GÜVENLİ AYGIT KONTROL PROTOKOLLERİ VE YEREL BİR UYGULAMA

Özge GÜNER
Yüksek Lisans Tezi

Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı
Mart – 2004

JÜRİ VE ENSTİTÜ ONAYI

Özge Güner'in **Güvenli Aygıt Kontrol Protokolleri Ve Yerel Bir Uygulama** başlıklı **Bilgisayar Mühendisliği** Anabilim Dalındaki, Yüksek Lisans tezi 18.03.2004..tarihinde, aşağıdaki jüri tarafından Anadolu Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliğinin ilgili maddeleri uyarınca değerlendirilerek kabul edilmiştir.

	Adı-Soyadı	İmza
Üye (Tez Danışmanı)	**: Prof.Dr.Yaşar HOŞCAN**	
Üye	**: Yrd.Doç.Dr.Ömer N. GEREK**	
Üye	**: Yrd.Doç.Dr.Cüneyt AKINLAR**	

Anadolu Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 13.10.2004 tarih ve ..36/1.... sayılı kararıyla onaylanmıştır.

Enstitü Müdürü

Prof. Dr. Altuğ İFTAR
Fen Bilimleri Enstitüsü
Müdürü

ÖZET

Yüksek Lisans Tezi

GÜVENLİ AYGIT KONTROL PROTOKOLLERİ
VE
YEREL BİR UYGULAMA

ÖZGE GÜNER

Anadolu Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman : Prof.Dr.Yaşar HOŞCAN
2004, 138 sayfa

Bu tezde uzaktan aygıt denetimi için bir uygulama katmanı protokolünün karşılaması gereken temel ve işlevsel gereksinimler belirlenmiş ve tanımlanmıştır. Mevcut Internet alt yapısında farklı amaçlarla kullanılmakta olan uygulama katmanı protokollerinden SNMP, SMTP ve SIP'nin belirlenmiş olan gereksinimleri ne derece karşıladıkları her bir protokol için sıra ile incelenmiştir. Bu esnada incelenen protokollerin destekleyemediği gereksinimlere yönelik olarak protokollerin standart tanımına uzantı olacak şekilde çözümler geliştirilmeye çalışılmıştır. İncelemeler sonucunda üç aday protokolden güvenli aygıt denetimi için en uygun ve yeterli özelliklere sahip olanının SIP olduğu belirlenmiştir. Son olarak da SIP kullanılarak yerel bir aygıt kontrolü uygulaması Java programlama dili kullanılarak geliştirilmiştir.

Anahtar Kelimeler: Aygıt Kontrolü, Aygıt Kontrol Gereksinimleri, SNMP, SMTP, SIP

ABSTRACT

Master of Science Thesis

RELIABLE APPLIANCE CONTROL PROTOCOLS AND A LOCAL APPLICATION

OZGE GUNER

Anadolu University
Graduate School of Natural and Applied Sciences
Computer Engineering Program

Supervisor :Prof.Yasar HOSCAN
2004, 138 pages

In this thesis, the main and functional requirements which must be met by an application layer protocol in appliance control are determined and defined. SNMP, SMTP and SIP which are chosen from among the protocols that functions in different purposes in the current Internet infrastructure are examined according to how they each meet the determined requirements. Meanwhile, solutions are tried to be developed for the requirements that can not be supported by the protocols as extensions to the standard definitions of the protocols. As a result of the examinations, SIP is determined to have the most convenient and adequate features for reliable appliance control among the three candidate protocols. Finally, by using SIP a local appliance control application is developed with the Java programming language.

Keywords: Appliance Control, Appliance Control Requirements, SNMP, SMTP, SIP

İÇİNDEKİLER

Sayfa

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

KISALTMALAR DİZİNİ

ADY	: Ara Düzey Yönetici
API	: Application Programming Interface
ASN	: Abstract Syntax Notation
BEEP	: Blocks Extensible Exchange Protocol
BER	: Basic Encoding Rules
DES	: Data Encryption Standart
DMP	: Device Messaging Protocol
DNS	: Domain Name System
ESMTP	: Extended SMTP
HTTP	: Hypertext Transfer Protocol
IANA	: International Assigned Numbers Authority
IETF	: Internet Engineering Task Force
IMAP	: Internet Mail Access Protocol
IP	: Internet Protocol
IPA	: Internet Personal Appliances
IPng	: IP next generation
IPX/SPX	: Internetwork Packet eXchange / Sequenced Packet eXchange
JMF	: Java Media Framework
JRE	: Java runtime environment
Mbone	: Multicast Backbone
MD5	: Message Digest 5
MDA	: Mail Delivery Agent
MIB	: Management Information Base
MIME	: Multipurpose Internet Mail Extensions
MTA	: Mail Transport Agent
MUA	: Mail User Agent
MX	: Mail Exchange
NA	: Networked Appliance

NAT	: Network Address Translation
NCP	: Network Control Program
NetBEUI	: Network BIOS Extended User Interface
NITS	: Network Independent Transport Service
OSI	: Open System Interconnection
PDA	: Personal Digital Assistant
PDU	: Protocol Data Unit
POP	: Post Office Protocol
RGW	: Residential Gateway
RTP	: Real-Time Transport Protocol
S/MIME	: Secure/MIME
SDP	: Session Description Protocol
SHA	: Secure Hash Algorithm
SIP	: Session Initiation Protocol
SLP	: Service Location Protocol
SMI	: Structure of Management Information
SMTP	: Simple Mail Transfer Protocol
SNA	: Systems Network Architecture
SNMPv3	: Simple Network Management Protocol version 3
TCP	: Transmission Control Protocol
UA	: User Agent
UAC	: User Agent Client
UAS	: User Agent Server
UDP	: User Datagram Protocol
UPS	: Uninterruptible Power Supply
URL	: Universal Resource Locator
USM	: User-Based Security Model
VACM	: View Based Access Control Protocol
VoIP	: Voice over IP
XML	: Extensible Markup Language

1. GİRİŞ VE AMAÇ

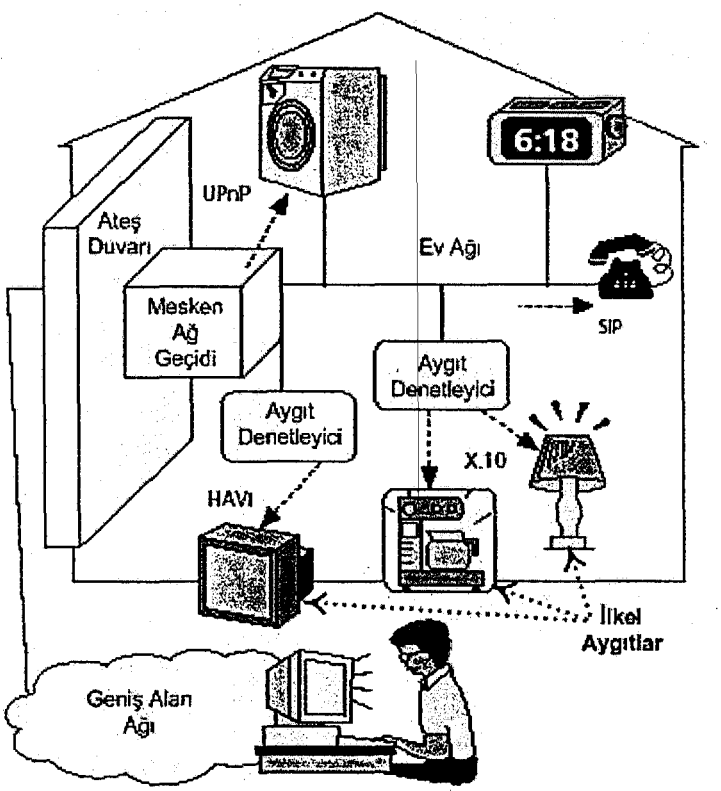
Günümüzde, aklımıza gelebilecek hemen her işi belirli bir cihaz yardımı ile gerçekleştirmek modern hayatın bir gereği haline gelmiştir. Bu cihazlar evdeki mikro dalga fırın, televizyon, çamaşır makinesi, bulaşık makinesi, klima sistemi, ışıklandırma sistemi, güvenlik sistemleri, elektrikli kapılar olabileceği gibi bir şirket binasının alarm sistemi ya da bir taşıtın ses sistemi, klima denetimi de olabilmektedir [1].

Bilişim ve iletişim teknolojilerindeki hızlı gelişim ile birlikte, günlük hayatımızda bulundukları ortamlar içerisinde fiziksel etkileşim ile kullandığımız her türlü cihazı çalışma ortamları (ev, ofis gibi) dışından kontrol edebilme fikri ortaya çıkmıştır. Şu an için sadece bilgisayar, yazıcı, tarayıcı, kamera gibi aygıtların oluşturduğu ev ağlarına televizyon, fırın, çamaşır makinesi, klima sistemi, kahve makinesi gibi klasik ev ve ofis aygıtlarının birer **Ağ Aygıtı (Networked Appliance-NA)** olarak bağlanması ile akıllı ortamlar yaratma yönünde çeşitli çalışmalar yapılmaya başlanmıştır. Yaratılan bu akıllı ağ ortamları sayesinde aygıtların mevcut işlevlerinin yanında kullanıcıya bir çok yeni kullanım olanağı sunacağı açıktır. Meyve rafi boşalan bir buzdolabının bağlı olduğu ev ağı aracılığı ile kullanıcısının cep telefonuna mesaj göndermesi, kullanıcının evindeki ağa bağlı dijital alarm saati uzaktan kurarak, saatin zaman dolduğunda yine evdeki akıllı klima sistemini çalıştırması, kullanıcının evindeki elektronik kapı zili çaldığında bu konuda bilgilendirilmesi, yine kullanıcının bu bilgilendirmeye karşılık kapıya bağlı kameradan görüntü isteyebilmesi gibi uygulama örnekleri gün geçtikçe artmaktadır.

Aslında bugün ağa aygıtlarının yerel alanları içerisinde kontrolüne yönelik olarak kullanılan çeşitli ağ teknolojileri (**UPnP - Universal Plug and Play, HAVi - Home Audio / Video Interoperability, X.10** gibi) mevcuttur (Şekil 1.1.). Buna karşın aygıtların yerel ortamlarının dışından, Internet üzerinden kontrol edilebilmesine ilişkin çalışmalar ise çok yakın bir geçmişte başlatılmıştır.

Denetim ortamının genişlemesi, ağ aygıtlarının kontrolünde kullanılacak olan protokollerde güvenlik, güvenilirlik, esneklik, ortamdan bağımsızlık gibi bazı önemli unsurların tam olarak karşılanabilmiş olmasının gerekliliğini ortaya çıkarmaktadır.

Günlük hayatımızda doğrudan etkileşimli olarak kullandığımız aygıtların yanlış kullanımları bile çeşitli problemlere neden oluyorken, bu işin bir de uzaktan gerçekleştirildiği düşünüldüğünde, kontrol sistemindeki herhangi bir eksikliğin ne gibi istenmeyen sonuçlara yol açacağını kestirmek oldukça güçtür.



Şekil 1.1. Örnek bir ev ağ aygıtları mimarisi [3]

Ağ aygıtı olarak adlandırılan akıllı aygıtlar, belirli bir işi yapması için tasarlanmış ve sadece o işi yapmasına yetecek kadar kısıtlı bir konfigürasyona sahip olan, ağ bağlantısına sahip aygıtlar olarak düşünülmelidir. Yani ağ aygıtları genel amaçlı kullanımı olmayan, ağ bağlantısına sahip aygıtlardır [2]. Daha biçimsel olarak bir ağ aygıtı, "En az bir adet ağ işlemcisine sahip olan adanmış işlevleri olan cihazlardır" [3]. Ağ aygıtları **Kişisel Internet Aygıtları (Internet Personal Appliances-IPA)** olarak da adlandırılırlar [1].

2. PROBLEM

Aygıt kontrolü en basit haliyle, bir ağ aygıtının, içinde bulunduğu yerel alanın dışından yönetilebilmesi olarak düşünülür. Buna karşın bu işlem gerçekleştirilirken en başta güvenlik ve gizlilik olmak üzere çeşitli gereksinimlerin de yerine getirilmesi gerekir. Mevcut uygulama katmanı protokolleri incelendiğinde çoğunun aygıt kontrolü bağlamında düşünülen işlev ve gereksinimleri kısmen de olsa karşılayabildikleri görülür. Ancak üretici firmalar bu durumdan yola çıkarak seçecekleri bir uygulama katmanı protokolünü aygıt kontrolü için geliştirip ürettikleri cihazlar için kullanmaya kalkarlarsa oluşacak karmaşadan dolayı gelecekte aygıt kontrolü bağlamında çeşitli zorluklar ve sorunlar ortaya çıkacaktır. Örneğin bir firmadan alınan ve içerisinde A protokolünü çalıştıran bir ağ güvenlik kamerası, başka bir firmadan alınan ve içerisinde B protokolü çalıştıran klima sistemi ile bir başka firmadan alınan ve C protokolünü destekleyen televizyonun bir ev ağı içerisinde çalışabilmelerinin imkanı yoktur. Çünkü bu durum farklı dilleri konuşan bir grup insanı bir odaya kapatıp iletişim kurmalarını beklemeye benzer. Aynı şekilde bu aygıtların oluşturduğu bir akıllı ortama D protokolünü destekleyen bir cep telefonu aracılığı ile erişip bu aygıtları kontrol etmek gibi bir imkana sahip olunamayacaktır. Bu karmaşayı ortadan kaldırabilmek için aygıt kontrolü bağlamında başta aygıt kontrol protokolü olmak üzere standart bir alt yapının oluşturulması şarttır. Bu sayede tüm firmalar kabul edecekleri standart protokol ve tanımlara bağlı kalarak aygıtlarını geliştirecekler ve böylece aygıt kontrolü evrensel anlamda tüm aygıtlar tarafından aynı protokol kullanılarak aynı süreçler çerçevesinde gerçekleştirilebilecektir.

Bir aygıtın ağ aygıtı olabilmesi için en başta bulunduğu yerel ağa ya da doğrudan Internet'e bir bağlantısının olması gerekir. Bu bağlantıyı sahip olduğu ağ arabirim kartı ve kendisine atanmış ağ adresini kullanarak gerçekleştirir. Eğer aygıt bir ağ arabirim kartı ve bir ağ adresine sahip değilse, o zaman ağ bağlantısını bir aracı birim tarafından yapması gerekir. Bu birim, **aygıt denetleyicisi (appliance controller)**, bu birime bağlı olan ağ aygıtları da **"legacy" (ilkel)** aygıtlar olarak adlandırılırlar. Bu aygıtlara örnek olarak günlük hayatta kullanmakta olduğumuz ve ağ desteği olmayan gece lambası, kahve makinesi, dijital çalar saat ...gibi aygıtlar

verilebilir (Şekil 1.1.). Aygıtların kontrolü doğal olarak bir kullanıcı tarafından doğrudan çalıştırılan ya da bazı koşullara göre otomatik olarak çalışması için ayarlanan başka bir ağ aygıtı kullanılarak gerçekleştirilir. Örneğin bir kullanıcının evindeki ağ çalar saatinin alarmını cep telefonu aracılığı ile belli bir saate kurması esnasında denetimi gerçekleştiren aygıt cep telefonuyken, çalar saatin ayarlanan zaman dolduğunda örneğin evdeki klimayı çalıştırması sırasında denetimi otomatik gerçekleştiren ağ aygıtı çalar saattir.

Şekil 1.1. 'de örneklenen iletişim senaryosunda da görüldüğü gibi örneğin gece lambası, kahve makinesi gibi bazı aygıtlar, ağ adresine sahip olamadıklarından yerel ağa aygıt denetleyicileri aracılığı ile bağlanmışlardır. Bununla birlikte yerel ağa dışarıdan erişimi, dolayısıyla da ağ aygıtlarının kontrol güvenliğini ve gizliliğini sağlamak için gerekli olan **Ateş Duvarı (Firewall) / NAT (Network Address Translation)** mekanizmalarını içeren ve ağ geçidi görevi gören birim de şekilde belirtilmiştir. Bu birime **Mesken Ağ Geçidi (Residential Gateway - RGW)** adı verilmektedir. Mesken ağ geçidinin aygıt kontrolü açısından görevi özetle, kendisine dışarıdan gelen iletileri incelemek, iletinin hedefi kendisine bağlı olan ağ aygıtlarından birisi ise iletiyi ona yönlendirmektir. Diğer taraftan bazı durumlarda ağ aygıtından gelen yanıt iletilerini de istekte bulunan dış birime iletmekle de yükümlüdür.

Bir ağ aygıtının sahip olduğu bazı temel işlevleri vardır:

- *Oturum **(Session)** denetimi:* İstemci oturum başlatma işiyle görevlidir. Oturum, genelde bir güvenlik ağ kamerasından **görüntü akışı (streaming)** almak gibi sürekliliği olan bir işlem için gerçekleştirilir. Sunucu ise oturum başlatma isteklerini ele almakla görevlidir.
- *Ağ aygıtı denetimi:* İstemci, ağ aygıtlarının sağladığı hizmetler için istekte bulunur ve bu hizmetlerin denetimini yapar. Buna örnek olarak ev ortamındaki klimanın çalıştırılması, sıcaklığının ayarlanması ya da bir araba içindeki açık unutulmuş müzik sisteminin kapatılması verilebilir. Burada sunucu bu istekleri ele almakla görevlidir.

- *Durum **(Status)** denetimi:* İstemci, ağ aygıtının o an bulunduğu durumu sorgulamak için kullanılır. Durum bilgisi, aygıtın sahip olduğu algılayıcılardan edinilen veriler olabileceği gibi, o an sahip olduğu çalışma konfigürasyonu bilgileri de olabilir. Sunucu bu bilgileri toplayıp istemcinin sorgusunu cevaplamakla görevlidir.
- *Olay denetimi:* İstemci, başka bir ağ aygıtında gerçekleşebilecek belirli bir olaya yönelik bildirim alma isteğinde bulunmak için kullanılır. Buna örnek olarak kapı zilinin çalması durumunda bu olayın bildirimi cep telefonundaki istemci aracılığı ile kapıdaki sunucudan istenebilir. Bu tür bildirim isteklerinin ele alınıp, olay gerçekleştiğinde ilgili istemciye bildirilmesi işlemleri de sunucu tarafından gerçekleştirilir.

Bir ağ aygıtının bu işlevlerinden en az birisini kesinlikle gerçekleştirmesi gerekir. Bunların dışında kalan keşif ve kayıt işlevleri seçimlik olup, bir ağ aygıtında bulunmaları şart değildir.

- *Keşif **(Discovery)**:* İstemci, bir yerel alan içindeki belirli **öz niteliklere (attribute)** ya da işlevlere sahip ağ aygıtlarının belirlenmesi için kullanılır. Örneğin evdeki televizyonlardan renkli olanlarının belirlenmesi istemci aracılığı ile gerçekleştirilir. İsteği alan aygıt aranan öznitelik bilgilerine uyuyorsa sunucusu aracılığı ile cevap verir.
- *Kayıt **(Registration)**:* İstemci, üzerinde çalıştığı ağ aygıtının adres, isim, desteklediği hizmetler gibi kayıt için gerekli bilgileri gönderir. Bu bilgiler sunucu tarafından bir aygıt kayıt veritabanına girilir.

Yukarıdaki ağ aygıt işlevleri kullanılan denetim protokolünün desteği ile, **istemci/sunucu (client/server)** ya da **eşler arası (peer-to-peer)** iletişim şeklini kullanan yazılımlar tarafından gerçekleştirilirler.

Internet ortamının mevcut genel işleyişine farklı görevlerle katkıda bulunan çeşitli protokoller yukarıda belirtilen ağ aygıtı işlevlerini tamamen ya da kısmen gerçekleştirebilecek imkanlara sahiptirler. Fakat her birinin asıl kullanım amacı ve

işleyişi farklı olduğundan, bu protokollerle denetlenen farklı ağ aygıtları arasındaki iletişimi sağlamak oldukça güç olacaktır. Bu zorluğu ortadan kaldırmak için ağ aygıt kontrolü konusunda bazı standartların oluşturulması gerekmektedir. Bunun için de önce ağ aygıt kontrolünün gereksinimleri üzerinde durulması, belirlenen gereksinimlere göre de mevcut Internet alt yapısındaki uygulama katmanı protokolleri incelenmesi gerekir. Burada amaç zaten hazır ve uzun süre denenmiş olan protokol altyapısını kullanabilmektir.

Bunun için şimdiye kadar, ağ aygıtlarının işlevlerinin ve kontrol gereksinimlerinin belirlenmesine yönelik bazı çalışmalar yapılmıştır. Schulzrinne ve ark'nın kişisel Internet aygıt kontrolü konusunda yaptıkları çalışmalar [1], Hughes Software Systems'in yapmış olduğu gereksinim araştırmaları [4], Telcordia Technologies Inc. bünyesinde çalışan Simon Tsang (**IETF (Internet Engineering Task Force)** Internet Personal Appliance Control çalışma grubunun başkanı) , Dave Marples, Stan Moyer'in araştırmaları sonucunda çıkardıkları makale ve **taslak**lar **(draft)** [3,6] ve yine Dave Marples, Stan Moyer ile birlikte Abhrajit Ghosh'un araştırmaları [5] bu çalışmaların başında gelmektedir.

Bu tezin temelini Schulzrinne ve ark'nın kişisel Internet aygıtlarının kontrolüne yönelik yaptıkları çalışma [1] oluşturmaktadır. Schulzrinne ve ark.'nın çalışmasında [1] kişisel Internet aygıtları ve işlevsel bileşenleri olarak aygıt denetimi, olay **üyeliği (subscrition)**, olay **bildirimi (notification)**, keşif, kayıt, durum denetimi ve oturum denetimi hakkında tanımlar yapılmaktadır. Bunun yanında bir aygıt kontrol protokolünün karşılaması gereken gereksinimler genel (zorunlu) ve isteğe bağlı ve bileşenlere yönelik olmak üzere iki grupta sıralanmaktadır. Bu tezde belirlen ve tanımlanan temel gereksinimler Schulzrinne ve ark.'nın çalışmasında bulunan genel gereksinimlere karşılık gelmektedir. Yalnız yine aynı kaynakta belirtilmiş olan "Zamanında ileti aktarım desteği" ile "Güvenilir iletişim desteği" başlıklı gereksinimler bu tezde "Güvenilir iletişim desteği" başlığı altında bir arada düşünülmüştür. Yine Schulzrinne ve ark.'nın [1] çalışmasında sıralanan bileşen gereksinimleri ise bu tezde işlevsel gereksinimler başlığı altında incelenmektedir. Schulzrinne ve ark. aynı kaynakta, mevcut uygulama katmanı protokollerinden

SNMPv3 (Simple Network Management Protocol version 3), SMTP (Simple Mail Transfer Protocol), SIP (Session Initiation Protocol), HTTP (Hypertext Transfer Protocol), SLP (Service Location Protocol) ve **BEEP'i (Blocks Extensible Exchange Protocol)** belirledikleri protokol gereksinimlerine uygunlukları bakımından yüzeysel olarak karşılaştırmışlardır. Hatta protokoller incelenirken bazı gereksinimleri karşılayıp karşılayamadıkları hakkında kesin sonuca varılamamıştır. Ayrıca aynı çalışma incelenen protokollerin destekleyemedikleri gereksinimleri karşılayabilmeleri için neler yapılabileceği ile ilgili tanımlar ve açıklamalar içermemektedir. Schulzrinne ve ark. tarafından [1] karşılaştırılan protokoller içerisinde SNMPv3, SMTP ve SIP protokolleri bu tezde "Geniş alan iletişim desteği", "Güvenilir iletişim desteği", "Ağ aygıtları konumdan bağımsız şekilde tek (unique) adreslenebilmeli ve bir ağ adresinde birden fazla mantıksal aygıt bulunabilmeli", Ağ aygıtları konumdan bağımsız şekilde tek (unique) adreslenebilmeli ve bir ağ adresinde birden fazla mantıksal aygıt bulunabilmeli", "Güvenlik, kimlik denetimi ve gizlilik desteği", "Aygıt taşınırlık desteği", "IPv6 desteği", "Esnek ileti yapısına sahip olma" ve "Ağ aygıt kontrol protokolünün yapısı karmaşık olmamalı" başlıkları ile tanımlanan *temel* gereksinimler ve "Aygıt denetimi", "Oturum denetimi", "Olay denetimi", "Durum denetimi" başlıkları ile tanımlanan *işlevsel* gereksinimler göz önünde tutularak incelenmiştir.

