtar tahsisi mesajlarını alır.

Grup-Odaklı Yeniden Anahtar Tahsisi olarak adlandırılan alternatif bir yaklaşımda ise, sunucu tüm yeni anahtarları içeren tek bir yeniden anahtar tahsisi mesajı oluşturur. Ardından, oluşturulan bu yeniden anahtar tahsisi mesajı, tüm gruba çoğa gönderim yöntemi ile yollanır. Bu şekilde hazırlanan bir yeniden anahtar tahsisi mesajının, diğer yöntemlerde oluşturulan yeniden anahtar tahsisi mesajlarına göre çok daha büyük olacağı ve her bir kullanıcının ihtiyacı olmayan bilgileri içereceği açıktır. Ancak, ölçeklenebilirlik bir sorun oluşturmaz. Çünkü, *n* kullanıcılı bir grup ve derecesi *d* olan bir anahtar ağacı için mesaj büyüklüğü $O(\log_d n)$'dir. [16]

Bu yaklaşım kullanılarak, anahtar ağaç grafına katılma işlemleri için gerekli protokol adımları aşağıdaki şekildedir:

(1) $u \rightarrow s$: katılma isteği

(2) $s \Leftrightarrow u$: u'yu asılla ve k_u'yu yolla

(3) s : bir katılma noktası bul ve k_u yu ekle;

x_j katılma noktasını göstermek ve x_0 kök ve i = 1,..... , j olmak üzere x_{i-1} x_i nin ebeveyni olsun.

K_{j+1} k_u yu göstermek ve K_0 ,..... ,K_j x_0 ,.... ,x_j nin eski anahtarları olmak üzere, rastgele olarak K_0' , ..., K_j' yeni anahtarları yarat.

(4) $s \rightarrow$ (userset(K_0)) : { K_0' }K_0 , ... , { K_j' }K_j

(5) $s \rightarrow u$: { K_0' , ... , K_j' }k_u

Şekil 6.4'te verilen anahtar ağaç grafını ele alalım. Kullanıcı u_9 un şeklin üst kısmında yer alan güvenli gruba katılması için, *s* sunucusunun, aşağıdaki iki yeniden anahtar tahsisi mesajını oluşturması ve yollaması gerekir. Birinci mesaj çoğa gönderim yöntemi ile tüm gruba yollanır. Diğeri ise teke gönderim yöntemi ile yalnızca yeni katılan kullanıcıya yollanır.

$s \rightarrow \{u_1$,....., $u_8\}$: $\{k_{1\text{-}9}\}k_{1\text{-}8}$, $\{k_{789}\}k_{78}$

$s \rightarrow \{u_9\}$ $: \{k_{1\text{-}9}, k_{789}\}k_9$

Grup-Odaklı yaklaşımın, Anahtar-Odaklı ve Kullanıcı-Odaklı yaklaşımlara göre bazı avantajları bulunmaktadır. Bunlardan birincisi, alt grup çoğa gönderimi ya da teke gönderim yerine çoğa gönderim kullanılabilir. İkincisi, daha az yeniden anahtar tahsisi mesajı ile sunucunun mesaj başına ek yükü azalıtır. Üçüncü olarak, katılma/ayrılma başına sunucunun ilettiği bayt (sekizli) sayısı, yeniden anahtar tahsisi mesajlarında, bazı bilgilerin ikinci kopyasını çıkaran Kullanıcı-Odaklı ve Anahtar-Odaklı yaklaşımlara oranla oldukça azdır.

Bu yaklaşım, yeniden anahtar tahsisi mesaj sayısını tek bir çoğa gönderim mesajı ve tek bir teke gönderim mesajına indirir. Bu sırada şifreleme maliyeti 2(h-1)'dir. Bu değer, Anahtar-Odaklı yaklaşımdaki değer ile aynıdır.

6.2.4. AĞAÇ ANAHTAR GRAFINDAN AYRILMA

Kullanıcı *u'*dan gelen ayrılma isteği onaylandıktan sonra, *s* sunucusu, *u* için yaratılan *u-düğümü* ve kullanıcı *u*'nun şahsi anahtarı k_u için yaratılmış olan *k-düğümü'*nü silerek anahtar grafını günceller. Kullanıcı *u*'nun şahsi anahtarının yer aldığı *k-düğümü'*nün ebeveyni olan *k-düğümü'*ne *ayrılma noktası* adı verilir.

Ayrılan kullanıcının ayrıldıktan sonraki grup iletişimine erişimini engellemek için, ayrılma noktasından kök düğümüne kadar yer alan tüm anahtarların değiştirilmesi gerekir. Bu yol üzerinde yer alan tüm *k-düğümleri* için yeni anahtarlar yaratıldıktan sonra, *s* sunucusunun geride kalan kullanıcılara bu anahtarları güvenli olarak dağıtması gereklidir. Bunu bir örnek üzerinde açıklayalım: Şekil 6.4'te verilen anahtar ağacını ele alalım. Kullanıcı u_9 un şeklin alt kısmında yer alan güvenli gruptan ayrılmak istediğini düşünelim. Bu durumda, alt kısımdaki anahtar grafında ayrılma noktası k_{789} için yaratılmış olan *k-düğümü* olacaktır. Bu düğümde yer alan anahtar, üst kısımdaki anahtar grafından da görülebileceği gibi, k_{78} olarak değiştirilecektir. Kökte yer alan grup anahtarı ise $k_{1\text{-}9}$ yerine $k_{1\text{-}8}$ olarak değiştirilecektir. Kullanıcı u_1,...., u_6 yalnızca yeni grup anahtarı $k_{1\text{-}8}$ e ihtiyaç duyarken, kullanıcı u_7 ve u_8 hem yeni alt grup anahtarı k_{78}'e hem de yeni grup anahtarı $k_{1\text{-}8}$ e ihtiyaç duyacaklardır.[16]

Ayrılma işleminden sonra, yeni anahtarları kullanıcılara güvenli olarak dağıtmak için, katılma işleminde incelediğimiz üç farklı yeniden anahtar tahsisi yaklaşımı kullanılacaktır. Bundan sonraki bölümde bu yaklaşımlar ele alınacaktır.

6.2.4.1. KULLANICI-ODAKLI YENİDEN ANAHTAR TAHSİSİ

Bu yaklaşımda, her bir kullanıcı ihtiyaç duyduğu tüm yeni anahtarların, sahip olduğu mevcut anahtarlar kullanılarak şifrelendiği bir yeniden anahtar tahsisi mesajı alır.

Ayrılma işlemi için, Kullanıcı-Odaklı Yeniden Anahtar Tahsisi mesajları şu şekilde oluşturulur. Anahtarı k yerine k' olarak değiştirilecek olan her bir *x k-düğümü* ve bu *x* düğümünün değişmeyen her bir çocuğu *y* için, anahtar sunucusu, bu *x k-düğümü'*nün yeni anahtarını ve bu düğümden köke kadar olan tüm ebeveyn düğümlerinde yer alan yeni anahtarları, *y k-düğümü'*nün *K* anahtarı ile şifreleyerek bir yeniden anahtar tahsisi mesajı oluşturur. Ardından, bu yeniden anahtar tahsisi mesajı, userset(*K*) kullanıcı alt kümesine çoğa gönderim yöntemi ile yollanır. [16]

Şekil 6.4'te verilen anahtar ağaç grafını ele alalım. Kullanıcı u_9 un şeklin alt kısmında yer alan güvenli gruptan ayrılması için, *s* sunucusunun, aşağıdaki dört yeniden anahtar tahsisi mesajını oluşturması ve yollaması gerekir.

$s \rightarrow \{u_1 , u_2 , u_3 \}$: $\{k_{1\text{-}8}\}k_{123}$

$s \rightarrow \{u_4 , u_5 , u_6 \}$: $\{k_{1\text{-}8}\}k_{456}$

$s \rightarrow u_7$: $\{k_{1\text{-}8} , k_{78} \}k_7$

$s \rightarrow u_8$: $\{k_{1\text{-}8} , k_{78} \}k_8$

Bu yaklaşım (*d*-1)(*h-1)* adet yeniden anahtar tahsisi mesajı gerektirir. Burada, *h* anahtar ağaç grafının yüksekliği, *d* ise anahtar ağaç grafının derecesidir. Sunucu için şifreleme maliyeti ise;

$$(d-1)(1+2+\ldots+h-1) = \frac{(d-1)h(h-1)}{2}$$

olacaktır. Yukarıdaki örneğe bakacak olursak, burada anahtar ağacının yüksekliğinin 3 ve derecesinin de 3 olduğunu görürüz. Bu durumda yeniden anahtar tahsisi mesaj sayısı $h = 3$ ve d = 3 olduğundan, (3-1)(3-1) = 4 olacaktır. Şifreleme maliyeti ise;

$$(3-1)(1+2) = \frac{(3-1)3(3-1)}{2} = 6$$

olacaktır. Yukarıda verilen mesaj sayısı ve mesajlar içerisinde şifrelenen anahtar sayısı, burada elde ettiğimiz sonuçların doğruluğunu göstermektedir.

6.2.4.2. ANAHTAR-ODAKLI YENİDEN ANAHTAR TAHSİSİ

Bu yaklaşımda, her bir yeni anahtar ayrı ayrı şifrelenir. [16]

Bu yaklaşım kullanılarak, anahtar ağaç grafından ayrılma işlemleri için gerekli protokol adımları aşağıdaki şekildedir:

(1) $u \rightarrow s$: {ayrılma isteği}k_u

(2) $s \rightarrow u$: {ayrılma isteği onaylandı}k_u

(3) s : ayrılma noktasını bul (k_u nun ebeveyni);

k_u yu ağaçtan sil;

x_{j+1} ağaçtan silinen k_u için yaratılmış k-*düğümü*'nü göstermek üzere, x_j ayrılma noktası, x_0 kök ve i = 1,..... , j olmak üzere x_{i-1} x_i nin ebeveyni olsun.

x_0 ,.... ,x_j nin yeni anahtarları olmak üzere, rastgele olarak K_0' , ..., K_j' yeni anahtarları yarat.

(4) for i = 0 upto j do

for each x_i nin y ($\neq x_{i+1}$) çocuğu do

y *k-düğümü*ndeki anahtar K ile gösterilmek üzere;

$s \rightarrow$ userset(K) : { K_i' }K , { K_{i-1}' } K_i' , ..., { K_0' } K_1'

Şekil 6.4'te verilen anahtar ağaç grafını ele alalım. Kullanıcı u_9 un şeklin alt kısmında yer alan güvenli gruptan ayrılması için, *s* sunucusunun, aşağıdaki dört yeniden anahtar tahsisi mesajını oluşturması ve yollaması gerekir.

$s \rightarrow \{u_1 , u_2 , u_3 \}$: $\{k_{1\text{-}8}\}k_{123}$

$s \rightarrow \{u_4 , u_5 , u_6 \}$: $\{k_{1\text{-}8}\}k_{456}$

$s \rightarrow u_7$: $\{k_{1\text{-}8}\}k_{78}$, $\{ k_{78} \}k_7$

$s \rightarrow u_8$: $\{k_{1\text{-}8}\}k_{78}$, $\{ k_{78} \}k_8$

Şifrelenmiş yeni anahtarların, farklı yeniden anahtar tahsisi mesajlarında kullanılması için saklaması ile, bu yaklaşımın şifreleme maliyeti $d(h\text{-}1)\text{-}1$ olacaktır. Bu sonuç, Kullanıcı-Odaklı yaklaşımda elde edilen şifreleme maliyetinden oldukça düşüktür. Yeniden anahtar tahsisi mesajlarının sayısı ise $(d\text{-}1)(h\text{-}1)$ olacaktır. Bu sayı Kullanıcı-Odaklı yaklaşımda elde edilen değer ile aynıdır.

Şekil 6.4'te verilen anahtar ağacının yüksekliği $h = 3$, derecesi ise $d = 3$'tür. Buradan yola çıkarak, elde ettiğimiz sonuçları yukarıda verilen örneğe uygulayacak olursak, yeniden anahtar tahsisi mesaj sayısı $(3\text{-}1)(3\text{-}1) = 4$ olacaktır. Şifreleme maliyeti ise $3(3\text{-}1)\text{-}1 = 5$'dir. Yukarıdaki paragrafda verilen örnek için çıkarılan gerekli yeniden anahtar tahsisi mesajları bu sonuçların doğru olduğunu göstermektedir.

6.2.4.3. GRUP-ODAKLI YENİDEN ANAHTAR TAHSİSİ

Bu yaklaşımda, tüm yeni anahtarları içeren tek bir yeniden anahtar tahsisi mesajı oluşturulur. [16]

Bu yaklaşım kullanılarak, anahtar ağaç grafından ayrılma işlemleri için gerekli protokol adımları aşağıdaki şekildedir:

(1) $u \rightarrow s$: {ayrılma isteği}k_u

(2) $s \rightarrow u$: {ayrılma isteği onaylandı}k_u

(3) s : ayrılma noktasını bul (k_u nun ebeveyni);

k_u yu ağaçtan sil;

x_{j+1} ağaçtan silinen k_u için yaratılmış *k-düğümü'*nü göstermek üzere, x_j ayrılma noktası, x_0 kök ve i = 1,..... , j olmak üzere x_{i-1} x_i nin ebeveyni olsun.

x_0 ,.... ,x_j nin yeni anahtarları olmak üzere, rastgele olarak K_0' , ..., K_j' yeni anahtarları yarat.

(4) for i = 0 upto j do

J_1 ,....., J_r yeni anahtar ağacında x_i nin çocuk düğümlerindeki anahtarları göstersin;

L_i : { K_i' }J_1 , ..., { K_i' } J_r olmak üzere;

$s \rightarrow$ userset(K_0') : L_o , ..., L_j

Şekil 6.4'te verilen anahtar ağaç grafını ele alalım. Kullanıcı u_9 un şeklin alt kısmında yer alan güvenli gruptan ayrılması için, *s* sunucusunun, aşağıdaki yeniden anahtar tahsisi mesajlarını oluşturması ve yollaması gerekir.

L_0 : $\{k_{1\text{-}8}\}k_{123}$, $\{k_{1\text{-}8}\}k_{456}$, $\{k_{1\text{-}8}\}k_{78}$

L_1 : $\{k_{78}\}k_7$, $\{k_{78}\}k_8$ olmak üzere

$s \rightarrow \{u_1$, ... , $u_8\}$: L_0 , L_1

Ayrılma işlemi için oluşturulan tek bir yeniden anahtar tahsisi mesajının, katılma işlemi için oluşturulan mesajdan *d* kat daha büyük olduğu görülmektedir. Burada *d,* bir *k-düğümü'*nün ortalama derecesini göstermektedir.

Bu yaklaşımda, yalnızca bir adet yeniden anahtar tahsisi mesajı kullanılır ve bu mesaj tüm gruba çoğa gönderim yöntemi ile yollanır. Şifreleme maliyeti, Kullanıcı-Odaklı yaklaşımda olduğu gibi $d(h\text{-}1)$-1'dir.

6.3. ŞİFRELEME VE ŞİFRE ÇÖZME MALİYETLERİ

Sunucu ve kullanıcıların yaklaşık olarak hesaplama maliyetleri, katılma/ayrılma isteği için gereken anahtar şifreleme ve şifre çözme sayıları ile ölçülür. Güvenli grup içerisindeki kullanıcı sayısı n olsun. Her bir katılma/ayrılma isteği için, katılma/ayrılma isteğinde bulunan kullanıcı *istekte bulunan kullanıcı*, grup içerisindeki diğer kullanıcılar ise *istekte bulunmayan kullanıcılar* olarak adlandırılır. Bir katılma/ayrılma isteği için istekte bulunan kullanıcının, istekte bulunmayan bir kullanıcının ve sunucunun maliyetleri Tablo 6.2'de sırası ile (a), (b) ve (c) kısımlarında gösterilmiştir. Bu maliyetler, yıldız ve ağaç anahtar grafları için yukarıda anlatılan protokoller kullanılarak elde edilmiştir (ağaç anahtar grafları için Anahtar-Odaklı ya da Grup-Odaklı yaklaşımın kullanıldığı kabul edilmiştir). [16]

Tablo 6.2 Bir katılma/ayrılma isteğinin maliyeti

(a)	İstekte bulunan kullanıcı		
	Yıldız	Ağaç	Tam
Katılma	1	$h-1$	$2^n - 1$
Ayrılma	0	0	0
(b)	İstekte bulunmayan bir kullanıcı		
	Yıldız	Ağaç	Tam
Katılma	1	$\frac{d}{d-1}$	2^{n-1}
Ayrılma	1	$\frac{d}{d-1}$	0
(c)	Sunucu		
	Yıldız	Ağaç	Tam
Katılma	2	$2(h-1)$	$2^{n+1} + 2$
Ayrılma	$n - 1$	$d(h-1)$	0

Bir anahtar ağacı için, *d* anahtar ağacının derecesini, *h* ise anahtar ağacının yüksekliğini ifade eder. Bu durumda, istekte bulunmayan *u* kullanıcısı için, katılma ya da ayrılma için *u*'nun ortalama maliyeti *d* / (*d*-1) den daha azdır ve ağacın büyüklüğünden bağımsızdır.

İsteğin eşit olasılıkla katılma ya da ayrılma olabileceği ve gruptaki kullanıcı sayısı *n*'nin çok büyük olduğu düşünüldüğünde, istek başına elde edilecek ortalama maliyetler Tablo 6.3'de verilmiştir.

Tablo 6.3 İstek başına ortalama maliyet

	Yıldız	Ağaç	Tam
Sunucunun maliyeti	$n / 2$	$(d + 2)(h - 1) / 2$	2^n
Kullanıcının maliyeti	$n - 1$	$d / (d\text{-}1)$	2^n

Tablo 6.3'e bakıldığında, tam anahtar graflarının kullanılmaması gerektiği açık olarak görülmektedir. Diğer taraftan, ölçeklenebilir grup anahtar yönetimi, ağaç anahtar grafları kullanılarak sağlanabilir. Dolu ve dengeli bir d-li ağaç için, ortalama sunucu maliyeti, $(d+2)(h-1) / 2 = (d+2)(\log_d (n)) / 2$'dir. Ancak her bir kullanıcı biraz daha fazla iş yapmak durumunda kalır (1 yerine $d / (d-1)$). $d = 4$ olması durumunda, bir kullanıcının ortalama olarak 1 yerine 1.33 anahtar şifresi çözme işlemi yapması gerekir (sunucu maliyetinin $d = 4$ olması durumunda minimize edildiği gösterilebilir. Başka bir deyişle anahtar ağacının optimal derecesi dörttür).

6.4. YENİDEN ANAHTAR TAHSİSİ MESAJLARININ İMZALANMASI

Bu çalışmada anlatılan katılma/ayrılma protokollerinde, her bir yeniden anahtar tahsisi mesajı, bir ya da birden fazla yeni anahtar içerir. Her bir yeni anahtar, bir grup kullanıcıya yönlendirilmiştir ve bu grup içerisindeki kullanıcılar ve sunucu tarafından bilinen bir anahtar kullanılarak şifrelenir. Burada herhangi bir kullanıcının, sunucunun kılığına girmesi ve diğer kullanıcılara yeniden anahtar tahsisi mesajları yollaması olasılığı vardır. Bu nedenle, eğer kullanıcılar güvenilir değillerse, her bir yeniden anahtar tahsisi mesajının sunucu tarafından dijital olarak imzalanması gerekir.[16]

Dijital imza işlemi, DES (Data Encryption Standard-DES) kullanılarak bir anahtarın şifrelenmesi işlemine göre, büyüklüğün iki katı daha yavaş bir işlemdir. Bu nedenle, her bir katılma/ayrılma işlemi için gereken dijital imza işlem sayısının azaltılması gerekir. Eğer, her bir yeniden anahtar tahsisi mesajının ayrı ayrı imzalandığı düşünülecek olursa, tüm kullanıcılar için katılma/ayrılma isteği başına tek bir yeniden anahtar tahsisi mesajının kullanıldığı Grup-Odaklı Yeniden Anahtar Tahsisi, katılma/ayrılma başına pek çok yeniden anahtar tahsisi mesajı gerektiren Anahtar-Odaklı ya da Kullanıcı-Odaklı Yeniden Anahtar Tahsisi yaklaşımlarından çok daha iyi sonuçlar verecektir.

M_1 , ... , M_m olmak üzere, m adet yeniden anahtar tahsisi mesajı olduğunu kabul edelim. Mesaj özetleri ise, $h()$ MD5 (Message Digest5-MD5) gibi bir güvenli mesaj özet fonksiyonu olmak üzere $d_i = h(M_i)$ $i = 1,...., m$ şeklinde hesaplansın. Asıllamayı sağlamak için standart yöntem, sunucunun her bir mesaj özetini kendi özel anahtarı ile imzalaması ve imzalanmış mesaj özetlerini mesaj ile birlikte yollamasıdır. Bu yöntem, m adet mesaj için m dijital imza işleminin yapılmasını gerektirecektir.

Bu çalışma kapsamında, farklı bir teknik kullanılması önerilmiştir. Buna göre, mesaj özetleri d_1 , d_2 , d_3 , d_4 olan dört adet M_1 , M_2 , M_3 , M_4 mesajı olsun. $d_{12} = h(d_1 , d_2)$, $d_{34} = h(d_3 , d_4)$ ve $d_{1\text{-}4} = h(d_{12} , d_{34})$ şeklinde hesaplansın. Sunucu, $d_{1\text{-}4}$ mesaj özetini kendi şahsi anahtarı ile imzalar ve imzalanmış mesaj özeti $sign(d_{1\text{-}4})$'ü; d_3 , d_{12} ve M_4 ile M_4 mesajına ihtiyaç duyan kullanıcıya yollar. Sunucu tarafından yollananlar alındığında, kullanıcı M_4 den d_4' yü, d_4' ve d_3 den d_{34}' yü hesaplar. Elde ettiği d_{34}' yü ve d_{12} yi kullanarak $d_{1\text{-}4}'$ yü bulur ve bu değeri kendisine gelen $sign(d_{1\text{-}4})$'ü doğrulamak için kullanır. Yukarıdaki örnek, m adet mesaj için rahatlıkla genişletilebilir.[16]

Bu tekniği kullanmanın yararları Anahtar-Odaklı ve Kullanıcı-Odaklı Yeniden Anahtar Tahsisi yaklaşımlarına göre Tablo 6.4'te gösterilmiştir. Grup-Odaklı Yeniden Anahtar Tahsisi yaklaşımı için böyle bir tekniğin kullanılmasına gerek yoktur. Çünkü, yalnızca bir tane yeniden anahtar tahsisi mesajı vardır. Tabloda yer alan sonuçların elde edilmesi için, başlangıçtaki kullanıcı sayısı 8192 olan bir grup alınmış, şifreleme için DES-CBC (Data Encryption Standard-Cipher Block Chaining-DES-CBC), mesaj özetleri için MD5 ve dijital imzalar için de RSA (encryption algorithm of Rivest, Shamir, and Adleman-

RSA) kullanılmıştır. Bu tekniğin kullanılması ile, Kullanıcı-Odaklı ve Anahtar-Odaklı yaklaşımlarda yeniden anahtar tahsisi mesajlarının imzalanması için gerekli işlem zamanında 1'e 10 oranında bir azalma söz konusu olmuştur. Ancak, ortalama mesaj büyüklüğünde bir miktar artış olmaktadır.