İncelenen protokollerden SNMPv3'ün "Ağ aygıtları konumdan bağımsız şekilde tek (unique) adreslenebilmeli ve bir ağ adresinde birden fazla mantıksal aygıt bulunabilmeli" başlıklı temel gereksinimi karşılayamadığı belirlenmiş, konumdan bağımsız tek (unique) adresler oluşturulabilmesi için bir tasarım gerçekleştirilmiştir. Aynı şekilde tasarlanan bu adresleme mantığı kullanılarak protokol için "Aygıt taşınırlık desteği" başlıklı gereksinimi karşılamaya yönelik de bazı eklentiler tasarlanmıştır. Ayrıca SNMPv3'ün standart ileti türlerinden yararlanılarak İşlevsel gereksinimleri karşılayabileceği belirlenmiştir.

SMTP protokolü incelendiğinde kısmen karşıladığı "Ağ aygıtları konumdan bağımsız şekilde tek (unique) adreslenebilmeli ve bir ağ adresinde birden fazla mantıksal aygıt bulunabilmeli" başlıklı temel gereksinimi tam olarak karşılayabilmesi

için standart SMTP bileşenlerine ek olarak iki adet yeni varlık tanımı yapılmıştır. Bu varlık tanımlarını da kullanarak protokolün desteğinin bulunmadığı "Aygıt taşınırlık desteği" başlıklı gereksinimi karşılamaya yönelik olarak yeni ileti başlık alanları ve mekanizmalar tasarlanmıştır. Aynı şekilde yeni ileti başlık alanları tanımlanarak, bu alanlar yardımıyla SMTP'nin işlevsel gereksinimleri desteklemesi sağlanmıştır.

İncelenen bir diğer protokol olan SIP'in standart tanımı ve aygıt kontrolü bağlamında tanımlanmış olan uzantıları sayesinde temel ve işlevsel gereksinimleri yeterince ve eksiksiz karşılayabildiği anlaşılmıştır. Son olarak da aygıt kontrolü için SNMPv3, SMTP ve SIP arasında en uygun protokol olarak belirlenen SIP kullanılarak yerel bir uygulama geliştirilmiştir. Bu uygulama ile bir elektronik sesli ajanda aygıtı benzetimi yapılmıştır. Bu sayede sesli ajanda cihazında çalışacağı farz edilen sunucu modülü ile Java applet'i şeklinde web ortamında çalışan istemci modülü arasındaki SIP iletişimi örneklenmiştir.

3. GEREKSİNİMLER

Ağ aygıtlarının iletişimi için kullanılacak olan ağ protokollerinin, aygıtlar arası iletişimi en başta güvenilir, emniyetli, hızlı ve etkin bir şekilde sağlaması beklenir.

Bu bölümde aygıt kontrolü için kullanılacak protokollerin zorunlu olarak karşılaması gerektiği düşünülen gereksinimler temel ve işlevsel gereksinimler olmak üzere iki ana başlık altında tanımlanmaktadır.

3.1. Temel Gereksinimler

Bu bölümde aygıt denetim protokolleri için belirlenen temel gereksinimler açıklanacaktır.

3.1.1. Geniş-alan iletişim desteği

Bir ağ aygıtının uzaktan, yani yerel etki alanı dışından denetiminin gerçekleştirilebilmesi için, denetim protokolünün bu aygıtla geniş alan ağı üzerinden iletişim kurabilmeye olanak veriyor olması gerekir. Günümüzde Internet omurgasını **TCP/IP (Transmission Control Protocol / Internet Protocol)** modelinin oluşturduğu düşünülürse, denetim protokollerinin IP tabanlı TCP ya da **UDP (User Datagram Protocol)** aktarım protokolleri üzerinden veri aktarımı yapması beklenir. Bununla birlikte, geniş alan desteğinin tam anlamıyla sağlanabilmesi için ilgili protokolün TCP/IP dışında kullanılmakta olan ağ ortamları (**IPX/SPX (Internetwork Packet eXchange)** - NetWare, **NetBEUI (Network BIOS Extended User Interface)** - Microsoft, AppleTalk - Macintosh, **SNA (Systems Network Architecture)** - IBM mainframe) üzerinde de çalışabiliyor olması gerekir.

3.1.2. Güvenilir iletişim desteği

Ağ ortamında güvenilir iletişim, aktarılan verilerin kaynaktan hedefe bozulmadan, hatasız bir şekilde aktarılmasını sağlayan mekanizmalarla gerçekleştirilir. Bu mekanizmalar, kullanılan aktarım protokolünün bünyesinde doğrudan bulunabileceği gibi, üst katmanlarda geliştirilen uygulamaların sorumluluğuna da bırakılabilmektedir. Aktarım protokolü güvenilir aktarıma imkan

vermeyen bir ortamda seçilen ağ aygıt kontrol protokolünün bu görevi yürütmesi gerekir.

3.1.3. Ağ aygıtları konumdan bağımsız şekilde tek (unique) adreslenebilmeli ve bir ağ adresinde birden fazla mantıksal aygıt bulunabilmeli

Ağ üzerindeki her aygıt iletişim kurabilmek için ayırt edici bir adrese ihtiyaç duyar. Bu adres onun ağ üzerinde tanınmasını, ona gönderilen iletilerin yerine ulaşabilmesini sağlar. Günlük hayatta farklı alanlarda kullanılan pek çok cihazın birer ağ aygıtı haline getirilebileceği düşünüldüğünde, bu aygıtların büyük bir kısmının kullanım konumunun zaman içinde değişkenlik gösterebileceği çok açıktır. **PDA (Personal Digital Assistant)**, cep telefonu gibi doğasında konumdan bağımsız kullanım özelliği olan aygıtların yanında buzdolabı, güvenlik kameraları, mikrodalga fırın gibi cihazların odadan odaya, ya da evden eve konum değiştirmeleri sonucunda takip edilebilmeleri için daha basit ve konumdan bağımsız bir şekilde adreslenebilmeleri şarttır.

Bunun yanında ev ya da ofis gibi bir yönetsel **alan (domain)** içerisinde kurulmuş olan yerel ağa bağlı aygıtların Internet'e tek bir mantıksal ağ adresi üzerinden (mesken ağ geçidinin adresi) açılması güvenlik açısından karşılanması gereken bir gereksinimdir. Böylece aygıtlar ile dış dünya arasındaki iletişim denetim altında tutulmuş olur. Bunun yanında günümüzde kullanılan ağ adresleme mantığı (IPv4) dünyadaki tüm ağ aygıtlarını ayrı ayrı adreslemeye yetecek yapıda olmadığından, ağ aygıtlarını belirli yönetimsel alanlar içerisinde düşünmek ve tek bir adres üzerinden Internet ile iletişimlerini sağlamak şu an için bir gerekliliktir.

Sonuç olarak ağ aygıt kontrolünde kullanılacak olan protokolün mantıksal adresi değişse bile ağ aygıtının **DNS (Domain Name System)** adı aracılığı ile tanınmasını sağlayan DNS mantığına benzer konumdan bağımsız bir adresleme yöntemini kullanıyor olması şarttır.

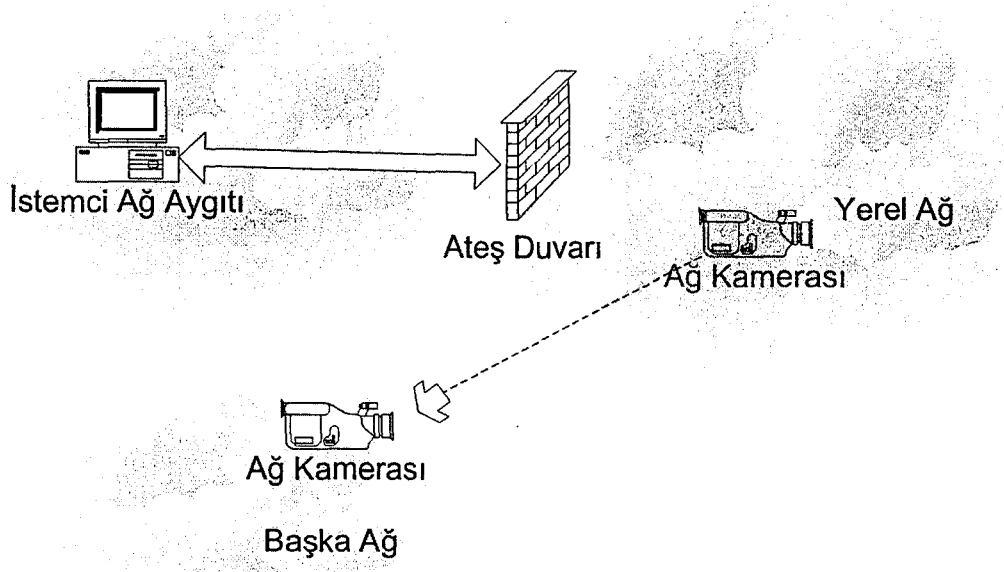
Ayrıca mantıksal ağ adresine sahip bir bağlantı noktasına birden fazla ağ aygıtının bağlanması da aygıt erişim güvenliğini desteklemesi bakımından ve mevcut

bir ortamda kapı zilinin bağlı olduğu aygıt denetleyiciye erişim için gerekli bilgileri elde etmiş olan bir izinsiz kullanıcı kapıyı, anahtar kullanmadan açabilir, evin sahibine yanlış ileti göndererek kapıyı onun açmasını sağlayabilir, evde aynı sisteme bağlı olan örneğin mikrodalga fırını çalıştırarak daha trajik sonuçlara bile neden olabilir.

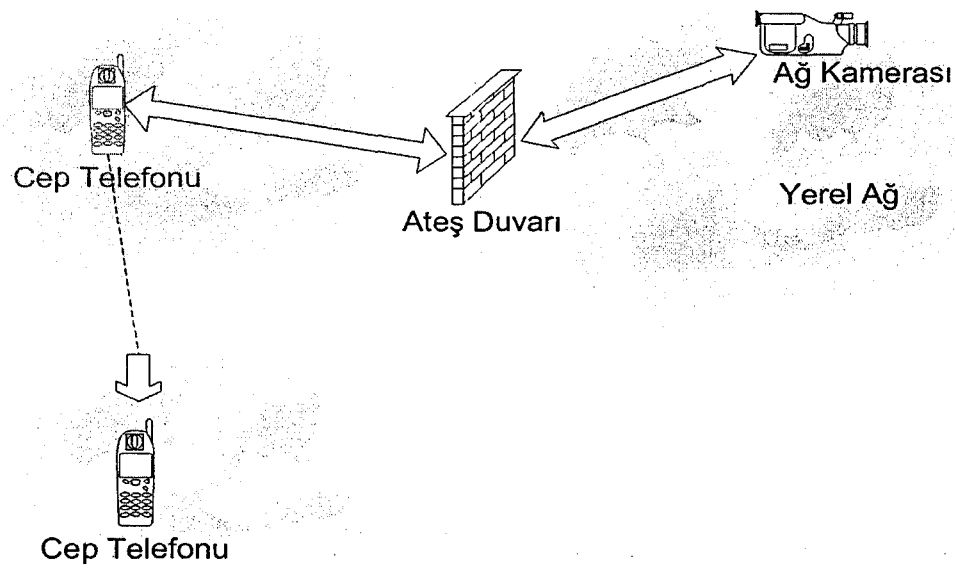
Bu yüzden seçilecek aygıt kontrol protokolünün ilgili aygıtlara erişimde kimlik denetimi, yetkilendirme, şifreleme gibi ağ aygıtının iletişim ve işletim bütünlüğünü koruyacak çeşitli mekanizmalara sahip olması şarttır.

3.1.5. Aygıt taşınırlık desteği

Aygıt taşınırlığı, bir ağ aygıtının alt ağlar arasında hareket etmesi olarak düşünülür [7]. Şekil 3.2. ve Şekil 3.3.'de örneklenen bu duruma göre ağ aygıtı bir alt ağdan diğerine geçtiğinde sahip olduğu mantıksal ağ adresini değiştirmek durumundadır. Bu değişikliklerin aygıtın kendi işlevlerini ve ağ üzerindeki mevcut bağlantılarını etkilememesi için, kullanılan aygıt kontrol protokolü kapsamında bazı işlevlerin gerçekleştirilmesi gerekir. Bu sayede kullanıcı bir ağ aygıtı aracılığı ile farklı konumlardaki diğer ağ aygıtları ile konum değişikliğinden etkilenmeden iletişim kurabilir. Örneğin evindeki klimaya arabada giderken cep telefonu aracılığı ile hava sıcaklığını 20 dereceye getirmesini söyleyebilir. Yalnız burada dikkat edilmesi gereken nokta, seçilen aygıt kontrol protokolünün aynı zamanda konumdan bağımsız ve tek adresleme yöntemine sahip olması gerekir ki konum değişikliğinde ilgili aygıt diğerlerinden ayırt edilebilsin.



Şekil 3.2. Yerel Etki Alanı içindeki ağ kamerasının farklı bir alana taşınması



Şekil 3.3. İstemci ağ aygıtının (cep telefonu) taşınırlığı

3.1.6. IPV6 desteği

Günümüzde ağ aygıtlarının adreslenmesinde IP adresleme mantığı kullanılır. Zaten Internet de IP tabanlı bir omurgaya sahiptir. IP adresleme mantığında mantıksal ağ adresleri 4 Byte'tan oluşan değerlerdir. Halen kullanılmakta olan bu adresleme yöntemi IPv4 olarak da adlandırılmaktadır.

Internet'in son 10 yıl içerisindeki büyüme hızının beklenenin çok üzerinde olması nedeniyle çok yakın zamanda 32 bitlik IP adreslerinin tükeneceği görüldü. Bunun sonucunda IP adresleme yönteminin, dolayısıyla bu yöntemi kullanan Internet Protokol'ünün (IP) yeni sürümü üzerinde çalışılmaya başlandı. Bu yeni sürüm 128 bit adreslemeye imkan verebilecek. Böylece dünyadaki tüm ağ aygıtları birer sabit ağ adresine kavuşabilecekler. Yeni sürüm IPv6 ya da **IPng (IP next generation)** olarak adlandırılmaktadır.

Geliştirilen bu yeni sürümün zaman içerisinde IPv4'ün yerini alması planlanmaktadır. Ancak bu geçiş parça parça, alt ağlar bazında düşünüldüğünden seçilecek aygıt kontrol protokolünün mevcut Internet yapısı içerisinde bu iki sürüme de uyumlu olarak çalışması gerekmektedir.

Bununla birlikte, yeni sürümün sağladığı oto konfigürasyon yöntemi ile ağ aygıtlarının ağ konfigürasyon işlemleri otomatik olarak gerçekleşebilecek. Böylece kullanıcının aygıtın temel işleyişi dışında bir şey bilmesine gerek kalmayacak. Oto konfigürasyon sayesinde aygıt alt ağ değişikliğinde otomatik olarak ağ adresini değiştirebilecek. Bu sayede de taşınırlık gereksinimine yönelik de kolaylık getirilmiş olacak.

3.1.7. Esnek ileti yapısına sahip olma

Mevcut uygulama katmanı protokollerinin genelinde kullanılmakta olan ileti formatı :

İleti Formatı = Başlık + **Gövde (payload : veri alanı)**

olmak üzere iki parçadan oluşan bir yapıdır. Başlık bölümünde temel ağ aygıt işlevlerinden amaca uygun olanı ile ilgili bilgi bulunur. Gövde bölümünde ise, iletinin gönderileceği aygıttan yapması beklenen işleme yönelik bir işlem tanımı olması

beklenir. Bu bir komut ya da veri olabilir. Bu bölümün çok esnek tutulması gerekir ki, ilgili aygıtın özelliklerine göre kolayca oluşturulabilsin. Buna örnek olarak aygıt kontrolü için seçilmiş olan protokolün, temel aygıt işlevlerinden ağ aygıt kontrolü için tanımladığı "YAP" adında bir yönteminin olduğunu varsayalım. Bu protokolün kullanıldığı ortamda bir kişi evindeki ağ aygıtlarından önce klima sisteminden 1 saatliğine evi 25 derece sıcaklıkta ısıtmasını, daha sonra da ortamdaki lambaların durumlarını öğrenmek için yapacağı sorgulama sonucunda açık unutulduğu ortaya çıkan banyo lambasını ilgili aygıt denetleyiciden kapatmasını isteyebilir. Protokol her iki istek için de ilgili aygıtlara ya da onların bağlı olduğu mesken ağ geçidine "YAP" yöntem iletisini gönderecektir. Ancak burada veri alanı bölümünün içeriği önem kazanır. Bu bölümde klima için farklı, lamba aygıt denetleyicisi için farklı komutlar ya da veriler olmak durumundadır. Bu örnekten de görüldüğü gibi seçilen protokolün kullanıldığı ortamdaki aygıtların işlevlerinden, özelliklerinden bağımsız olarak çalışması gerekir. Yani veri alanı bölümünü tanımlanmaya yönelik çok fazla kısıtlamaya sahip olmaması gerekir.

3.1.8. Ağ aygıt kontrol protokolünün yapısı karmaşık olmamalı

Günlük hayatta kullanılan pek çok cihazı birer ağ aygıtı haline getirmek mümkündür. Bu yüzden de bu cihazların işlevlerinin yanında boyutları ile de orantılı olarak işlem güçleri de değişkenlik gösterir. Bir cihazın ağ aygıtı olabilmesi için aslında küçük bir bilgisayar gibi çalışması gerekir. Yani en başta belleğinin, işlemcisinin, ağ arabirim kartının bulunması gerekir. Ancak ağ aygıtlarının çoğu bir genel amaçlı işletim sistemi içeremeyecek kadar kısıtlı kaynağa ve işlem gücüne sahiptir. Bu yüzden de seçilen denetim protokolünün bellekte olabildiğince az yer kaplaması ve karmaşık işlemler gerçekleştirmemesi, yani **yalın (lightweight)** bir yapıya sahip olması gerekir.

3.2. İşlevsel Gereksinimler

Bir ağ aygıtının yukarıda belirtilen temel gereksinimleri dışında bazı işlevsel gereksinimleri de bulunmaktadır. Bunlar aslında daha önce belirtilen ağ aygıtı temel işlevlerine karşılık gelmektedir. Bunlardan zorunlu olanlar sırasıyla incelenirse:

3.2.1. Aygıt kontrol protokolünde aygıt denetimi

Ağ aygıt kontrol protokolünün ağ aygıtlarının aralarındaki hizmet isteme/sağlama iletişimlerini sağlaması gerekir. Yani örneğin "Sokak kapısını aç" şeklindeki bir denetim cümlesinin ve bu cümleye karşılık verilecek olumlu/olumsuz yanıtın protokol bünyesinde karşılıklarının olması gerekir.

3.2.2. Aygıt kontrol protokolünde durum denetimi

Protokolün ağ aygıtının sahip olduğu algılayıcılardan elde edilen ya da o anki çalışma durumu ile ilgili bilgilere erişebilmeye imkan verebilmesi gerekir. Yani örneğin "Evin sıcaklığı nedir?" gibi durum denetim sorularına ve cevaplarına yönelik destek sağlaması gerekir.

3.2.3. Aygıt kontrol protokolünde olay denetimi

İstemci tarafın başka bir ağ aygıtından belirli bir olayın olması durumunda bildirim alma isteğinde bulunabilmesini ve olay meydana geldiğinde bildirimin gerçekleşmesi için destek sağlaması gerekir. Yani "Kapı zili çalınca bana bildir" cümlesinin karşılığı ve bu bildirim işleminin gerçekleşmesi protokol tarafından desteklenmelidir.

3.2.4. Aygıt kontrol protokolünde oturum denetimi

Protokolün sürekliliği olan bir işlem için oturum başlatma ve denetleme özelliklerine sahip olması gerekir. Yani protokolün "Sokak kapısının kamerasını göster" cümlesindeki gibi görüntü akışına yönelik bir oturumu kurma ve yönetme yeterliliğinde olması gerekir.

4. ÇÖZÜM SEÇENEKLERİ

Bu bölümde sırası ile SNMPv3, SIP ve SMTP protokollerinin her biri için önce protokol yapısı ve işleyişi hakkında genel bilgiler verilecektir. Daha sonra da ilgili protokolün 3. bölümde tanımlanmış olan temel ve işlevsel gereksinimlere uygunluğu incelenecektir.

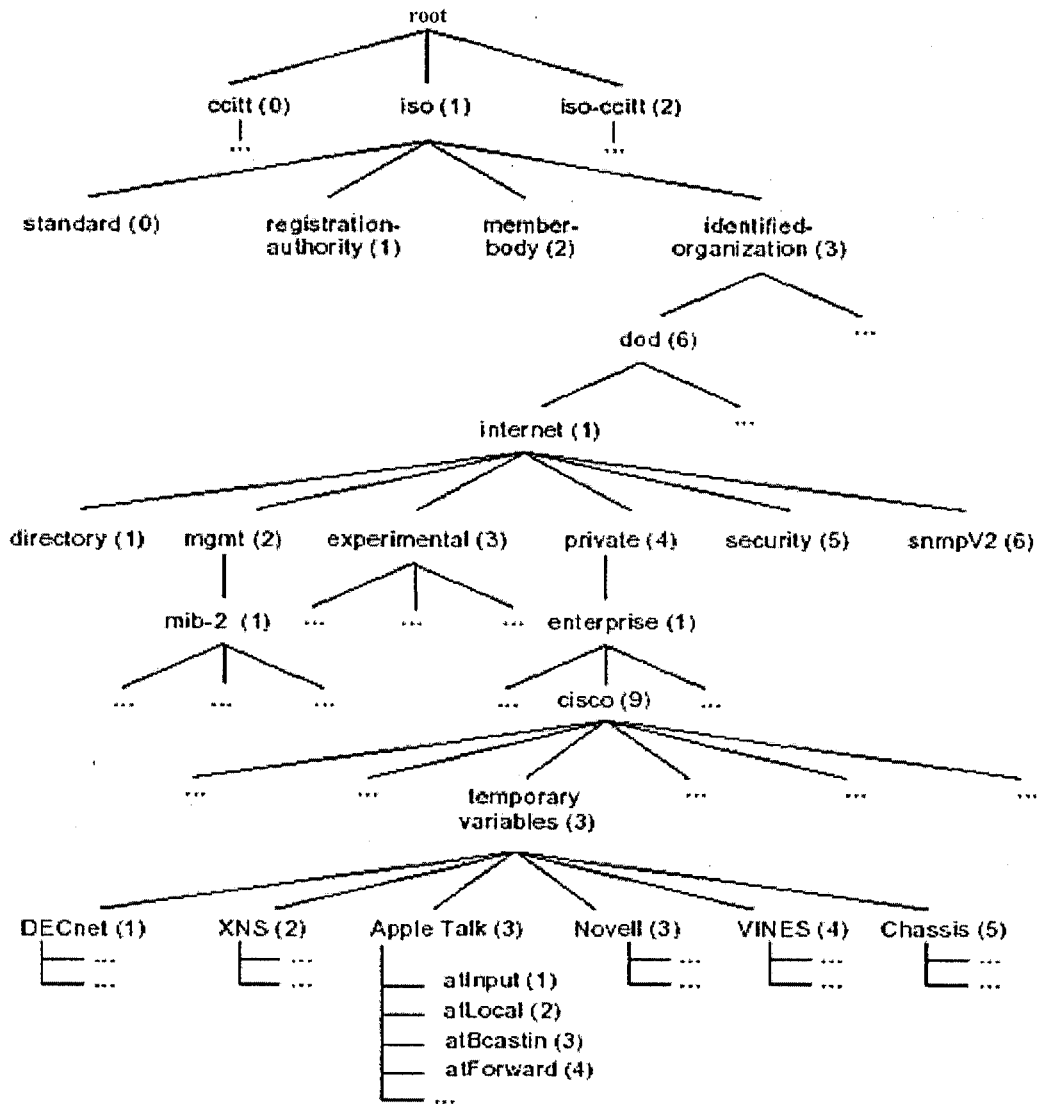
4.1. SNMP (Simple Network Management Protocol)

Bu bölümde SNMP'nin tarihçesi, yapısı ve işleyişi hakkında bilgiler verilecektir. Daha sonra da bu protokolün tanımlanmış olan temel ve işlevsel gereksinimlere uygunluğu incelenecektir.

4.1.1. SNMP'nin tarihçesi ve yapısı

SNMP, ilk defa 1988 yılında IETF tarafından standart haline getirilmiş bir ağ yönetim protokolüdür. Görevi kısaca ağ aygıtları arasında yönetim bilgilerinin alışverişini kolaylaştırmaktır. İlk sürümü SNMPv1 olarak adlandırılmış, günümüze gelene kadar çeşitli zamanlarda, mevcut gereksinimler göz önünde tutularak güncellenmiştir. Deneysel amaçla çıkarılmış olanlar dışında SNMPv1, SNMPv2 ve yakın zamanda standart haline getirilmiş olan SNMPv3, protokolün temel sürümleridir. Aslında ağ yönetimi **izleme (monitoring)** ve **denetim (control)** olmak üzere iki işlevden oluşsa da, SNMPv1 sürümünün günümüzde kullanılması durumunda ağ yönetiminin sadece izleme işlevi gerçekleştirilebilir. Bunun sebebi ise SNMPv3'e kadar, bazı deneysel sürümler dışında SNMP protokolünün hiçbir sürümümde güvenlik mekanizmalarının bulunmayışıdır. Zaten en son geliştirilen SNMPv3 sürümünün diğer sürümlere göre en önemli üstünlüğü de sağladığı güvenlik mekanizmalarıdır.

Internet'i oluşturan çok üreticili ağ ortamlarındaki iletişim ve eşgüdümü kolay bir şekilde sağlayabilmek için SNMP protokolünün de kullandığı yaklaşım, ağ ortamını birbirleri ile işbirliği içinde iletişim kuran varlıkların oluşturduğu bir bütün olarak görmektir. Burada, yönetilen uç birimler ve yönetim uç birimleri olmak üzere



Şekil 4.2. MIB sıra düzensel ağaç yapısı [8]

Yönetilen uç birimler kendi donanım, konfigürasyon parametreleri, performans istatistikleri gibi bilgileri bir dizi **yönetilen nesne** içinde tutarlar. Yönetilen nesneler **Yönetim Bilgi Tabanı (Management Information Base - MIB)** olarak adlandırılan sıradüzensel yapıdaki bilgi veritabanları içerisinde tek **olgu/değişken (scalar object)** ya da aynı nesnenin olgularının oluşturduğu **tablo yapıları (tabular object)** şeklinde tutulurlar (Şekil 4.2.). Ağ yönetim protokolü tarafından erişilen bu nesneler **SMI**

(Structure of Management Information) [9] kullanılarak tanımlanıp MIB'leri oluştururlar.

SMI yönetilen nesnelerin veri türlerini, söz dizimlerini ve isimlendirilme ölçütlerini belirleyen kurallar bütünüdür. Yönetilen nesnelere örnek olarak **atInput** verilebilir. Bu nesne bir yönlendirici ara yüzüne gelen AppleTalk paketlerinin toplam sayısını tutan bir sayısal değişkendir ve ilgili yönlendiricinin MIB'si içinde yer alır.

MIB içerisinde bir nesnenin bir ya da daha çok **olgusu** bulunabilir. Olgular aynı nesnenin farklı değerlerini tutan değişkenler olarak da düşünülebilir. Bu olguların her biri aynı yönetsel alan içindeki bir **bağlam**'a **(context)** aittir.

Bağlam aynı alan içindeki fiziksel ya da mantıksal aygıtlardan birisi olabileceği gibi bir aygıt grubu, bir aygıtın ya da aygıt grubunun alt kümesi de olabilir. Bu yüzden bir yönetsel alan içinde bir nesnenin olgusunun hangi bağlama ait olduğunu belirleyebilmek için o nesnenin nesne türü ve olgu belirteci (nesnenin kaçıncı olgusu/değişkeni) ile birlikte *contextEngineID* ve *contextName* bilgilerine bakılır.

- **contextEngineID :** Bir yönetsel alan içinde bir SNMP varlığını ayırt etmek için kullanılır. contextName ile birlikte aynı yönetsel alan içindeki bağlamları belirler.
- **contextName :** Bir bağlamı isimlendirmek için kullanılır. Bir SNMP varlığı içindeki her bir contextName değeri tek olmalıdır.

Örneğin, **ifDescr** [10] türü yönetilen nesne, bir ağ ara yüzünü tanımlamak için kullanılır. Aygıt-X ağ aygıtının ilk ağ ara yüzünü tanımlayabilmek için dört parça bilgiye ihtiyaç vardır [10]:

- Aygıt-X'in yönetim bilgilerine erişimi sağlayan SNMP varlığının snmpEngineID değerine,
- Aygıt-X'in contextName değerine = Aygıt-X,
- Yönetilen nesnenin türüne = ifDescr,
- Nesnenin kaçıncı olgusu olduğu bilgisine = 1 (ilk ağ ara yüzü).

Burada da görüldüğü gibi Aygıt-X'in tüm ağ ara yüzlerinin oluşturduğu yapı bir bağlamdır.

SNMP'nin MIB tanımları ve SMI için kullandığı söz dizim kuralları **ASN.1** olarak adlandırılır. ASN.1 OSI tarafından söz dizim tanımlama amacı ile geliştirilmiş olan **ASN (Abstract Syntax Notation)** adlı üst düzeyli **biçimsel (formal)** bilgisayar dilinin bir alt kümesidir. MIB'ler içinde tanımlı olan nesnelerin değerleri ise iletişim esnasında **BER (Basic Encoding Rules)** kullanılarak kodlanır. BER CCIT (X.209) ve ISO (ISO 8825) tarafından geliştirilen ve standartlaştırılan bir kodlama modelidir ve ASN.1 ile tanımlanmış farklı türdeki nesne değerlerini sekiz bitlik veri katarı şekline dönüştürmek için kullanılır.

Bir yönetilen nesne, nesne adı ya da bunun eş değeri olan **nesne belirleyicisi (object identifier/ object ID/ OID)** ile ayırt edilebilir. MIB sıradüzensel ağaç yapısı düşünüldüğünde, bu ağacın Şekil 4.2.'de görüldüğü gibi bir isimsiz **kök (root)** birimi, ve farklı kuruluşlar tarafından tanımlanmış olan katmanları bulunur. Ağacın en üst katmanındaki nesne belirleyicileri farklı standart kuruluşlarına ait iken, daha alt katmanlardaki nesne belirleyicileri onlarla ilişkili kuruluşlar tarafından tanımlanırlar. Satıcı firmalar ise kendi ürünlerine yönelik olarak yönetilen nesnelerden oluşan özel dallar tanımlayabilirler. Henüz standartlaşmamış MIB'ler ise **"deneysel" (experimental)** olarak adlandırılmış dalda tutulurlar. Şekil 4.2.'de de görüldüğü gibi **atInput** nesnesini ayırt etmek için kökten itibaren oluşturulan nesne ismi **"iso.identified-organization.dod.internet.private.enterprise.cisco.temporary - variables.AppleTalk.atInput"** ya da yine aynı şekilde oluşturulan nesne belirleyicisi **(OID) 1.3.6.1.4.1.9.3.3.1.** kullanılabilir.

Ajan varlıkların iki temel işlevi bulunmaktadır:

- Yönetici varlıkların MIB içindeki nesne değerlerini elde etmeye ya da değiştirmeye yönelik isteklerine yanıt vermek.
- Yönetilen uçta meydana gelen bir olaydan (sistem çökmesi, trafik yoğunluğunun sınır değeri geçmesi gibi...), **trap (bildirim)** türünde iletiler oluşturarak yöneticiyi haberdar etmek.

Bir ağ aygıtına TCP/IP paketi ile birlikte SNMP kurulması, o aygıtta artık bir SNMP ajanının çalışmaya başlaması anlamına gelir. Bir ağ üzerinde bulunan ağ aygıtlarının tamamının SNMP'yi doğrudan desteklemesi gerekmez. SNMP desteği

olmayan ağ aygıtlarının SNMP yöneticileri ile iletişim kurabilmesi için SNMP **vekil ajan**lardan yararlanılır. Bu varlıklar yönetici ile SNMP desteği olmayan ağ aygıtı arasında çevirmenlik görevi görürler. Ağ aygıt kontrolü bağlamında bu tür ajan varlıklar aygıt denetleyicisi olarak görev yapan birimler tarafından çalıştırılırlar.

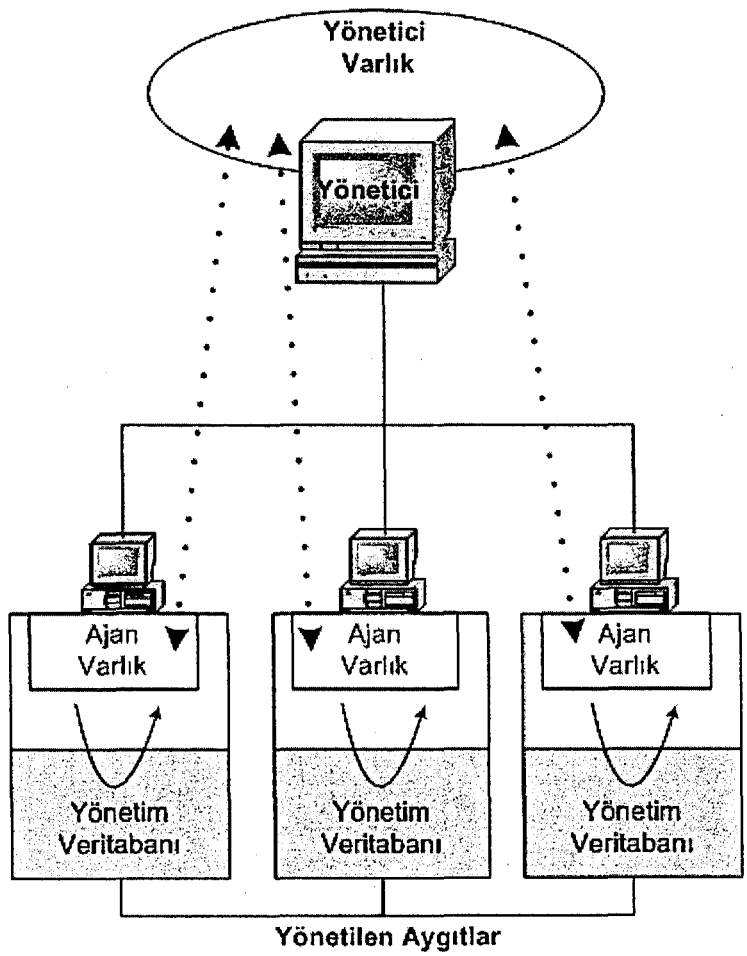
4.1.2. SNMP'nin çalışma mantığı

Ağ içerisindeki yöneticiler, ajanlarla istek iletileri aracılığı ile iletişim kurarlar. Ağ üzerinde birden fazla yönetici varsa, ajan varlık tüm yöneticilerin isteklerini ele almak durumundadır. SNMP yöneticilerinin istekte bulunacakları ajanları çalıştıran uç birimlerin iç yapıları hakkında detaylı bilgi sahibi olmalarına gerek yoktur. Aynı şekilde ajan varlığın da istek iletisini gönderen yönetici birimin iç yapısı ve iletinin bağlamı hakkında detaylı bilgi edinmesi gerekmez. Ajan sadece gelen iletiyi inceleyip onaylar. Onaylanan iletinin istediği işlem gerçekleştirilir. İşlem sonunda ajan tekrar, yeni bir istek iletisini bekleme durumuna geçer. Ajana birden fazla yöneticiden istek iletileri gelmesi durumunda, iletiler geliş sırasına göre işleme alınır.

SNMP protokolünün genel iletişim mantığı, yöneticilerin ajanlardan bilgi edinme ya da değiştirme isteğinde bulunması (Şekil 4.3.), ajanların da buna karşılık yanıt iletisi oluşturması şeklinde olsa da, bazı özel durumlarda ajan varlıkların hiçbir istek iletisi mevcut olmamasına rağmen yönetici varlığı bilgilendirmek için ileti göndermeleri söz konusu olabilir. Gönderilen bu tür iletilere **trap** adı verilir. Özel durumlara örnek olarak ağ üzerindeki bir yönlendiricinin ara yüzlerinden birisinde bağlantı sorununun olması, ağa bağlı bir PC'nin sisteminin çökmesi gibi olaylar verilebilir.

Yöneticiler ve ajanlar arasında gerçekleşen iletişimin temelini yönetilen nesne tarafından saklanan MIB'ler içerisinde bulunan nesne olguları/değişkenleri oluşturur. Yani yöneticiler SNMP MIB'leri içindeki bu değişkenler üzerinde değer edinme ve değiştirme işlemleri yaparak ağ aygıtlarının işleyişi hakkında bilgi edinip, bazı durumlarda da işleyişlerini yönlendirmiş olurlar. MIB'ler **standart** ve **kuruluş (enterprise)** MIB'leri olmak üzere ikiye ayrılırlar.

- **Standart MIB'ler :** Kuruluştan bağımsız olarak, ağın genel işleyişinin yönetimine yönelik geliştirilen veri tabanlarıdır. Bu türe örnek olarak **MIB-II** verilebilir. **MIB-II** TCP/IP ağlarını yönetmek için kullanılan ortak nesnelere yönelik tanımlar içerir.
- **Kuruluş MIB'leri :** Çeşitli kuruluşların ürettikleri ürünlere (nesneler) yönelik olarak geliştirdikleri veri tabanlarıdır. Bu tür MIB'lere örnek olarak CISCO systems, Cabletron, ve IBM'in kendi donanımları için geliştirdikleri veri tabanları verilebilir.



Şekil 4.3. Yönetici ve ajanlar arası iletişim[8]

için bir SNMP varlığı birden fazla ileti işleme modelini içerebilir. İleti işleme modeli ileti formatlarını (v1, v2, v3) tanımlar. İlgili formatın hangi sürüme ait olduğu ise ileti başlığı içindeki **version (sürüm)** alanı ile belirlenir.

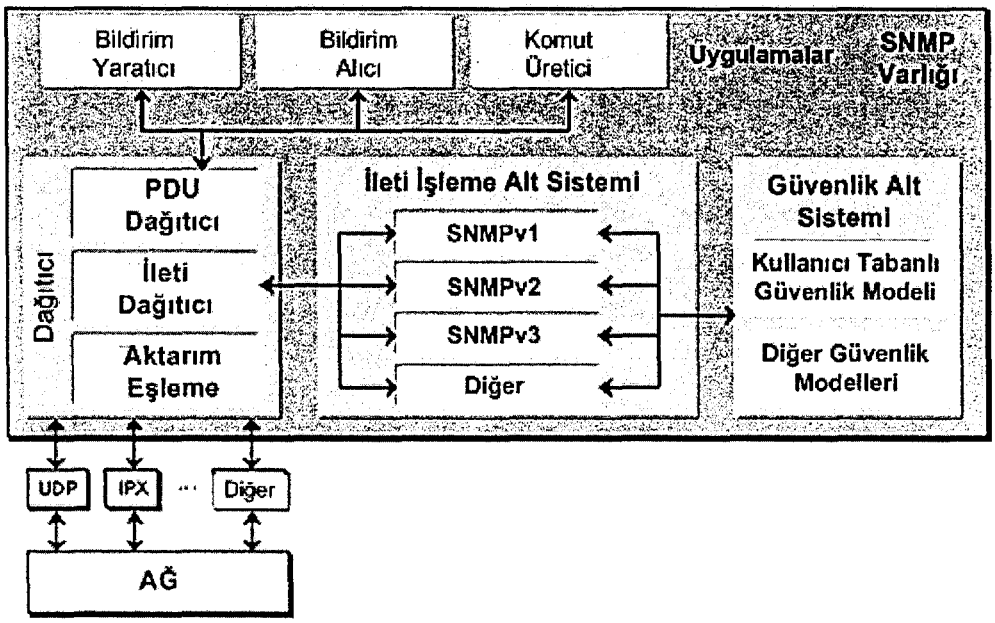
- **Güvenlik Alt Sistemi (Security Subsystem):** Kimlik denetimi, kodlama ve bütünlük mekanizmaları içerir.
- **Erişim Denetim Alt Sistemi (Access Control Subsystem):** Yönetilen kaynaklara erişim denetim işlevini yerine getirir.

Bu alt sistemleri içeren SNMP motoru genel olarak ileti alış verişi, kimlik denetimi ve kodlama ile birlikte yönetilen nesnelere erişim işlemlerinde rol oynar. SNMP motoru ve onu içeren SNMP varlığı snmpEngineID değeri ile ayırt edilirler.

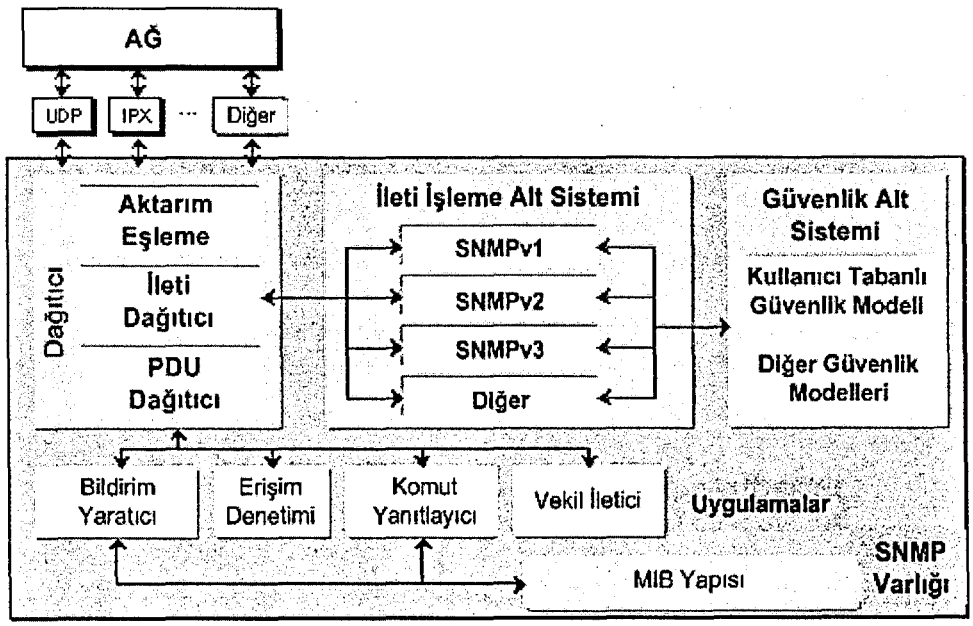
SNMP varlığı, SNMP motoru ile birlikte belirli işlemleri yerine getiren uygulama modülleri içerir. Bunlar (Şekil 4.5.):

- **Komut Üretici (Command Generator)** : Yönetim verisini izleme ve işleme işlemlerini yapar.
- **Komut Yanıtlayıcı (Command Responder)** : Yönetim verilerine (MIB içeriğine) erişimi sağlar.
- **Bildirim Yaratıcı (Notification Originator)** : Zamanuyumsuz ileti oluşturur.
- **Bildirim Alıcı (Notification Receiver)** : Gelen zamanuyumsuz iletileri işler.
- **Vekil İletici (Proxy Forwarder)** : Varlıklar arasında iletilerin aktarılmasını sağlar. Bu varlıklar farklı yönetsel alanlar içerisinde bulunabilir.

SNMPv3 çatısını anlatan örnek bir mimari geliştirilmiştir [9]. Bu mimaride tasarlanan SNMP yönetici varlıkların iç yapısı Şekil 4.5.'de, SNMP ajan varlıkların yapısı Şekil 4.6.'da gösterilmektedir [11].



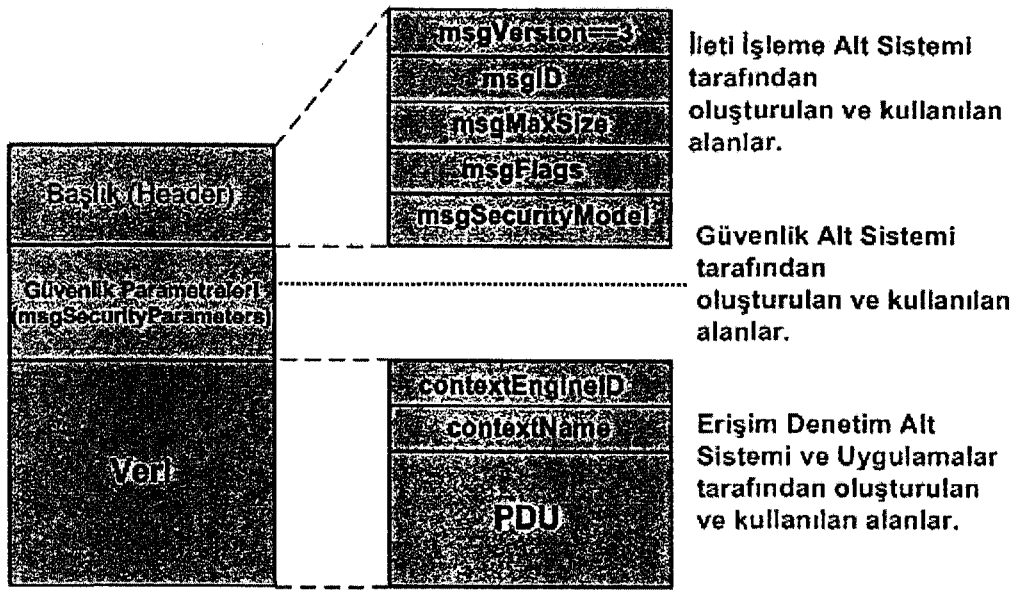
Şekil 4.5. SNMPv3 yönetici yazılımının iç yapısı [11]



Şekil 4.6. SNMPv3 ajan yazılımının iç yapısı [11]

4.1.4. SNMPv3 ileti formatı

SNMPv3 iletileri Şekil 4.7.'de de görüldüğü gibi bazı bölümlerden oluşur. Bunlar:



Şekil 4.7. SNMPv3 ileti formatı

i. Başlık

msgVersion : SNMP protokolünün sürüm bilgisini taşıyan alandır. 0 değeri SNMPv1'i, 1 değeri SNMPv2c adlı ara sürümü, 2 değeri SNMPv2'yi ve 3 ise SNMPv3 sürümünü belirtir. Bu sayede ilgili iletiyi işleyecek ileti işleme modeli seçilir.

msgID : SNMP iletilerinin belirleyicisidir, bu yüzden değeri her ileti için tektir. msgID değeri, iletişim sırasında aktarılan istek iletilerini yanıt iletileri ile ilişkilendirmek amacı ile kullanılır.

msgMaxSize : İstekte bulunan SNMP varlığının izin verdiği azami SNMPv3 ileti büyüklüğünü belirler.

msgFlags : Bu alan SNMPv3 ileti güvenlik seviyesini belirtir. Alanın,

- 0. biti iletinin kimlik denetimli olup olmadığını,
- 1. biti ileti üzerinde gizlilik işlemleri yapılıp yapılmadığını,

- 2. biti iletiyi alan SNMP varlığının, aldığı iletiye karşılık bir rapor iletisini geri döndürüp döndürmemesi gerektiğini gösterir. Bu rapor iletisi istekte bulunan varlığı iletinin düştüğüne ya da yanıt iletisinin gönderilemeyeceğine dair bilgilendirmek amacı ile kullanılır.

msgSecurityModel : Bu alan iletiyi oluştururken hangi güvenlik modelinin kullanıldığını gösterir. SNMPv3 standardı **User Based Security Model'in (USM - Kullanıcı Tabanlı Güvenlik Modeli)** kullanılmasını önermektedir. Bu model ileride daha ayrıntılı açıklanacaktır. Eğer USM kullanılıyorsa bu alanın değeri 3 olacaktır.

ii. Güvenlik Parametreleri (msgSecurityParameters)

Bu alan da güvenlik parametreleri bölümünde bulunur ve seçilen güvenlik modeline özel parametreler içerir. USM güvenlik modelinde bu alan kimlik denetim ve gizlilik parametrelerini içerir.

iii. Veri

contextEngineID : Bir yönetsel alan içinde bulunan bir SNMP varlığını belirler. Bu varlık tek **contextName** değerlerine sahip bir ya da daha çok bağlam içerebilir.

contextName : Bir bağlamın adını içerir. Bu değerler bir SNMP varlığı içinde tektir.

PDU (Protokol Veri Birimi) : SNMP varlıkları arasında iletişim kurmak amacı ile kullanılan veri birimleridir ve iletiyi alan SNMP motorunun yapması gereken işlemleri tanımlarlar. İşlevlerine göre farklı isimler alırlar. Bunlar üç ana grupta incelenir:

İstek (Request) PDU türleri:

- *Get PDU:* Bir yönetici tarafından bir ajandan, bir MIB nesnesinin ilgili olgusunun içerdiği değeri istemek amacı ile kullanılır. Ajan buna **Response-PDU** ile cevap verir.
- *GetNext PDU:* Bir yönetici tarafından bir ajandan MIB içinde bir sonraki sırada bulunan tablo kaydı/nesne olgusu hakkında bilgi almak amacıyla kullanılır. Böylece yönetici MIB'nin yapısını dinamik olarak

keşfetme imkanı bulduğu gibi kayıtları bilinmeyen tablolar üzerinde arama yapabilir.