Tablo 6.4 Ortalama yeniden anahtar tahsisi mesaj uzunluğu ve sunucu işlem zamanı

			Kullanıcı-odaklı	Anahtar-odaklı	Grup-odaklı
Mesaj başına tek imza	Mesaj büyüklüğü (bayt)	katılma	263.1	303.0	525.5
		ayrılma	233.8	270.9	1005.7
	İşlem zamanı (msec)	katılma	76.7	76.3	11.9
		ayrılma	204.6	203.8	140.1
		ortalama	140.6	140.1	11.9
Tüm mesajlar için tek imza	Mesaj büyüklüğü (bayt)	katılma	312.8	352.8	525.5
		ayrılma	306.9	344.0	1005.7
	İşlem zamanı (msec)	katılma	13.6	13.1	11.9
		ayrılma	17.1	15.9	12.0
		ortalama	15.3	14.5	11.9

6.5. BAŞARIM KARŞILAŞTIRMALARI

Bu çalışmada anlatılan yeniden anahtar tahsisi stratejilerinin ve yeniden anahtar tahsisi mesajlarının imzalanması için önerilen tekniğin başarımının değerlendirilmesi için çok sayıda deneme yapılmıştır. Sunucu, başlangıç grup büyüklüğünün, yeniden anahtar tahsisi stratejisinin, anahtar ağaç derecesinin, şifreleme algoritmasının, mesaj özet algoritmasının, dijital imza algoritmasının vb. yer aldığı bir tanım dosyasından başlatılır. *Katılma, katılma-onay, ayrılma, ayrılma-onay* mesajları ile birlikte, gerçek yeniden anahtar tahsisi mesajları sunucu ile istemci arasında yollanmaktadır. [16]

Her bir denemede, başlangıç grup büyüklüğü *n* olmak üzere, istemci, öncelikle *n* adet katılma mesajı yollamış ve sunucu bir anahtar ağacı oluşturmuştur. Ardından istemci simülatörü 1000 adet katılma/ayrılma mesajı yollamıştır. 1000 adet katılma/ayrılma mesajı rastgele olarak üretilmiştir. Her bir deneme, üç farklı 1000 adet katılma/ayrılma sekansı ile gerçekleştirilmiştir. Adil bir karşılaştırma (farklı yeniden anahtar tahsisi stratejileri, farklı dereceden ağaçlar vb. arasında) için, aynı grup büyüklükleri ve aynı üç farklı sekans kullanılmıştır. Sunucu, dolu ve dengeli bir anahtar ağacı oluşturmak ve sağlamak için buluşsal bir yöntem kullanır. Ancak, katılma/ayrılma istekleri rastgele olarak üretildiğinden, ağacın gerçekten dolu ve dengeli olması olası değildir.

Farklı yeniden anahtar tahsisi stratejilerini ve yeniden anahtar tahsisi mesajlarının imzalanması tekniğini değerlendirmek için, mesaj büyüklükleri (bayt olarak) ve mesaj sayıları ölçülmüştür. Burada, katılma/ayrılma başına sunucu tarafından harcanan işlem zamanı (msec) da ölçülebilir. Katılma/ayrılma başına sunucu işlem zamanı şu parçalardan oluşur: Birincisi, sunucu gelen isteği ayrıştırır; anahtar ağacının üzerinden geçerek hangi anahtarların değiştirileceğini tespit eder; yeni anahtarları üretir ve anahtar grafını günceller. İkincisi, sunucu yeni anahtarların şifrelenmesini sağlar ve yeniden anahtar tahsisi mesajlarını üretir. Üçüncüsü, eğer mesaj özeti tanımlanmış ise, sunucu yeniden anahtar tahsisi mesajlarının mesaj özetlerini hesaplar. Dördüncüsü ise, eğer dijital anahtar tanımlamış ise, sunucu bu çalışmada tanımlanan tekniği kullanarak mesaj özetlerini ve dijital imzaları hesaplar. Son olarak, sunucu yeniden anahtar tahsisi mesajlarını yollar. [16]

Tablo 6.5'te sunucu tarafından yollanan yeniden anahtar tahsisi mesajlarının sayıları ve büyüklükleri verilmiştir. Burada mesaj içerisine, şifreleme ve imza dahildir.

Tablo 6.5 Sunucu tarafından yollanan yeniden anahtar tahsisi mesajlarının (şifreleme ve imza ile birlikte) sayısı ve büyüklüğü (Başlangıç grup büyüklüğü 8192)

Anahtar ağaç derecesi 4			Kullanıcı-odaklı	Anahtar-odaklı	Grup-odaklı
Yeniden anahtar tahsisi mesaj büyüklüğü (bayt)	katılma başına	ortalama	312.8	352.8	525.5
		min	196	212	356
		max	552	616	564
	ayrılma başına	ortalama	306.9	344.0	1005.7
		min	228	244	968
		max	412	476	1076
Yeniden anahtar tahsisi mesaj sayısı	katılma başına	ortalama	7.00	7.00	1.00
		min	6	6	1
		max	7	7	1
	ayrılma başına	ortalama	19.02	19.02	1.00
		min	18	18	1
		max	20	20	1

Anahtar ağaç derecesi 8			Kullanıcı-odaklı	Anahtar-odaklı	Grup-odaklı
Yeniden anahtar tahsisi mesaj büyüklüğü (bayt)	katılma başına	ortalama	287.3	319.3	464.5
		min	196	212	284
		max	496	544	492
	ayrılma başına	ortalama	285.9	314.3	1293.1
		min	228	244	1256
		max	356	404	1364

Yeniden anahtar tahsisi mesaj sayısı	katılma başına	ortalama	5.00	5.00	1.00
		min	4	4	1
		max	5	5	1
	ayrılma başına	ortalama	29.01	29.01	1.00
		min	28	28	1
		max	30	30	1

Anahtar ağaç derecesi 16			Kullanıcı-odaklı	Anahtar-odaklı	Grup-odaklı
Yeniden anahtar tahsisi mesaj büyüklüğü (bayt)	katılma başına	ortalama	274.0	302.0	427.8
		min	180	196	248
		max	452	492	456
	ayrılma başına	ortalama	282.4	306.6	1869.1
		min	244	260	1832
		max	344	384	1940
Yeniden anahtar tahsisi mesaj sayısı	katılma başına	ortalama	4.00	4.00	1.00
		min	3	3	1
		max	4	4	1
	ayrılma başına	ortalama	46.01	46.01	1.00
		min	45	45	1
		max	47	47	1

Tablo 6.6'te ise istemci tarafından alınan yeniden anahtar tahsisi mesajlarının sayısı ve büyüklüğü verilmiştir. Sadece ortalama mesaj büyüklüğü verilmiştir; çünkü, minimum ve maksimum değerler Tablo 6.5'te verilen değerler ile aynıdır. Burada, her üç yeniden anahtar tahsisi stratejisi kullanıldığında da, istemcinin yalnızca bir adet yeniden anahtar tahsisi mesajı alıyor olduğuna dikkat edilmelidir.

Tablo 6.6 İstemci tarafından alınan yeniden anahtar tahsisi mesaj (şifreleme ve imza ile birlikte) sayısı ve büyüklükleri (Başlangıç grup büyüklüğü 8192)

Anahtar ağaç derecesi 4	Yeniden anahtar tahsisi mesaj büyüklüğü (bayt)		Katılma/ayrılma başına yeniden anahtar tahsisi mesaj sayısı
	Katılma başına Ortalama	Ayrılma başına ortalama	
Kullanıcı-odaklı	209.3	237.4	1
Anahtar-odaklı	227.9	256.0	1
Grup-odaklı	525.5	1005.7	1
Anahtar ağaç derecesi 8	Yeniden anahtar tahsisi mesaj büyüklüğü (bayt)		Katılma/ayrılma başına yeniden anahtar tahsisi mesaj sayısı
	Katılma başına ortalama	Ayrılma başına ortalama	
Kullanıcı-odaklı	200.0	242.0	1
Anahtar-odaklı	217.2	259.2	1
Grup-odaklı	464.5	1293.1	1
Anahtar ağaç derecesi 16	Yeniden anahtar tahsisi mesaj büyüklüğü (bayt)		Katılma/ayrılma başına yeniden anahtar tahsisi mesaj sayısı
	Katılma başına ortalama	Ayrılma başına ortalama	
Kullanıcı-odaklı	197.8	246.7	1
Anahtar-odaklı	214.3	263.2	1
Grup-odaklı	427.8	1896.1	1

Tablo 6.5'te, sunucu tarafından yollanan yeniden anahtar tahsisi mesajlarının sayısı ve büyüklüğü verilmektedir. Kullanıcı-Odaklı Yeniden Anahtar Tahsisi ve Anahtar-Odaklı Yeniden Anahtar tahsisinde, istek başına birden çok yeniden anahtar tahsisi mesajı

kullanılırken (alt gruba çoğa gönderim ya da teke gönderim yöntemi ile yollanır), Grup-Odaklı Yeniden Anahtar Tahsisinde, istek başına tek ve büyük bir yeniden anahtar tahsisi mesajı kullanılır (grup çoğa gönderim yöntemi ile yollanır). Sunucu tarafından, katılma/ayrılma başına yollanan toplam bayt sayısı, Grup-Odaklı Yeniden Anahtar Tahsisi yaklaşımına göre Anahtar-Odaklı ve Kullanıcı-Odaklı Yeniden Anahtar Tahsisi yaklaşımlarında çok daha fazladır.

Buradaki sonuçlar, Grup-Odaklı Yeniden Anahtar Tahsisi yaklaşımının, sunucu tarafında en iyi başarıma sahip olduğunun, ancak, istemci tarafında Anahtar-Odaklı ve Kullanıcı-Odaklı yaklaşımlara nazaran daha büyük mesajların işlenmesini gerektirdiği için, daha fazla işin yapılmasını gerektirdiğini göstermektedir. Ortalama yeniden anahtar tahsisi mesaj sayısı, istemci tarafında, Kullanıcı-Odaklı Yeniden Anahtar Tahsisi yaklaşımında en azdır. [16]

6.6. ANAHTAR YÖNETİMİ KONUSUNDA YAPILAN DİĞER ÇALIŞMALAR İLE BU YAKLAŞIMIN KARŞILAŞTIRILMASI

Literatürde, tek bir kaynaktan bir grup kullanıcıya güvenli olarak veri yollama problemi üzerine yapılmış pek çok çalışma bulunmaktadır. Bu çalışmalardan önceki bölümlerde bir miktar bahsedilmiştir. Bu çalışmaların hepsinde, grup anahtarının yeni katılan kullanıcıya nasıl güvenli olarak yollanacağı incelenmiştir. Grup içerisinden ayrılan bir kullanıcı olması durumunda, grup anahtarının nasıl değiştirileceği konusu için, yeni bir güvenli grubun oluşturulması haricinde, herhangi bir çözüm ortaya sürülmemiştir. Yeni bir güvenli grubun oluşturulması yaklaşımı ise açık bir şekilde ölçeklenebilir değildir. [16]

Sıklıkla katılma ve ayrılmaların yaşandığı büyük gruplar için, grup anahtar yönetiminin ölçeklenebilirliği, ilk olarak, Mittra tarafından ortaya atılan Iolus sistemi [17] ile adreslenmiştir. Bu çalışmada da öne sürüldüğü gibi, Iolus sisteminde de ölçeklenebilirlik problemi hiyerarşi kullanılarak çözümlenmiştir. Ancak, buradaki yaklaşım ile Iolus arasındaki benzerlik bu kadardır. Sistem mimarileri birbirinden oldukça farklıdır.

Iolus ağacında hiyerarşi, yapraklarda kullanıcılar ve kullanıcıların üzerinde grup güvenlik temsilcilerinden oluşur. Her bir ağaç düğümü ve bu düğüme (grup güvenlik

temsilcisi) bağlı olan çocukları (kullanıcılar ya da daha alt seviye grup güvenlik temsilcileri) ile birlikte bir alt grup oluşturur ve bir alt grup anahtarını paylaşırlar. Dolayısı ile bir alt gruba katılma ya da ayrılma diğer alt grupları etkilemez. Sadece katılan ya da ayrılan kullanıcının bulunduğu alt grubun, alt grup anahtarının değiştirilmesi gerekir.

Bu çalışmada anlatılan ağaç hiyerarşisinde, yapraklarda şahsi anahtarlar, kökte grup anahtarı ve ara katmanlarda alt grup anahtarları yer alır. Tüm kullanıcılar için tek bir anahtar sunucusu vardır. Grup güvenlik temsilcileri yoktur ve her bir kullanıcıya kendi şahsi anahtarı, grup anahtarı ve bazı alt grupların anahtarları verilir.

Bu iki yaklaşımın karşılaştırılması için başarım, güven ve güvenilirlik başlıklarının ele alınması gerekir.

6.6.1. BAŞARIM

Her iki yaklaşımda hiyerarşiden yararlandığı için, her ikisi de O(n) problemini O(log(n)) problemine dönüştürmeye çalışır. Burada n, grup büyüklüğünü göstermektedir. Ancak, iki yaklaşım birbirinden, bir istemci güvenli gruba katılmak ya da ayrılmak istediğinde, güvenli yeniden anahtar tahsisinin sağlanması için nerede ve ne zaman işlem gerektirdiği konusunda farklılaşır.

Ayrılma sonrasında güvenli yeniden anahtar tahsisi, katılma sonrasında yeniden anahtar tahsisine oranla daha fazla işlem gerektirir. Çünkü, katılmanın tersine, önceki grup anahtarı kullanılamaz ve n adet yeniden anahtar tahsisi mesajı gerekir. Bu problem [17]'de "*1 eşit değildir n"* problemi olarak ifade edilmiştir. Bu, her iki yaklaşımda da hiyerarşi kullanılarak çözümlenmiş olan bir problemdir.

Iolus ile buradaki yaklaşım arasındaki asıl farklılık "*1 n'i etkiler"* probleminin nasıl adreslendiği noktasında ortaya çıkmaktadır. Buradaki yaklaşımda, istemcinin güvenli gruba her katılma ya da ayrılmasında, tüm grubu etkileyen bir yeniden anahtar tahsisi işlemi gerekir. Ancak bu bir ölçeklenebilirlik sorunu oluşturmaz; çünkü, sunucu maliyeti O(log(n)), istemci maliyeti ise O(1)'dir.

Iolus sisteminde, tüm grup tarafından paylaşılan bir grup anahtarı yoktur. İstemci, güvenli gruba katılmak ya da ayrılmak istediğinde, yalnızca alt grup anahtarının

değiştirilmesi gerekir. Ancak, bir istemcinin tüm gruba güvenli olarak bir mesaj yollaması için, istemcinin mesajı şifrelemek için bir *mesaj anahtarı* oluşturması ve bu mesaj anahtarının grup güvenlik temsilcileri aracılığı ile güvenli olarak tüm gruba dağıtılması gerekir. Her bir temsilci, mesaj anahtarını elde etmek için, bir alt grup anahtarını kullanarak mesajın şifresini çözer ve mesajı başka bir alt grup anahtarı ile şifreleyerek ileriye yollar.

Burada da anlatıldığı gibi, "*1 n'i etkiler"* probleminin ele alınması için yapılması gerekenlerin çoğu, Iolus sisteminde, bir istemcinin tüm gruba gizli olarak bir mesaj yollamak istemesi durumunda ortaya çıkacaktır. Bu çalışmada anlatılan yaklaşımda ise, "*1 n'i etkiler"* tipi problemi ortadan kaldırmak için yapılması gerekenlerin çoğu, istemcilerden birinin güvenli gruba katılmak ya da ayrılmak istemesi durumunda ortaya çıkacaktır.[16]

6.6.2. GÜVEN

Bu çalışmada anlatılan yaklaşım tek bir sadık birim gerektirir. O da anahtar sunucusudur. Güvenilirlik ve başarım iyileştirmesi için, anahtar sunucusu, birden çok sadık birimin gerekli olduğu durumlarda kopyalanabilir. Her bir sadık birim, güçlü güvenlik önlemleri ile korunmalıdır. (Örneğin; fiziksel güvenlik, çekirdek güvenliği) Ancak Iolus sisteminde, pek çok temsilci vardır ve tüm temsilciler sadık birimdir. Dolayısı ile, böyle sistem birimlerinin gerektirdiği güven seviyesi, bu çalışmada anlatılan yaklaşıma göre oldukça fazladır.[16]

6.6.3. GÜVENİLİRLİK

Iolus'da, temsilcilerin mesaj anahtarlarını güvenli olarak ileriye yollamaları gereklidir. Temsilcilerden birisi aksadığında, bir yedek gerek olmaktadır. Güvenilirliği arttırmak için, tek bir anahtar sunucusunun yedeklenmesinin, oldukça fazla sayıda temsilcinin yedeklenmesine nispeten daha kolay olduğu açıktır. [16]

7. ANAHTAR GRAFLARI KULLANILARAK GÜVENLİ ÇOĞA GÖNDERİM UYGULAMASI

Bu bölümde, altıncı bölümde anlatılmış olan anahtar ağaç grafları kullanılarak, güvenli çoğa gönderim mimarisinin uygulanması sonucunda elde edilmiş olan yazılım tanıtılacak ve bu yazılım aracılığı ile elde edilen başarım sonuçları verilecektir. Elde edilen başarım sonuçları ile Kullanıcı-Odaklı Yeniden Anahtar Tahsisi, Anahtar-Odaklı Yeniden Anahtar Tahsisi ve Grup-Odaklı Yeniden Anahtar Tahsisi yöntemleri karşılaştırılacaktır.

Çalışma kapsamında iki farklı yazılım geliştirilmiştir. Bunlardan birincisi, ağaç grafları kullanılarak güvenli çoğa gönderim için kullanılabilecek, yukarıda isimleri verilmiş olan yeniden anahtar tahsisi yöntemlerinin birbirleri ile karşılaştırılmasını sağlayacak başarım sonuçlarını elde etmek amacıyla geliştirilmiş benzetim programıdır. Diğer yazılım ise, anahtar ağaçları kullanılarak güvenli çoğa gönderim yapılmasını sağlayan, gerçek bir çoğa gönderim uygulamasıdır.

7.1 BENZETİM YAZILIMI

Benzetim yazılımı "Microsoft Visual Studio" içerisinde yer alan "Microsoft Visual C++" geliştirme ortamı kullanılarak geliştirilmiştir. Tek bir "C" dosyasından oluşur. Program; Kullanıcı-Odaklı, Anahtar-Odaklı ve Grup-Odaklı Yeniden Anahtar Tahsisi yöntemlerinin her üçünü de içermektedir. Programın çalışması ile, öncelikle, anahtar ağacının başlangıç kullanıcı sayısının girilmesi istenir. Ardından, yeniden anahtar tahsisi için kullanılacak yöntemin seçilmesi istenir. Seçilen yeniden anahtar tahsisi yöntemi kullanılarak, verilen başlangıç kullanıcı sayısı kadar kullanıcıdan, bir ikili anahtar ağacı oluşturulur.

Yazılımın geliştirilmesi aşamasında, ikili anahtar ağaçlarının kullanılması tercih edilmiştir. Yöntemlerin uygulanması açısından, anahtar ağacının derecesi konusunda

hiçbir sınırlama söz konusu değildir. Analiz neticesinde elde edilecek nihai değerler ağaç derecesine bağlı olarak değişiklik gösterebilir. Ancak, burada yapılan analizlerde, yöntemlerin birbirleri ile karşılaştırılması söz konusu olduğundan, bağıl değerler önemli olmaktadır. Bu nedenle, analiz sırasında kullanılan ağaçların derecesi önem taşımamaktadır. Dolayısı ile bu çalışmada, bir önceki bölümde verilen analiz sonuçlarının 2'den daha fazla dereceli ağaçlar için olması ve ikili ağaçlarda anahtar ağaç grafının yüksekliğinin çok çabuk artıyor olmasının mesaj sayıları ve şifreleme maliyetini küçük kullanıcılar için bile kayda değer oranda değiştiriyor olması nedeniyle, böyle bir tercih yapılmıştır.

Programın geliştirilmesi aşamasında, anahtar graflarının oluşturulması için, bağlantılı liste yapısı kullanılmıştır. Anahtar ağacının her bir düğümünün (u-düğümleri ve k-düğümleri) oluşturulması için aşağıdaki "structure" yapısı kullanılmıştır:

```
struct node
{
        char *no;
        char *userno;
        struct node *left;
        struct node *right;
};
```

Bu yapıda "no" alanı, bu düğümde yer alan anahtarın numarasını ifade eder. Bu düğümde yer alan anahtar herhangi bir kullanıcının şahsi anahtarı ise, bu kullanıcının numarası "userno" alanında yer alır. Böylelikle, bu anahtar düğümünün bir ara düğüm mü olduğu, yoksa herhangi bir kullanıcının şahsi anahtarını tutan bir yaprak düğümü mü olduğu bu alanda yer alan değer ile anlaşılabilir. "left" alanı, düğümün sol tarafında yer alan düğümü işaret eder. "right" alanı ise, ağaçta düğümün sağ tarafında yer alan düğümü ifade eder. Her iki alan da NULL değer taşıyor ise, bu düğüm bir yaprak demektir. Her yaprağın "userno" alanında boşluktan farklı bir değer bulunmalıdır.

Başlangıç kullanıcı sayısı ile bir ikili anahtar ağacı yaratıldıktan sonra, başarım sonuçlarının elde edilmesi için rasgele olarak kullanıcıların ağaçtan çıkarılması ve yeni kullanıcıların ağaca eklenmesi sağlanır. Kullanıcıların eklenmesi sırasında başlangıç

kullanıcı sayısının aşılmamasına dikkat edilmiştir. Kullanıcıların rasgele olarak eklenmesi ve silinmesi için, sıfır ile bir arasında "random" olarak bir sayı üretilir. Bu sayının sıfır (sıfır) olması durumunda ağaçtan bir kullanıcının çıkarılması sağlanır. Hangi kullanıcının çıkarılacağının belirlenmesinde de bir rasgelelik söz konusudur. "Random" fonksiyonu ile üretilen kullanıcı numarasına sahip olan kullanıcının ağaçtan silinmesi sağlanır. Ekleme ve silme işlemi için üretilen numara "1" (bir) ise, ağaca yeni bir kullanıcının eklenmesi sağlanır. Bütün bu işlemler yapılırken, başlangıçta seçilen anahtar tahsisi yöntemine göre üretilen mesaj sayısı ve şifreleme maliyetleri hesaplanır.