- *GetBulk PDU:* Bir yönetici tarafından bir ajandan bir bilgi yığınını almak amacıyla kullanılır. Bu sayede büyük veri öbekleri (Örneğin tek bir erişimle ajan MIB'sinin içindeki bir grup nesne olgusunun değerini elde etmek gibi) daha etkin bir erişimle elde edilmiş olur. Ancak bu yöneticiye gelen verinin miktarı **yanıt paketinin (response packet)** büyüklüğü ile sınırlıdır. Bu PDU türünün formatı diğer PDU'lardan farklıdır (Şekil 4.8a.).
- *Set PDU:* Yönetici tarafından bir ajan MIB'sindeki bir tablo satır değerinin güncellenmesi, yeni satır eklenmesi, satırın ortadan kaldırılması ya da ajan varlığa bir işlem yaptırılması amacı ile kullanılır.
- *Inform PDU:* Bir yöneticinin başka bir yöneticiyi önemli bir olay hakkında talep etmediği halde bilgilendirmesi için kullanılır. Bu PDU da bir Response-PDU ile onaylanır.

Bildirim (Trap) türü PDU:

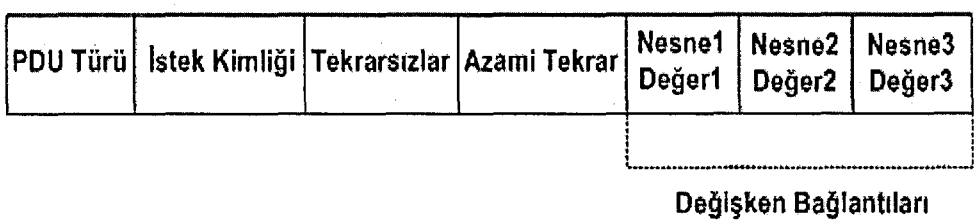
- *SNMPv2-Trap-PDU:* Trap PDU'ları belirli bir olay meydana geldiğinde ajan tarafından yöneticiyi bilgilendirmek üzere kullanılan **zamanuyumsuz (asynchronous)** bilgilendirme paketleridir. SNMPv3, SNMPv2 trap mekanizmasını kullanmaktadır.

Yanıt (Response) türü PDU:

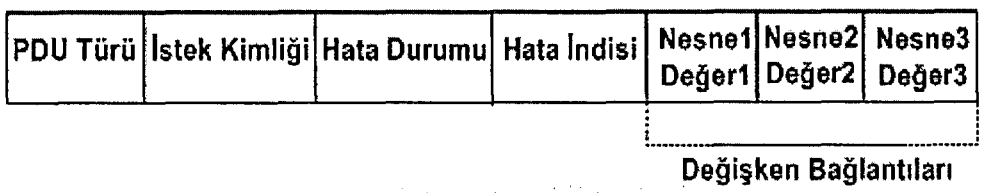
Trap PDU dışında tüm istek PDU türlerini onaylamak için kullanılır. Eğer istenen işlem herhangi bir nedenle gerçekleşemezse, meydana gelen hata ve sebebi ile ilgili bilgileri geri döndürür [11]. Sahip olduğu format istek PDU'ları ile aynıdır.

Şekil 4.8b.'de gösterilen GetRequest, GetNextRequest, InformRequest, SetRequest ve Trap PDU yapısını oluşturan alanlar:

- ***PDU Türü (PDU Type)*** – PDU türünü belirtir. (Get, GetNext, Inform, Response, Set ya da Trap).
- ***İstek Kimliği (Request ID)*** – SNMP istek PDU'larını yanıt PDU'ları ile ilişkilendirir.
- ***Hata Durumu (Error Status)*** – SNMP protokolünün tanımladığı hata türlerinden birisini belirtir. Bu alan sadece yanıt işlemi sırasında kurulur. Diğer işlemler ise bu alanı sıfırlarlar.
- ***Hata İndisi (Error Index)*** – Bir hatayı belirli bir nesne olgusu ile ilişkilendirir. Bu alan sadece yanıt işlemi sırasında kurulur. Diğer işlemler ise bu alanı sıfırlarlar.
- ***Değişken Bağlantıları (Variable Bindings)*** – SNMP PDU'sundaki veri alanı olarak kullanılır. Her bir değişken bağlantı alanı belli bir nesne olgusunu, mevcut değeri ile ilişkilendirir (Get ve GetNext istek PDU'larında değer alanları boş bırakılır).



Şekil 4.8a. GetBulk PDU formatı

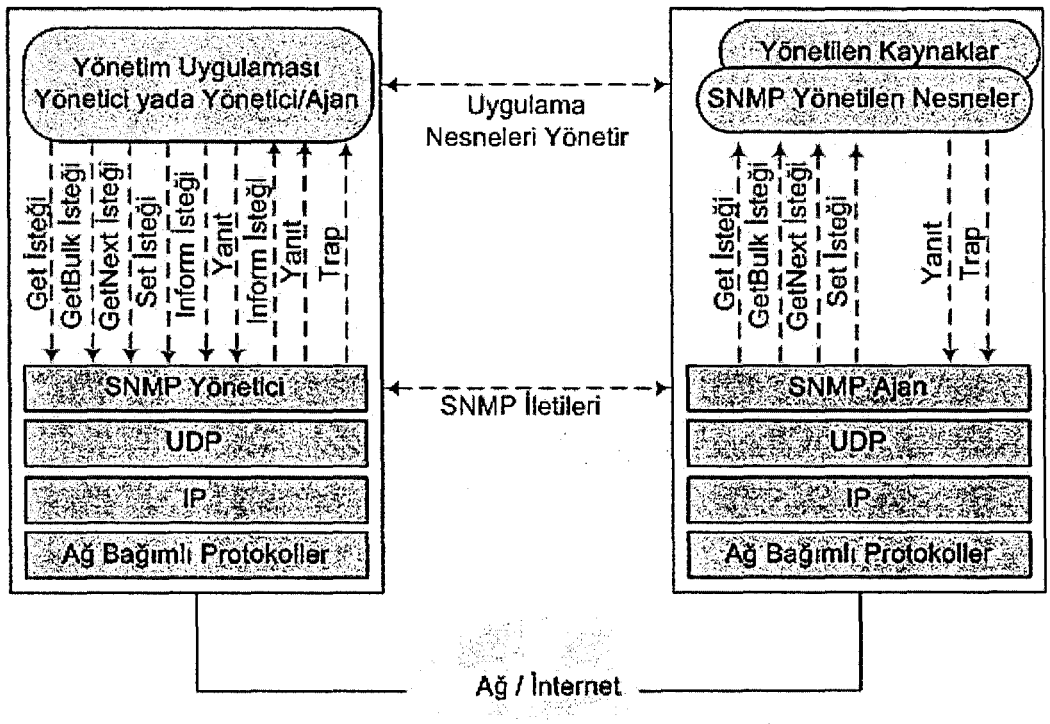


Şekil 4.8b. Get, GetNext, Inform, Set ve Trap PDU yapısı

Şekil 4.8a.'da gösterilen GetBulk PDU türünün diğer istek PDU'larından farklı bazı alanları bulunmaktadır. Bunlar :

- ***Tekrarsızlar (Non-repeaters)*** – Değişken bağlantı alanında bulunan tek olgulu nesnelerin sayısını belirtmek için kullanılır.
- ***Azami Tekrar (Max-repetitions)*** – Tek olgulu nesneler dışında kalan nesnelerin erişilmek istenen azami olgu sayısını tanımlar.

Şekil 4.9.'da SNMPv3 protokolünün yukarıda anlatılan PDU türlerini kullanarak oluşturduğu iletişim modeli gösterilmektedir.



Şekil 4.9. SNMPv3 iletişim modeli [11]

4.1.5. SNMP protokolünde temel gereksinimler

Bu bölümde SNMP 3. bölümde tanımlanan temel gereksinimlere göre incelenmektedir.

4.1.5.1. SNMPv3 protokolünde geniş alan iletişim desteği

SNMP başlangıçta sadece LAN aygıtlarının denetimi için küçük çaplı ağ ortamlarında çalışması için geliştirilmiş olan bir ağ denetim protokolüydü. Zaman

- IPX (IPX/SPX paketi) [14, 15]
- DDC (Appletalk paketi) [15,16]
- X25 protokolü üzerine kılıflama
- ATM Cell üzerine kılıflama

verilebilir. Aynı şekilde bu ağ ortamlarının ve bu ortamlardaki ağ aygıtlarının ve çeşitli iletişim mekanizmalarının denetimi için de MIB tanımları yapılmıştır.

Aygıt MIB'lerine örnek olarak:

- Modem [17]
- Yazıcı [18]
- UPS (Uninterruptible Power Supply) [19]

verilebilir.

SNMP'nin standart tanımında yönetici birimlerinin 50-100 arası ajan varlığı yönetebilmesi öngörülmüştür. Bu durum protokolün genişleyebilirliğine bir kısıt getiriyor olsa da, bu sorun dağıtılmış yönetim stratejisi ile çözülmeye çalışılmıştır. İlk olarak SNMPv2 tarafından sağlanan bu merkezi olmayan yönetim mantığına göre **Mid-Level Manager (Ara Düzey Yönetici - ADY)** [20] kullanılarak yöneticiler arasında bir sıradüzensel yapı oluşturulur. Bu ağaç yapısının yapraklarında ajanlar, kök noktasında ise ana yönetici olarak çalışan **Kuruluş Yöneticisi (Enterprise Manager)** bulunur [11,20]. ADY'ler kendi sorumluluklarına verilmiş olan bir grup ağ aygıtını izlerler ve denetlerler. Aynı şekilde kendilerinden sıradüzensel yapıda bir üst seviyede bulunan yönetici varlıklar tarafından yönetilirler ve onlara bilgi sağlarlar. Bu iki işlevi yerine getirmeleri için hem ajan hem de yönetici işlevlerine sahip olarak tasarlanmışlardır. ADY'ler aynı seviyedeki ve bir üst seviyedeki ADY'leri belirli özel durumlara karşı uyarabilmek için ise **Inform** PDU'larından yararlanırlar.

Bu sıradüzensel yapı sayesinde SNMP protokolü ile geniş çaplı ağ sistemlerinin yönetimi gerçekleştirilebilir. Ayrıca bu yapı ile, ADY'lerin ağ trafiğini kendi alanları içerisinde yerelleştirmeleri sayesinde büyük boyutlu ağların toplam trafiğinde de belirli bir azalma sağlanmış olur [11,20].

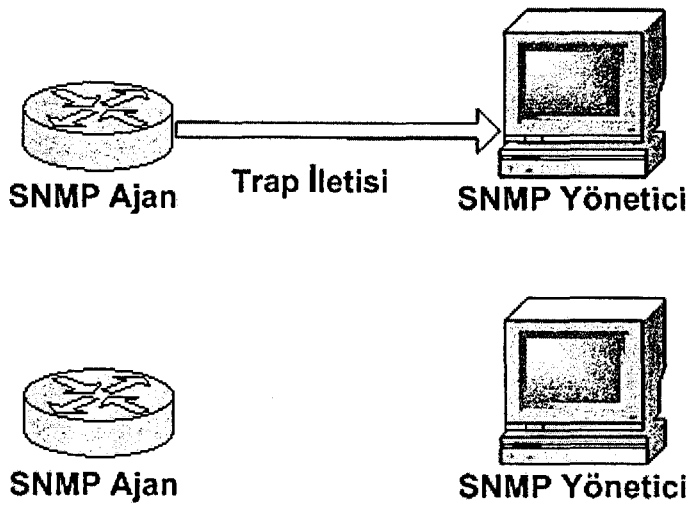
Aygıt kontrolü bağlamında, denetim yapacak olan istemci birimlerin ADY olarak çalışması gerekmektedir. Denetlenecek olan varlıların ise sadece ajan

işlevlerini gerçekleştirmesi yeterli olacaktır. Ayrıca belli bir yönetsel alan içerisinde bulunan ağ aygıtlarının Internet ortamına ağ geçidi görevi de görecek olan bir ADY sayesinde tek bir noktadan bağlı olması hem güvenlik problemleri hem de mevcut ağ adresleme mantığının yetersizliğinden dolayı daha uygun olacaktır.

4.1.5.2. SNMPv3 protokolünde güvenilir iletişim desteği

Aktarım eşleme tanımları yapılmış olan aktarım protokollerinden UDP, IPX ve **OSI (Open System Interconnection)** CLTS protokollerini içeren bir bölümü bağlantısız, OSI'nin CONS protokolünü içeren diğer bölümü ise bağlantılı iletişim hizmeti sunmaktadır [15]. Bunun dışında Schoenwaelder tarafından [12] SNMP protokolünün, TCP/IP'nin bağlantılı TCP protokolünü kullanabilmesi için gereken **eşleme (mapping)** tanımlamaları yapılmıştır. SNMP iletişimi için tanımı yapılmış olan aktarım protokollerinden sadece bağlantılı hizmet verenlerin kullanımına izin verilseydi, iletişim güvenilirliği aktarım protokolü tarafından sağlanacağından SNMP protokolünün ek güvenilirlik mekanizmalarına sahip olmasına gerek yoktu. Ancak SNMP protokolünün standart tanımında da belirtildiği gibi iletişim için bağlantısız ve dolayısıyla güvenilir olmayan bir aktarım ortamı tercih edilmiştir. Bu yüzden de SNMP gerçekleştirimlerinde en çok kullanılan aktarım protokolü UDP'dir. UDP protokolü güvenilir aktarım hizmeti veremediğinden SNMP protokolü bu eksikliği istek kimliği (*Request ID*) ileti alanı ve tekrar gönderme işlemi ile çözmeye çalışmaktadır. Aslında protokol tekrar gönderme ile ilgili de herhangi bir yöntem tanımı içermemektedir. Tekrar gönderme işleminin gerçekleştirilip gerçekleştirilmemesi, sıklığı ve miktarı ile ilgili kararlar yönetici varlığın inisiyatifine bırakılmıştır. Yönetici varlığın gönderdiği istek iletisine karşılık bir alındı iletisi gelene kadar isteği yinelemesi yöntemi **Trap** işlemi dışındaki tüm işlemler için geçerlidir. Trap PDU'ları Şekil 4.11.'de de görüldüğü gibi tanımları gereği alındısı olmayan ayrıca ajandan yöneticiye aktarılan iletilerdir. Bu yüzden yönetici varlığa Trap iletisi gönderen bir ajan iletinin yerine ulaşıp ulaşmadığını öğrenemez. Bu sorunu gidermek için SNMP yönetici tasarımlarında en çok kullanılan yöntem **tarama (polling)** yöntemidir. Bu yönteme göre SNMP yöneticisi belirli zaman

aralıklarında sahip olduğu yönetsel alan içindeki tüm ağ aygıtlarını sırayla tarayarak ağ üzerinde iletişimi engelleyecek herhangi bir problem olup olmadığını araştırır. Böylece herhangi bir problemden dolayı bilgilendirme amaçlı gönderilen bir trap iletisi yerine ulaşmasa bile yönetici bu problemi tarama işlemi ile öğrenmiş olur.



Şekil 4.11. Trap İletisine karşılık alındı iletisi oluşturulmaz

Yönetici varlıklar tarafından ajan varlıklara aktarılan istek iletilerini numaralamak amacı ile kullanılan istek kimliği (*Request ID)* ileti alanı kayıp istek ya da yanıt iletilerini belirleyebilmek için de kullanılabilir. Bu alandaki değer ilgili istek iletisini belirler ve ilgili istek iletisi ile yanıt iletisini ilişkilendirmek amacı ile kullanılır.

4.1.5.3. SNMPv3 protokolünde ağ aygıtları konumdan bağımsız şekilde tek (unique) adreslenebilmeli ve bir ağ adresinde birden fazla mantıksal aygıt bulunabilmeli

SNMPv3 protokolünde snmpEngineID nesnesi bir yönetsel alan içindeki SNMP motorlarını, dolayısı ile SNMP varlıklarını ayırt etmek için kullanılır. Her ağ aygıtının bir SNMP varlığını çalıştırdığı düşünülürse, snmpEngineID değeri aslında ağ aygıtlarını ayırt etmek amacı ile de kullanılmış olur. Aynı snmpEngineID (bir varlığa gönderilen iletide bulunan contextEngineID değeri ile iletinin gönderildiği

varlığın motoruna ait snmpEngineID değeri aynıdır) değerini farklı yönetsel alanlar içerisinde kullanabilme imkanı bulunmaktadır [9]. Bu durum snmpEngineID içeriğinin tanımlanma şekline bağlıdır. Ağ aygıtlarının yönetsel alanlar arasında bile tek adreslere sahip olabilmeleri için snmpEngineID tanımının bu amaca uygun olarak yapılması gerekir.

snmpEngineID değerinin en öncelikli (soldan ilk) biti geri kalan verinin nasıl oluşturulduğunu ifade eder [9]. Buna göre 0 ve 1 değerlerini alan iki seçenek mevcuttur:

0 – SNMPv3'den önce varolan yöntemlerle sabit boyutta tanımlama : Bu durumda 12 sekizli uzunluğunda olan snmpEngineID değerinin ilk 4 sekizlisi **International Assigned Numbers Authority (IANA)** tarafından atanan özel kuruluş numarası için kullanılır. Örneğin Acme Ağları kuruluşunun numarası 696 (Onaltılık tabandaki karşılığı = 000002b8H)'dır. Geriye kalan 8 sekizli boyutundaki alan ise kuruluşa özgü bazı yöntemlerle belirlenir. Bu yöntemler yönetsel alan içinde ajan varlıkların tek adreslenebilme olasılığını arttırmaya yönelik olarak belirlenir.

1 – Değişken boyutta tanımlama : Soldan ilk dört sekizli dışında kalan alanın içeriği beşinci sekizli içerisindeki değere bağlıdır. Bu değerler şunlar olabilir :

0 – Rezerve, kullanılmayan değer

1 – IPv4 adresi (4 sekizli)

2 – IPV6 adresi (16 sekizli)

3 – MAC adresi (6 sekizli)

4 – Yönetsel amaçlı atanan metin (azami uzunluğu 27 sekizli)

5 – Yönetsel amaçlı atanan sekizli dizisi (azami uzunluğu 27 sekizli)

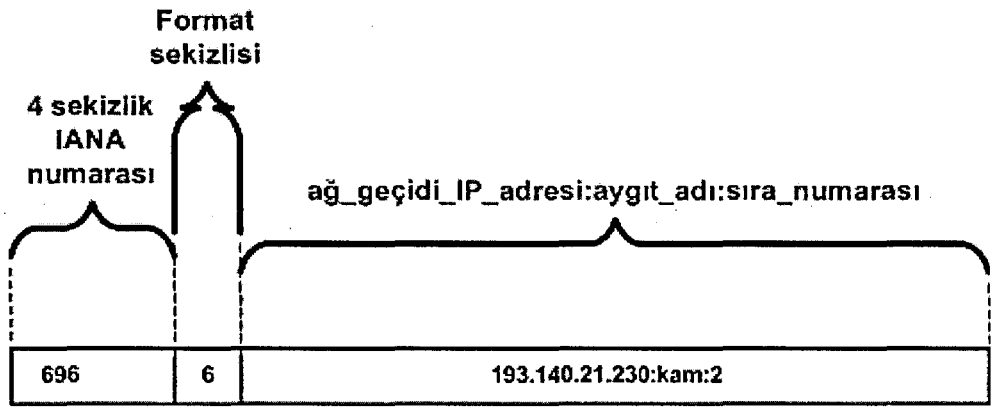
6-127 – Rezerve, kullanılmayan değer

128-255 – Kuruluş tarafından tanımlanan değer (azami uzunluğu 27 sekizli)

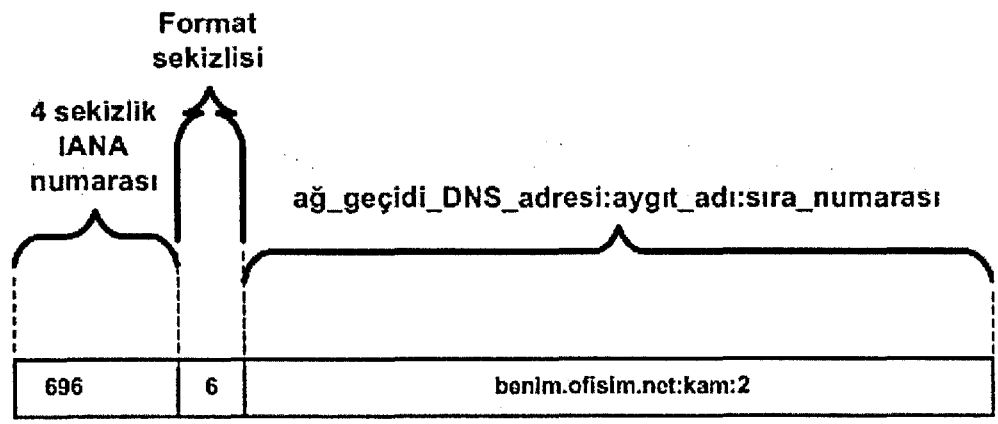
Ağ aygıtlarının evrensel olarak tek adreslenebilmesi için veri alanı içeriğini değişken boyutta tanımlama yöntemini seçmek uygun olacaktır. Burada bir yönetsel alanın içerisindeki ağ aygıtlarının tek bir ADY birime, yani tek bir mantıksal ağ

adresine bağlı olarak çalıştığı varsayılmıştır. Böylece tüm ağ aygıtlarının denetimi ve dış dünya ile iletişimleri bu birim tarafından sağlanmaktadır.

Geliştirilmek istenen çözüm yolu kapsamında beşinci sekizli için 6-127 arasındaki kullanılmayan değerlerden birisi, örneğin **6** seçilebilir. Bu değere karşılık, geri kalan veri alanına ilgili aygıtın yönetsel alanından sorumlu ADY varlığın IP adres ya da DNS adresi değerinin geleceği varsayıldığında bir yönetsel alan içindeki tüm aygıtların aynı snmpEngineID değerini alacağı görülür. Harrington'un çalışmasındaki snmpEngineID tanımı gereği bu değerin bir yönetsel alan içinde tek olması gerekmektedir. Bu koşulu sağlayabilmek için ADY IP adres/alan adı değeri yanında aynı alan içindeki aygıtlara genel bir isim ve bir sıra numarası verilmesi yeterli olacaktır. Şekil 4.12.'de içeriği tasarlanan snmpEngineID'nin sahip olduğu alanlar ve bu alanlara karşılık gelen örnek değerler görülmektedir.



Şekil 4.12. snmEngineID yapısı ve örnek değerler (IP adresli)

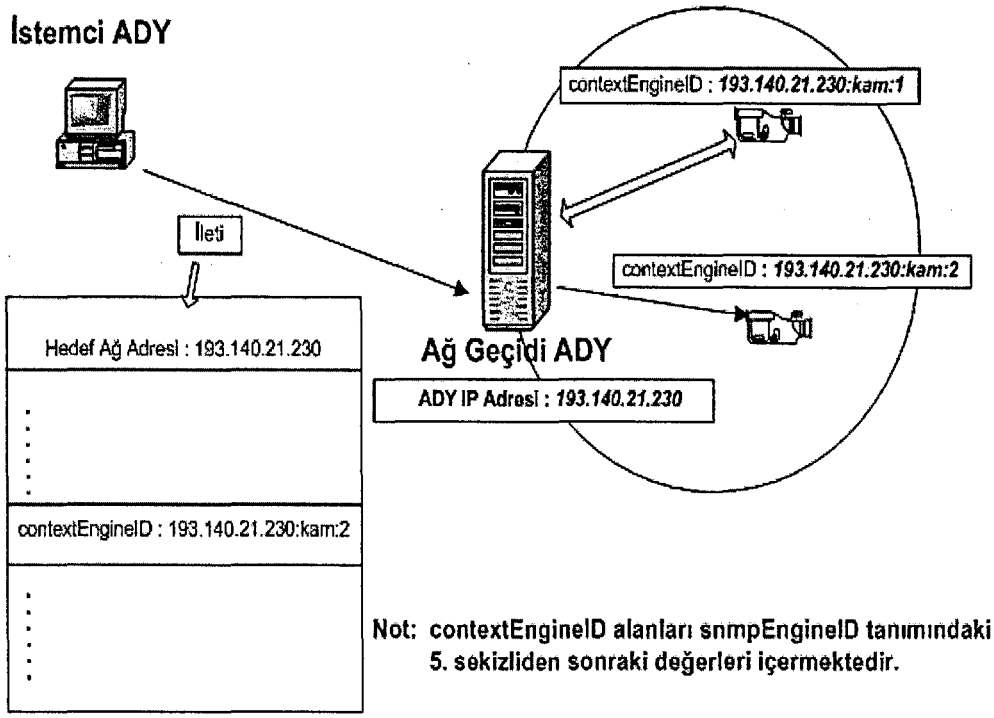


Şekil 4.13. snmEngineID yapısı ve örnek değerler (DNS adresli)

Şekil 4.12.'de birbirlerinden ":" ile ayrılan alanlardan **aygıt_adı** olarak adlandırılan değer bir aygıt türüne verilen karakter katarı türünde simgesel bir isimdir (örneğin kamera için kam). **sıra_numarası** olarak adlandırılan değer ise, bir ağ aygıtına, bir yönetsel ağ içerisinde çalışmaya başladığı anda bağlı olduğu ADY tarafından, aygıt türüne göre verilen sıra numarasıdır. Örneğin bir yönetsel alan içerisinde çalıştırılmak istenen bir kamera, o ana kadar aynı alanda çalışmaya başlamış olan kameraların toplamından bir fazla değeri ya da arada boşalmış olan bir değeri (bir nedenle işletimi sona ermiş olan bir aygıta atanmış olan sıra numarası) kendi aygıt sıra numarası olarak alır. Şekil 4.12.'deki **ağ_geçidi_IP_adresi** değeri ilgili yönetsel alan için ağ geçidi görevi de gören ADY birimin ağ adresine, Şekil 4.13.'deki **ağ_geçidi_DNS_adresi** ise ilgili ağ geçidi ADY'nin DNS adresine eşittir.

Bu yapıda adresleme yapıldığında bir yönetsel alan içerisinde çalışmaya başlamış olan bir ağ aygıtı hem alan içi hem de alanlar arası eşi olmayan bir adrese sahip olur. Ancak Şekil 4.13.'deki yapıda, yani DNS adresi kullanılarak oluşturulan snmpEngineID değerleri kullanılarak konumdan bağımsızlık tam olarak sağlanabilir. Çünkü diğer durumda snmpEngineID değerlerinin içerisinde ADY ağ adresi de bulunduğundan aygıtlar o ağ adresine bağımlı olarak adreslendirilmiş olur.

Şekil 4.14'de tasarlanan adresleme mantığı ile gerçekleştirilen bir iletişim örneği görülmektedir. Bu örneğe göre istemci yönetici içinde ADY varlığı çalışan bir PC olarak görülmektedir. Bu birim, dış dünya ile *193.140.21.230* IP adresine sahip ağ geçidi ile iletişim kuran bir yönetsel alan içindeki 2 sıra numaralı aygıt olan kameraya erişmek istiyor. Bunun için gönderdiği istek iletisinin hedef ağ adresi ilgili ağ geçidine ait olup, erişilmek istenen aygıtın contextEngineID değerinin adres bölümü (5. sekizliden sonraki geriye kalan bölüm) ise tasarımı yapılan snmpEngineID yapısına uygun şekilde *193.140.21.230:kam:2* olarak görülmektedir. Burada *193.140.21.230* adresli ağ geçidine bağlı 2 numaralı kameraya erişilmek istendiği anlaşılmaktadır. Bu örnekte snmpEngineID değerlerini oluştururken IP adresinden (*193.140.21.230*) yararlanılmıştır. Bunun yerine varsa ağ geçidinin DNS adres değeri de kullanılabilirdi.



Şekil 4.14. snmpEngineID tasarımı ile gerçekleştirilen iletişim örneği

4.1.5.4. SNMPv3 protokolünde güvenlik, kimlik denetimi ve gizlilik desteği

SNMPv3 protokolünde, bir SNMP varlığının güvenlik ve gizlilik işlemlerini gerçekleştirmek üzere SNMP motorunun birer parçası olarak **Güvenlik Alt Sistemi** ve **Erişim Denetim Alt Sistemi** adında iki yazılım modülü görev yapar. Bu iki alt sistem aslında şu sorulara cevap aramak için kullanılmaktadır :

- İleti zamanında ve değişmeden gelmiş mi?
- İleti içindeki işlemi yaptırmak isteyen kim?
- İletideki tanımlı işlemin erişeceği nesneler hangileri?
- İstemcinin işlemde kullanılacak nesneler bazında hakları nelerdir?

Güvenlik Alt Sistemi:

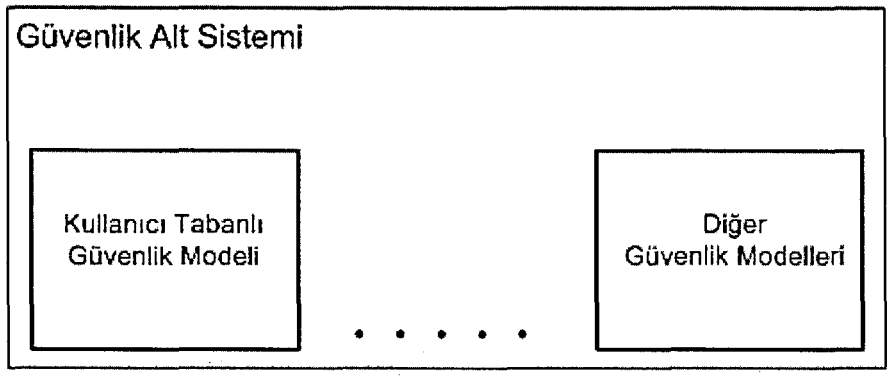
Güvenlik alt sisteminin sağladığı temel hizmetler şunlardır:

- İletilerin kimlik denetimini gerçekleştirmek
- Gizlilik amacı ile iletileri şifrelemek/ çözmek.

Bu hizmetler aslında farklı işlevleri olan güvenlik modellerinden birisi tarafından sağlanır. Bir güvenlik modelinin tanımlaması gereken temel konular :

- Koruma sağladığı güvenlik tehditleri,
- Sağladığı hizmetler,
- Kimlik denetimi ve gizlilik gibi konularda kullanılan güvenlik protokolleri.

Şekil 4.15.'de de görüldüğü gibi güvenlik alt sistemi içerisinde birden fazla güvenlik modeli tanımlanıp kullanılabilmektedir.



Şekil 4.15. Güvenlik Alt Sistemi ve içerdiği güvenlik modelleri

SNMPv3 güvenlik alt sistemi gerçekleştirime bağlı olarak birden fazla güvenlik modeli içerebilmektedir. Bu esnek yapı zaman içerisinde değişen güvenlik gereksinimlerini yeni güvenlik modelleri ile karşılayabilmek için düşünülmüştür. Bunun da ötesinde desteklenen güvenlik protokollerinin SNMP varlığı tarafından eş zamanlı olarak kullanılmasına da izin verilmektedir. SNMPv3 iletilerinin başlık bölümü içerisinde Şekil 4.7.'de gösterilen **msgSecurityModel** adlı alan aktarılan iletinin hangi güvenlik modeli ile incelenmesi gerektiğini belirtir.

Farklı SNMPv3 varlık gerçekleştirimleri arasında uyumlu bir iletişim sağlayabilmek için SNMPv3 varlıklarının güvenlik alt sistemlerinde en az bir güvenlik modelinin ortak olarak tanımlı olması gerekir. SNMPv3 mimarisi içerisine bu amaçla yerleştirilmiş olan güvenlik modelinin adı **User Based Security Model (USM)**'dir.

User Based Security Model (USM) [21]:

USM ileti-seviyesinde sağladığı mekanizmalar sayesinde SNMPv3 varlıklarını şu temel tehditlere karşı korumaktadır :

- *Bilginin değiştirilmesi* ***(Modification of Information)*** *:* SNMP iletilerinin aktarım esnasında bir yetkisiz varlık (makine ya da kullanıcı) tarafından değiştirilmesi.
- *Kılık değiştirme* ***(Masquerade)*** *:* Bir yetkisiz varlığın, yetkili varlığın kimlik bilgilerini üzerine alarak bir işlem gerçekleştirmeye çalışması.
- *İleti akışının değiştirilmesi* ***(Message Stream Modification)*** *:* İletilerin gecikme sürelerinin ya da aktarım tekrarı sayılarının belirlenmiş olan sınır değerlerin üzerine çıkarılması.
- *Açığa çıkarma* ***(Disclosure)*** *:* Yetkisiz varlıkların SNMP veri alışverişi içeriğini izlemeleri.

Bunların dışında, tam olarak tehdit özelliği taşımayan ve önlenmesi neredeyse imkansız olan **trafik analizi (traffic analysis)** ve **hizmet reddi (denial of service)** olaylarına karşı USM tarafından bir korunma yöntemi öngörülmemiştir.

Kullanıcı Tabanlı Güvenlik Modeli temelde üç modülden oluşmaktadır :

- ***Kimlik Denetimi Modülü*** – Veri bütünlüğü ve veri kaynağı kimlik denetimini sağlar.
- ***Zamanlılık (Timeliness) Modülü*** – İleti gecikmesi ve tekrarlarına karşı koruma sağlar.
- ***Gizlilik Modülü*** – İletinin **veri alanının (payload)** açığa çıkmasını önleyici mekanizmalar içerir.

Aygıt kontrolü bağlamında düşünüldüğünde, bir kullanıcı sahip olduğu ağ aygıtına kullanıcı adı ve parolası ile oturum açıp başka bir ağ aygıtını denetlemek için ileti göndermek isterse, gönderilecek olan PDU'ya kimlik denetimi modülünde bulunan **doğrama (hashing)** algoritmalarından (**MD5 (Message Digest 5)** ya da **SHA (Secure Hash Algorithm)**) bir tanesi bir anahtar değeri ile birlikte uygulanır. Bu değer kullanıcının kullanıcı adı ve parolasıyla ilişkilidir. Doğrama işlevinden

sonra üretilen ve **parmak izi** olarak adlandırılan değer ise aktarılacak olan PDU'nun içerisine yerleştirilir [22]. PDU'yu alan sunucu aygıt PDU içerisindeki parmak izi değerini çıkarıp yerine aynı anahtar değeri koyar ve PDU aynı doğrama işlemine sokulur. Sonuçta oluşan değer gelen PDU içerisindeki parmak izi değeri ile karşılaştırılır. Bu sayede, aktarılan ileti **bilginin değiştirilmesi** tehdidine karşı korunmuş olur. Ayrıca kimlik denetiminin kapsamı iletinin kaynağını da içerdiğinden, yani kimlik denetimi iletiyi üreten birimden itibaren yapılmaya başlandığından (parola bilgisi ile) **kılık değiştirme** tehdidinin de önüne geçilmiş olur. Böylece hem ileti bütünlüğü hem de iletiyi gönderenin kimliğinin doğruluğu denetlenebilmektedir [22].

USM'nin zamanlılık modülü sayesinde zaman uyumluluk ve zaman penceresi denetimi teknikleri kullanılarak SNMP PDU'larının aktarım zamanlılığı denetlenebilir. Bu da değiştirilme ihtimaline karşı geciken iletilerin saptanmasına yardımcı olur [23].

İletilerin veri alanlarının gizlenmesi için ise **DES (Data Encryption Standart)** [24, 25] şifreleme algoritması kullanılır. Bu algoritmaya göre SNMPv3 ajan ve yönetici varlıklar bir gizli şifreleme anahtarını paylaşırlar. Bu anahtar kimlik denetiminde anahtar oluşturmak için kullanılan paroladan farklı bir parola değerine bağlı olup, elde edilme yöntemi kimlik denetimindeki ile aynıdır.

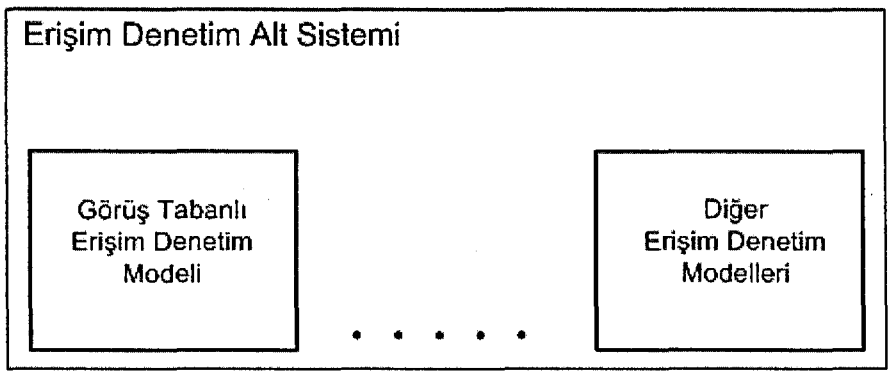
Erişim Denetim Alt Sistemi :

Erişim denetimi alt sisteminin (Şekil 4.16.) temel görevi, bir SNMP varlığının sakladığı yönetilen nesnelere erişimleri denetlemektir. Erişim Denetim Sistemi içerisinde de güvenlik alt sisteminde olduğu gibi birden fazla erişim denetim modeli tanımlanıp kullanılabilmektedir. Ancak SNMPv3 standart tanımında **View Based Access Control Model (VACM - Görüş Tabanlı Erişim Denetim Modeli)** [26] önerilmektedir.

VACM kullanıcı kimliklerine dayalı olarak yönetim bilgilerine (yönetilen nesnelere) erişimi denetlemek için tasarlanmıştır. Bunun için farklı kullanıcıların MIB içeriğinin her bir bölümüne farklı seviyelerde (okuma, yazma, bildirim)

erişimlere izin verilmektedir. Bir ağ yöneticisi SNMP sunucusunda kimliğini belirtip oturum açtığında ağa göndereceği her komut iletisinin içerisinde kendi kimlik bilgileri de bulunur. Bu iletileri alan SNMP ajanları ileti içerisinde belirtilen komuta yönelik olarak herhangi bir işleme başlamadan önce kullanıcı ile ilgili kimlik bilgilerini kendi sistemlerinde önceden oluşturulmuş erişim denetim veritabanını kullanarak kontrol ederler. Bu sayede her bir ağ yöneticisine farklı erişim hakları verilebilir.

Aygıt kontrolü kapsamında düşünüldüğünde de bir aygıta uzaktan bazı işlemler yaptırmak isteyen bir kullanıcı, aygıt ya da aygıtın bağlı olduğu ağ geçidi ADY içerisinde tutulan erişim veritabanında bulunan ve kendi kimlik bilgilerine karşılık gelen erişim hakları çerçevesinde ilgili aygıta erişebilir.



Şekil 4.16. Erişim Denetim Alt Sistemi ve içerdiği güvenlik modelleri

4.1.5.5. SNMPv3 protokolünde aygıt taşınırlık desteği

SNMPv3 protokolünün standart tanımı ağ yönetimi bağlamında bir yerel ağ içerisindeki sabit ağ aygıtlarının denetimini kapsamaktadır. Bu yüzden, ağ aygıt kontrolü kapsamında taşınır ağ aygıtlarının denetimine yönelik bir uygulama katmanı desteğine sahip değildir.

Bu desteği sağlayabilmek için bölüm 4.1.5.3.'de tanımlanan aygıt adresleme yapısından yararlanılabilir. Bir yönetsel alan içerisinde çalışmaya başlayan bir ağ aygıtının ağ iletişimine katılabilmesi için en az bir gerçek ya da sanal ağ adresine ihtiyacı vardır. Ağ aygıtlarında güvenli iletişimi sağlayabilmek için dış ortama

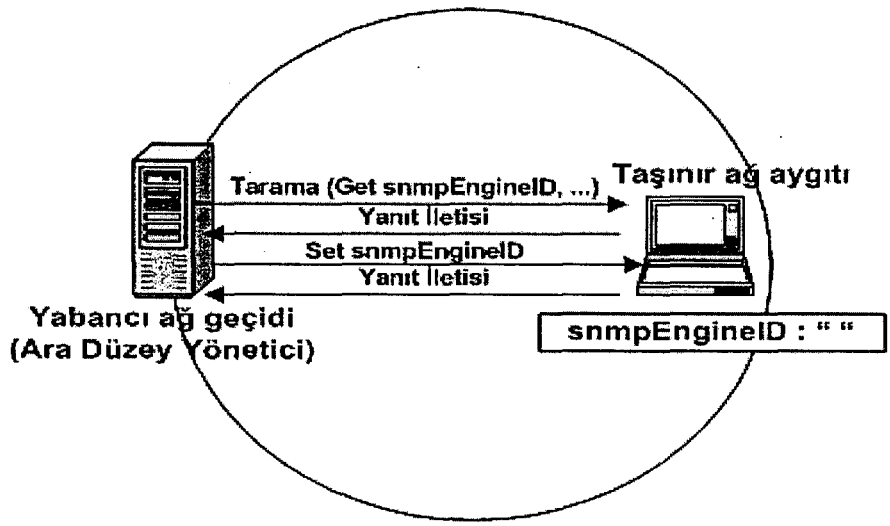
(Internet) tek noktadan çıkış mantığı göz önünde tutulduğundan (Ara Düzey Yöneticiler - ADY) ağ aygıtlarının sanal ağ adresleri kullanmaları daha uygun olacaktır. Bu adresler ADY tarafından tanımlanmış bir aralık içerisinden verilmelidir ki atanan adresler üzerinde denetim sağlanabilsin.

Bir yönetsel alanın denetiminden sorumlu olan yönetici birimin (ADY) ilgili alan içerisinde hangi aygıtların ağa bağlı olduğunu, dolayısıyla çalışmakta olduğunu anlayabilmesi için belirli zaman aralıklarında kendisinde tanımlı olan ağ adres aralığındaki tüm aygıtları taramalıdır. Bu tarama işlemini yerel ağa, hedef snmpEngineID, yani contextEngineID alanı içerisindeki **aygıt_adi** değeri "**hepsi**", **sıra_numarası** değeri ise **0** olan bir Get iletisi göndererek gerçekleştirebilir. Bu ileti içerisinde bulunan değişken bağlantıları bölümünde ise aygıtla ilgili bilgileri ve snmpEngineID değerini tutan nesneler bulunmalıdır.

Bu tarama işlemi sonucunda cevap üreten adresler ilgili MIB tablosunda güncellenmelidir. Eğer aygıt ilk olarak başka bir yönetsel alan içinde çalıştırılmadıysa snmpEngineID nesne değerinin boş olacağı varsayılmaktadır. Böylece ADY dönen iletilerden hangi aygıtların yerel alanlarında bulunduğunu hangilerinin ise yabancı bir yönetsel alandan geldiklerini belirlemiş olur. Bunun için gelen yanıt iletisi içindeki snmpEngineID nesne değerini kullanır:

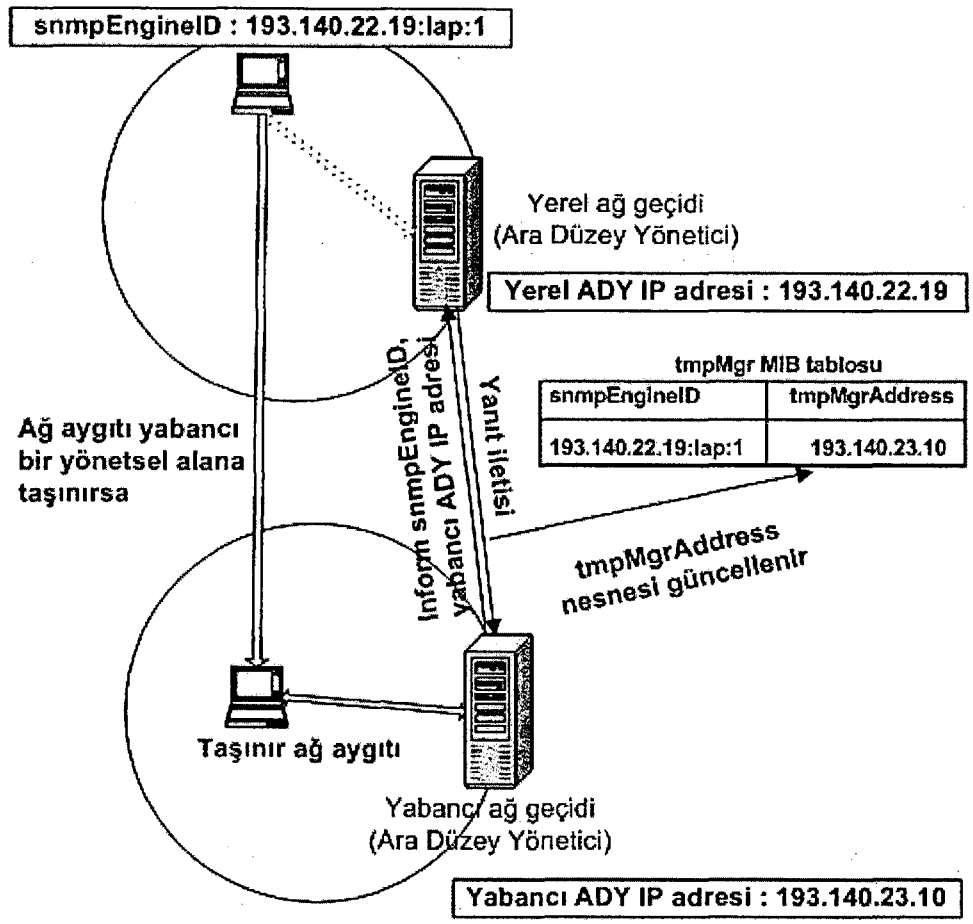
- Değer boş karakter katarı ise ya da değerin içerisindeki **ağ_geçidi_IP_adres** ya da **ağ_geçidi_DNS_adresi** alanı kendi ağ adres değerine eşitse ilgili aygıt kendisine bağlıdır.
- Değer içerisindeki **ağ_geçidi_IP_adres** ya da **ağ_geçidi_DNS_adresi** alanı kendi ağ adres değerinden farklı ise ilgili aygıt yabancı bir yönetsel alandan gelmiştir.

ADY, tarama sonucunda belirlediği boş snmpEngineID değerli aygıtların bu adres bilgilerini aygıtların türlerini (kamera, klima ...gibi) göz önünde tutarak oluşturur. Ve aygıtlar içerisindeki ilgili nesne içeriği bu değerlerle Set PDU'lar kullanılarak güncellenir (Şekil 4.17.).



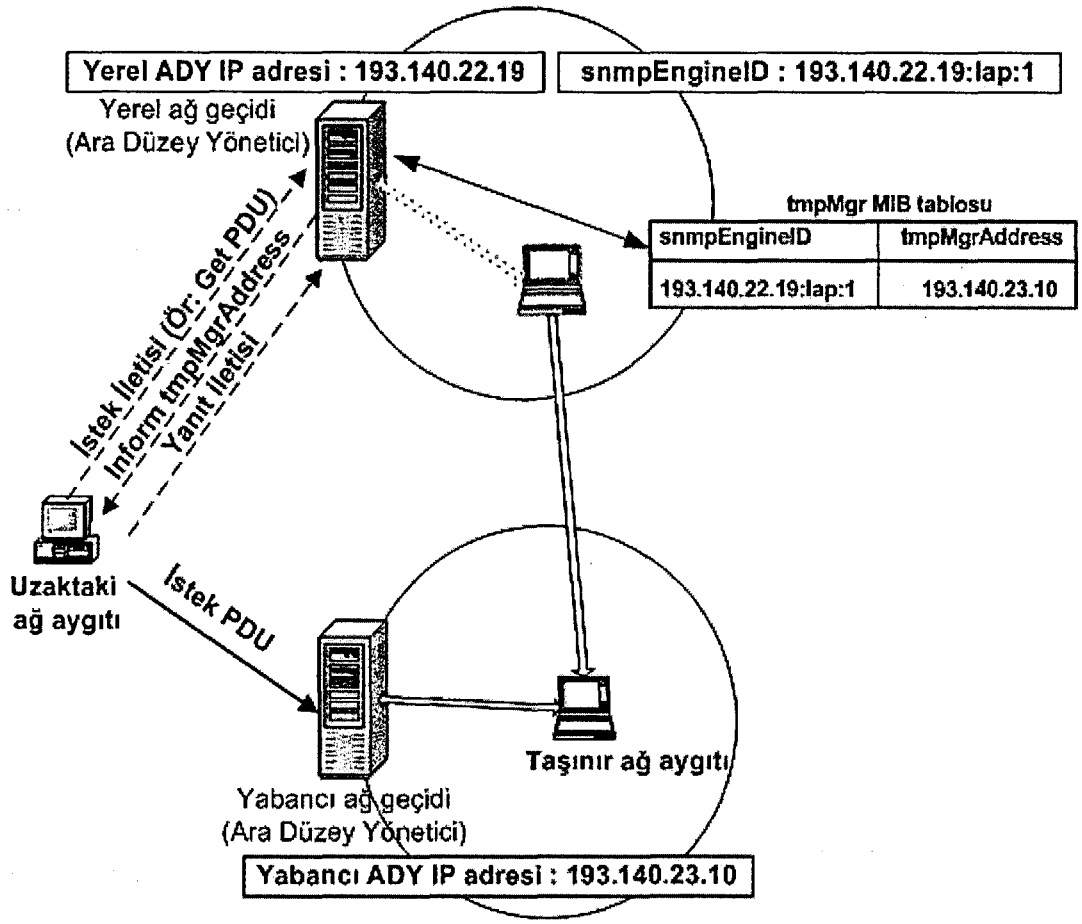
Şekil 4.17. Bir yönetsel alanda ilk defa çalışacak olan ağ aygıtının snmpEngineID değerini edinmesi

Tarama sonucunda farklı yönetsel alanlardan gelme ağ aygıtlarına rastlanırsa, bu aygıtların iletişimlerini sürdürebilmeleri için bağlı oldukları yönetsel alanın ara düzey yöneticisinin bu konuda bilgilendirilmesi gerekir. Bunun için de Şekil 4.18.'de örneklenmiş olan yöntem kullanılabilir. Buna göre ADY bir ağ aygıtının yabancı bir yerel alandan geldiğini saptadığında ilgili aygıtın snmpEngineID değeri ile kendi ağ adres değerini bir Inform iletisi içerisindeki değişken bağlantıları bölümüne koyarak yerel ADY'ye gönderir. Bu iletiyi alan ADY güvenlik açısından önce ilgili snmpEngineID'ye sahip olan ağ aygıtının eski ağ adresinde olup olmadığını bir Get iletisi ile sınayabilir. Get iletisine belirli bir süre içerisinde cevap gelmezse aygıtın yerel yönetsel alanın dışında olduğu belirlenmiş olur. Daha sonra yerel ADY sahip olduğu ağ aygıtlarının o an bulundukları yabancı yönetsel alanı yöneten yabancı ADY'lerin ağ adreslerini tutan bir MIB tablosunu kendisine Inform iletisi ile gelen snmpEngineID ve ADY ağ adresi değerleri ile günceller.