Üç farklı yöntemin adil olarak karşılaştırılabilmesi için kullanılabilecek iki parametre vardır: Bunlar:

- Yeniden anahtar tahsisi için yollanan mesaj sayısı,
- Her bir yeniden anahtar tahsisi mesajının uzunluğu,
- Yeniden anahtar tahsisi mesajlarının içerisinde yer alan anahtarların şifreleme maliyeti.

Burada şifreleme maliyetinden kasıt, anahtar sunucusunun farklı anahtarlar kullanarak gerçekleştirdiği şifreleme sayısıdır.

Bunlar haricinde, akla gelebilecek bazı parametreler, tüm kullanıcıların yeniden anahtarlanması için geçen süre, başlangıç anahtar tahsisi süresi vb. dir. Ancak, kullanılan sunucunun donanım kapasitesi, sunucu ve istemcilerin içinde bulundukları ağın özellikleri (bant genişliği, gecikme zamanı vb.), istemcilerdeki işleme kapasitesi gibi pek çok unsur bu değerlerin belirlenmesinde etkili olabileceği için, kullanıcı sayısı ve ağaç özelliklerine bağlı parametreler olmadıkları düşünülerek, burada yapılacak olan karşılaştırmalarda göz önüne alınmayacaklardır.

Yazılım aracılığı ile, her bir katılma ya da ayrılma sonucunda elde edilen yeniden anahtar tahsisi mesaj sayısı, yeniden anahtar tahsisi mesaj uzunlukları ve şifreleme maliyetlerinin bir dosyaya yazılması sağlanır. Böylelikle, programın çalışması tamamlandığında, bu dosyada yer alan değerlerin incelenmesi sağlanmış olur.

Üç yeniden anahtar tahsisi yönteminin karşılaştırılması için başlangıç kullanıcı sayısı değiştirilerek çeşitli denemeler yapılmıştır. Grup-Odaklı Yeniden Anahtar Tahsisi

yöntemi kullanıldığında, farklı başlangıç kullanıcı sayıları için elde edilen yeniden anahtar tahsisi mesaj sayıları, yeniden anahtar tahsisi mesaj büyüklükleri ve şifreleme maliyetleri Ekler bölümünde Tablo A.1, Tablo A.2, Tablo A.3, Tablo A.4 ve Tablo A.5'te verilmiştir.

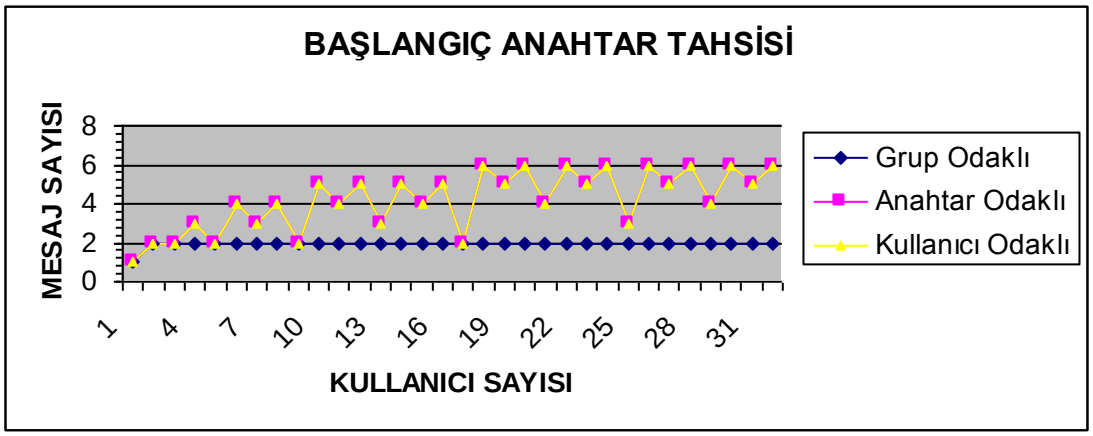
Ekler bölümünde Tablo A.6, Tablo A.7, Tablo A.8, Tablo A.9 ve Tablo A.10'da Anahtar-Odaklı Yeniden Anahtar Tahsisi yaklaşımı kullanıldığında, farklı başlangıç kullanıcı sayıları için elde edilen yeniden anahtar tahsisi mesaj sayıları, yeniden anahtar tahsisi mesaj büyüklükleri ve şifreleme maliyetleri verilmiştir.

Ekler bölümünde Tablo A.11, Tablo A.12, Tablo A.13, Tablo A.14 ve Tablo A.15'te Kullanıcı-Odaklı Yeniden Anahtar Tahsisi yaklaşımı kullanıldığında, farklı başlangıç kullanıcı sayıları için elde edilen yeniden anahtar tahsisi mesaj sayıları, yeniden anahtar tahsisi mesaj büyüklükleri ve şifreleme maliyetleri verilmiştir.

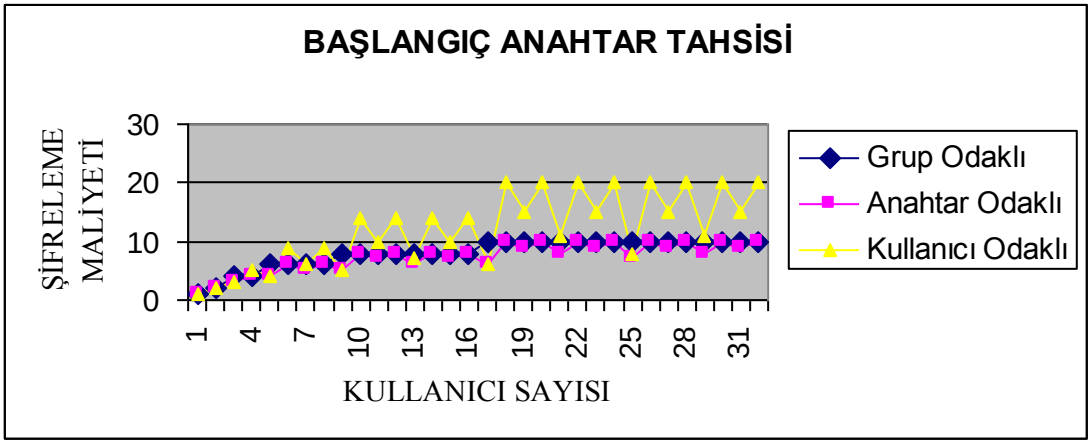
Ekler bölümünde verilen tablolardaki değerler, bu üç yöntem arasından Grup-Odaklı Yeniden Anahtar Tahsisi yaklaşımının şifreleme maliyeti ve mesaj sayısı bakımından daha üstün olduğunu göstermektedir.

Üç yöntemin birbirleri ile daha kolay karşılaştırılması için aşağıda verilen grafiklerden yararlanılacaktır.

İlk olarak, kullanıcıların birer birer gruba dahil olmaları ve sonuçta 32 kullanıcının yer aldığı güvenli bir grubun oluşturulması için, başlangıç anahtar tahsisi sırasında her üç yöntemin uygulanması durumunda elde edilen mesaj sayıları ile kullanıcı sayısı arasındaki ilişki Şekil 7.1'de, şifreleme maliyeti ve kullanıcı sayısı arasındaki ilişki ise Şekil 7.2'de verilmiştir.

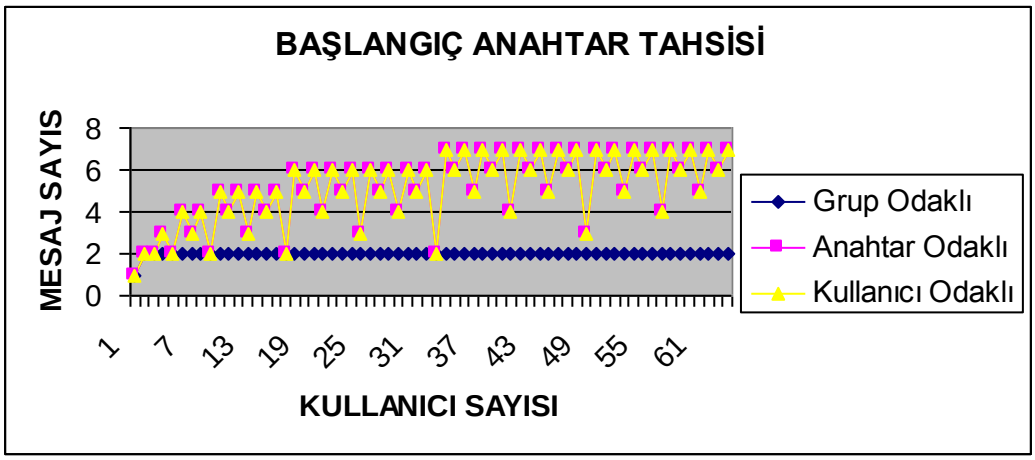


Şekil 7.1 Başlangıç Anahtar Tahsisi Mesaj Sayıları (Kullanıcı Sayısı = 32)

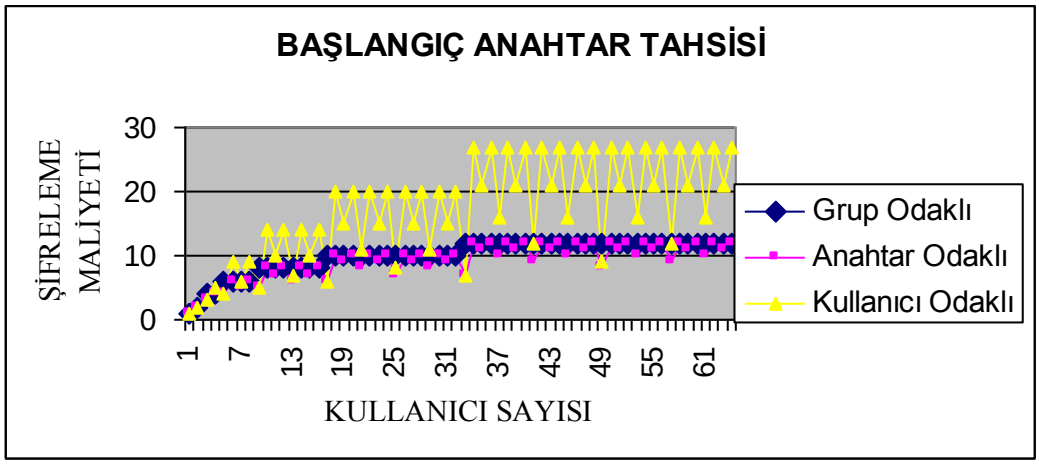


Şekil 7.2 Başlangıç Anahtar Tahsisi Şifreleme Maliyetleri (Kullanıcı Sayısı = 32)

Şekil 7.3 ve Şekil 7.4'te, kullanıcıların birer birer gruba dahil olmaları ve sonuçta 64 kullanıcının yer aldığı güvenli bir grubun oluşturulması için, başlangıç anahtar tahsisi sırasında her üç yöntemin uygulanması durumunda elde edilen mesaj sayıları ile kullanıcı sayısı arasındaki ilişki ve şifreleme maliyeti ile kullanıcı sayısı arasındaki ilişki verilmiştir.

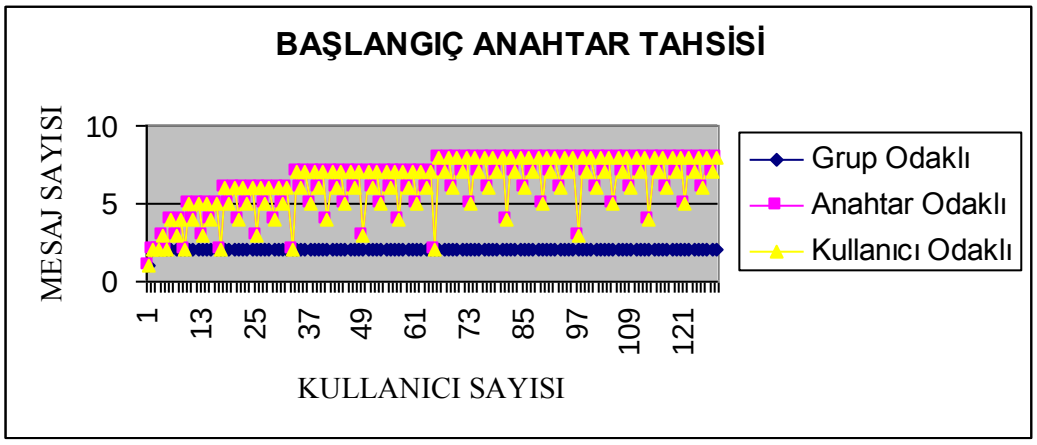


Şekil 7.3 Başlangıç Anahtar Tahsisi Mesaj Sayıları (Kullanıcı Sayısı = 64)

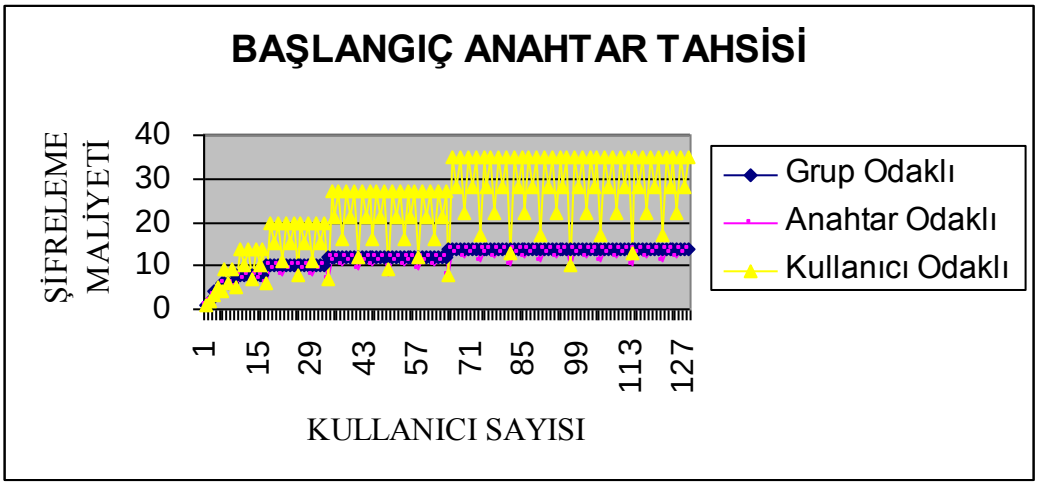


Şekil 7.4 Başlangıç Anahtar Tahsisi Şifreleme Maliyetleri (Kullanıcı Sayısı = 64)

Son olarak kullanıcıların birer birer gruba dahil olmaları ve sonuçta 128 kullanıcının yer aldığı güvenli bir grubun oluşturulması için, başlangıç anahtar tahsisi sırasında her üç yöntemin uygulanması durumunda elde edilen mesaj sayıları ile kullanıcı sayısı arasındaki ilişki Şekil 7.5'te ve şifreleme maliyeti ile kullanıcı sayısı arasındaki ilişki ise Şekil 7.6'da verilmiştir.



Şekil 7.5 Başlangıç Anahtar Tahsisi Mesaj Sayıları (Kullanıcı Sayısı = 128)

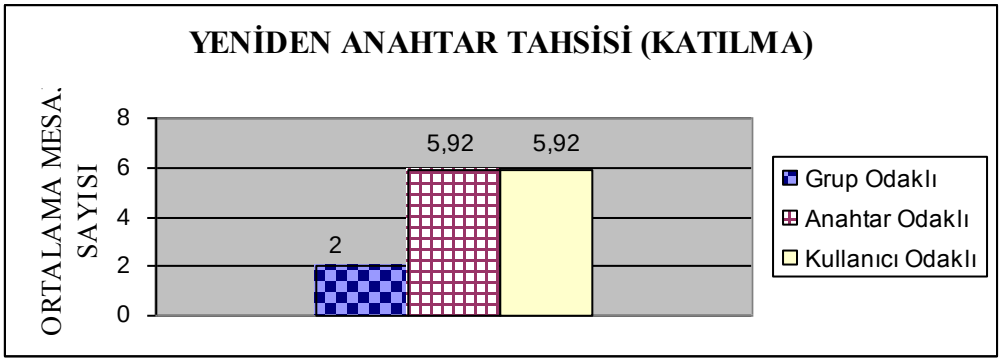


Şekil 7.6 Başlangıç Anahtar Tahsisi Şifreleme Maliyeti (Kullanıcı Sayısı = 128)

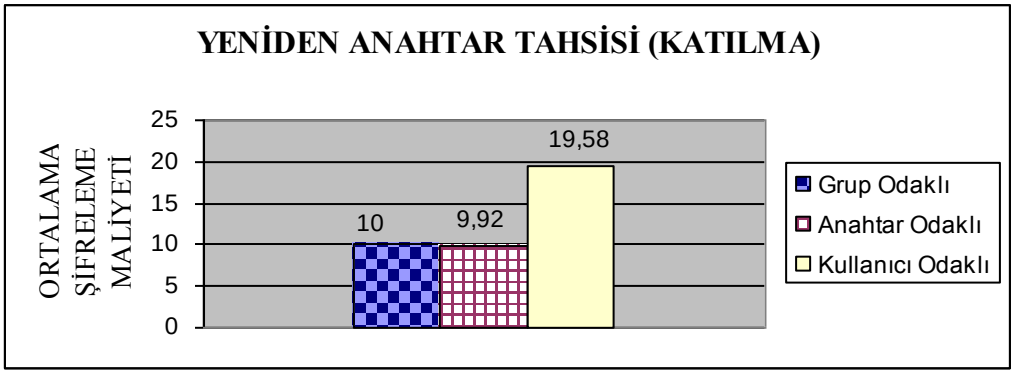
Yukarıdaki şekillerden açıkça görüldüğü üzere Grup-Odaklı Yeniden Anahtar Tahsisi yaklaşımı, şifreleme maliyeti açısından Anahtar-Odaklı Yeniden Anahtar Tahsisi yaklaşımı ile benzerlik gösterse de, mesaj sayısını oldukça azalttığı için avantajlı bir başlangıç anahtar tahsisi imkanı sunmaktadır.

Diğer taraftan, başlangıç anahtar tahsisinin yanı sıra, grup oluşturulduktan sonra gruba yeni üyelerin katılması ya da gruptan ayrılan kullanıcıların olması durumunda yeniden anahtar tahsisi başarımının incelenmesi de bu yöntemlerin sağlıklı olarak karşılaştırılabilmesi için gerekli olmaktadır.

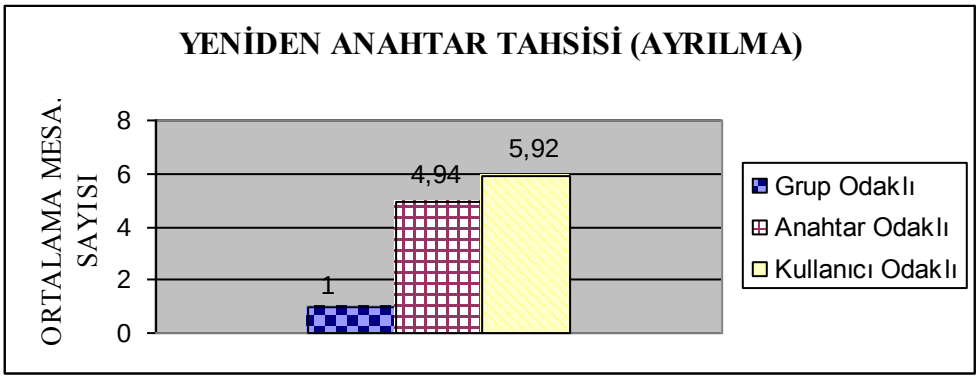
Şekil 7.7 ve Şekil 7.8'de, başlangıç kullanıcı sayısı 32 olan bir güvenli grupta rasgele kullanıcı ayrılması ve katılmaların olması durumunda, katılma başına elde edilen yeniden anahtar tahsisi ortalama mesaj sayıları ve ortalama şifreleme maliyetleri verilmiştir. Şekil 7.9 ve Şekil 7.10 ise, ayrılma başına ortalama mesaj sayıları ve ortalama şifreleme maliyetini ifade etmektedir.



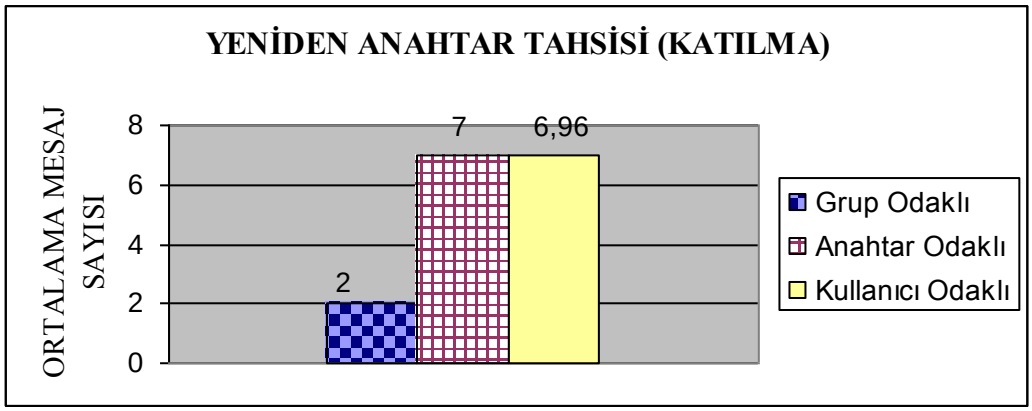
Şekil 7.7 Yeniden Anahtar Tahsisi Ortalama Mesaj Sayıları (Katılma) (Kullanıcı Sayısı = 32)



Şekil 7.8 Yeniden Anahtar Tahsisi Ortalama Şifreleme Maliyetleri (Katılma) (Kullanıcı Sayısı = 32)

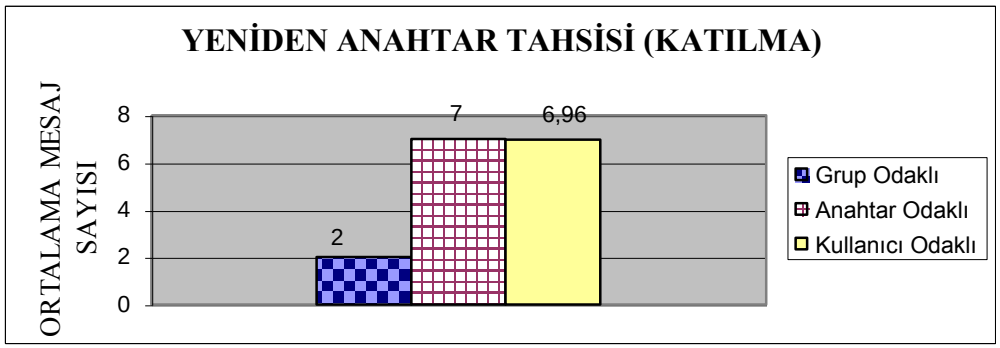


Şekil 7.9 Yeniden Anahtar Tahsisi Ortalama Mesaj Sayıları (Ayrılma) (Kullanıcı Sayısı = 32)

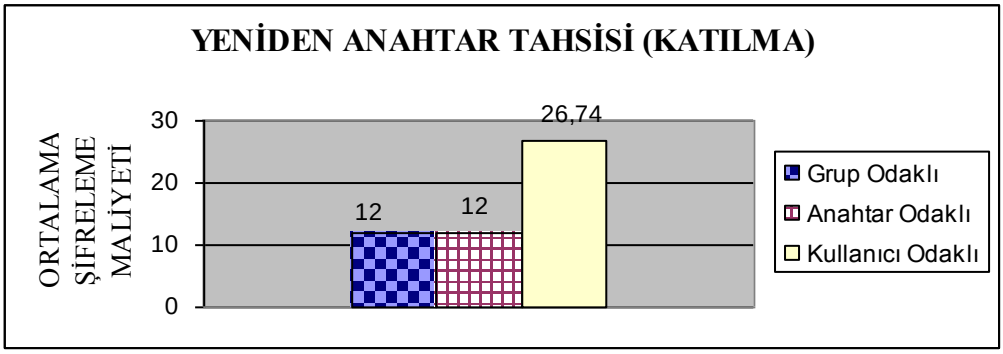


Şekil 7.10 Yeniden Anahtar Tahsisi Ortalama Şifreleme Maliyetleri (Katılma) (Kullanıcı Sayısı = 32)

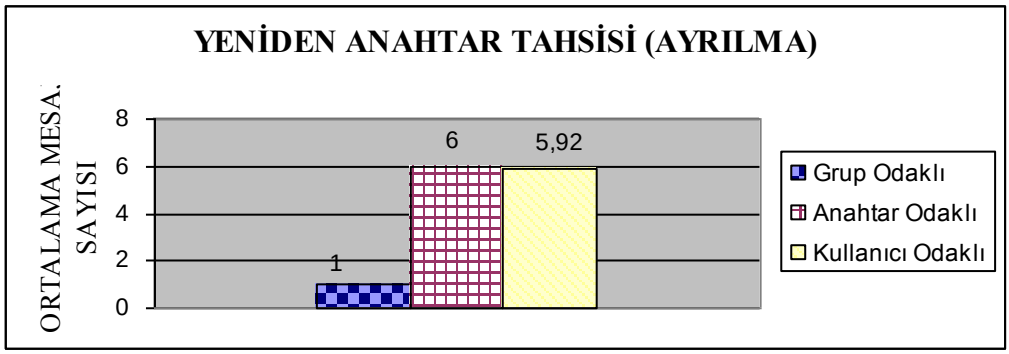
Şekil 7.11 ve Şekil 7.12'de, başlangıç kullanıcı sayısı 64 olan bir güvenli grupta, rasgele kullanıcı ayrılması ve katılmaların olması durumunda katılma başına elde edilen yeniden anahtar tahsisi ortalama mesaj sayıları ve ortalama şifreleme maliyetleri verilmiştir. Şekil 7.13 ve Şekil 7.14 ise ayrılma başına ortalama mesaj sayıları ve ortalama şifreleme maliyetini ifade etmektedir.



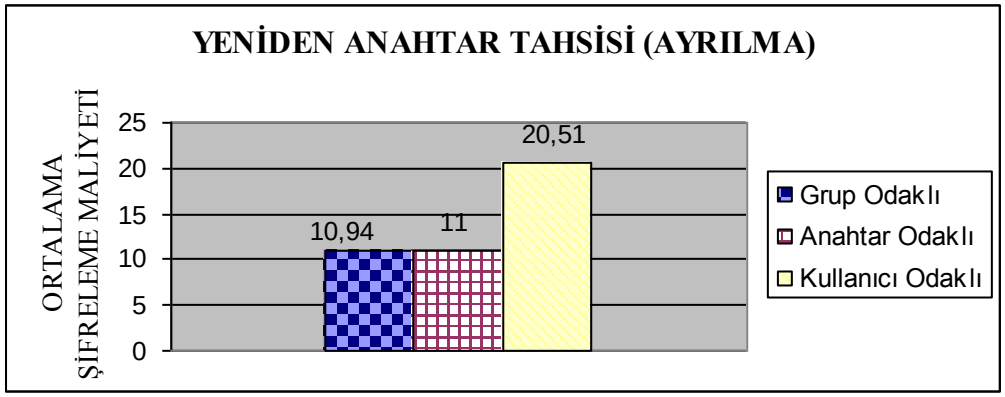
Şekil 7.11 Yeniden Anahtar Tahsisi Ortalama Mesaj Sayıları (Katılma) (Kullanıcı Sayısı = 64)



Şekil 7.12 Yeniden Anahtar Tahsisi Ortalama Şifreleme Maliyetleri (Katılma) (Kullanıcı Sayısı = 64)

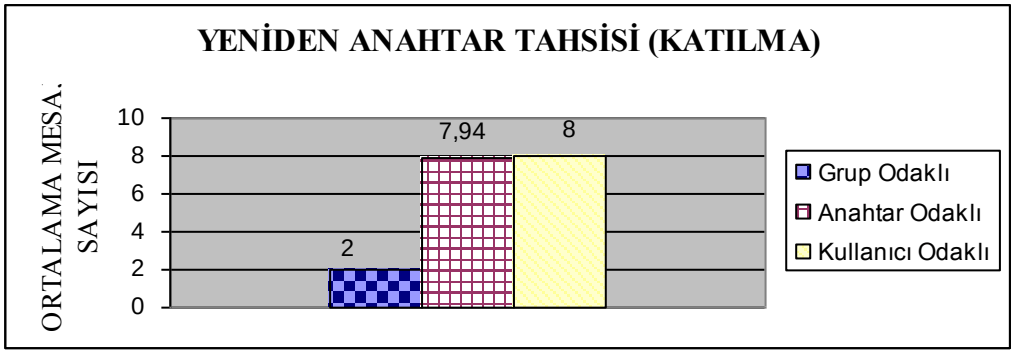


Şekil 7.13 Yeniden Anahtar Tahsisi Ortalama Mesaj Sayıları (Ayrılma) (Kullanıcı Sayısı = 64)

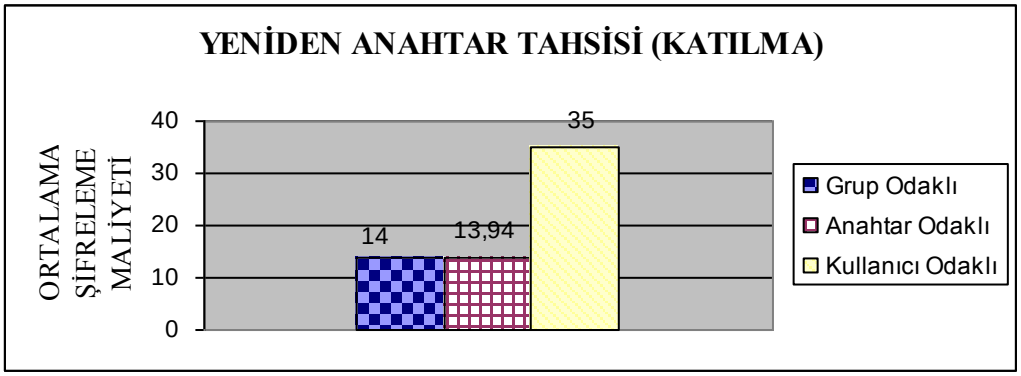


Şekil 7.14 Yeniden Anahtar Tahsisi Ortalama Şifreleme Maliyetleri (Ayrılma) (Kullanıcı Sayısı = 64)

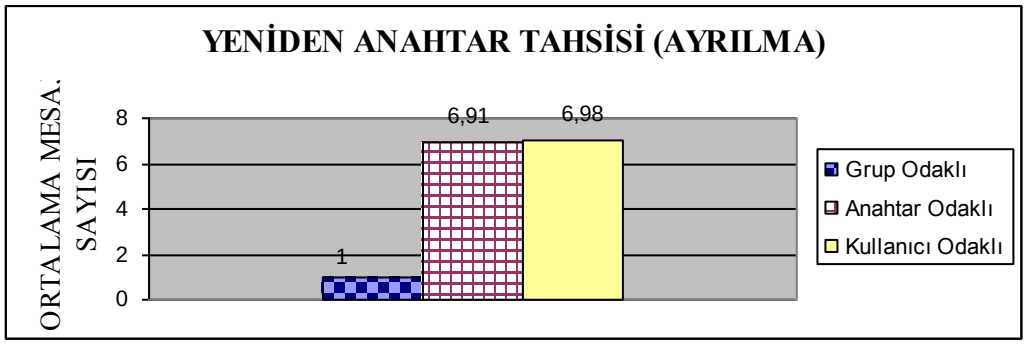
Şekil 7.15 ve Şekil 7.16'da, başlangıç kullanıcı sayısı 128 olan bir güvenli grupta, rasgele kullanıcı ayrılması ve katılmaların olması durumunda katılma başına elde edilen yeniden anahtar tahsisi ortalama mesaj sayıları ve ortalama şifreleme maliyetleri verilmiştir. Şekil 7.17 ve Şekil 7.18 ise ayrılma başına ortalama mesaj sayıları ve ortalama şifreleme maliyetini ifade etmektedir.



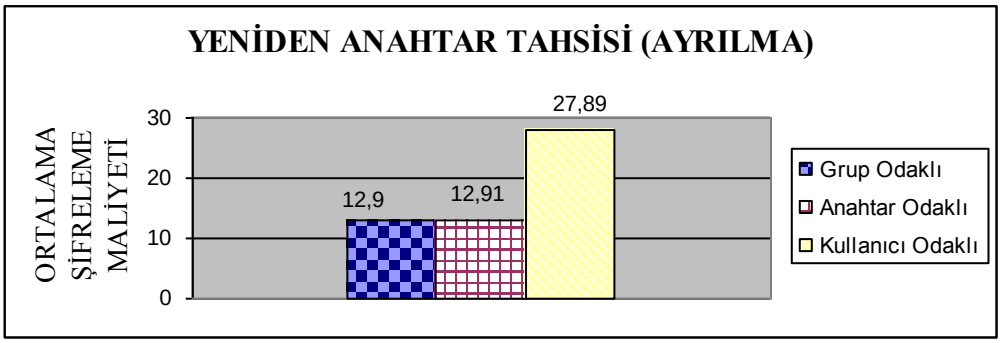
Şekil 7.15 Yeniden Anahtar Tahsisi Ortalama Mesaj Sayıları (Katılma) (Kullanıcı Sayısı = 128)



Şekil 7.16 Yeniden Anahtar Tahsisi Ortalama Şifreleme Maliyetleri (Katılma) (Kullanıcı Sayısı = 128)



Şekil 7.17 Yeniden Anahtar Tahsisi Ortalama Mesaj Sayıları (Ayrılma) (Kullanıcı Sayısı = 128)



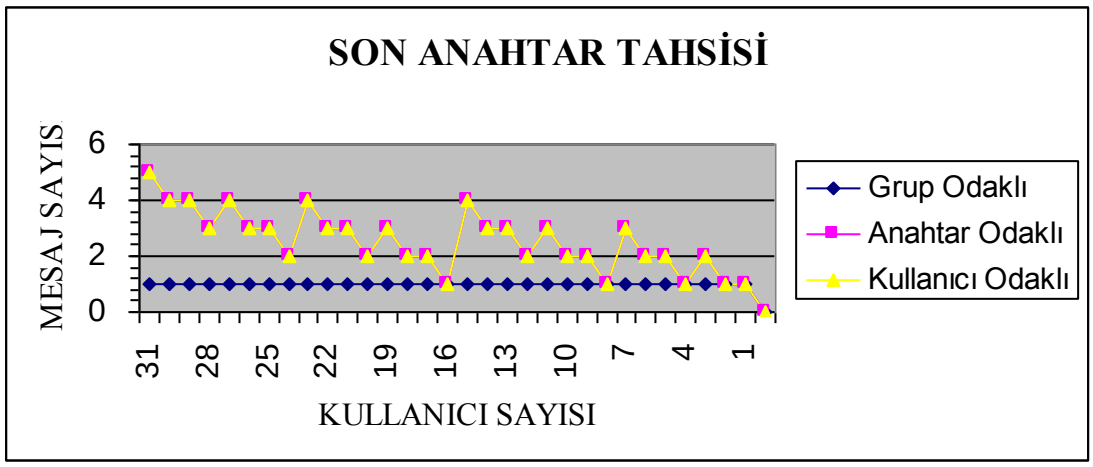
Şekil 7.18 Yeniden Anahtar Tahsisi Ortalama Şifreleme Maliyetleri (Ayrılma) (Kullanıcı Sayısı = 128)

Yukarıdaki grafikler incelendiğinde, her üç grup büyüklüğü için de, Grup-Odaklı Yeniden Anahtar Tahsisi yaklaşımının diğer yöntemlerden üstün olduğu ve avantajlı bir yeniden anahtar tahsisi imkanı sunduğu görülmektedir. Örneğin; kullanıcı sayısının 128 olduğu durumu ele alalım: Şekil 7.15'te her üç yöntemin yeniden anahtar tahsisi için kullanılması durumunda, katılma başına elde edilen ortalama mesaj sayıları; Şekil 7.17'de ise, ayrılma başına ortalama mesaj sayıları verilmiştir. Kullanıcı-Odaklı Yeniden Anahtar Tahsisi yaklaşımı kullanılması durumunda; ortalama mesaj sayısı katılma başına 7, ayrılma başına ise 6,98 olmaktadır. Anahtar-Odaklı Yeniden Anahtar Tahsisi yaklaşımı kullanıldığında ise; katılma başına ortalama mesaj sayısı 7,94, ayrılma başına ise 6,91 olmaktadır. Grup-Odaklı Yeniden Anahtar Tahsisi yaklaşımında ise; ortalama mesaj sayısı katılma başına 2, ayrılma başına ise 1 olmaktadır. Burada Kullanıcı-Odaklı Yeniden Anahtar Tahsisi yaklaşımının verdiği sonuçların diğer iki yöntemin oldukça üzerinde olduğu görülmektedir. Bunun yanı sıra, Grup-Odaklı Yeniden Anahtar Tahsisi yaklaşımı ise, mesaj sayısını katılma başına 2, ayrılma başına ise 1 mesaja düşürmektedir. Dolayısı ile, mesaj sayısı bakımından oldukça avantajlı bir çözüm olarak görülmektedir.

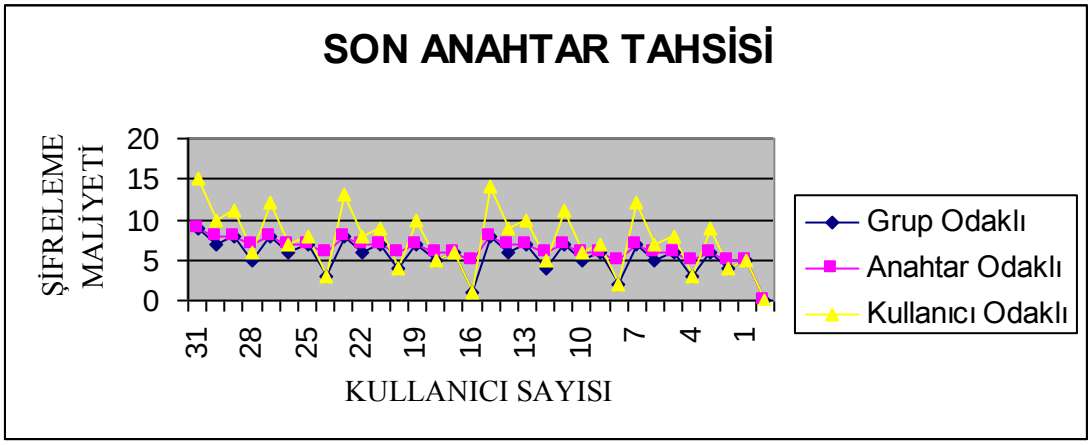
Şifreleme maliyetlerine bakacak olursak; Kullanıcı-Odaklı Yeniden Anahtar Tahsisi yaklaşımı kullanıldığında; katılma başına 35, ayrılma başına ise 27,89 anahtar uzunluğu olmaktadır. Anahtar-Odaklı Yeniden Anahtar Tahsisi yaklaşımı kullanıldığında ise; katılma başına 13,94, ayrılma başına ise 12,91 anahtar uzunluğu elde edilmektedir. Bu yaklaşımın Kullanıcı-Odaklı yaklaşıma oranla şifreleme maliyetini %60 oranında düşürdüğü gözlemlenmiştir. Bunun yanı sıra, Grup-Odaklı Yeniden Anahtar Tahsisi

Yaklaşımı da, Anahtar-Odaklı Yeniden Anahtar Tahsisi yaklaşımında elde edilen değerlere yaklaşık sonuçlar vermiştir. Katılma başına şifreleme maliyeti 14, ayrılma başına ise 12,9 olmaktadır. Buradan, Anahtar-Odaklı Yeniden Anahtar Tahsisi yaklaşımının, şifreleme maliyeti açısından avantajlı bir çözüm sunduğu görülmektedir. Ancak, mesaj sayısı ile birlikte ele alındığında, Grup-Odaklı Yeniden Anahtar Tahsisi yaklaşımının diğer yöntemlere nazaran daha avantajlı olduğu görülmektedir.

Bir diğer inceleme, başlangıçta 32 kullanıcının bulunduğu bir güvenli gruptan, kullanıcıların birer birer ayrılmaları durumunda, son anahtar tahsisi sırasında her üç yöntemin uygulanmasıyla elde edilen mesaj sayıları ve şifreleme maliyetleri üzerinde yapılabilir. Şekil 7.19'da kullanıcı sayısı ile mesaj sayıları arasındaki ilişki, Şekil 7.20'de ise şifreleme maliyeti ile kullanıcı sayısı arasındaki ilişki verilmiştir.

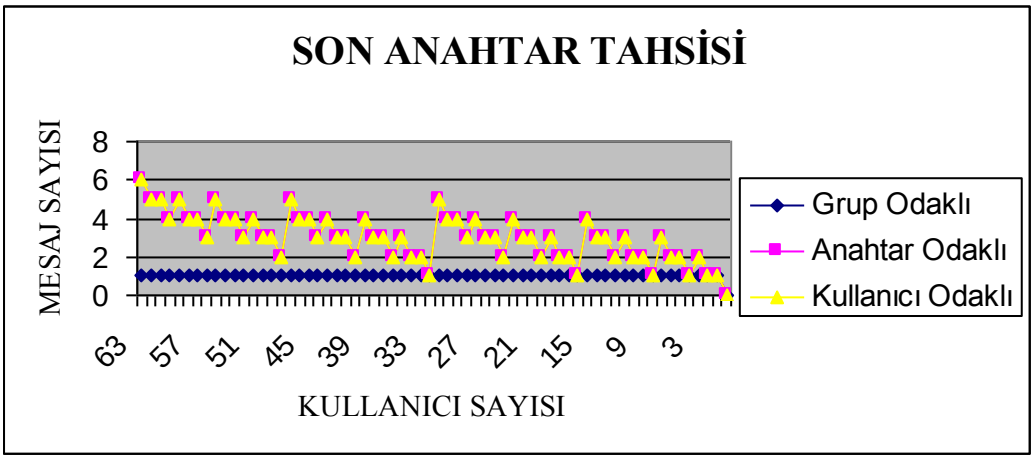


Şekil 7.19 Son Anahtar Tahsisi Mesaj Sayıları (Kullanıcı Sayısı = 32)

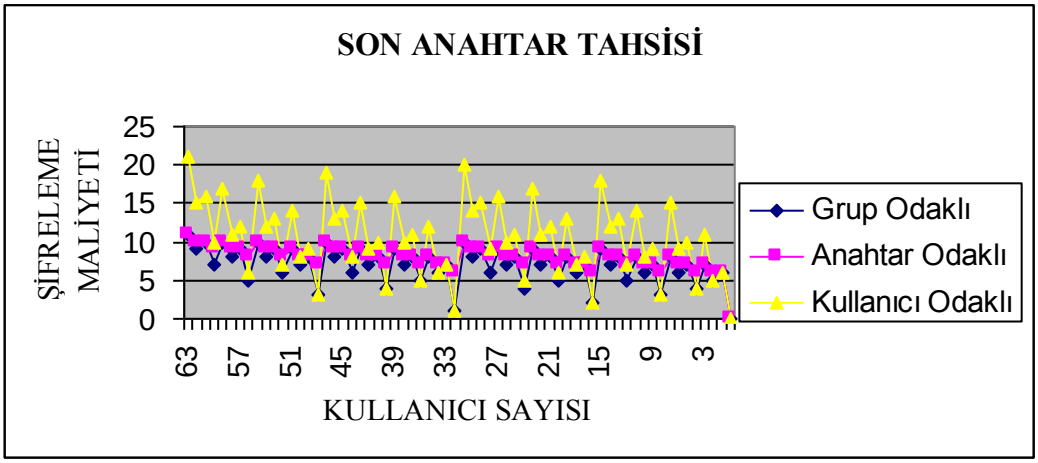


Şekil 7.20 Son Anahtar Tahsisi Şifreleme Maliyetleri (Kullanıcı Sayısı = 32)

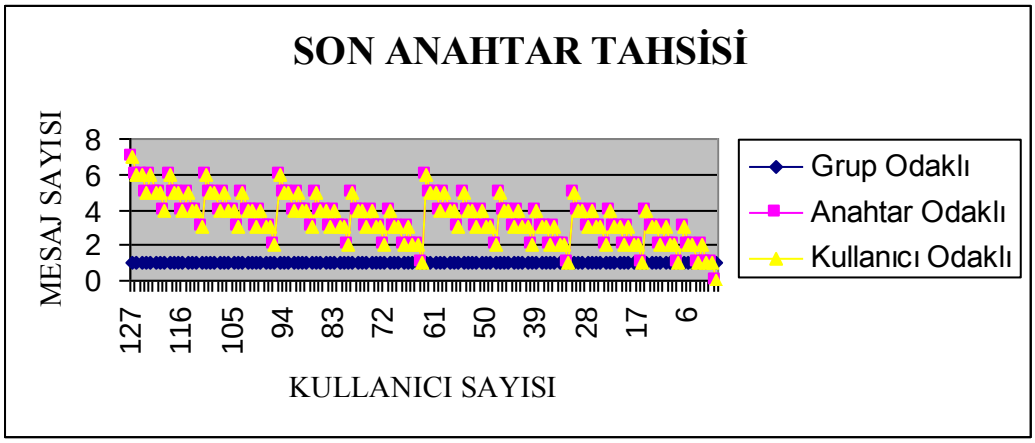
Başlangıçta, sırasıyla 64 ve 128 kullanıcının bulunduğu bir güvenli gruptan, kullanıcıların birer birer ayrılmaları durumunda, son anahtar tahsisi sırasında her üç yöntemin uygulanmasıyla elde edilen mesaj sayıları ve şifreleme maliyetleri ile kullanıcı sayıları arasındaki ilişkiler ise sırasıyla Şekil 7.21, Şekil 7.22, Şekil 7.23 ve Şekil7.24'de verilmiştir.