Şekil 4.18. Yabancı bir yönetsel alana taşınmış olan ağ aygıtının yerel ara düzey yöneticisinin güncellenmesi

Şekil 4.18.'de bu tablo **tmpMgr** olarak yabancı ADY'nin ağ adresini tutacak olan alan ise **tmpMgrAddress** olarak adlandırılmıştır. Böylece uzaktaki bir ağ aygıtı ilgili ağ aygıtı ile iletişim kurmak istediğinde yerel ADY sahip olduğu **tmpMgrAddress** değerini uzaktaki istemci aygıta göndererek iletişimi bu adres üzerinden yapması gerektiğini belirtebilecektir. Bu işlem ise Şekil 4.19.'da örneklenmiştir.



Şekil 4.19. Yabancı bir yönetsel alanda bulunan bir ağ aygıtı ile iletişim kurmak isteyen uzaktaki bir ağ aygıtı

Şekil 4.19.'da görüldüğü gibi, uzaktaki bir ağ aygıtı bir yerel alan içerisindeki ağ aygıtına erişmek istemektedir. Ancak ilgili ağ aygıtı o anda başka bir yönetsel alan içerisinde bulunduğundan yerel ADY aracılığı ile iletişim kuramayacaktır. Bu yüzden yerel ADY'nin yabancı yönetsel alanın ADY ağ adres bilgisini **(tmpMgrAddress)** istemci ağ aygıtına iletmesi gerekir. Yerel ADY bu bilgi iletimini bir Inform PDU oluşturarak gerçekleştirir. Yalnız şekildeki örnekte yerel ADY'nin biraz önce anlatılan ve Şekil 4.18.'de de örneklenen yöntemle güncellenmiş olduğu varsayılmaktadır.

Inform PDU aracılığı ile asıl iletişim kurması gereken ADY'nin ağ adresini elde eden istemci ağ aygıtı bu adrese isteğini yineleyerek ilgili ağ aygıtına erişebilecektir.

Yabancı ADY tarama işlemleri esnasında başka bir yönetsel alandan gelen ağ aygıtına atadığı ağ adresinden yanıt alamazsa, bu durumu bir Inform PDU ile ilgili aygıtın yerel ADY birimine bildirmek zorundadır. Böylece yerel ADY içerisinde tutulan **tmpMgr** tablosu sürekli güncel bilgilere sahip olur.

Önerilen yöntemler sayesinde aygıt ve oturum taşınırlığı sağlanmış olur. Bu örnekte aygıt snmpEngineID değerleri ilgili yerel ADY'nin IP adresi kullanılarak yapılmıştır. Bunun yerine varsa ADY'nin DNS adresi ile de snmpEngineID değerleri oluşturulabilir.

4.1.5.6. SNMPv3 protokolünde IPv6 desteği

SNMPv3 protokolünün standart tanımı IPv4 protokolü üzerinde çalışacak şekilde yapılmış olsa da, yakın zamanda bu protokolün sağladığı ağ adres aralığı yetersiz kalacağı için IPv6 desteğine yönelik olarak da çalışmalar süre gelmektedir. Bu çalışmaların temelinde IPv6 ve onunla ilgili veri yapılarının (metinsel kuralların) tanımlanması ve bu veri yapıları kullanılarak da gerekli yönetilecek nesnelerin tutulacağı MIB'lerin tanımlanması bulunmaktadır [27]. Kullanılan aktarım adreslerinin metinsel kurallarının tanımlandığı bazı çalışmalarda IPv6 adresleri hakkında da tanımlar bulunmaktadır [28].

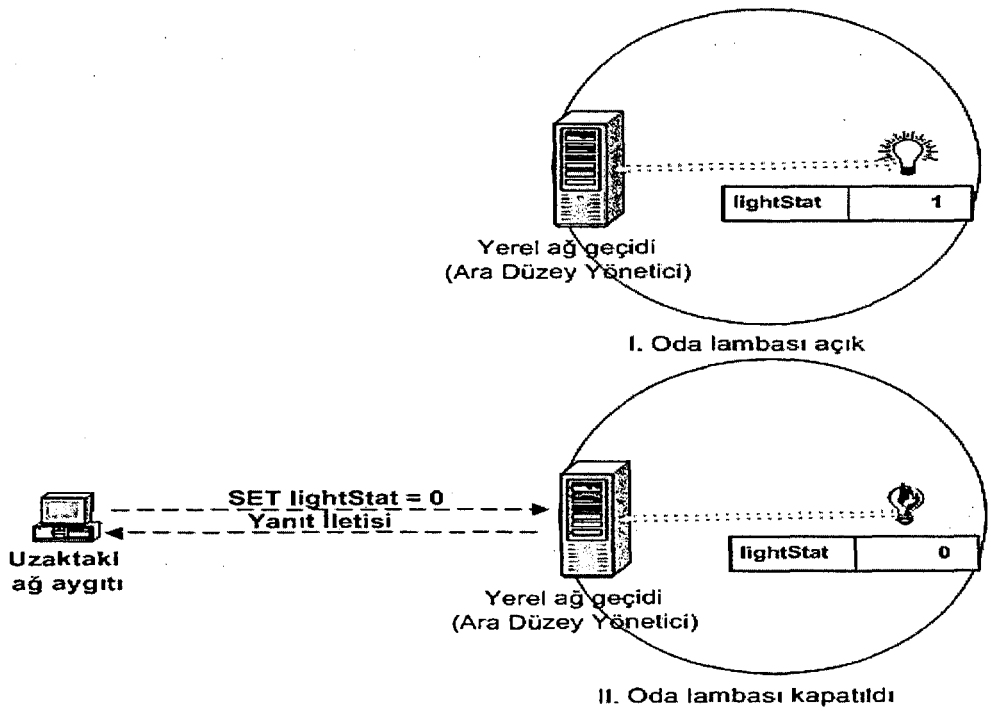
Bununla birlikte bölüm 4.1.5.3.'de önerilmiş olan aygıt adresleme mantığı da IPv6 ile sorunsuz çalışabilecek durumdadır çünkü gerekli alan değişken uzunlukta tanımlanmıştır.

4.1.5.7. SNMPv3 protokolü esnek ileti yapısına sahip olmalı

SNMPv3 protokolünün ileti yapısı Şekil 4.7.'de de görüldüğü gibi başlık, güvenlik parametreleri ve veri alanından oluşur. İletişim esnasında yapılması istenen işlemler ve aktarılması gereken veriler aslında PDU olarak adlandırılan veri alanın bir bölümü içerisinde bulunur (Şekil 4.8a., 4.8b.). İletişim kurulmak istenen aygıtın protokol adres bilgisi ise yine veri alanı içerisinde ancak PDU'dan bağımsız olarak (contextEngineID ve contextName) tutulurlar. SNMPv3 protokolünde tanımlı temel işlemler gerçekleştirilirken daima MIB nesneleri temel alınır. MIB'ler kullanım

amaçlarına ve ortamlarına göre SMI olarak adlandırılan veri tanımlama kuralları kullanılarak protokolden bağımsız olarak gerçekleştirildiğinden protokol ve aktarılan veri birbirlerinden ayrılmış olur. Bu da aktarılacak veriler, türleri ve üzerlerinde yapılacak işlemlerin çeşitliliği konusunda büyük esneklik sağlar. Ağ aygıt kontrolü çerçevesinde düşünüldüğünde ağ aygıtının belli bir işlemi yerine getirmesi için, bu işi ifade eden sahip olduğu bir MIB nesne değerinde değişiklik yapmak üzere bir istek iletisi göndermek yeterli olacaktır. Buna örnek olarak Şekil 4.20.'de bir oda lambasının açık durumdan kapalı duruma geçirilmesi basit bir şekilde gösterilmektedir.

Lambanın bir aygıt denetleyicisi ile ilgili ağ geçidine bağlı olduğu düşünülürse, işlevleri ile ilgili MIB tablolarının ağ geçidinde bulunması gerektiği görülür. Bu tablonun örnekte **lightStat** olarak adlandırılan nesne değeri üzerinde değişiklik yaparak lambayı açma/kapama işlemi gerçekleştirilebilir. Bunun için **lightStat** nesnesi 0 değeri aldığında lambanın kapalı/kapanacak olduğu, 1 değeri aldığında açık/açılacak olduğu varsayılmıştır.



Şekil 4.20. Yanmakta olan bir oda lambasının kapatılması

4.1.5.8. SNMPv3 protokolünün yapısı karmaşık olmamalı

SNMPv3 modüler yapıda tasarlanmış bir mimariye sahiptir (Şekil 4.4.). Bu özelliği sayesinde oldukça esnek ve karmaşık olmayan gerçekleştirimlere olanak vermektedir. İlk sürümünden itibaren SNMP protokolünün tasarımında dikkat edilen en önemli ölçütlerden birisi, isminden de (Basit Ağ Yönetim Protokolü) anlaşılacağı gibi yapısının olabildiğince basit tutulmasıydı. Bu sayede gerçekleştirimi ve kurulumu kolay, işlemci ve ağ kaynaklarını asgari düzeyde kullanan bir protokol tasarlanmış oldu.

SNMPv3 protokolünün modüler yapısı sayesinde, ağ aygıt kontrolü bağlamında SNMPv3 protokolü bir ağ aygıtına (ajan/yönetici), o aygıtın işlevlerine uygun şekilde kurulabilir. Yani aygıtın işlevlerine yönelik SNMPv3 modülleri seçilebilir ya da yenileri eklenebilir. Böylece aygıtın sahip olduğu kaynakların ve işlem gücünün gereksiz kullanımı önlenmiş olur.

SNMPv3 protokolünün sahip olduğu komut sayısının oldukça sınırlı tutulmuş olması, iletilecek PDU türlerinin ve bu türler üzerinde yapılacak özel işlemlerin de basit ve az sayıda olmasını sağlamıştır.

4.1.6. SNMPv3 protokolünde işlevsel gereksinimler

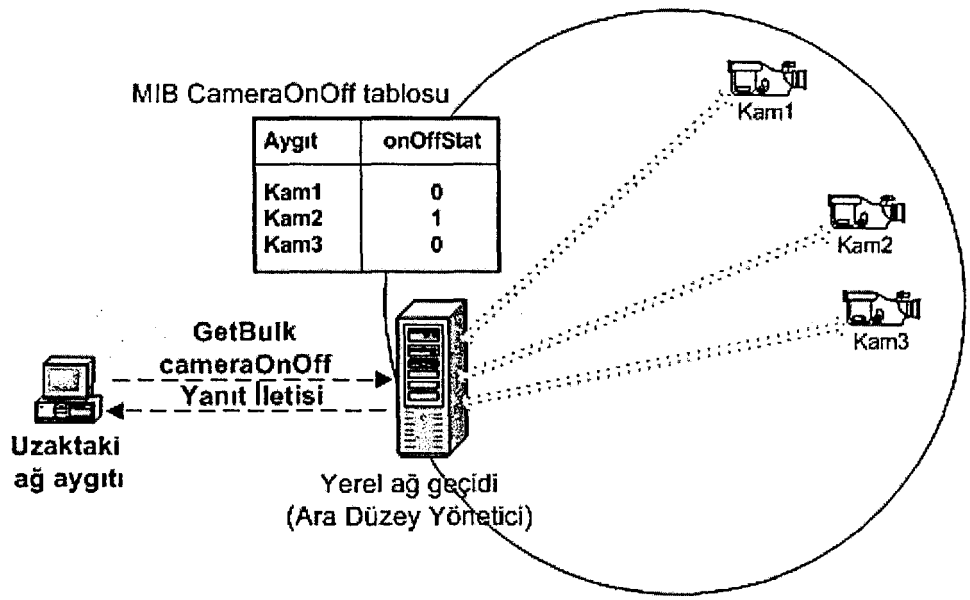
Bu bölümde SNMP 3. bölümde tanımlanan işlevsel gereksinimlere göre incelenmektedir.

4.1.6.1. SNMPv3 protokolünde aygıt denetimi

SNMPv3 ağ aygıt kontrolü bağlamında ağ aygıtlarına bazı işlevleri yapmalarına yönelik olarak istekte bulunmak için Set PDU'ları kullanılabilir. Bununla birlikte ağ aygıtının işlevlerine, sahip olduğu özelliklere yönelik SNMPv3 MIB nesne tanımlarının yapılmış olması gerekir. Böylece SET PDU'ları aracılığı ile tanımlı nesnelerin değerlerinde değişiklikler yapılarak aygıtın belli bir işlevi yerine getirmesi sağlanabilir (Şekil 4.20.).

4.1.6.2. SNMPv3 protokolünde durum denetimi

Durum denetimi için SNMPv3 protokolünün sahip olduğu Get, GetNext, GetBulk türü istek PDU'larından yararlanılabilir. Get PDU ile tek bir MIB nesne değerine erişilebileceği gibi GetBulk PDU yardımı ile örneğin bir nesnenin aynı ağ içerisindeki aynı türden ağ aygıtlarına atanmış olgu değerlerinin hepsine ya da bir bölümüne erişmek de mümkündür. Buna örnek olarak Şekil 4.21.'de bir yönetsel alan içerisindeki 3 adet kameranın açık olup olmadığı bilgisini tutan CameraOnOff MIB tablosunun içeriği bir GetBulk iletisi ile ilgili ADY ağ geçidinden istenmektedir. Geriye dönen yanıt iletisinde ise bu tablonun içeriği bulunacaktır.



Şekil 4.21. GetBulk PDU ile CameraOnOff MIB tablosunun içeriğinin elde edilmesi

4.1.6.3. SNMPv3 protokolünde olay denetimi

SNMPv3 bünyesinde ajan varlıkların yönetici varlıklara belli bir olayın bildirimi için kullandıkları Trap PDU ve yöneticiler arasında olay ve durum bildirimi için kullanılan Inform PDU türleri bulunmaktadır. Ancak bildirim istemine yönelik bir tanım bulunmamaktadır.

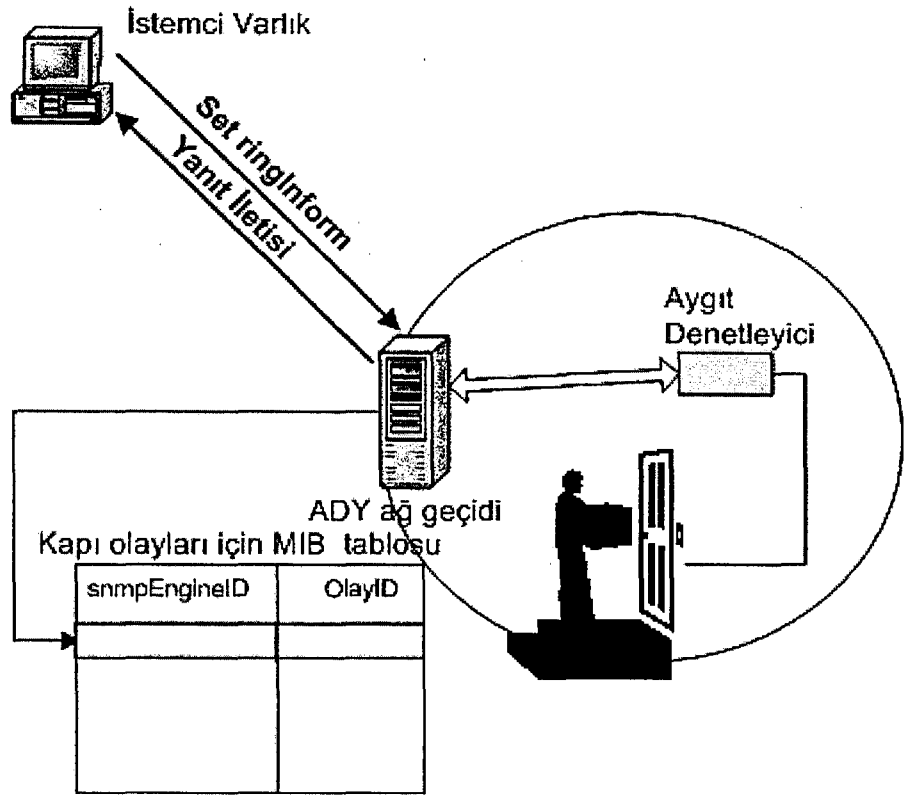
Bir yönetici birim belli bir yönetsel alan içerisinde bulunan bir ağ aygıtının belli bir olaya yönelik olarak kendisine bildirimde bulunmasını sağlamak için SNMP

protokolünün her işlevinde olduğu gibi MIB nesnelerinden yararlanabilir. Bunun için istemci ajan ve ilgili yönetici varlık bünyesindeki MIB'ler içerisinde ağ aygıtının desteklediği olaylar ve bildirim durumları ile ilgili nesne tanımlarının bulunması gerekir.

İstemci yönetici varlık bir aygıtla ilgili bir olayın gerçekleşmesi durumunda kendisine bildirilmesi için ilgili olayın bildirimi ile ilişkili olan nesnenin değerini kurarak SET PDU içerisinde aygıtın bağlı olduğu yerel ADY'ye gönderir.

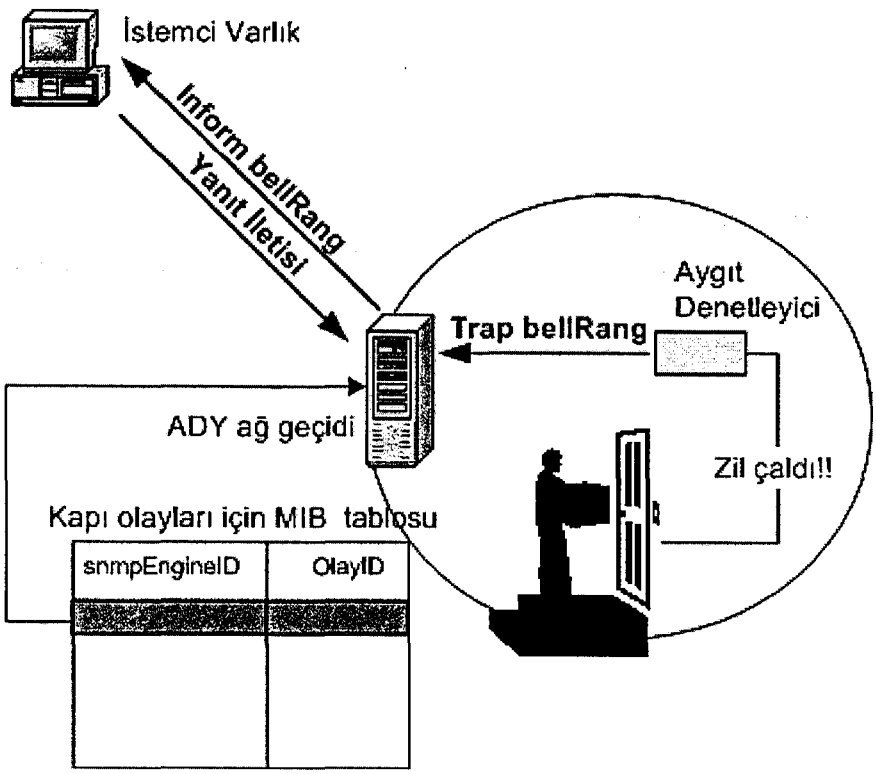
Olayın gerçekleşmesi durumunda ajan varlığın durumu anlayabilmesi için bu olaya yönelik olarak tuttuğu MIB nesnesini kuracak bir uygulamanın ajan varlık içerisinde çalışıyor olması gerekir. Ajan varlık ilgili nesnenin değeri değiştiği anda bir Trap PDU oluşturarak nesne değerini yerel ADY'ye gönderir. Yerel ADY (ağ geçidi olarak da çalışan) sahip olduğu bildirim tablosundan bu nesnenin belirttiği olaya karşılık bildirim isteminde bulunulup bulunulmadığını araştırır. Bulduğu kayıtlardaki snmpEngineID değerlerine sahip varlıklara olay bildiriminde bulunmak için ilgili nesne değerini INFORM PDU içerisine yerleştirir. Bu iletiyi alan istemci varlık bildirim isteminde bulunduğu olayın gerçekleştiğini anlar ve buna karşılık yapması gereken işlemleri yerine getirir.

Şekil 4.22.'deki örnekte kapı zili ilkel bir ağ aygıtı olarak düşünüldüğünden ilgili ADY'ye bir aygıt denetleyici ile bağlanmıştır. Uzaktaki bir ağ aygıtını kullanan kullanıcı yerel ADY aracılığı ile aygıt denetleyiciden kapı zilinin çalma olayının bildirilmesini istemektedir. Bunu da ADY'ye gönderdiği Set PDU içerisinde **ringInform** adı verilen nesnenin değerini kurma isteğinde bulunarak gerçekleştirmektedir. Burada **ringInform**'un bir uzak yönetici birim için (bu örnekte uzaktaki bir bilgisayar) kapı zilinin çalınma bildirim isteminin yapılıp yapılmadığını tutan bir MIB nesnesi olduğu varsayılmaktadır. Set PDU'yu alan ADY sahip olduğu MIB tablosuna bu değerin kurulu olduğu ve ilgili istemci varlığın snmpEngineID değerinin bulunduğu yeni bir kayıt ekler. Olay belirleyicisi ve olayın tanımı da başka bir MIB tablosunda tutulabilir. Bildirim istek kaydı yapılmış olan yönetici varlık ADY'den bildirim beklemeye başlar. Şekil 4.23.'de kapı zilinin çalması sonucunda gerçekleşecek işlemler örneklenmiştir. Buna göre kapı ziline bağlı aygıt denetleyici



Şekil 4.22. Kapı çalma olayına karşı bildirim isteğinde bulunulması

durumu bu olayla ilgili nesne değerini içeren bir Trap PDU oluşturarak ADY'ye bildirir. Bildirim için, kapı zilinin çalınma olayının gerçekleştiğini belirten MIB nesnesinin (örnekte **bellRang** olarak adlandırılmıştır) değeri kurularak aktarılacak olan Inform PDU içerisine yerleştirilir. Inform iletisini alan istemci yönetici varlık önce bir Response PDU ile bildirimi aldığını yerel ADY'ye haber verir sonra da kendi MIB'si içerisindeki **bellRang** nesnesinin değerini PDU içerisindeki değerle günler. Böylece eğer bu değerin günlenmesi ile yapılması gereken işlemler varsa, bu işlemlerden sorumlu olan uygulama çalışmaya başlar.