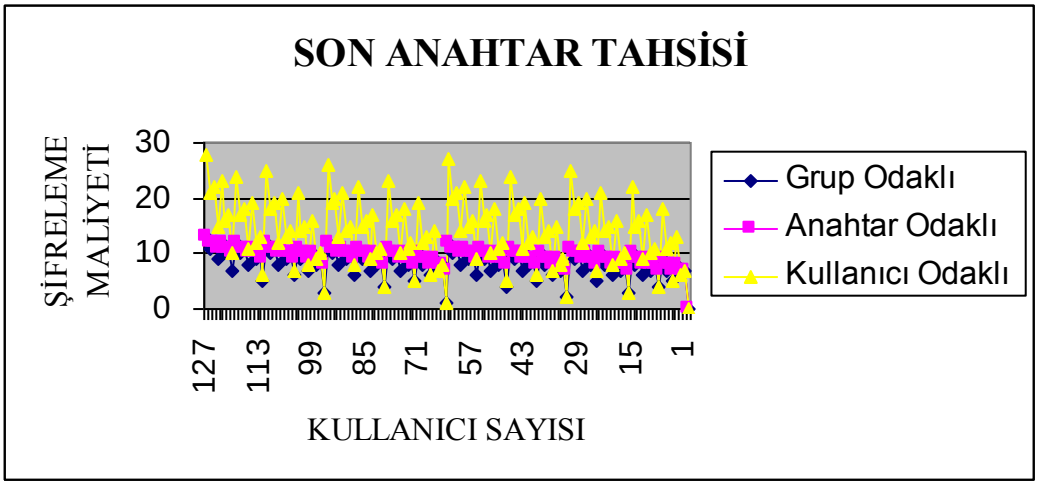
Şekil 7.21 Son Anahtar Tahsisi Mesaj Sayıları (Kullanıcı Sayısı = 64)



Şekil 7.22 Son Anahtar Tahsisi Şifreleme Maliyetleri (Kullanıcı Sayısı = 64)



Şekil 7.23 Son Anahtar Tahsisi Mesaj Sayıları (Kullanıcı Sayısı = 128)



Şekil 7.24 Son Anahtar Tahsisi Şifreleme Maliyetleri (Kullanıcı Sayısı = 128)

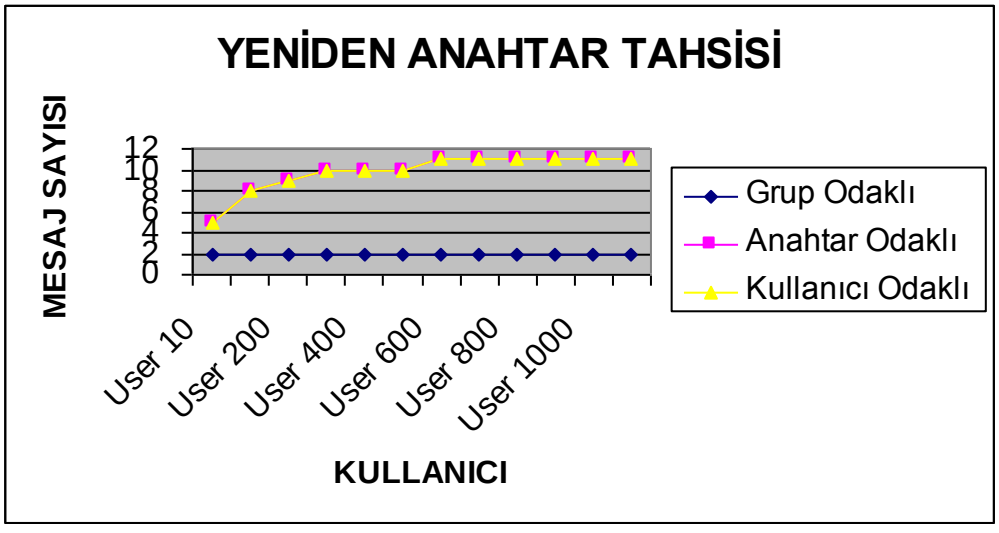
Yukarıdaki şekillerden de açıkça görüldüğü üzere, Grup-Odaklı Yeniden Anahtar Tahsisi yaklaşımı şifreleme maliyeti açısından Anahtar-Odaklı Yeniden Anahtar Tahsisi yaklaşımı ile benzerlik gösterse de, mesaj sayısını oldukça azalttığı için avantajlı bir son anahtar tahsisi imkanı sunmaktadır.

Son olarak yapılan analizde, kullanıcılar birer birer güvenli gruba katılarak, sonunda 1024 kullanıcıdan oluşan bir güvenli grup oluşturulması hedeflenmiştir. Bu katılmalar sırasında, belirli noktalarda gereken yeniden anahtar tahsisi mesaj sayıları ve şifreleme maliyetleri ölçümlenmiş ve bir tablo haline getirilmiştir. Bu değerler Tablo 7.1'de verilmiştir.

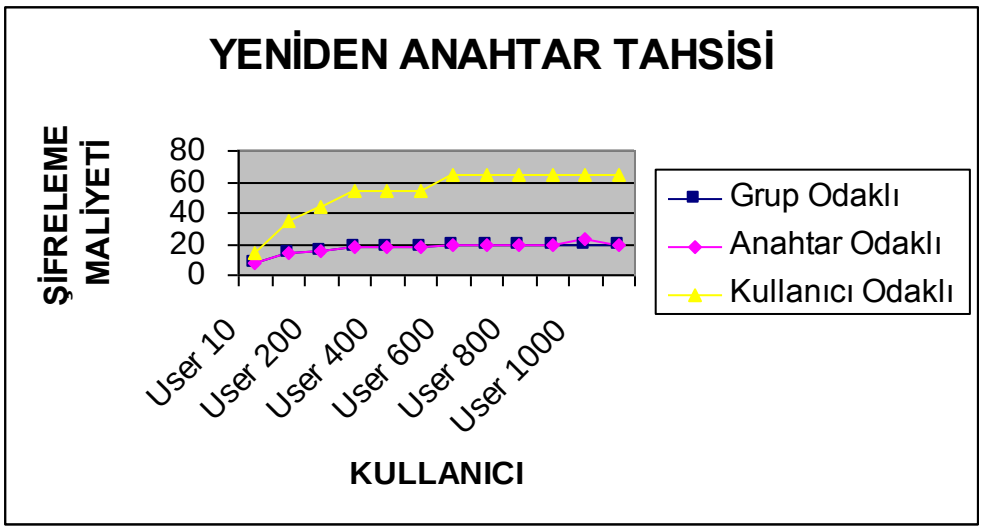
Tablo 7.1 Çeşitli kullanıcılar için mesaj sayıları ve şifreleme maliyetleri (Her üç yöntem için)

Kullanıcı	Grup Odaklı		Anahtar Odaklı		Kullanıcı Odaklı	
	Mesaj Sayısı	Şifreleme Maliyeti	Mesaj Sayısı	Şifreleme Maliyeti	Mesaj Sayısı	Şifreleme Maliyeti
10	2	8	5	8	5	14
100	2	14	8	14	8	35
200	2	16	9	16	9	44
300	2	18	10	18	10	54
400	2	18	10	18	10	54
500	2	18	10	18	10	54
600	2	20	11	20	11	65
700	2	20	11	20	11	65
800	2	20	11	20	11	65
900	2	20	11	20	11	65
1000	2	20	11	20	11	65
1024	2	20	11	20	11	65

Bu tabloda yer alan değerleri kullanarak "Mesaj Sayıları" ve "Şifreleme Maliyetleri" ile oluşturulan ilişki Şekil 7.25 ve Şekil 7.26'da verilmiştir.



Şekil 7.25 Yeniden Anahtar Tahsisi Mesaj Sayıları (Kullanıcı Sayısı = 1024)



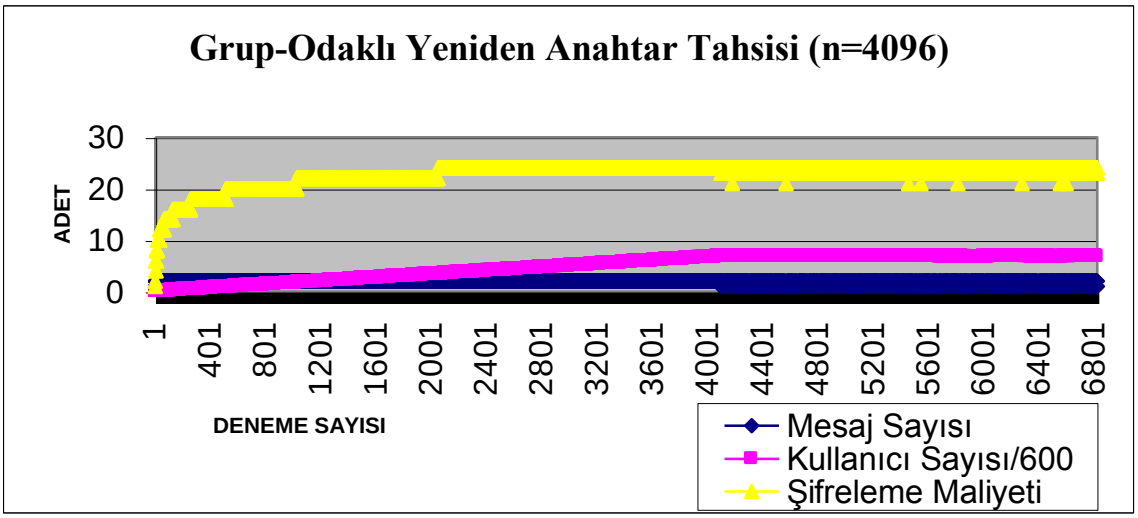
Şekil 7.26 Yeniden Anahtar Tahsisi Şifreleme Maliyetleri (Kullanıcı Sayısı = 1024)

Burada da görüldüğü gibi, Grup-Odaklı Yeniden Anahtar Tahsisi yaklaşımı şifreleme maliyeti açısından Anahtar-Odaklı Yeniden Anahtar Tahsisi yaklaşımı ile benzer sonuçlar verse de, mesaj sayısını oldukça azalttığı için diğer yöntemlerden daha avantajlıdır. Ancak, hangi yöntemin kullanılması gerektiği uygulamanın kullanılacağı ağ alt yapısına, sunucu ve istemcilerin donanım ve yazılım kapasitelerine göre değişiklik gösterebilir.

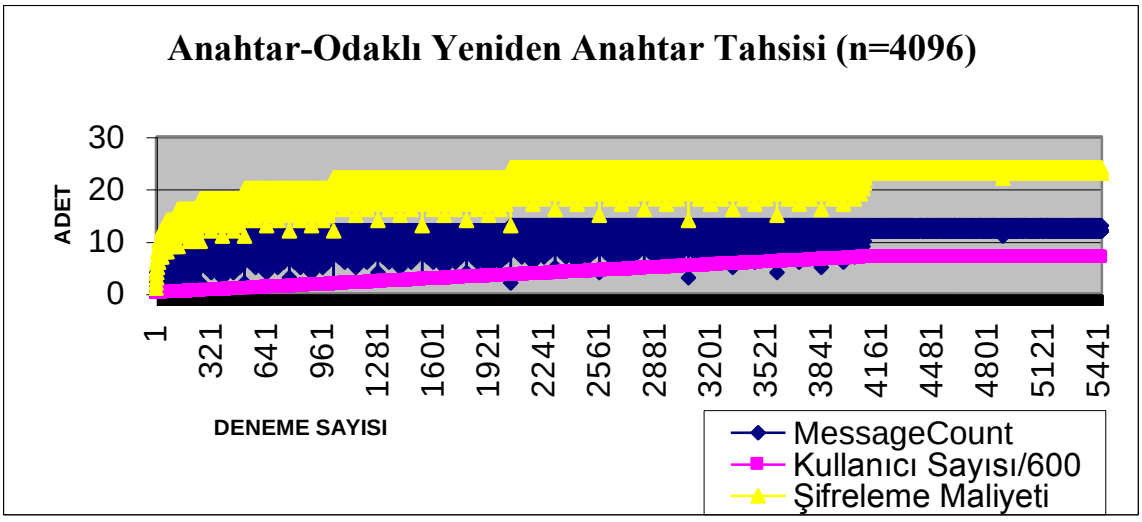
Yöntemlerin ölçeklenebilir olduğunu göstermek için çok fazla sayıda kullanıcı için de denemeler yapılmıştır. Bu denemelerden elde edilen sonuçlar ile elde edilen ilişkiler aşağıdaki gibidir.

İlk olarak kullanıcı sayısınnın 4092 olduğu durum ele alınmıştır. Başlangıçta sırası ile 4092 kullanıcının güvenli gruba girme istekleri kabul edilmiş ve 4092 kullanıcıdan oluşan bir ikili ağaç oluşturulmuştur. Ardından rasgele olarak kullanıcıların güvenli gruptan ayrılmaları ve yeni kullanıcıların katılması sağlanmıştır. Bu işlem 1000 kez gerçekleştirilmiştir. Bunun sonucunda her üç yeniden anahtar tahsisi yaklaşımı için elde edilen Kullanıcı Sayısı ile Deneme Sayısı arasındaki ilişki, Mesaj Sayısı ile Deneme Sayısı arasındaki ilişki ve Şifreleme Maliyeti ile Deneme Sayısı arasındaki ilişki Şekil 7.27, Şekil 7.28 ve Şekil 7.29'da verilmiştir.

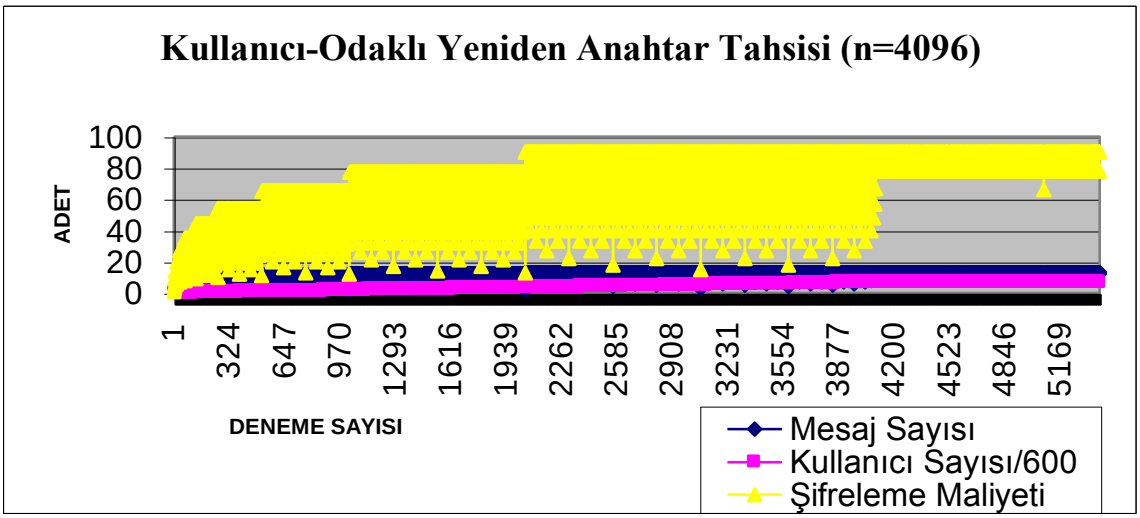
Bu ilişkilerin çizilmesi sırasında karşılaştırma yapılmasına imkan vermek amacıyla diğer değişkenler ile skalanın tutarlılığının sağlanabilmesi amacıyla Kullanıcı Sayısı gerçek değeri ile değil 600'e bölünmüş hali ile gösterilmektedir.



Şekil 7.27 Grup-Odaklı Yeniden Anahtar Tahsisi (Kullanıcı Sayısı = 4096)



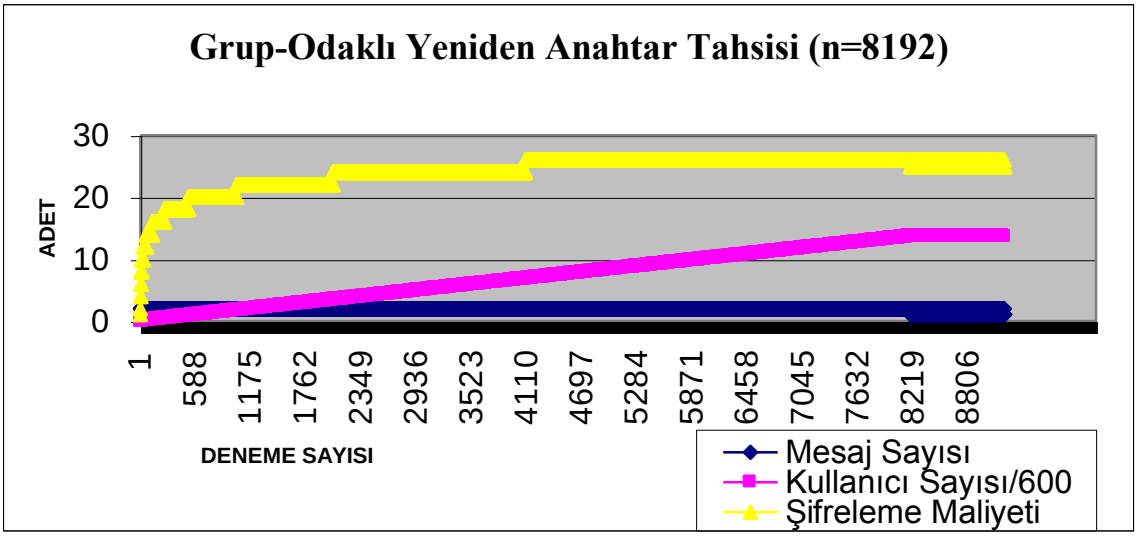
Şekil 7.28 Anahtar-Odaklı Yeniden Anahtar Tahsisi (Kullanıcı Sayısı = 4096)



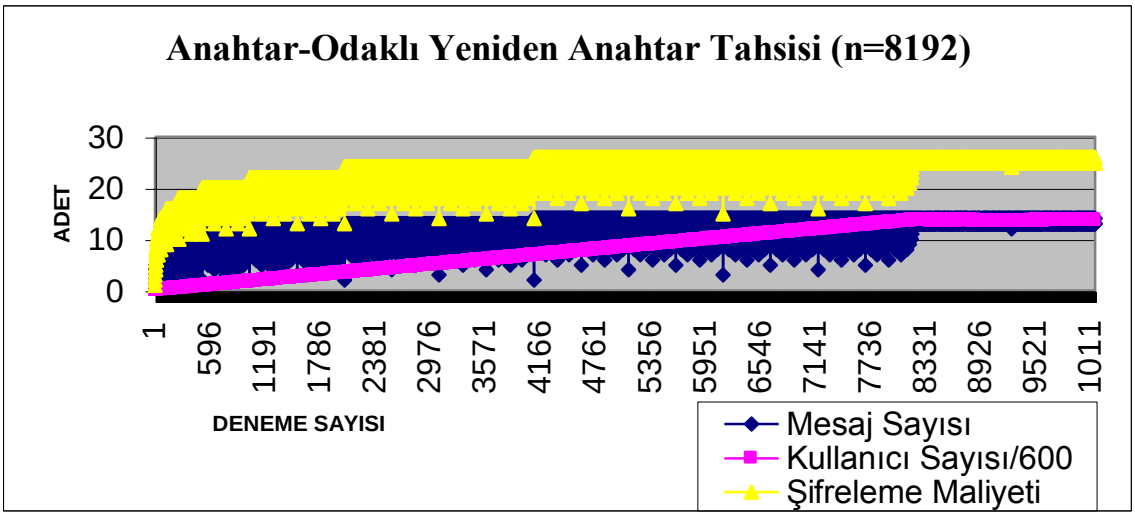
Şekil 7.29 Kullanıcı-Odaklı Yeniden Anahtar Tahsisi (Kullanıcı Sayısı = 4096)

Kullanıcı sayısı 4096'ya çıktığı halde gerekli yeniden anahtar tahsisi mesaj sayısı 2'yi geçmemektedir. Benzer şekilde Şifreleme Maliyeti ise en kötü durumda 25 olmaktadır.

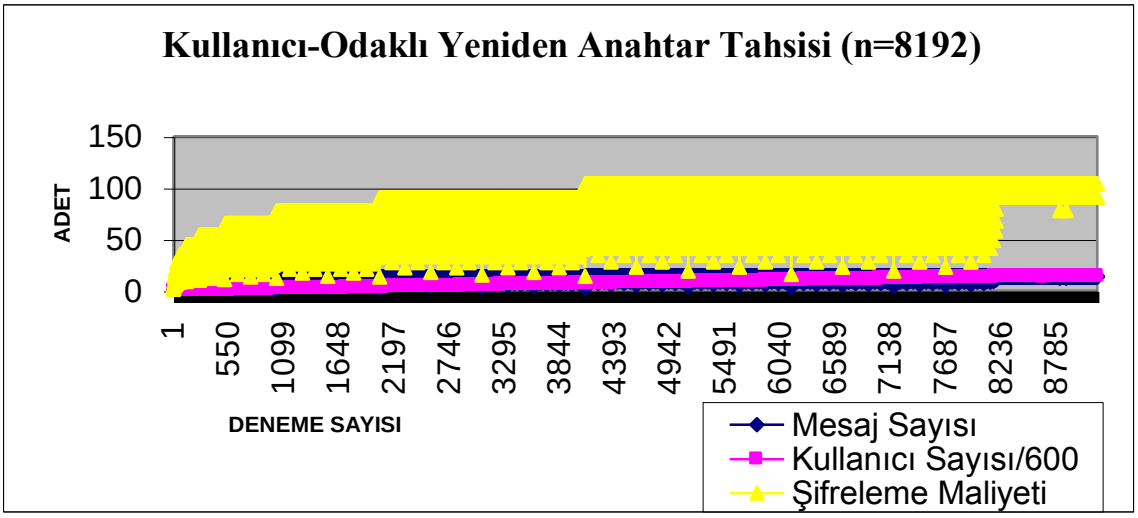
Kullanıcı sayısını biraz daha arttırarak 8192 yaptığımızı düşünelim. Her üç yeniden anahtar tahsisi yöntemi için Kullanıcı Sayısı ile Deneme Sayısı arasındaki ilişki, Mesaj Sayısı ile Deneme Sayısındaki ilişki ve Şifreleme Maliyeti ile Deneme Sayısı arasındaki ilişki sırası ile Şekil 7.30, Şekil 7.31 ve Şekil 7.32'de verilmiştir.