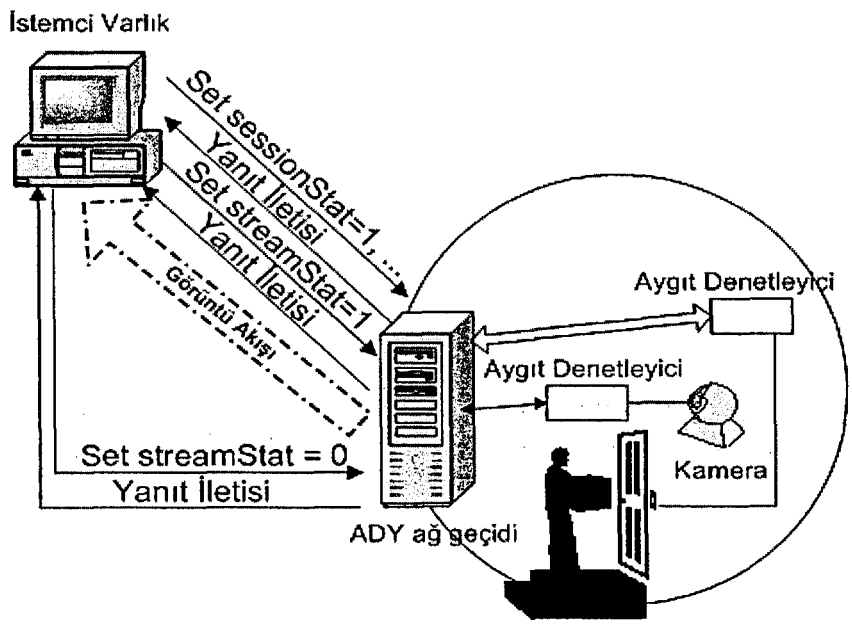


Şekil 4.23. Kapı zili çaldığında Trap-PDU ile bildirim yapılması

4.1.6.4. SNMPv3 protokolünde oturum denetimi

SNMPv3 protokolünde işlemler sadece MIB nesne değerlerinin alışverişi şeklinde gerçekleştiğinden sürekliliği olan işlemlere yönelik destek bulunmamaktadır. Ancak bu destek MIB nesneleri ve ek uygulamalarla gerçekleştirilebilir. Şekil 4.22. ve Şekil 4.23.'de kapı zili olayının bildirim istemi ve bildirim işlemleri örneklenmiştir. Bu örneğe ek olarak bildirimi alan istemci varlığın kapıda bulunan kamera ile bir görüntü akışı oturumu açmak istediği düşünüldüğünde bu işlem de yine MIB nesneleri aracılığı ile gerçekleştirilebilir (Şekil 4.24.). Zil çalındı bildirimini alan aygıt kapı zili ile aynı yönetsel alan içerisinde bulunan kapı kamerasına bir Set PDU gönderir. Bu ileti içerisine kameranın oturum durumunu belirleyen bir MIB nesnesi değeri (Şekil 4.24.'de sessionStat = 1) kurulmuş halde yerleştirilir. Bunun yanında istenen görüntü akış parametreleri ve gereksinimleri ile ilgili MIB nesne değerleri de bu iletide yer almalıdır. Bu değerler kameranın sahip olduğu özelliklerle

karşılaştırılacak, uygun olursa görüntü akışı onaylanacaktır. Oluşturulan Set PDU'yu alan ADY gönderildiğinde iletideki contextEngineID değeri ile aynı değere sahip olan kameraya iletiyi aktarır. Kameranın aygıt denetleyicisi iletideki nesnelerden birincisinin çalışmaya başlama istemi ile ilgili nesne olduğunu gördüğünde geri kalan nesne değerleri ile kendi nesne değerlerini karşılaştırmak için ilgili değerleri tanımlanan nesnelerin yedek olgularına aktarır. Nesne değerleri arasındaki karşılaştırma işlemi sonucunda değerler uyuşuyorsa aygıt denetleyici olumlu bir yanıt iletisi oluşturup istemci yönetici varlığa gönderir. Buna karşılık istemci varlık içinde bulunan ve görüntü alımı ile ilgilenen uygulama çalıştırılarak görüntü akışı beklenmeye başlanır. Bu arada istemci oluşturacağı bir Set PDU ile de kameranın görüntü akışına başlamasını sağlaması gerekmektedir. Bu PDU içerisinde kameranın görüntü aktarım durumunu tutan MIB nesnesi (streamStat = 1) ve onun kamerayı çalıştırmaya yönelik değeri bulunur. Set PDU'yu alan kameranın aygıt denetleyicisi tarafından kendi nesne değeri, aktarılan bu değerle günlenir. Bu nesnenin değerine bağlı olarak çalışan uygulama kameranın desteklediği özelliklere göre istemci ağ aygıtına görüntü akışını başlatır.



Şekil 4.24. Kapı kamerasından görüntü akışı sağlanması

Akış esnasında istemci varlık bağlantıyı kesmek istediğinde yine bir Set PDU kullanılır. Bu PDU içerisinde, örnekte streamStat olarak adlandırılan akış durum bilgisini tutan MIB nesnesinin değeri akışın durdurulmasına yönelik olarak sıfırlanır. İletiyi alan kamera aygıt denetleyicisi de kendi streamStat nesne değerini sıfırlar. Bu nesne değerinin sıfırlandığını algılayan görüntü akış uygulaması da akış işlemini sonlandırır. Bununla birlikte oturum denetimi için tutulan geçici nesnelerin ve oturum durumunu belirten sessionStat nesnelerinin değerleri de sıfırlanır.

4.2. SIP (Session Initiation Protocol)

Bu bölümde geliştirilmekte olan SIP isimli bir uygulama katmanı protokolü hakkında bilgiler verilecektir. Daha sonra da bu protokolün tanımlanmış olan temel ve işlevsel gereksinimlere uygunluğu incelenecektir.

4.2.1. SIP'in tarihçesi ve yapısı

SIP, IETF bünyesinde bulunan SIP çalışma grubu tarafından halen geliştirilmekte olan bir uygulama katmanı protokolüdür. İlk olarak 1996 sonlarında **Multicast Backbone (Mbone)** adlı konferans ağ yapısının sahip olduğu hizmet ve protokol kümesinin bir elemanı olarak kullanılmaya başlandı. **Mbone** genel Internet mimarisinin en üstünde bulunacak olan deneysel bir çoklu saçım ağ yapısı idi. Sesli ve görüntülü çoklu ortam içeriklerinin dağıtımı için kullanılmaktaydı.

SIP Mbone işlevlerinin en önemlilerinden birisini, yani Internet üzerinde başlatılacak olan ya da devam eden bir çoklu ortam oturumuna kullanıcıları davet etme görevini yerine getirmekteydi [29].

Protokol aynı dönemlerde popülerlik kazanan Internet telefonculuğu – IP Telefonculuğu düşüncesine yönelik olarak de denenmiş ve başarı sağlanmıştır. Böylece **VoIP (Voice over IP)** üzerinde de çalışabildiği kanıtlanmıştır. Bunun sonucu olarak da SIP protokolünün varlığı 1998 yılı sonlarında çıkarılan RFC 2543 ile de resmi olarak kabul edilmiş oldu. 1999 – 2000 yılları içerisinde sahip olduğu esneklik, genişletilebilirlik ve ölçeklendirilebilirlik özellikleri nedeniyle servis sağlayıcıların ilgisini çeken ve benimsenen protokol ITU, ETSI TIPHON, IMTC,

Softswitch Consortium ve JAIN gibi çeşitli standart kuruluşları tarafından da incelenmeye başlandı.

2001 yılından itibaren Moyer ve ark. ağ aygıtlarının denetimine yönelik olarak SIP protokolünün özellikleri ve yeterlilikleri ile ilgili araştırmalar yapmaya başladılar [3,30]. IETF SIP çalışma grubu bünyesinde yayınladıkları yazı ve taslaklarla SIP protokolünün Internet üzerinden aygıt kontrolü konusunda bir standart haline gelmesi için çalışmaktadırlar.

SIP temel olarak kullanıcılar arasında oturum başlatma, devam eden oturumu denetleme ve sonlandırma görevlerini yerine getirmektedir. Oturum çift yönlü bir telefon görüşmesi olabileceği gibi, bir çoklu ortam konferans görüşmesi gibi daha karmaşık bir yapıda da olabilir.

Yapısı HTTP ve SMTP protokollerine benzeyen metin tabanlı bir uygulama katmanı protokolü olan SIP'in mimarisi temel olarak iki bileşenden oluşmaktadır:

- **Kullanıcı Ajanlar :** Son birimler olarak düşünülen iş istasyonları, IP telefonlar, otomatik yanıt servisleri, ağ geçitleri ...gibi ağ aygıtlarında SIP iletişim desteği sağlamak amacı ile çalışan SIP yazılım parçasıdır. Kullanıcı ajanlar hem istemci hem de sunucu özelliklerine sahip olmalarını sağlayan iki işlevsel elamandan oluşurlar. Bunlar:
  - **İstemci Eleman :** SIP kullanıcı ajanının hizmet istek iletileri üreten bölümüdür. **UAC (User Agent Client)** olarak adlandırılırlar.
  - **Sunucu Eleman :** Bir SIP hizmet istek iletisi aldığında buna olumlu ya da olumsuz yanıt üreten yazılım bölümüdür. **UAS (User Agent Server)** olarak adlandırılırlar.

  Bu iki işlevsel parçanın bir aygıt içerisinde çalışıyor olması, aygıtların gerektiğinde hem istemci hem de sunucu olarak görev yapabilmesini, aynı zamanda istemci sunucu arasındaki iletişimlerin eşler arası iletişim mantığında gerçekleşmesini sağlamaktadır.
- **SIP Ağ Sunucuları :** Bu SIP yazılım biriminin amacı, SIP iletişimi esnasında isim çözme, kullanıcı konumu belirleme (İstemci birimin hedef birimin ağ adresini ya da makine adını bilmediği durumlarda) ve kendilerine

gelen SIP iletilerini yönlendirme protokollerinden yararlanarak diğer SIP sunucularına gönderme işlevlerini yerine getirmektir. SIP sunucular **durumlu (stateful)** ve **durumsuz (stateless)** olmak üzere iki farklı çalışma şekline sahiptirler. Aralarındaki tek fark durumlu şekilde çalışan bir SIP sunucusunun kendisine gelen istek iletilerini ve bunlara karşı aktardığı yanıtları, aynı zamanda aktarımında aracılık yaptığı istek iletilerini hatırlayabilmesidir. Durumsuz çalışan sunucular durumlu sunucuların aksine aktardıkları iletiler hakkında bilgi saklamazlar. Durumsuz sunucuların bu yüzden SIP ağlarının omurga altyapısını oluşturmada, durumlu sunucuların ise belirli kullanıcı alanlarını denetlemek için kullanıcı ajanlara yakın kullanılmaları daha uygundur.

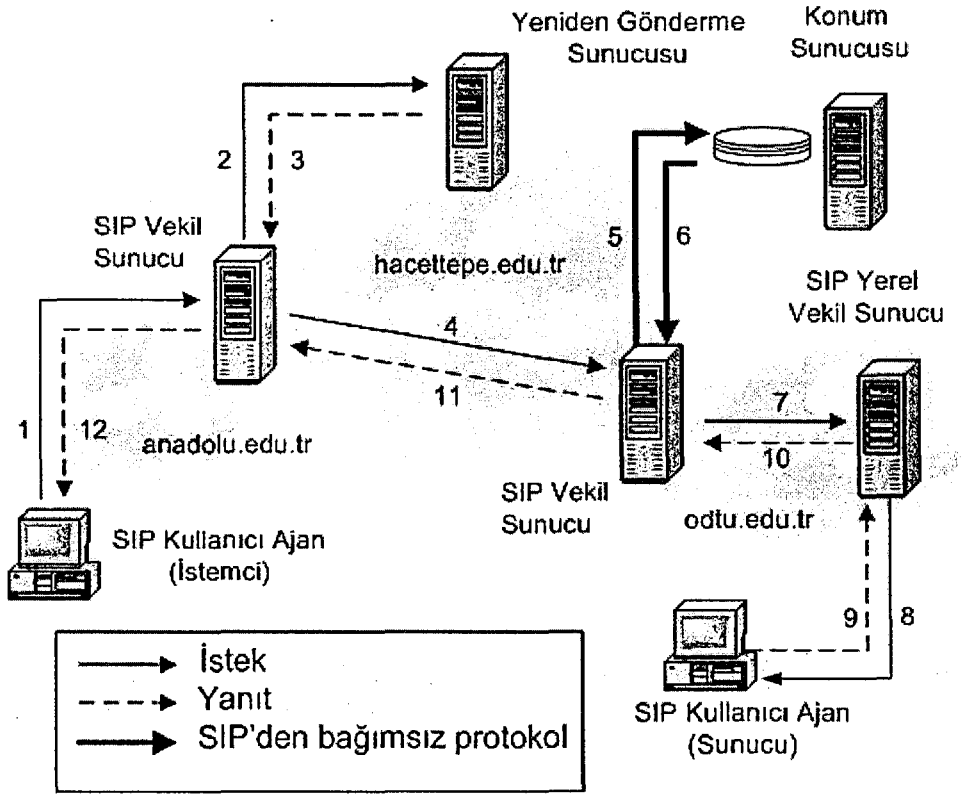
SIP sunucuları gerçekleştirdikleri işlemlere göre dört gruba ayrılırlar:

- *Durumlu Vekil Sunucu* ***(Stateful Proxy Server)*** *:* Durumlu şekilde çalışan SIP sunucularıdır. Görevleri kendilerine gelen iletileri hedeflerine ulaşabilmeleri için geçmeleri gereken bir sonraki ağ birimine yönlendirmektir. Bu birim hedef kullanıcı ajan olabileceği gibi başka bir SIP sunucu (İletinin yolu üzerinde birden fazla SIP sunucusu bulunabilir) da olabilir. Durumlu çalışmasından dolayı kendisine gelen iletileri hedef kullanıcının tüm olası konumlarına koşut olarak göndererek, gelen yanıt iletilerinden en uygununu belirleyip istemci ajana aktarabilir.
- *Durumsuz Vekil Sunucu* ***(Stateless Proxy Server)*** *:* Durumsuz çalışma şekline sahip SIP sunucularıdır. Yani görevleri kendilerine gelen iletilerle ilgili hiçbir bilgi tutmadan hedefleri üzerindeki bir sonraki ağ birimine yönlendirmektir. Durumlu vekil sunucu ile aralarındaki tek fark çalışma şekilleridir. Durumsuz şekilde çalıştıklarından durumlu sunuculara göre görevlerini daha hızlı gerçekleştirirler. Bu yüzden SIP ağ omurga altyapısında kullanılırlar.
- *Yönlendirme Sunucusu :* Kendisine gelen isteğin hedef adresine karşılık sıfır ya da daha çok sayıda yeni adres eşleştirmesi yapar ve bu adres

değerlerini istemciye gönderir. Böylece vekil sunucunun aksine istekleri başka bir sunucuya aktarmaz, gitmeleri gereken yeni adresleri belirleyip istemciyi bu konuda bilgilendirir.

- *Kayıt Sunucusu* ***(Registrar Server)*** *:* Kullanıcı ajanlar bazı durumlarda bağlı oldukları yerel vekil ya da yönlendirme sunucusuna mevcut konumları hakkında bilgi verme gereksinimi duyarlar. Konum bilgilerini kayıt eden ve ilgili vekil ya da yönlendirme sunucusuna konum belirleme hizmeti sunan sunuculara **kayıt sunucuları** adı verilir.

Şekil 4.25.'de SIP protokolünün sahip olduğu temel bileşenler bir ağ yapısı içerisinde gösterilmektedir.



Şekil 4.25. Temel bileşenler üzerinden SIP iletişimi [29]

4.2.2. SIP ileti türleri

SIP diğer birçok uygulama katmanı protokolü gibi istek/yanıt mantığı ile çalışmaktadır. Bu yüzden de iki temel ileti türüne sahiptir:

- İstek iletileri : İstemci ajandan sunucu ajana gönderilen iletilerdir.
- Yanıt iletileri : Sunucu ajandan istemci ajana gönderilirler.

İletilerin veri alanları SIP'nin standart tanımında **SDP (Session Description Protocol)** adı verilen bir protokol ile oluşturulurken, ağ aygıt kontrolü kapsamında tanımlanan uzantılar sayesinde bu kısıt ortadan kaldırılmıştır. Veri alanına, mevcut herhangi bir **MIME (Multipurpose Internet Mail Extensions)** türüne sahip veri konulabildiği gibi halen geliştirilmekte olan ve **DMP (Device Messaging Protocol)** olarak adlandırılan **XML (Extensible Markup Language)** tabanlı [34] yeni bir MIME türü de veri tanımlama amacı ile kullanılabilecektir.

İstek iletileri SIP işlemlerinin gerçekleşmesi için kullanılan komutları içerirler. Bu komutlar SIP mimarisinde **yöntem** olarak adlandırılırlar. Altı adet standart SIP istek yöntemi mevcuttur [32]. Bunlar:

i. *INVITE:* Bir kullanıcı ajanın başka bir ajana belli bir işleme yönelik olarak oturum açma isteğinde bulunması için kullanılır.

ii. *ACK:* INVITE iletisi için verilen yanıt iletisini onaylamak için kullanılır.

iii. *BYE:* Mevcut bir oturumu sonlandırmak için kullanılır.

iv. *CANCEL:* Belli durumlarda (genelde yanıt iletisinin gecikmesi üzerine) sunucu ajandan işlemekte olduğu istek iletisine yönelik işlemleri sonlandırması istenebilir. Bu istek CANCEL yöntemi kullanarak gerçekleştirilir.

v. *OPTIONS:* Bir UAC'ın başka bir UAC ya da vekil sunucunun yeteneklerini sınaması için kullanılır. Böylece istemci ajan, sunucu ajanın desteklediği yöntemleri, veri içerik türlerini, uzantıları ...vb oturum isteğinde bulunmadan öğrenebilir.

vi. *REGISTER:* Bir UAC tarafından bir vekil ya da yönlendirme sunucusundan mevcut konum bilgisinin kaydedilmesi isteminde bulunmak için kullanılır. Kayıt bilgisi istemciye erişim için kullanılabilecek adres ya

da adresleri içerir. İlgili vekil ya da yönlendirme sunucusu kayıt sunucusunu kullanarak bu adres değerlerini kaydeder. Gerektiğinde vekil ya da yönlendirme sunucusuna konum belirleme mekanizması bağlamında bu bilgileri sağlar.

SIP'in bu standart yöntemlerinin yanında ağ aygıt kontrolüne yönelik olarak aşağıdaki istek yöntemleri de uzantı olarak sonradan tanımlanmışlardır:

i. *DO [31]:* Standart bir SIP yöntemi değildir. INVITE yöntemi kullanılarak başlatılan bir oturum üzerinden iletişim kurmak yerine DO yöntemi ile oturum kurmaya gerek kalmadan ağ aygıtından istekte bulunulabilmekte ve standart bir SIP yanıt iletisi ile cevap alınabilmektedir. Yani DO ve INVITE arasındaki ilişki UDP ve TCP arasındaki ilişkiye benzetilebilir. Bunun yanında DO yöntemi, diğer standart istek yöntemlerinden farklı olarak ileti gövdesinde örneğin INVITE yönteminin kullandığı SDP tanımları yerine başta DMP olmak üzere farklı MIME türlerinde veriler de taşıyabilir.

ii. *SUBSCRIBE [32]:* Bir ağ aygıtında meydana gelebilecek bir işlem (olay) hakkında haberdar olabilmek için ilgili aygıta yapılan olay üyelik isteği için kullanılır. Bu sayede ilgili olay gerçekleştiğinde aygıt aşağıda anlatılan "NOTIFY" yöntemi sayesinde olayın gerçekleştiğini istemciye bildirir.

iii. *NOTIFY [32]:* Bir ağ aygıtından "SUBSCRIBE" yöntemi kullanılarak bildirimi istenen olayın gerçekleştiğini ilgili istemci kullanıcı ajanına haber vermek için kullanılır.

SIP protokolü yanıt iletileri için sayısal kodlardan yararlanır. Bunlar genel olarak Çizelge 4.1a.'da gösterildiği gibi sınıflandırılmaktadır (Çizelgede "xx" olarak verilen kod parçaları 00 ile 99 arasında değişen değerler alabilmektedir.).

Çizelge 4.1a. SIP yanıt ileti kod türleri

1xx	Bilgi niteliğinde	İstek alındı, isteği işlemeye devam ediliyor
2xx	Sonuç	İşlem başarılı bir şekilde alındı, anlaşıldı ve onaylandı.
3xx	Yeniden gönderme	İsteği karşılayabilmek için ek bir işlem yapılması gerekir.
4xx	İstemci hatası	İstek yanlış söz dizime sahip ya da sunucu tarafından yerine getirilememiş olabilir.
5xx	Sunucu hatası	Sunucu açıkça geçerli olan bir isteği yerine getirmekte başarısız olmuş.
6xx	Genel hata	İstek hiçbir sunucuda yerine getirilememiş.

Bunlardan 1xx koduna sahip olanlar geçici yanıtlardır. Sunucu tarafından işlemin sürdürüldüğüne dair bilgilendirmek amacı ile kullanılırlar. Geri kalan yanıt kodları sonuç yanıtlarıdır. Sunucu tarafından herhangi bir nedenden dolayı işlemlerin sonlandırılacağını istemciye bildirmek için kullanılırlar. Çizelge 4.1b, Çizelge 4.1c, Çizelge 4.1d ve Çizelge 4.1e'de sırası ile 1xx, 2xx, 3xx, 4xx, 5xx ve 6xx türündeki standart SIP yanıt kodları ve anlamları gösterilmektedir. Buradaki xx gösterimleri sayısal değerler almaktadır.

Çizelge 4.1b. SIP 1xx ve 2xx yanıt ileti kodları

100	*Devam ediliyor*	**200**	***Tamam (OK)***
180	***Çalıyor (ringing)***		

Çizelge 4.1c. SIP 3xx yanıt ileti kodları

300	*Çoktan seçme*
301	***Kalıcı yerdeğiştirme (moved permanently)***
302	***Geçici yerdeğiştirme (moved temporarily)***

Çizelge 4.1d. SIP 4xx yanıt ileti kodları

400	*Yanlış istek*	**481**	*İletişim ayağı bulunamadı*
401	*Yetkisiz*	**482**	*Döngü saptandı*
403	*Yasak*	**483**	*Aşırı adım sayısı*
404	*Bulunamadı*	**484**	*Eksik adres*
407	*Vekil kimlik denetimi gerekli*	**485**	*Belirsiz*
408	*İstek zaman aşımı*	**486**	*Meşgul*
420	*Yanlış uzantı*	**487**	*İstek iptali*
480	*Geçici kullanım dışı*	**488**	*Kabul edilemez*

Çizelge 4.1e. SIP 5xx ve 6xx yanıt ileti kodları

500	*Sunucu iç hatası*	**600**	*Meşgul*
501	*Gerçekleştirilmedi*	**601**	*Ret*
502	*Yanlış ağ geçidi*	**604**	*Mevcut değil*
503	*Servis kullanım dışı*	**606**	*Kabul edilemez*
504	*Ağ geçidi zaman aşımı*		
505	*Sürüm desteği yok*		

4.2.3. SIP İleti Formatı

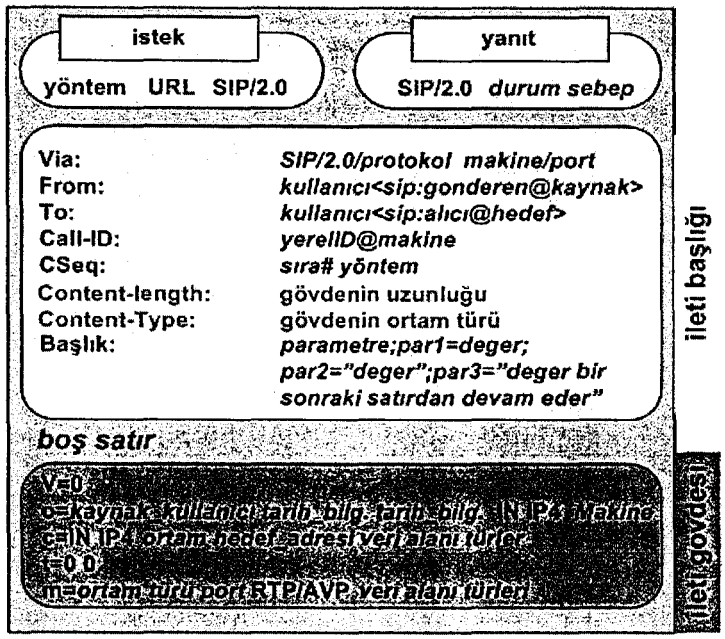
SIP iletileri üç ana parçadan oluşurlar (Şekil 4.26.):

i. *Başlangıç satırı :* Tüm SIP iletilerinde bulunan bu satır aktarılan iletinin türünü (istek iletilerinde ileti türünü, yanıt iletilerinde yanıt kodunu) ve protokol sürümü bilgilerini taşır. Başlangıç satırı, İstek satırı ya da Durum satırı olarak kullanılır: İstek satırı olarak kullanıldığında hedef **URL (Universal Resource Locator)** değerini içerir. Durum satırı ise bir durum kodu ile bu kodun metinsel tanımını içerir.

ii. *İleti başlığı :* Bu bölüm ileti öznitelik bilgilerini tutmak ve iletinin anlamını değiştirmek için kullanılır. HTTP başlık alanları ile benzer sözdizimsel ve anlamsal yapıya sahiptirler.