Şekil 7.30 Grup-Odaklı Yeniden Anahtar Tahsisi (Kullanıcı Sayısı = 8192)

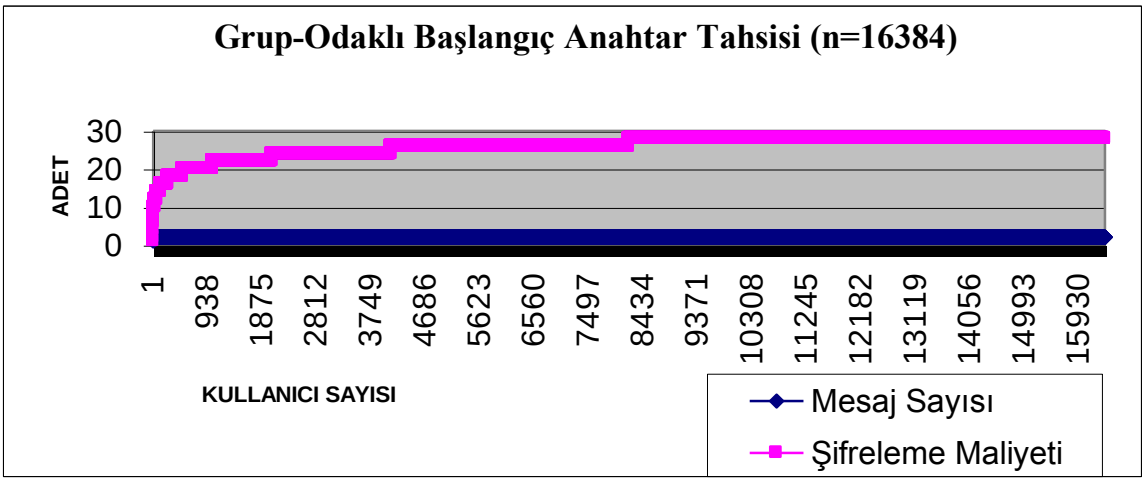


Şekil 7.31 Anahtar-Odaklı Yeniden Anahtar Tahsisi (Kullanıcı Sayısı = 8192)

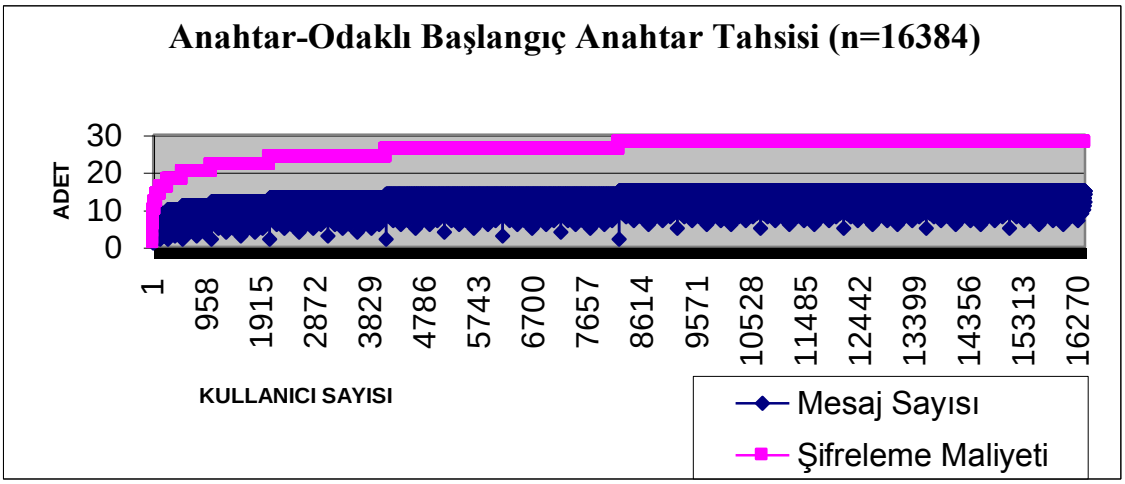


Şekil 7.32 Kullanıcı-Odaklı Yeniden Anahtar Tahsisi (Kullanıcı Sayısı = 8192)

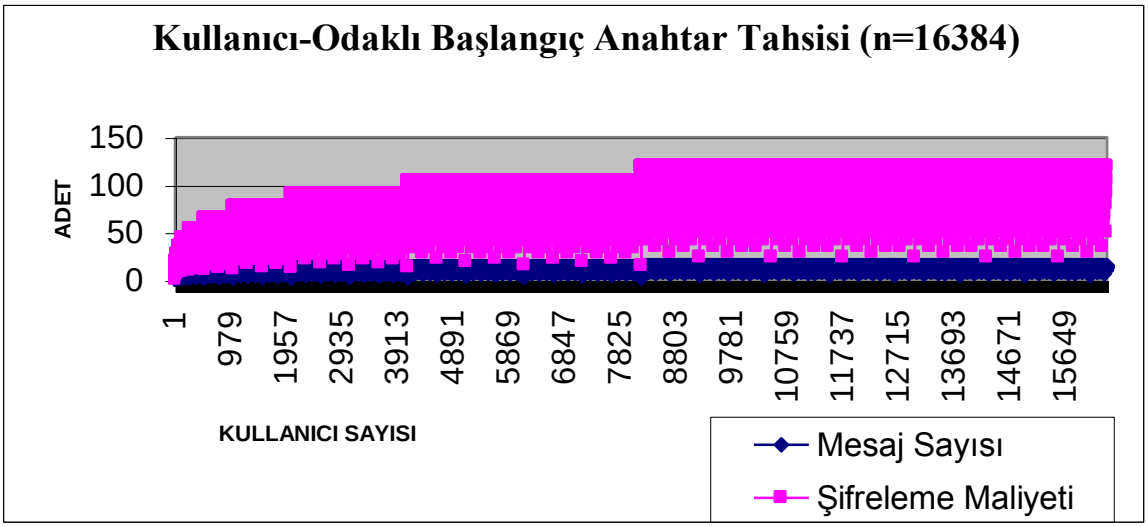
Son olarak 16384 kullanıcı icin deneme yapılmıştır. Bu denemede başlangıç anahtar tahsisinin incelenmesi sağlanmıştır. Burada elde edilen sonuçlar yukarıda verilen sonuçlar ile paralellik göstermektedir. Her üç yeniden anahtar tahsisi yaklaşımı için elde edilen Mesaj Sayısı ile Kullanıcı Sayısı ve Şifreleme Maliyeti ile Kullanıcı Sayısı ilişkiler Şekil 7.33, Şekil 7.34 ve Şekil 7.35'de gösterilmiştir.



Şekil 7.33 Grup-Odaklı Başlangıç Anahtar Tahsisi (Kullanıcı Sayısı = 16384)



Şekil 7.34 Anahtar-Odaklı Başlangıç Anahtar Tahsisi (Kullanıcı Sayısı = 16384)



Şekil 7.35 Kullanıcı-Odaklı Başlangıç Anahtar Tahsisi (Kullanıcı Sayısı = 16384)

Burada da görüldüğü üzere özellikle sunucu tarafında Grup-Odaklı yaklaşım oldukça iyi sonuçlar vermekte ve diğer yöntemler de büyük kullanıcı sayıları için bile ölçeklenebilir olduklarını verilen ilişkiler ile göstermektedir. Kullanıcı sayısını ikinin derecesi şeklinde arttırdığımızda Mesaj Sayısı ve Şifreleme Maliyeti değerleri Kullanıcı Sayısının logaritması ile orantılı olarak değişiklik göstermektedir. Bu da ölçeklenebilirliğin diğer bir yansımasıdır.

7.2 ÇOĞA GÖNDERİM ANAHTAR YÖNETİMİ YAZILIMI

Çalışma kapsamında, yöntemlerin birbirleri ile karşılaştırılabilmeleri için gerekli sonuçların elde edilebilmesini sağlayan benzetim yazılımının yanı sıra, yöntemlerin fiziksel olarak uygulandığı ve "Çoğa Gönderim Anahtar Yönetimi Uygulaması (Multicast Key Management Application-MKMA)" olarak adlandırılan bir yazılım geliştirilmiştir. Böylelikle, yöntemlerin fiziksel olarak uygulanabilirliği gözlemlenmiştir.

Yazılım, "Microsoft Visual Studio" içerisinde yer alan "Microsoft Visual C++" geliştirme ortamı kullanılarak yazılmıştır.

Programın geliştirilmesi aşamasında ikili ağaçların kullanılması tercih edilmiştir. Bu çalışma kapsamında geliştirilen yazılım açısından, anahtar ağacının derecesi belirleyici bir özellik taşımamaktadır. Bu nedenle, ikili ağaçların kullanılmasının tercih edilmesinin yazılım açısından bir etkisi yoktur.

Anahtar graflarının oluşturulması için "*pointer"* ve *"linked list"* yapısından yararlanılmıştır. Bu yapılır aracılığı ile ikili bir ağacın kurulması ve bakımı sağlanmıştır. Anahtar ağacının her bir düğümünün oluşturulması için aşağıdaki "structure" yapısı kullanılmıştır:

```
struct node
{       char    *no;
        char    *userno;
        char    key[17];
        char    oldkey[17];
        SOCKET      sock;
        struct  node *left;
        struct  node *right;
};
```

Burada “no” alanı bu düğüme yer alan anahtara verilen ID’yi ifade etmektedir. Daha öncede bahsedildiği gibi *keyno* değeri, kullanılan anahtarın diğerlerinden ayırt edilmesi ve İstemci ile Anahtar Sunucu arasında değişen anahtarlar üzerinde anlaşılabilmesini sağlamak için kullanılır. “Userno” alanı, düğüm eğer bir yaprak ise dolu olacak ve o yaprakta yer alan kullanıcının “KullanıcıID” değerini içerecektir. “KullanıcıID” herhangi bir istek geldiğinde bu isteğin hangi kullanıcıdan geldiğinin anlaşılması için kullanılan, Anahtar Sunucusu ve İstemci tarafından bilinen değerdir. “OldKey” ve “Key” alanlarında sırası ile bu düğümde yer alan anahtarın eski ve yeni değerleri olacaktır. “sock” alanı ise düğüm bir yaprak ise dolu olacak ve bu yaprakta yer alan kullanıcı ile Anahtar Sunucusu arasında iletişimin sağlandığı *socket* tanımlayıcısını içerecektir. Bunun haricinde kalan “left” ve “right” alanları ağaçta bu düğümün solunda ve sağında yer alan dalların işaretçileridir. Düğüm bir yaprak ise “left” ve “rigth” alanları NULL değerini taşır.

İletişim için “Windows Socket” leri kullanılmaktadır. Progam içerisinde Wsock32.lib kütüphane dosyası içerisinde yer alan komutlar kullanılır. “socket” komutu ile yaratılacak olan socket’in tipi “SOCK_STREAM” olarak seçilmelidir. Ardından “bind” komutu aracılığı ile verilen IP adresi ve port numarası ile yaratılan socketin ilişkilendirilmesi sağlanır. “listen” komutu ile bu sockete yapılacak olan başvuruların dinlenmesi sağlanmış olacaktır. Daha sonra yaratılacak olan bir “thread” içerisinde “Anahtar Sunucusu”nun istemcilerden gelen bağlantı isteklerini “accept” komutu ile kabul etmesi sağlanmış olur.

Bir istemci bağlantısı kurulduktan sonra istemci ile Anahtar Sunucusu arasındaki socket bağlantısında meydana gelecek “Read” ve “Close” olaylarının asenkron olarak dinlenmesi sağlanır. “Read” olayı meydana geldiğinde istemci tarafından yollanan mesaj okunur. Eğer bu bir katılma isteği ise anahtar ağacı üzerinde gerekli eklemeler ve değişiklikler yapılır ve yeni kullanıcı ile birlikte grup üyelerine güvenli olarak yeniden tahsis edilmesi sağlanır. Eğer gelen mesaj bir ayrılma mesajı ise, bu taktirde ayrılma isteğinde bulunan kullanıcının güvenli gruptan ayrılması sağlanır ve ayrılan kullanıcının sahip olduğu anahtarların değiştirilmesi ve tüm değişen anahtarlar ile geride kalan güvenli gruba yeniden anahtar tahsisi edilmesi gerçekleştirilir.

“Close” olayı oluşur ise istemcinin bağlantıyı kopardığı anlaşılır ve yine bu kullanıcının güvenli gruptan çıkarılması işlemi gerçekleştirilir.

MKMA yazılımı kapsamında üç ayrı program geliştirilmiştir. Bunlar sırası ile; Veri Sunucu (Data Server), Anahtar Sunucusu (Key Server) ve İstemci (Client) programıdır.

7.2.1 ANAHTAR SUNUCUSU

MKMA yazılımı kapsamında gerçekleştirilen modüllerden birisi de “Anahtar Sunucusu” dur. Anahtar Sunucusu, çoğa gönderim iletişimini güvenli hale getirmek için kullanılan grup anahtarının ve yeniden anahtar tahsisi mesajlarının şifrelenmesinde kullanılan şifreleme anahtarlarının tutulduğu anahtar grafının oluşturulmasından, yeni anahtarların yaratılmasından, yeni anahtar düğümlerinin ağaca eklenmesinden, ağaçtan düğüm çıkarılmasından ve yeniden anahtar tahsisi mesajlarının ilgili yönteme göre üretilip yollanmasından sorumludur.

Anahtarlar 16 karakter uzunluğunda alfanümerik değişkenlerdir. Anahtarların oluşturulması aşamasında aşağıda verilen algoritmadan yararlanılır:

```
#define getrandom(min, max) ((rand() % (int) (((max) + 1) - (min))) + (min))

void GenerateKey(char pKey[17])
{
        int i,j;

        memset(pKey,'\0',17);

        srand( (unsigned)time( NULL ) );

        for(i=0;i<16;i++) {
                j = getrandom(48,122);
                pKey[i] = toascii(j);
        }
        pKey[16] = '\0';
}
```

Anahtarın oluşturulması için 48 ve 122 arasındaki sayılardan rastgele bir değer üretilmesi sağlanır. Ardından bu sayının ascii kod tablosunda karşı geldiği karakter "toascii" fonksiyonu aracılığı ile bulunarak anahtar "*string"* ine eklenir. Bu işlem 16 kez yapılarak sonuçta rasgele bir anahtar yaratılmış olacaktır.

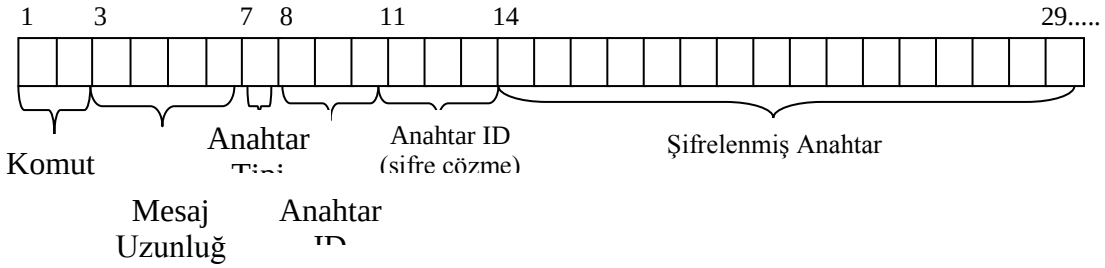
Programın çalıştırılması ile "Anahtar Sunucusu"nun çalışacağı "Port Numarası"nın kullanıcı tarafından girilmesi istenir. Böylelikle, Anahtar Sunucusu çalıştığında, kendisine gelen katılma ve ayrılma isteklerini bu port aracılığı ile dinleyebilecektir. Port numarasının yanı sıra, kullanıcının Anahtar Sunucusunun çalışması sırasında kullanılacak olan yeniden anahtar tahsisi stratejisinin de Anahtar-Odaklı ve Grup-Odaklı yaklaşımlar arasından seçilmesi istenir.

"Anahtar Sunucusu"nun çalışması ile, kendisine gelen katılma ve ayrılma isteklerini dinlemesi için bir "thread" yaratılır. Bu "thread" seçilmiş olan port numarasına gelen mesajları dinler.

Anahtar Sunucusu çalışması sırasında belli bir aşamada kapatılabilir. Bu durumda, ilgili istemcilere ve veri sunucusuna Anahtar Sunucusunun kapandığına dair bir mesaj iletilir.

Anahtar Sunucusuna bir kullanıcının katılma isteği geldiğinde, öncelikle bu kullanıcının şahsi anahtarı olacak bir anahtarın yaratılması sağlanır. Ardından, anahtar ağacının boş olan ilk yaprağına bu kullanıcı için yaratılan şahsi anahtar düğümü eklenir. Bu kullanıcının eklenmesi ile anahtar ağaç grafında değiştirilmesi gereken anahtarlar işaretlenir. Ardından, değiştirilmesi gereken anahtarlar için yeni anahtarlar üretilir ve seçilmiş olan yeniden anahtar tahsisi stratejisine göre, bu değişiklikten haberdar olması gereken kullanıcılara ilgili yeniden anahtar tahsisi mesajlarının yollanması sağlanır.

Bir yeniden anahtar tahsisi mesajı içerisinde birden fazla değişen anahtar yollanabileceği için bir yeniden anahtar tahsisi mesajı alt birimlere bölünmüştür. Her bir alt birim 23 karakterden oluşur. Alt birimlerinin birleşiminden oluşan yeniden anahtar tahsisi mesajının başına bu mesajın hangi yeniden anahtar tahsisi yöntemine göre hazırlandığı ve içerisinde kaç adet alt birimin yer aldığını ifade eden bir başlık eklenir. Bu şekilde oluturulan bir yeniden anahtar tahsisi mesaj yapısı aşağıda verilmektedir:



Şekil 7.36 Yeniden anahtar tahsisi mesaj yapısı

Şekil 7.36'de verilen mesaj yapısında, "Komut" alanında yer alabilecek değerler şunlardır:

Tablo 7.2 Yeniden Anahtar Tahsisi mesaj yapısı komutları

Katılma	~G	Grup-Odaklı Yeniden Anahtar Tahsisi
	^G	Anahtar-Odaklı Yeniden Anahtar Tahsisi (Grup Anahtarı)
	^K	Anahtar-Odaklı Yeniden Anahtar Tahsisi (Anahtar Şifreleme Anahtarı)
	^^	Anahtar-Odaklı Yeniden Anahtar Tahsisi (Katılma isteğinde bulunan kullanıcı)
Ayrılma	~L	Grup-Odaklı Yeniden Anahtar Tahsisi
	^L	Anahtar-Odaklı Yeniden Anahtar Tahsisi

Sadece Anahtar-Odaklı Yeniden Anahtar Tahsisi (Katılma) işleminde katılma isteğinde bulunan kullanıcı haricinde diğer kullanıcılara değişen her anahtar ayrı ayrı mesajlar içerisinde yollanır. Bu durumda oluşturulan yeniden anahtar tahsisi mesajının başlığında "Mesaj Uzunluğu" alanı yer almayacak ve "Komut" alanında da tek karakter olacaktır. Bu değerler yukarıdaki tabloda ifade edilmiştir. Katılma isteğinde bulunan kullanıcı için Şekil 7.36'de verilen yeniden anahtar tahsisi mesaj yapısı geçerlidir.

Şekil 7.36'de verilen mesaj yapısında "Anahtar ID" alanı, yeniden Anahtar Tahsisi mesajı içerisinde yollanan anahtarın ID'sini belirtmek için kullanılır. Sunucu ile istemci arasında değiş tokuş edilen anahtarların anlaşılabilmesi için, anahtarların her birine bir ID atanmıştır. Yeniden anahtar tahsisi mesajının başına bu ID yerleştirilir. Böylelikle, istemci kendisine gelen mesajın içerisinde hangi anahtarın yer aldığını anlayabilecektir.

“Anahtar ID (şifre çözme)” alanı, yollanan anahtarın hangi anahtar ile şifrelendiğini belirtmek için kullanılır. Böylelikle, istemci kendi anahtar kümesi içerisinde yer alan bu alanda ifade edilen ID’ye sahip anahtarı kullanarak, yeni gelen anahtarın şifresinin çözülmesini sağlar.

“Anahtar Tipi” alanı şifrelenerek yollanmış olan anahtarın Grup Anahtarı mı yoksa bir anahtar şifreleme anahtarı mı olduğunu ifade etmek için kullanılan alandır. Bu alanda yer alan değer “G” ise mesaj içerisinde Grup Anahtarı, “K” ise de bir anahtar şifreleme anahtarı yer alıyor demektir.

“Mesaj uzunluğu” alanı ise bu yeniden anahtar tahsisi mesajı içerisinde alt birimlerde yer alan karakter sayısını ifade eder. Bu değer 23 ve katlarından oluşmalıdır.

“Şifrelenmiş Anahtar” alanında ise, yeniden anahtar tahsisi mesajı içerisinde yollanan değiştirilmiş ve şifrelenmiş anahtar yer alacaktır.

Anahtar Sunucusu, grup anahtarının değişmesi durumunda, bu değişiklikten “Veri Sunucusu”nu yollayacağı bir değişiklik mesajı ile haberdar eder. “Veri Sunucusu”, grup anahtar değişikliğini bildiren mesajı aldıktan sonra, gruba yollayacağı mesajları bu anahtarı kullanarak şifreler.

Anahtar Sunucusu çalıştırıldığında “Veri Sunucusu”nun IP adresi ile iletişim kuracağı “Port No”sunun da girilmesini ister. Grup anahtarının her değişikliğinde bu değişiklikten Veri Sunucusunu bu port üzerinden iletişim kurarak haberdar eder.

7.2.2 VERİ SUNUCUSU

“Veri Sunucusu”, güvenli grubun oluşturulmasından sonra, bu gruba, grup anahtarının son hali ile şifrelenmiş mesajların yollanmasını sağlayan modüldür. Veri sunucusu çalıştırıldığında, veri yollayacağı “Sınıf D” tipi çoğa gönderim adresinin ve port numarasının kullanıcı tarafından girilmesi istenir. “Veri Sunucusu”, çalıştırıldıktan sonra girilen metinleri grup anahtarı ile şifreleyerek çoğa gönderim adresine yollar.

7.2.3 İSTEMCİ

“İstemci” modülü “Çoğa Gönderim Anahtar Yönetimi Uygulaması”nın son parçasıdır. Güvenli çoğa gönderim grubuna katılmak isteyen kullanıcılar bu program aracılığı ile

istekte bulunurlar. Programın çalıştırılması ile, ilk olarak, güvenli çoğa gönderim grubuna katılmak için istekte bulunulur. Bu istek için anahtar sunucusunun IP adresinin ya da sunucu isminin ve port numarasının kullanıcı tarafından girilmesi istenir. Katılma isteğinin anahtar sunucusu tarafından kabul edilmesinin ardından, kullanıcının ihtiyaç duyduğu anahtarlar ve grup anahtarı yollanır. Ardından, istemci, çoğa gönderim adresinden veri alabilmek için çoğa gönderim oturumuna katılma isteğini bildirir. Bu isteğin yollanabilmesi için çoğa gönderim IP adresinin ve port numarasının verilmesi gerekir. Bu istek de onaylandıktan sonra, istemci, güvenli çoğa gönderim grubundan veri almaya hazır hale gelmiştir. "Veri Sunucusu" tarafından yollanan mesajlar istemci tarafından alınır ve elinde bulundurduğu grup anahtarı ile şifresi çözülerek işlenir.

"İstemci" güvenli gruptan ayrılmak istediğinde, "Anahtar Sunucusu"na çoğa gönderim oturumundan ayrılmak istediğini bildirilen bir mesaj yollar. Üyelerden birinin ayrılma isteğini alan anahtar sunucusu gerekli yeniden anahtar tahsisi ları yapar ve yeni grup anahtarını veri sunucusuna bildirir. Böylelikle, bundan sonra yapılacak olan iletişime ayrılma isteğinde bulunan istemcinin erişimi engellenir. İstemci yeniden gruba katılmak istediğinde, bu işlemlere baştan başlaması gereklidir.

İstemci programında, yeniden anahtar tahsisi mesajı için uygulanan algoritma şu şekildedir:

Katılma yeniden anahtar tahsisi mesajı alındığında, bu mesaj içerisinde "Anahtar ID (şifre çözme)" alanında yer alan değerin kendi anahtar kümesi içerisinde yer alıp almadığına bakar. Eğer bu anahtar kendi anahtar kümesi içerisinde yer almıyor ise, bu mesaj kendisini ilgilendirmediği için, gelen mesajı atar. Eğer "Anahtar ID (şifre çözme)" alanında yer alan anahtarı kendi anahtar kümesi içerisinde bulur ise, gelen şifreli mesajın şifresini bu anahtar ile çözer. İstemci, ardından, "Anahtar ID" alanında yer alan değeri kendi anahtar kümesi içerisinde arar. Eğer bulamaz ise, şifresini çözdüğü anahtarı yeni bir anahtar olarak anahtar kümesi içerisine ekler. Ancak, kendi anahtar kümesi içerisinde bu anahtarı bulur ise, yeni gelen anahtar ile bu anahtarı değiştirir. Böylelikle, kendi anahtar kümesini güncellemiş olur.

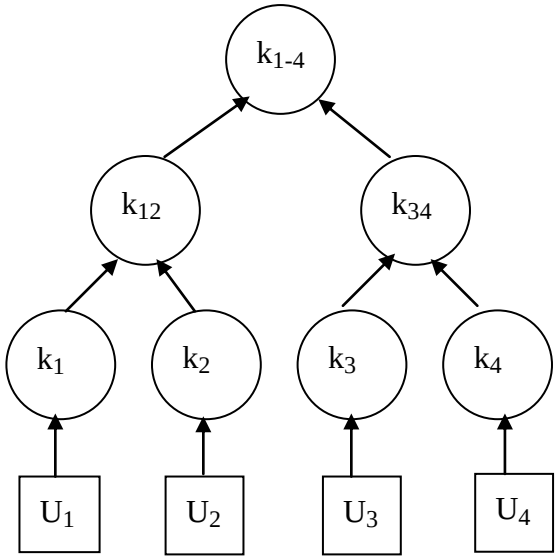
Ayrılma yeniden anahtar tahsisi mesajı alındığında, "Anahtar ID (şifre çözme)" alanında ID'si yer alan anahtarın kendi anahtar kümesi içerisinde yer alıp almadığına bakar. Bu

anahtarı bulamaz ise, bu anahtar ile şifrelenmiş olan yeniden anahtar tahsisi mesajını atar. Anahtarı bulması durumunda, "Anahtar ID" alanında yer alan değere bakar. Bu anahtarı da kendi anahtar kümesi içerisinde bulamaz ise, yeniden anahtar tahsisi mesajını atacaktır. Eğer anahtar kümesi içerisinde bu ID'ye sahip olan bir anahtar var ise, gelen mesajın şifresini çözer ve yeni gelen anahtar ile eskisini değiştirir.

Grup-odaklı yeniden Anahtar Tahsisi yaklaşımında, aynı mesaj içerisinde birden fazla yeni anahtar bulunabileceği ve bu anahtarlardan bazılarının aynı yeniden anahtar tahsisi mesajı içerisinde yer alan anahtarlar ile şifrelenebileceği göz önünde bulundurulmalı ve öncelikle "Anahtar ID (şifre çözme)" alanında yer alan değerin, aynı yeniden anahtar tahsisi mesajı içerisinde yer alan anahtarlardan olmadığı mesajdan başlanmalıdır.

Buraya kadar anlattıklarımızı bir örnek ile açıklamak yararlı olacaktır:

Dört kullanıcının oluşturduğu bir ikili anahtar ağacı ele alınsın. Şekil 7.37'da bu ağaç gösterilmektedir.



Şekil 7.37 Dört kullanıcıdan oluşan anahtar ağaç grafı

Bu grafta yer alan U_4 kullanıcısı güvenli gruptan ayrılmak istediğinde, anahtar-odaklı yeniden anahtar tahsisi ve grup-odaklı yeniden Anahtar Tahsisi yaklaşımlarına göre oluşturulacak olan yeniden anahtar tahsisi mesajları şu şekildedir:

Anahtar-Odaklı Yeniden Anahtar Tahsisi :

$s \rightarrow \{ u_1, u_2\} : \{k_{1-3}^{*}\}\, k_{12}$

$s \rightarrow \{ u_3\}$: $\{k_{1\text{-}3}^{*}\}\, k_{34}^{*}, \{k_{34}^{*}\}\, k_3$

Grup-Odaklı Yeniden Anahtar Tahsisi :

$s \rightarrow \{ u_1, u_2, u_3\} : \{k_{1\text{-}3}^{*}\}\, k_{12}\, , \{k_{1\text{-}3}^{*}\}\, k_{34}^{*}\, , \{k_{34}^{*}\}\, k_3$

Örneğin, U_3 kullanıcısının, yukarıda verilen mesajlardan Anahtar-Odaklı yeniden anahtar tahsisi mesajını aldığı düşünülsün. Bu durumda, "İstemci" programı öncelikle k_3 ile şifrelenmiş $\{k_{34}^{*}\}$ anahtarının şifresini çözmeli ve ardından k_{34}^{*} ile şifrelenmiş olan $\{k_{1\text{-}3}^{*}\}$ anahtarının şifresini çözmelidir.

Grup-Odaklı yeniden anahtar tahsisi mesajı alındığında ise, istemci şu şekilde davranacaktır: k_{12} ile şifrelenmiş olan $\{k_{1\text{-}3}^{*}\}$ yeniden anahtar tahsisi mesajı, U_3 kullanıcısının anahtar kümesi içerisinde k_{12} anahtarı bulunmadığından atılacaktır. k_{34}^{*} anahtarı ile şifrelenmiş olan $\{k_{1\text{-}3}^{*}\}$ mesajının şifresinin çözülmesinden önce, k_3 anahtarı ile şifrelenmiş olan $\{k_{34}^{*}\}$ mesajının şifresi çözülmelidir.

Benzer şekilde, grup-odaklı yeniden anahtar tahsisi mesajının U_2 kullanıcısı tarafından alınması durumunda, k_{12} anahtarı U_2 kullanıcısının kümesi içerisinde yer aldığından, bu anahtar ile şifrelenmiş olan $\{k_{1\text{-}3}^{*}\}$ anahtarı çözülecek ve "İstemci" tarafından tutulan eski hali ile değiştirilecektir. k_{34}^{*} anahtarı, kullanıcının anahtar kümesi içerisinde yer almadığından bu anahtar ile şifrelenmiş $\{k_{1\text{-}3}^{*}\}$ mesajı silinecektir. Benzer şekilde, k_3 anahtarı, kullanıcının anahtar kümesi içerisinde olmadığından bu anahtar ile şifrelenmiş olan $\{k_{34}^{*}\}$ anahtar mesajı da silinecektir.

8. SONUÇLAR VE TARTIŞMA

Noktadan-noktaya oturum için, oturum kurulum ve anahtar dağıtım maliyetleri sadece bir kere oturumun başında gerçekleşir. Diğer taraftan bir grup oturumunda bu maliyetler üyelerin katılmaları ve ayrılmaları nedeniyle daha uzun sürebilir. Bu nedenle, grup anahtarının sürekli olarak değiştirilmesi gerekir. Yüksek seviyede güvenliğin sağlanması için grup anahtarının her bir katılma ve ayrılma sonrasında değiştirilmesi gerekir. Böylelikle ayrılan grup üyesi kendisinden sonraki iletişime, yeni katılan grup üyesi de kendisinden önceki iletişime ulaşamamış olacaktır.

Her bir katılma ve ayrılmadan sonra yeni bir grup anahtarını yaratan bir güvenilir anahtar sunucusu olduğunu düşünelim. Bir katılma sonrasında yeni grup anahtarı yeni katılan grup üyesine teke gönderim yolu ile (şahsi anahtarı ile şifrelenerek), diğer grup üyelerine ise çoğa gönderim yolu ile (eski grup anahtarı ile şifrelenerek) yollanır. Dolayısı ile katılma sonrasında grup anahtarının değiştirilmesi için çok fazla işlem yapılması gerekli olmamaktadır. Ancak, bir ayrılma sonrasında, önceki grup anahtarı kullanılamaz ve yeni grup anahtarı her bir grup üyesi için, grup üyelerinin şahsi anahtarları kullanılarak şifrelenmek durumunda kalınır. Dolayısı ile ayrılma sonrasında grup anahtarını güvenli olarak değiştirmek için işlem ve iletişim maliyetleri *n*'le orantılı olarak gerçekleşir. Dolayısı ile bu durum, üyeleri sıklıkla ayrılan ve katılan çok büyük gruplar için ölçeklenebilirlik problemi ortaya çıkarmaktadır.

Bu çalışmada, Grup Anahtar Yönetiminin ölçeklenebilirlik problemini adreslemek için anahtar ağaç graflarının nasıl kullanılabileceği incelenmiştir. *Kullanıcı-odaklı*, *anahtar-odaklı* ve *grup-odaklı* olmak üzere üç farklı yeniden anahtar tahsisi yöntemi incelenmiş, her biri için katılma/ayrılma protokolleri tanımlanmıştır. Yeniden anahtar tahsisi yöntemleri biri benzetim yazılımı, diğeri Çoğa Gönderim Anahtar Yönetimi yazılımı olmak üzere iki ayrı yazılımda uygulanmış ve sonuçlar elde edilmiştir.

Çalışmada öncelikle grup iletişimi tanımlanmış ve grup iletişimi tipleriden bahsedilmiştir. Daha sonra IP ortamında çoğa gönderimin nasıl yapıldığından bahsedilmiştir. Güvenli çoğa gönderim tanımlanmış ve güvenli çoğa gönderim kriterleri belirlenmiştir. Bir sonraki kısımda Çoğa Gönderimde anahtar yönetiminin nasıl yapılacağından bahsedilmiş ve anahtar dağıtma mimarileri anlatılmış ve birbirleri ile karşılaştırılmıştır. Anahtar grafları kullanılarak güvenli çoğa gönderimin nasıl gerçekleştirilebileceği anlatılmış ve üç farklı yeniden anahtar tahsisi yaklaşımı tanıtılmıştır. Bu üç farklı yöntemin uygulandığı benzetim yazılımı tanıtılmış ve bu yazılımdan elde edilen sonuçlar ile yöntemlerin birbirleri ile karşılaştırılması gerçekleştirilmiştir. Son olarak her üç yöntemin uygulanması ile gerçekleştirilmiş Çoğa Gönderim Anahtar Yönetimi yazılımı tanıtılmıştır.

Yapılan benzetim çalışmaları ve testlerden elde edilen sonuçlar bu üç yeniden anahtar tahsisi yaklaşımının her birinin büyük gruplar ve sık katılma/ayrılma istekleri için ölçeklenebilir olduğunu göstermektedir. Özellikle, katılma/ayrılma başına ortalama sunucu işlem zamanı, grup büyüklüğünün logaritması ile doğrusal olarak artmaktadır. Ayrıca yapılan çalışmalar sonucunda eniyi anahtar ağaç derecesinin dört olduğu görülmüştür.

Sunucu tarafında, Grup-Odaklı Yeniden Anahtar Tahsisi en iyi başarımı sağlamaktadır. İkinci sırada Anahtar-Odaklı Yeniden Anahtar Tahsisi; üçüncü sırada ise Kullanıcı-Odaklı Yeniden Anahtar Tahsisi yer alır. İstemci tarafında ise, Kullanıcı-Odaklı Yeniden Anahtar Tahsisi en iyi başarımı sağlamaktadır. İkinci sırada Anahtar-Odaklı Yeniden Anahtar Tahsisi; üçüncü sırada ise Grup-Odaklı Yeniden Anahtar Tahsisi yaklaşımı bulunmaktadır. Özellikle, istemcileri düşük-hızlı bağlantılar (modemler) ile ağa bağlı olan çok büyük gruplar için, Anahtar-Odaklı ya da Kullanıcı-Odaklı yaklaşım Grup-Odaklı yaklaşıma göre daha uygundur.

Çalışma boyunca üç yeniden anahtar tahsisi yaklaşımı tarafından yaratılan ağ trafiği incelenmemiştir. Grup-Odaklı yaklaşım ile katılma/ayrılma başına tek bir yeniden anahtar tahsisi yaklaşımı tüm gruba çoğa gönderim yolu ile yollanır. Yaratılan ağ yükü ağ düzenleşimine (yerel alan ağı, kampus ağı, geniş-alan ağı vb.) ve grubun coğrafik dağılımına göre değişiklik gösterecektir. Anahtar-odaklı ve kullanıcı-odaklı yeniden

anahtar tahsisi yaklaşımları ile katılma/ayrılma başına pek çok küçük yeniden anahtar tahsisi mesajları alt gruplara yollanır. Eğer yeniden anahtar tahsisi mesajları teke gönderim yolu ile yollanır ise (ağ alt grup çoğa gönderimine destek vermiyor ise), yaratılan ağ yükü grup-odaklı yeniden anahtar tahsisi yaklaşımından çok daha fazla olur.

Çalışma sırasında geliştirilen Çoğa Gönderim Anahtar Yönetimi yazılımının geliştirilmesinde Microsoft Visual Studio içerisinde yer alan Visual C++ geliştirme ortamı kullanılmıştır. İletişim için kullanılan socketler Windows Socket'leridir. Anahtarların tutulması için kurulacak ağaç yapısının oluşturulmasında array yapılarından yararlanılmıştır. Yazılımında çoğa gönderim trafiğini güvenli hale getirmek için kullanılan anahtarların yollanan verileri şifrelemesinde DES algoritması kullanılmıştır. Bu çalışmada anlatılan yöntemlerde kullanılan anahtar sistemlerinin simetrik olması gerekmektedir. Bunun sebebi, kamusal anahtar düzeninin kullanılması durumunda istemci ve sunucu tarafında anahtar saklama alanının oldukça fazla artması ve başlangıç anahtar tahsisi için gerekli olan sürenin artmasıdır. Çalışmada, simetrik şifreleme algoritmalarından DES algoritması tercih edilmiştir. Diğer simetrik algoritmalar da kullanılabilir.

Çoğa Gönderim Anahtar Yönetimi yazılımında anahtar ağaç grafının oluşturulması aşamasında bir eniyileme mekanizması kullanılmamaktadır. Katılmak için gelen kullanıcı ağaçta boş olan ilk yere yerleştirilmektedir. Yeniden anahtar tahsisi mesajları etkileyen faktörlerden biri de kullanıcının anahtar ağaç grafında eklendiği yer olmaktadır. Bu nedenle daha sonra yapılacak olan çalışmalarda yeniden anahtar tahsisi mesajlarının en iyi olması için kullanıcının ağaçta nereye yerleştirileceği konusunda bir eniyileme mekanizmasının tespit edilip kullanılabilir.

Çoğa Gönderim Anahtar Yönetimi yazılımı kapsamında istemcilere düz metin mesajların yollanması sağlanmıştır. İleriki aşamalarda bu programın bir beyaz tahta uygulamasına çevrilmesi, ya da dosya aktarımı için kullanılması sağlanabilir.

Burada unutulmaması gereken şeylerden biri şudur: bir uygulama için en iyi olan çözüm başka bir uygulamanın ihtiyaçlarını karşılamayabilir. Bu nedenle bir güvenlik çözümü belirlemeden önce uygulamanın güvenlik yönetimini ve gereksinimlerini çok iyi anlamak gerekir.

KAYNAKLAR

[1] **Çölkesen, R. ve Örencik, B.,** 1999. Bilgisayar Haberleşmesi ve Ağ Teknolojileri, *Papatya Yayıncılık*, İstanbul

[2] **Witthmann, R. and Zitterbart M.,** 2001. Multicast Communication Protocols And Applications, *Morgan Kaufmann Publishers Academic Press,* San Diego

[3] **Diot, C., Dabbous, W. and Crowcraft J.,** 1997 Multipoint communications: A survey of protocols, functions and mechanisms. *IEEE Journal on Selected Areas in Communications (JSAC),* 15(3):227-290, April 1997

[4] **Maufer, T., and Semeria, C.**, 1997, Introduction to IP Multicast Routing , IETF *Network Working Group Internet Draft, July 1997*

[5] **Kruus, P.S.**,1998, A Survey of Multicast Security Issues And Architecture, *http://download.nai.com/products/media/nai/pdf/1998kruus-NISSC98.pdf,* 1998

[6] **Deering, S.**, 1999, Host Extensions for IP Multicast, *RFC 1112,*

[7] **Quinn, B.**, 1998, Ip Multicast Applications: Challanges and Solutions, *Internet Draft draft-quinn-multicast-apps-00.txt, Nov.1998*

[8] **Ballardie, A.,** 1996, Scalable Multicast Key Distribution, *RFC-1949, May. 1996*

[9] **Ballardie, A.,** 1997, Core Base Trees (CBT) Multicast Routing Architecture, *Internet Draft, May. 1997*

[10] **Schulzrinne, H.,** 1996, RTP: A Transport Protocol for Real-Time Applications, *RFC-1889, January 1996*

[11] **Handley, M.,** 1996, SAP: Session Announcement Protocol, *Internet Draft, Nov1996*

[12] **Wallner, D., Harder, E., and Agee, R.,** 1997, Key Management for Multicast: Issues and Architecture, *Internet Draft, draft-wallner-key-arch-00.txt, July 1997*

[13] **Briscoe, R., and Bagnall, P.,** 1997, Taxonomy of Communication Requirements for
Large-scale Multicast Group, *Internet Draft, July 1997*

[14] **Chiou, G.H., and Chen, W.T.,** 1989, Secure Broadcasting Using the Secure Lock, *IEEE Transactions on Software Engineering, v. SE-15, n. 8, August 1989, pp. 929-934*

[15] **Oppliger, R., and Albanese, A.,** 1996, Distributed Registration and Key Distribution, *Proceedings of the 12th International Conference on Information Security (IFIF SEC '96), Island of Samos (Greece), May 21-24, 1996, Chapman & Hall, London, pp. 199-208*

[16] **Wong, C.K., Gouda M., and Lam, S.S.,** 2000, Secure Group Communications Using Key Graphs, *IEEE/ACM Transactions on Networking,February 2000,*

[17] **Mittra, S.,** 1997, Iolus: A framework for scalable secure multicasting , *ACM SIGCOMM'97,* 1997, *pp. 277-288*

[18] **The Internet Engineering Task Force,** (2002), *The Internet Engineering Task Force,* <http://www.ietf.cnri.reston.va.us/home.html>

[19] **The IP Multicast Initiative,** (2002), *The IP Multicast Initiative,* <http://www.ipmulticast.com>

[20] **Hot Topics in Networking References,** (2002), *Hot Topics in Networking References,* <http://www.cis.ohio-state.edu/~jain/refs/all_refs.htm>

[21] **IETF Secure Multicast Group (SMuG),** (2002), *IETF Secure Multicast Group (SMuG)* <http://www.securemulticast.org>

[22] **Atkinson, Randall,** 1995, Security Architecture for the Internet Protocol, *Internet Draft, RFC 1825, August 1995*

[23] **Deering, S.E.,** 1989, Host Extensions for IP Multicasting, *RFC 1112, August 1989*

[24] **Fenner, W.,** 1997, Internet Group Management Protocol, Version 2, *RFC 2236 on IETF, November1997*

[25] **Cain, B., Deering, S., and Thyagarajan, A.,** 1999, Internet Group Management Protocol, Version 2, *Internet Draft, draft-ietf-idmr-igmp-v3-01.txt, February1999*

[26] **D. Maughan, M. Schertler.** *Internet Security Association and Key Management Protocol (ISAKMP),* Work in Progress draft-ietf-ipsec-isakmp-04, National Security Agency, February 1996

[27] **A. Aziz, T. Markson, H. Prafullchandra.** *Simple Key-management For Internet Protocols (SKIP)*, Work in Progress draft-ietf-ipsec-skip-06, Sun Microsystems, Inc., December 1995

[28] **H. K. Orman.** *The Oakley Key Determination Protocol*, Work in Progress draft-ietf-ipsec-oakley-00, University of Arizona, February 1996

[29] **P. Karn and W. Simpson.** *The Photuris Session Key Management Protocol*, Work in Progress draft-ietf-ipsec-photuris-09, Qualcomm, Inc., Daydreamer, February 1996.

[30] **H. Krawczyk**. *SKEME: A Versatile Secure Key Exchange Mechanism for Internet*, SNDSS'96 Proceedings, February 1996

EKLER

BENZETİM PROGRAMINDAN ÇEŞİTLİ DENEMELERDE ELDE EDİLEN SONUÇLAR

Tablo A.1 Grup-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı = 4)

Ağaç Derecesi = 2	Kullanıcı Sayısı	Ağaç Yüksekliği	Mesaj Sayısı	Mesaj Uzunluğu	Şifreleme Maliyeti
JOIN=User 1	1	2	1	1	1
JOIN=User 2	2	2	2	1; 1	2
JOIN=User 3	3	3	2	2; 2	4
JOIN=User 4	4	3	2	2; 2	4
LEAVE=User 4	3	3	1	3	3
LEAVE=User 1	2	3	1	3	3
JOIN=User 4	3	3	2	2; 2	4
LEAVE=User 4	2	3	1	3	3

Tablo A.2 Grup-Odaklı YenidenAnahtar Tahsisi (Başlangıç Kullanıcı Sayısı = 8)

Ağaç Derecesi = 2	Kullanıcı Sayısı	Ağaç Yüksekliği	Mesaj Sayısı	Mesaj Uzunluğu	Şifreleme Maliyeti
...					
JOIN = User 5...User 8	5...8	4	2	3; 3	6
LEAVE=User 1	7	4	1	5	5
LEAVE=User 7	6	4	1	5	5
JOIN=User 1	7	4	2	3; 3	6

Tablo A.3 Grup-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı = 16)

Ağaç Derecesi = 2	Kullanıcı Sayısı	Ağaç Yüksekliği	Mesaj Sayısı	Mesaj Uzunluğu	Şifreleme Maliyeti
....					
JOIN=User 9...User 16	9...16	5	2	4; 4	8
LEAVE=User 9	15	5	1	7	7
LEAVE=User 1	14	5	1	7	7
LEAVE=User 7	13	5	1	7	7
JOIN=User 9	14	5	2	4; 4	8
JOIN=User 1	15	5	2	4; 4	8
JOIN=User 7	16	5	2	4; 4	8
LEAVE=User 5	15	5	1	7	7
JOIN=User 5	16	5	2	4; 4	8
LEAVE=User 5	15	5	1	7	7
LEAVE=User 7	14	5	1	7	7
LEAVE=User 14	13	5	1	7	7
LEAVE=User 9	12	5	1	7	7

Tablo A.4 Grup-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı = 32)

Ağaç Derecesi = 2	Kullanıcı Sayısı	Ağaç Yüksekliği	Mesaj Sayısı	Mesaj Uzunluğu	Şifreleme Maliyeti
...					
JOIN=User 17...User 32	17...32	6	2	5; 5	10
LEAVE=User 20	31	6	1	9	9
LEAVE=User 26	30	6	1	9	9
JOIN=User 20	31	6	2	5; 5	10
LEAVE=User17	30	6	1	9	9
JOIN=User 26	31	6	2	5; 5	10
LEAVE=User 12	30	6	1	9	9
LEAVE=User 14	29	6	1	9	9

LEAVE=User 25	28	6	1	7	7
LEAVE=User 27	27	6	1	8	8
LEAVE=User 30	26	6	1	9	9
LEAVE=User 11	25	6	1	7	7
LEAVE=User 32	24	6	1	9	9
JOIN=User 17	25	6	2	5; 5	10
JOIN=User 12	26	6	2	5; 5	10
JOIN=User 14	27	6	2	5; 5	10
JOIN=User 25	28	6	2	5; 5	10
LEAVE=User 20	27	6	1	9	9

Tablo A.5 Grup-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı = 64)

Ağaç Derecesi = 2	Kullanıcı Sayısı	Ağaç Yüksekliği	Mesaj Sayısı	Mesaj Uzunluğu	Şifreleme Maliyeti
...					
JOIN=User 33...User 64	33...64	7	2	6; 6	12
LEAVE=User 1	63	7	1	11	11
LEAVE=User 10	62	7	1	11	11
LEAVE=User 64	61	7	1	11	11
LEAVE=User 57	60	7	1	11	11
JOIN=User 1	61	7	2	6; 6	12
LEAVE=User 54	60	7	1	11	11
LEAVE=User 37	59	7	1	11	11
LEAVE=User 13	58	7	1	11	11
LEAVE=User 47	57	7	1	11	11
JOIN=User 10	58	7	2	6; 6	12
LEAVE=User 55	57	7	1	11	11
LEAVE=User 30	56	7	1	11	11
LEAVE=User 17	55	7	1	11	11
JOIN=User 64	56	7	2	6; 6	12
JOIN=User 57	57	7	2	6; 6	12

JOIN=User 54	58	7	2	6; 6	12
JOIN=User 37	59	7	2	6; 6	12
JOIN=User 13	60	7	2	6; 6	12
JOIN=User 47	61	7	2	6; 6	12
JOIN=User 55	62	7	2	6; 6	12
LEAVE=User 32	61	7	1	11	11
JOIN=User 30	62	7	2	6; 6	12
LEAVE=User 28	61	7	1	11	11
JOIN=User 17	62	7	2	6; 6	12
JOIN=User 32	63	7	2	6; 6	12
LEAVE=User 12	62	7	1	11	11

Tablo A.6 Anahtar-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı = 4)

Ağaç Derecesi = 2	Kullanıcı Sayısı	Ağaç Yüksekliği	Mesaj Sayısı	Mesaj Uzunluğu	Şifreleme Maliyeti
JOIN=User 1	1	2	1	1	1
JOIN=User 2	2	2	2	1; 1	2
JOIN=User 3	3	3	2	1; 2	3
JOIN=User 4	4	3	3	1; 1; 2	4
LEAVE=User 3	3	3	2	1; 2	3
JOIN=User 3	4	3	3	1; 1; 2	4
LEAVE=User 2	3	3	2	1; 2	3
JOIN=User 2	4	3	3	1; 1; 2	4

Tablo A.7 Anahtar-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı = 8)

Ağaç Derecesi = 2	Kullanıcı Sayısı	Ağaç Yüksekliği	Mesaj Sayısı	Mesaj Uzunluğu	Şifreleme Maliyeti
...					
JOIN=User 5	5	4	2	1; 3	4
JOIN=User 6	6	4	4	1; 1; 1; 3	6

JOIN=User 7	7	4	3	1; 1; 3	5
JOIN=User 8	8	4	4	1; 1; 1; 3	6
LEAVE=User 2	7	4	3	1; 2; 3	5
LEAVE=User 7	6	4	3	1; 2; 3	5
LEAVE=User 1	5	4	2	1; 4	4
LEAVE=User 5	4	4	3	1; 2; 3	5

Tablo A.8 Anahtar-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı = 16)

Ağaç Derecesi = 2	Kullanıcı Sayısı	Ağaç Yüksekliği	Mesaj Sayısı	Mesaj Uzunluğu	Şifreleme Maliyeti
....					
JOIN=User 9	9	5	2	1; 4	5
JOIN=User 10	10	5	5	1; 1; 1; 1; 4	8
JOIN=User 11	11	5	4	1; 1; 1; 4	7
JOIN=User 12	12	5	5	1; 1; 1; 1; 4	8
JOIN=User 13	13	5	3	1; 1; 4	6
JOIN=User 14	14	5	5	1; 1; 1; 1; 4	8
JOIN=User 15	15	5	4	1; 1; 1; 4	7
JOIN=User 16	16	5	5	1; 1; 1; 1; 4	8
LEAVE=User 12	15	5	4	1; 2; 3; 4	7
JOIN=User 12	16	5	5	1; 1; 1; 1; 4	8
LEAVE=User 11	15	5	4	1; 2; 3; 4	7
JOIN=User 11	16	5	5	1; 1; 1; 1; 4	8
LEAVE=User 7	15	5	4	1; 2; 3; 4	7
LEAVE=User 5	14	5	4	1; 2; 3; 4	7
LEAVE=User 13	13	5	4	1; 2; 3; 4	7
LEAVE=User 11	12	5	4	1; 2; 3; 4	7
LEAVE=User 10	11	5	4	1; 2; 3; 4	7
LEAVE=User 5	10	5	3	1; 2; 6	6
JOIN=User 7	11	5	4	1; 1; 1; 4	7

Tablo A.9 Anahtar-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı = 32)

Ağaç Derecesi = 2	Kullanıcı Sayısı	Ağaç Yüksekliği	Mesaj Sayısı	Mesaj Uzunluğu	Şifreleme Maliyeti
....					
JOIN=User 17	17	6	2	1; 5	6
JOIN=User 18	18	6	6	1; 1; 1; 1; 1; 5	10
JOIN=User 19	19	6	5	1; 1; 1; 1; 5	9
JOIN=User 20	20	6	6	1; 1; 1; 1; 1; 5	10
JOIN=User 21	21	6	4	1; 1; 1; 5	8
JOIN=User 22	22	6	6	1; 1; 1; 1; 1; 5	10
JOIN=User 23	23	6	5	1; 1; 1; 1; 5	9
JOIN=User 24	24	6	6	1; 1; 1; 1; 1; 5	10
JOIN=User 25	25	6	3	1; 1; 5	7
JOIN=User 26	26	6	6	1; 1; 1; 1; 1; 5	10
JOIN=User 27	27	6	5	1; 1; 1; 1; 5	9
JOIN=User 28	28	6	6	1; 1; 1; 1; 1; 5	10
JOIN=User 29	29	6	4	1; 1; 1; 5	8
JOIN=User 30	30	6	6	1; 1; 1; 1; 1; 5	10
JOIN=User 31	31	6	5	1; 1; 1; 1; 5	9
JOIN=User 32	32	6	6	1; 1; 1; 1; 1; 5	10
LEAVE=User 4	31	6	5	1; 2; 3; 4; 5	9
LEAVE=User 1	30	6	5	1; 2; 3; 4; 5	9
JOIN=User 4	31	6	6	1; 1; 1; 1; 1; 5	10
LEAVE=User 19	30	6	5	1; 2; 3; 4; 5	9
LEAVE=User 15	29	6	5	1; 2; 3; 4; 5	9
LEAVE=User 24	28	6	5	1; 2; 3; 4; 5	9
JOIN=User 1	29	6	6	1; 1; 1; 1; 1; 5	10
JOIN=User 19	30	6	6	1; 1; 1; 1; 1; 5	10
LEAVE=User 30	29	6	5	1; 2; 3; 4; 5	9
LEAVE=User 2	28	6	5	1; 2; 3; 4; 5	9
JOIN=User 15	29	6	6	1; 1; 1; 1; 1; 5	10

LEAVE=User 6	28	6	5	1; 2; 3; 4; 5	9
LEAVE=User 17	27	6	5	1; 2; 3; 4; 5	9
JOIN=User 24	28	6	6	1; 1; 1; 1; 1; 5	10
LEAVE=User 8	27	6	5	1; 2; 3; 4; 5	9
JOIN=User 30	28	6	6	1; 1; 1; 1; 1; 5	10
JOIN=User 2	29	6	6	1; 1; 1; 1; 1; 5	10
LEAVE=User 23	28	6	5	1; 2; 3; 4; 5	9

Tablo A.10 Anahtar-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı = 64)

Ağaç Derecesi = 2	Kullanıcı Sayısı	Ağaç Yüksekliği	Mesaj Sayısı	Mesaj Uzunluğu	Şifreleme Maliyeti
.....					
JOIN=User 45	45	7	5	1; 1; 1; 1; 6	10
JOIN=User 46	46	7	7	1; 1; 1; 1; 1; 1; 6	12
JOIN=User 47	47	7	6	1; 1; 1; 1; 1; 6	11
JOIN=User 48	48	7	7	1; 1; 1; 1; 1; 1; 6	12
JOIN=User 49	49	7	3	1; 1; 6	8
JOIN=User 50	50	7	7	1; 1; 1; 1; 1; 1; 6	12
JOIN=User 51	45	7	6	1; 1; 1; 1; 1; 6	11
JOIN=User 52	52	7	7	1; 1; 1; 1; 1; 1; 6	12
.....					
JOIN=User 62	62	7	7	1; 1; 1; 1; 1; 1; 6	12
JOIN=User 63	63	7	6	1; 1; 1; 1; 1; 6	11
JOIN=User 64	64	7	7	1; 1; 1; 1; 1; 1; 6	12
LEAVE=User 24	63	7	6	1; 2; 3; 4, 5; 6	11
JOIN=User 24	64	7	7	1; 1; 1; 1; 1; 1; 6	12
LEAVE=User 27	63	7	6	1; 2; 3; 4; 5; 6	11
LEAVE=User 16	62	7	6	1; 2; 3; 4; 5;6	11
LEAVE=User 34	61	7	6	1; 2; 3; 4; 5; 6	11
LEAVE=User 45	60	7	6	1; 2; 3; 4; 5; 6	11
LEAVE=User 35	59	7	6	1; 2; 3; 4; 5; 6	11

LEAVE=User 62	58	7	6	1; 2; 3; 4; 5; 6	11
JOIN=User 27	59	7	7	1; 1; 1; 1; 1; 1; 6	12
LEAVE=User 50	58	7	6	1; 2; 3; 4; 5; 6	11
JOIN=User 16	59	7	7	1; 1; 1; 1; 1; 1; 6	12
LEAVE=User 13	58	7	6	1; 2; 3; 4; 5; 6	11
LEAVE=User 49	57	7	5	1; 2; 3; 4; 10	10
JOIN=User 34	58	7	7	1; 1; 1; 1; 1; 1; 6	12
JOIN=User 45	59	7	7	1; 1; 1; 1; 1; 1; 6	12
LEAVE=User 14	58	7	6	1; 2; 3; 4; 5; 6	11
JOIN=User 35	59	7	7	1; 1; 1; 1; 1; 1; 6	12
JOIN=User 62	60	7	7	1; 1; 1; 1; 1; 1; 6	12
JOIN=User 50	61	7	7	1; 1; 1; 1; 1; 1; 6	12
JOIN=User 13	62	7	6	1; 1; 1; 1; 1; 6	11

Tablo A.11 Kullanıcı-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı = 4)

Ağaç Derecesi = 2	Kullanıcı Sayısı	Ağaç Yüksekliği	Mesaj Sayısı	Mesaj Uzunluğu	Şifreleme Maliyeti
JOIN=User 1	1	2	1	1	1
JOIN=User 2	2	2	2	1; 1	2
JOIN=User 3	3	3	2	1; 2	3
JOIN=User 4	4	3	3	1; 2; 2	5
LEAVE=User 2	3	3	2	1; 2	3
JOIN=User 2	4	3	3	1; 2; 2	5
LEAVE=User 2	3	3	2	1; 2	3
JOIN=User 2	4	3	3	1; 2; 2	5

Tablo A.12 Kullanıcı-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı = 8)

Ağaç Derecesi = 2	Kullanıcı Sayısı	Ağaç Yüksekliği	Mesaj Sayısı	Mesaj Uzunluğu	Şifreleme Maliyeti
...					
JOIN=User 5	5	4	2	1; 3	4
JOIN=User 6	6	4	4	1; 2; 3; 3	9
JOIN=User 7	7	4	3	1; 2; 3	6
JOIN=User 8	8	4	4	1; 2; 3; 3	9
LEAVE=User 8	7	4	3	1; 2; 3	6
JOIN=User 8	8	4	4	1; 2; 3; 3	9
LEAVE=User 6	7	4	3	1; 2; 3	6
LEAVE=User 7	6	4	3	1; 2; 3	6

Tablo A.13 Kullanıcı-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı = 16)

Ağaç Drecesi = 2	Kullanıcı Sayısı	Ağaç Yüksekliği	Mesaj Sayısı	Mesaj Uzunluğu	Şifreleme Maliyeti
....					
JOIN=User 9	9	5	2	1; 4	5
JOIN=User 10	10	5	5	1; 2; 3; 4; 4	14
JOIN=User 11	11	5	4	1; 2; 3; 4	10
JOIN=User 12	12	5	5	1; 2; 3; 4; 4	14
JOIN=User 13	13	5	3	1; 2; 4	7
JOIN=User 14	14	5	5	1; 2; 3; 4; 4	14
JOIN=User 15	15	5	4	1; 2; 3; 4	10
JOIN=User 16	16	5	5	1; 2; 3; 4; 4	14
LEAVE=User 5	15	5	4	1; 2; 3; 4	10
JOIN=User 5	16	5	5	1; 2; 3; 4; 4	14
LEAVE=User 15	15	5	4	1; 2; 3; 4	10
LEAVE=User 16	14	5	3	1; 2; 3	6
LEAVE=User 12	13	5	4	1; 2; 3; 4	10

JOIN=User 15	14	5	5	1; 2; 3; 4; 4	14
LEAVE=User 15	13	5	4	1; 2; 3; 4	10
JOIN=User 16	14	5	5	1; 2; 3; 4; 4	14
JOIN=User 12	15	5	4	1; 2; 3; 4	10
JOIN=User 15	16	5	5	1; 2; 3; 4; 4	14
LEAVE=User 2	15	5	4	1; 2; 3; 4	10
JOIN=User 2	16	5	5	1; 2; 3; 4; 4	14

Tablo A.14 Kullanıcı-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı = 32)

Ağaç Drecesi = 2	Kullanıcı Sayısı	Ağaç Yüksekliği	Mesaj Sayısı	Mesaj Uzunluğu	Şifreleme Maliyeti
...					
JOIN=User 17	17	6	2	1; 5	6
JOIN=User 18	18	6	6	1; 2; 3; 4; 5; 5	20
JOIN=User 19	19	6	5	1; 2; 3; 4; 5	15
JOIN=User 20	20	6	6	1; 2; 3; 4; 5; 5	20
JOIN=User 21	21	6	4	1; 2; 3; 5	11
JOIN=User 22	22	6	6	1; 2; 3; 4; 5; 5	20
JOIN=User 23	23	6	5	1; 2; 3; 4; 5	15
JOIN=User 24	24	6	6	1; 2; 3; 4; 5; 5	20
JOIN=User 25	25	6	3	1; 2; 5	8
JOIN=User 26	26	6	6	1; 2; 3; 4; 5; 5	20
JOIN=User 27	27	6	5	1; 2; 3; 4; 5	15
JOIN=User 28	28	6	6	1; 2; 3; 4; 5; 5	20
JOIN=User 29	29	6	4	1; 2; 3; 5	11
JOIN=User 30	30	6	6	1; 2; 3; 4; 5; 5	20
JOIN=User 31	31	6	5	1; 2; 3; 4; 5	15
JOIN=User 32	32	6	6	1; 2; 3; 4; 5; 5	20
LEAVE=User 12	31	6	5	1; 2; 3; 4; 5	15
LEAVE=User 31	30	6	5	1; 2; 3; 4; 5	15
JOIN=User 12	31	6	6	1; 2; 3; 4; 5; 5	20

LEAVE=User 19	30	6	5	1; 2; 3; 4; 5	15
LEAVE=User 11	29	6	5	1; 2; 3; 4; 5	15
JOIN=User 31	30	6	6	1; 2; 3; 4; 5; 5	20
JOIN=User 19	31	6	6	1; 2; 3; 4; 5; 5	20
JOIN=User 11	32	6	6	1; 2; 3; 4; 5; 5	20
LEAVE=User 11	31	6	5	1; 2; 3; 4; 5	15
LEAVE=User 9	30	6	5	1; 2; 3; 4; 5	15
LEAVE=User 5	29	6	5	1; 2; 3; 4; 5	15
LEAVE=User 31	28	6	5	1; 2; 3; 4; 5	15
JOIN=User 11	29	6	5	1; 2; 3; 4; 5; 5	20
LEAVE=User 32	28	6	4	1; 2; 3; 4	10
JOIN=User 9	29	6	6	1; 2; 3; 4; 5; 5	20
LEAVE=User 23	28	6	5	1; 2; 3; 4; 5	15

Tablo A.15 Kullanıcı-Odaklı Yeniden Anahtar Tahsisi (Başlangıç Kullanıcı Sayısı = 64)

Ağaç Derecesi = 2	Kullanıcı Sayısı	Ağaç Yüksekliği	Mesaj Sayısı	Mesaj Uzunluğu	Şifreleme Maliyeti
.....					
JOIN=User 45	45	7	5	1; 2; 3; 4; 6	16
JOIN=User 46	46	7	7	1; 2; 3; 4; 5; 6; 6	27
JOIN=User 47	47	7	6	1; 2; 3; 4; 5; 6	21
JOIN=User 48	48	7	7	1; 2; 3; 4; 5; 6; 6	27
JOIN=User 49	49	7	3	1; 2; 6	9
JOIN=User 50	50	7	7	1; 2; 3; 4; 5; 6; 6	27
JOIN=User 51	51	7	6	1; 2; 3; 4; 5; 6	21
JOIN=User 52	52	7	7	1; 2; 3; 4; 5; 6; 6	27
.....					
JOIN=User 62	62	7	7	1; 2; 3; 4; 5; 6; 6	27
JOIN=User 63	63	7	6	1; 2; 3; 4; 5; 6	21
JOIN=User 64	64	7	7	1; 2; 3; 4; 5; 6; 6	27
LEAVE=User 28	63	7	6	1; 2; 3; 4; 5; 6	21

JOIN=User 28	64	7	7	1; 2; 3; 4; 5; 6; 6	27
LEAVE=User 9	63	7	6	1; 2; 3; 4; 5; 6	21
JOIN=User 9	64	7	7	1; 2; 3; 4; 5; 6; 6	27
LEAVE=User 51	63	7	6	1; 2; 3; 4; 5; 6	21
JOIN=User 51	64	7	7	1; 2; 3; 4; 5; 6; 6	27
LEAVE=User 50	63	7	6	1; 2; 3; 4; 5; 6	21
LEAVE=User 55	62	7	6	1; 2; 3; 4; 5; 6	21
JOIN=User 50	63	7	7	1; 2; 3; 4; 5; 6; 6	27
JOIN=User 55	64	7	7	1; 2; 3; 4; 5; 6; 6	27
LEAVE=User 29	63	7	6	1; 2; 3; 4; 5; 6	21
JOIN=User 29	64	7	7	1; 2; 3; 4; 5; 6; 6	27
LEAVE=User 43	63	7	6	1; 2; 3; 4; 5; 6	21
LEAVE=User 62	62	7	6	1; 2; 3; 4; 5; 6	21
LEAVE=User 31	61	7	6	1; 2; 3; 4; 5; 6	21
JOIN=User 43	62	7	7	1; 2; 3; 4; 5; 6; 6	27
LEAVE=User 53	61	7	6	1; 2; 3; 4; 5; 6	21
LEAVE=User 14	60	7	6	1; 2; 3; 4; 5; 6	21
LEAVE=User 10	59	7	6	1; 2; 3; 4; 5; 6	21
LEAVE=User 55	58	7	6	1; 2; 3; 4; 5; 6	21

ÖZGEÇMİŞ

Sibel YÜKSEL, 1976 yılında Niğde'de doğdu. İlk ve orta öğrenimini Niğde'de tamamladıktan sonra; 1993 yılında Niğde Bor Şehit Nuri Pamir Lisesi'nden mezun oldu. Aynı yıl İTÜ Elektrik-Elektronik Fakültesi Kontrol ve Bilgisayar Mühendisliği Bölümü'ne girmeye hak kazandı. Temmuz 1998'de bölüm birincisi ve üniversite ikincisi derecesi ile mezun oldu. Aynı yıl İTÜ Kontrol ve Bilgisayar Mühendisliği Ana Bilim Dalı'nda yüksek lisans programına başladı.

Sibel YÜKSEL, 1998 yılından beri özel bir kuruluşta CRM, CTI, haberleşme, akıllı kartlar, veri tabanı, web ve finans uygulamaları projelerinde uzman olarak çalışmaktadır.