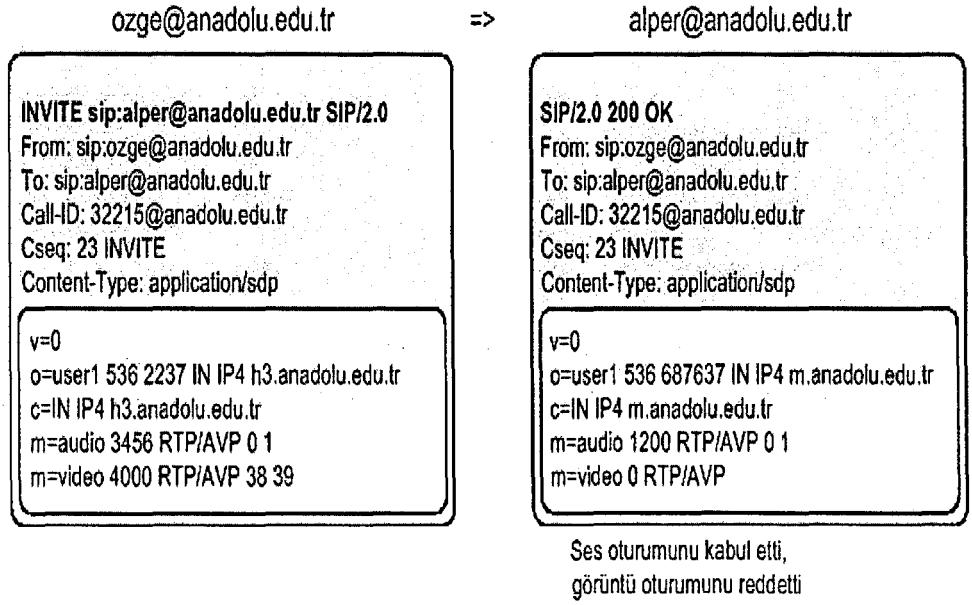
iii. *İleti gövdesi :* İleti gövdesi başlatılmak istenen oturum ile doğrudan ya da dolaylı ilgili tanımları (bir çoklu ortam oturumunda ses ve görüntü kodeklerinin seçimi ...gibi) içerir. Bu bölüm istek ve yanıt iletilerinin her

ikisinde de bulunur. Gövde içerisindeki tanımlar SDP, MIME, XML ...vb gibi çeşitli yöntemlerle gerçekleştirilebilmektedir.



Şekil 4.26. SIP ileti formatı

Şekil 4.27.'de istek ve yanıt iletileri örneklenmektedir.



Şekil 4.27. SIP istek ve yanıt iletisi örneği

4.2.4. SIP'in çalışma mantığı

SIP kullanıcılar arasında etkileşimli iletişim oturumları başlatmak, yönetmek ve sonlandırmak için tasarlanmış bir uygulama katmanı protokolüdür. Başlatılan oturumun içeriği hakkında bir denetim mekanizmasına sahip değildir, ancak iletileri bu bilgileri taşıyabilir. Oturumla ilgili bu bilgilerin tanımlanmasında web ve e-posta hizmetlerinde aktarılan verinin içerik tanımını (HTML, ses, görüntü, resim ...gibi) yapmak için sıkça kullanılan MIME türlerinden yararlanır. SIP'de veri tanımlamada en çok kullanılan MIME türü SDP'dir.

SIP istemci/sunucu mimarisine sahip bir protokoldür. Bir oturum başlatmak için istemcinin bir INVITE iletisi hazırlaması gerekir (Şekil 4.27.). Bu iletiyi alan sunucu iletinin içerisindeki oturum sebebi ve parametreleri ile ilgili bilgileri inceler ve başlangıç satırında sonuç durum bilgisini temsil eden bir sayısal kod bulunduran yanıt iletisini istemciye gönderir.

SIP'nin standart tanımında nesneleri yani makineleri ya da kullanıcıları adresleyebilmek için yapısı telnet ya da e-posta URL yapısına benzer SIP URL'lerinden yararlanılır. SIP URL yapısı genel olarak aşağıdaki gibidir:

<varlık>@<konum>

Burada

varlık bölümüne : telefon numarası ya da kullanıcı ismi,

konum bölümüne: ilgili makinenin alan adı ya da ağ adresi yerleştirilir (Örn: *sip:oguner@pc.com*). SIP iletileri doğrudan hedef adrese gönderilebileceği gibi çoğunlukla bir vekil sunucuya gönderilir. Vekil sunucu kendisine gelen iletilerinin yönlendirilip hedeflerine ulaştırılmasından sorumludur.

Bir vekil sunucu kendisine gelen bir istek iletisini birden fazla adrese gönderme yeteneğine sahiptir. Bu işleme **çatallama (forking)** adı verilir ve bu sayede aranmakta olan aygıtın ya da kullanıcının birden fazla konuma gönderilen aynı istek iletisi sayesinde bulunma olasılığı artar.

Şekil 4.25.'de görülen SIP bileşenlerinin oluşturduğu ağ yapısında bir istemci ve bir sunucu SIP elemanı arasındaki iletişim örneklenmiştir. Burada örnek olarak "anadolu.edu.tr" alan adına sahip alt ağda bulunan istemci birim örneğin bir veri akış

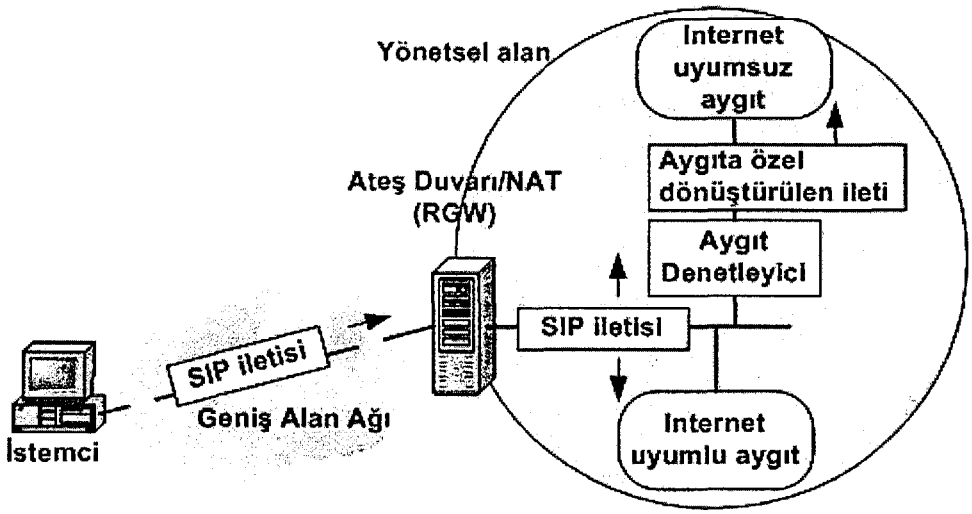
oturumu gerçekleştirmek için "bbm723@hacettepe.edu.tr" adresli kullanıcıya erişmek istemektedir. Bunun için oluşturduğu istek iletisini kendisinde adresi kayıtlı olan vekil sunucuya gönderiyor. Vekil sunucu ise hedef adresinin bir parçası olan "hacettepe.edu.tr" adlı alanda bulunan sunucuya iletiyi aktarıyor. Bu sunucu bir yönlendirme sunucusu olduğundan SIP hedef adresinin kullanıcı bölümünü ("bbm723") sahip olduğu veri tabanında arıyor ve bu kullanıcının o an "odtu.edu.tr" alan adlı alt ağda bulunduğunu öğreniyor. Bu bilgiyi kendisine istek iletisini gönderen vekil sunucuya aktarıyor. Böylece vekil sunucu hedef adresi "bbm723@odtu.edu.tr" olarak değiştirip iletiyi "odtu.edu.tr" alt ağındaki sunucuya gönderiyor. Buradaki sunucu bir vekil sunucu olduğundan kendi veri tabanını araştırıp kullanıcının bulunduğu asıl konum adresine ("bbm723@baum.odtu.edu.tr") istek iletisini aktarır. "baum.odtu.edu.tr" alanındaki vekil ajan ise kullanıcının hangi bilgisayarda çalıştığını ise SIP'nin "REGISTER" iletileri sayesinde bulur. "bbm723" kullanıcısı bilgisayarındaki SIP istemci kullanıcı ajanı çalıştırdığında, ajan çalıştığı makinenin adresi ("bilg1.baum.odtu.edu.tr") ile kullanıcıyı ilişkilendirmesi için vekil sunucuya "REGISTER" iletisi gönderir. Vekil sunucu ise sahip olduğu veri tabanını bu bağlantı bilgisi ile günceller. Böylece kendisine gelen tüm istek iletilerini "bbm723@bilg1.baum.odtu.edu.tr" adresine gönderir.

İstek iletisini alan "bbm723" kullanıcısının o anda çalıştığı bilgisayarın ("bilg1") bu isteğe karşılık oluşturduğu SIP yanıt iletisi ise 9, 10, 11 ve 12 olarak numaralanmış adımları takip ederek istemci bilgisayara ulaşır. Eğer gönderilen istek iletisi INVITE yöntemini taşıyorsa, yanıt iletisine karşı aynı yolu takip ederek istemciden sunucuya bir "ACK" iletisi gönderilir ve oturum başlatılmış olur.

Ağ aygıt kontrolü kapsamında ağ aygıtlarına erişim güvenliğini arttırmak ve kısıtlı sayıdaki ağ adreslerini daha etkin kullanabilmek için bir grup ağ aygıtına tek bir noktadan (ağ adresinden) erişme fikri benimsenmiştir. Bu ağ adresine sahip SIP elemanı **Mesken Ağ Geçidi (RGW-Residential Gateway)** olarak adlandırılır. Ateş duvarı ve NAT olarak da görev yapan bu birim yönettiği alan içerisindeki ağ aygıtlarının geniş alan ağına güvenli bağlantı kurmalarını sağlar. NAT sayesinde ağ aygıtlarına sanal ağ adresleri (IP) atanarak erişimin RGW üzerinden gerçekleşmesi

sağlanır. RGW'ler SIP vekil sunucu yazılımlarını da içerip bir vekil sunucu olarak da çalışabilirler.

Bazı ağ aygıtları SIP kullanıcı ajanlarını çalıştıracak kadar yeni ve yeterli teknolojiye sahip olmadığından ve ağ bağlantı yetenekleri olmadığından bu aygıtların ilgili RGW'ye bağlanıp iletişim kurabilmesi için aracı görevi gören SIP aygıtlardan yararlanılır. Bu aygıtlara **aygıt denetleyici** adı verilir (Şekil 4.28.). Bu SIP elemanları kendilerine bağlı olan ağ aygıtının desteklediği fiziksel sistem ve iletişim protokol işlemleri ile SIP protokol işlemleri arasında dönüştürme işlemi gerçekleştirir.



Şekil 4.28. RGW ve aygıt denetleyici içeren SIP ağ yapısı [29]

4.2.5. SIP protokolünde temel gereksinimler

Bu bölümde SIP 3. bölümde tanımlanan temel gereksinimlere göre incelenmektedir.

4.2.5.1. SIP protokolünde geniş alan iletişim desteği

SIP farklı konumlarda bulunan kullanıcılar arası çoklu ortam oturumları kurmak ve denetlemek için tasarlanmış bir istek/yanıt protokolü olduğu için her türlü güvenilir ya da güvenilir olmayan aktarım protokolleri (UDP, SCTP ve TCP ...gibi) kullanarak iletilerini aktarabilmesi sağlanmıştır. Yani SIP aktarım katmanı protokollerinden büyük ölçüde bağımsız olarak çalışabilmektedir [33]. SIP standart

tanımında önerilen aktarım protokolleri TCP ve UDP olsa da ATM AAL5, IPX, frame relay ya da X.25 gibi ağ teknoloji ve protokolleri ile de çalışma imkanı vardır. Bununla birlikte SIP'nin adresleme yapısı da geniş alan iletişimine izin verecek şekilde düşünülmüştür. Mevcut Internet ortamında kullanılan bu adresleme mantığı ile geniş alanda nesnelerin ayırt edilebilmesi ve birbirleri ile iletişim kurabilmesi sağlanmıştır.

4.2.5.2. SIP protokolünde güvenilir iletişim desteği

SIP bir istek/yanıt protokolü olduğundan istemciden sunucuya doğru gönderilen istek iletilerine sunucudan mutlaka yanıt iletileri döndürülür. Bir istemciden bir sunucuya gönderilen ve tekrarlanan bir istek iletisi ile bu iletiye karşılık dönen yanıt iletilerinin tümüne birden bir SIP **hareketi (transaction)** adı verilir.

SIP protokolünde güvenilir iletişim denetimi kullanılan aktarım protokolünün güvenilir olup olmama özelliğine göre değişir.

Kullanılan aktarım protokolü güvenilir bir protokol* (TCP gibi) *ise: Güvenilir aktarım protokolü aracılığı ile kurulan her bağlantı bir ya da daha çok SIP hareketi içerebilir. İstemci sunucu ile arasındaki bağlantıyı, aykırı durumlar dışında, bir sonuç iletisi (sonuç yanıt kodu içeren yanıt iletisi) alana kadar sürdürmek zorundadır. Eğer istemci bağlantıyı ilk sonuç iletisi gelmeden kapatıp yeniden kurarsa, sunucu bu durumda kendisine bir CANCEL iletisi gönderilmiş gibi davranarak ilgili bağlantıyı kullanan tüm işlemleri sonlandırır. Böylece istemcilerin yanlış çalışmalarından dolayı sunucu tarafındaki bağlantı durumlarının süresiz olarak açık halde kalması engellenmiş olur.

Kullanılmakta olan bir bağlantıyı kapatma görevi aslında istemci ajanın olmasına rağmen, sunucunun da sonuç yanıt iletisini daha önceden göndermiş olması şartıyla bağlantıyı sonlandırma hakkı vardır. Eğer sunucu bağlantıyı sonuç yanıt iletisini istemciye göndermeden önce kapatırsa, istemci bu durumu 500 kodlu (Sunucu İç Hatası) (Çizelge 4.1e.) hata olarak yorumlamalıdır.

SIP hareketleri sonucunda sunucu bağlantıyı açık bırakırsa, gerektiğinde istemci bu bağlantıyı yeni SIP hareketleri ya da benzer protokol ailesinden (HTTP, SMTP, SCTP ...gibi) istek iletilerini göndermek için kullanabilmektedir. Bununla birlikte bağlantı sona erdikten sonra sunucu istemciye bir yanıt iletisi göndermek isterse kendisine gelen iletilerdeki **Via** başlık alanındaki adres bilgisini kullanarak bağlantı kurabilir [33].

Kullanılan aktarım protokolü güvenilir olmayan bir protokol* (UDP gibi) *ise:
Güvenilir aktarım protokollerinin yeniden gönderme görevini SIP'in yapması gerekir. Bu işlemi üstel geri çekilme yöntemini kullanarak gerçekleştirir. Yani T gibi belli bir zaman aralığı sınırına kadar tekrar gönderilecek olan ileti bir önceki gönderimden bir Δt zamanı kadar sonra gönderilir. Yani :

T : Tekrar gönderme yapılabilecek azami zaman sınırı

Δt : İleti tekrarları arasındaki bekleme süresine eklenen zaman miktarı.Böylece her tekrardan önce gerekli bekleme süresi her tekrardan sonra Δt kadar artırılmış olur.

Tekrar gönderme işlemi karşı taraftan geçici yanıt iletileri dışında bir yanıt iletisi alınana ya da T sınır değerine ulaşılana kadar sürdürülür. Herhangi bir yanıt iletisi alınmadıysa bundan sonra tekrarlar T kadar sabit zaman aralıkları ile gerçekleştirilmeye başlanır. Tekrar gönderme işlemi standart olarak 11 kere yapılmasına karşın hala bir kesin yanıt iletisi alınamadıysa aktarıma son verilir. Bu yöntem içerisinde kullanılan zaman parametrelerinin varsayılan en düşük standart değerleri Δt=*500 ms (milisaniye) ve T=4 s (saniye)*'dir.

Sunucu kendisine gelen her istek tekrarına karşılık yanıt iletisini tekrar gönderir. İstemciye gönderdiği sonuç yanıt iletisinin yerine ulaşıp ulaşmadığını anlayamayacağından, istek iletisinin tekrarlanma olasılığına karşı istek iletisinin sonuç bilgilerini standart olarak en az $10 * T$ süresi kadar saklamalıdır [32].

4.2.5.3. SIP protokolünde ağ aygıtları konumdan bağımsız şekilde tek (unique) adreslenebilmeli ve bir ağ adresinde birden fazla mantıksal aygıt bulunabilmeli

SIP tarafından varlıkları (kullanıcı, makine) belirlemek için kullanılan adresler URL şeklinde kodlanmaktadır. E-posta sistemi düşünüldüğünde bir kullanıcıya servis sağlayıcı tarafından atanan e-posta adresi dünya çapında tek olmak durumundadır. SIP de e-posta sistemindekine çok benzer bir adresleme mantığına sahip olduğundan ağ aygıtlarını tek adresleyebilme özelliğine sahiptir.

SIP adresleri genel olarak:

sip:<varlık>@<konum>

yapısına sahiptir. Bu yapıda adresin konum bölümü ilgili ağ aygıtının ilk kullanıma başladığında bulunduğu konumun alan adı ya da ağ adresi olabilmektedir. Bilindiği gibi alan adları ve gerçek ağ adresleri dünya çapında tek değerlerdir. SIP adreslerinin varlık bölümü ise ilgili varlığa ilk bulunduğu konumda kendisi ile aynı türde olan aygıtların isimlendirilmesine uygun olacak şekilde atanan bir isimdir. Buna örnek olarak evin bir odasına yeni bir lamba alındığı düşünüldüğünde bu lambayı isimlendirmek için örneğin sıra numarası kullanılabilir. Yani evdeki tüm lambalar sıra ile *lambaX* yapısında X yerine sıra numarası gelecek şekilde isimlendirilebilir. Böylece yeni alınan lambanın adresi şöyle olabilir:

sip:lamba12@ev.net *ya da*

sip:lamba12@193.130.45.32

Bunun yerine daha yapısal bir adresleme mantığı da kullanılabilir. Buna göre bir şirketin tüm odalarında birer klima olduğu düşünülürse, bu aygıtlar bulundukları odanın örneğin numarasına göre şu şekilde adreslenebilmektedir:

sip:klima.45@ofis.net

Ağ aygıtlarının denetimi kapsamında SIP'nin e-posta adres yapısına benzer adresleme mantığı üzerinde bazı değişiklikler önerilmiştir [31]. SIP adresinin varlık bölümü için **SLP (Service Location Protocol)** [35, 36] tarafından tanımlanmış olan isimlendirme yapısı kullanılmıştır. Buna göre örneğin

[slp:/d=lamba,r=mutfak,u=ozge]@ev.net

şeklinde tanımlanmış bir SIP adresinde

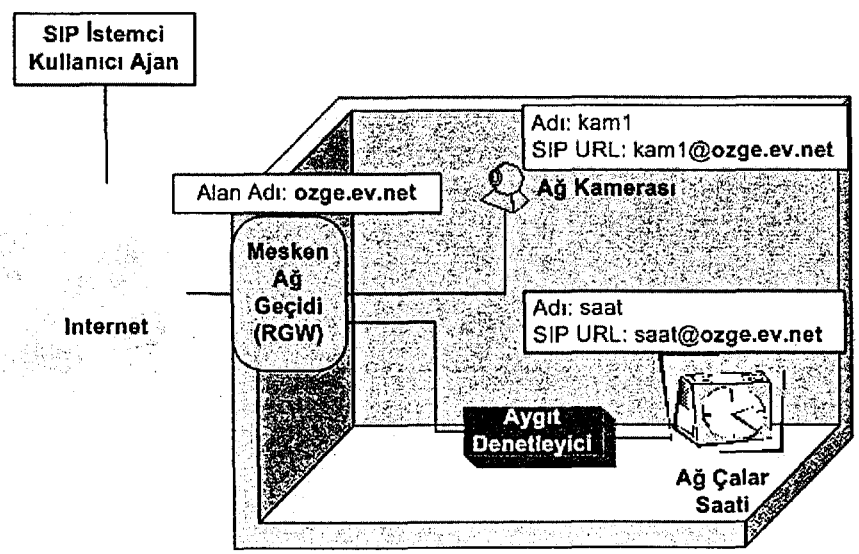
d: aygıt (device)

r: oda (room)

u: kullanıcı (user)

olarak düşünülür (Burada köşeli parantezler içerisindeki bölüm ileti içerisine gizlilik açısından Base64 şifreleme yöntemi ile şifrelenip yerleştirilecektir). Böylece bu adresin *"ozge"* kullanıcısının evindeki (*ev.net* alan adına sahip) mutfak lambasına ait olduğu anlaşılmaktadır. Bu adres yapısı kullanılarak bir alan içerisinde bulunan fakat adresleri bilinmeyen ağ aygıtlarının sağladıkları hizmetlere göre belirlenmesi oldukça kolaylaşmaktadır.

SIP bünyesinde ağ aygıtlarına bulundukları alan dışından doğrudan (aygıtın bir gerçek ağ adresinin olması gerekir) erişilebileceği gibi, bir denetleyici birim üzerinden de erişilebilmektedir. İkinci durumda bir alan içerisindeki ağ aygıtlarının dışarı ile iletişiminden sorumlu olan SIP elemanı mesken ağ geçidi olarak da adlandırılan bir vekil sunucudur (Şekil 4.29.). SIP mesken ağ geçidi sayesinde tek bir ağ adresi (ya da alan adı) kullanılarak bir yönetsel alan içerisindeki ağ aygıtlarına erişilebilmektedir. Şekil 4.29.'de görüldüğü gibi ilgili alan içerisindeki ağ aygıtlarına *ozge.ev.net* alan adı kullanılarak erişilebilmektedir.



Şekil 4.29. SIP Mesken ağ geçidi kullanımı

4.2.5.4. SIP protokolünde güvenlik, kimlik denetimi ve gizlilik desteği

Bilgisayar ağlarının mevcut yapısı (adresleme yöntemi, iletişim protokolleri) göz önünde tutulduğunda bir aygıta doğrudan erişimin çeşitli güvenlik problemleri doğuracağı açıktır. Bu yüzden ağ aygıt kontrolü bağlamında bir alan (oda, ev, ofis ...gibi) içerisindeki ağ aygıtlarına erişimlerin tek bir noktadan sağlanması için mesken ağ geçitlerinden yararlanılmaktadır (Şekil 4.29.). Genelde vekil sunucu olarak da çalışan bu SIP elemanları sağladıkları yönetsel alan içerisindeki ağ aygıtlarının dışarı ile iletişimini denetler ve yönlendirirler. Bunun için güvenlik ve gizlilik modellerinden yararlanırlar [33]. Ancak SIP'in herhangi iki taraf arasındaki iletişim oturumlarını denetlemek amacı ile geliştirildiği düşünülürse, ağ aygıt kontrolü kapsamında temel tanımlara göre bazı farklılıkların olması gerektiği açıktır. Genel SIP tanımında güvenliği sağlamak için açık anahtar teknolojisinin kullanılmasının gerekliliği üzerinde durulmaktadır [33]. Bunun sebebi ise SIP oturumlarının genelde aralarında hiçbir bağlantı olmayan birimler arasında gerçekleşmekte olduğunun düşünülmesidir. Ancak ağ aygıt kontrolünde ağ aygıtları, bu aygıtlara uzaktan erişmek isteyen kullanıcı ya da diğer ağ aygıtları ile bir şekilde bağlantılı olmak durumundadır. Bu yüzden de açık anahtar yerine paylaşılan gizli anahtar kullanmak güvenlik ve verimlilik seviyesini arttıracaktır [34]. Bununla birlikte bazı unsurlara bağlı olarak açık anahtar kullanımının da tercih edilebileceği düşünülerek her iki yöntemin de gerçekleştirimde yer alması daha anlamlıdır.

SIP'de gizlilik ve kimlik denetimini sağlamak amacı ile **Uçlar Arası Şifreleme (End-to-End Encryption)** ve **Adımlar Arası Şifreleme (Hop-by-Hop Encryption)** olmak üzere iki yöntem tanımlanmıştır. Bu yöntemler kullanılarak SIP iletilerinin gövde ve bazı başlık alanları şifrelenebilmektedir.

Uçlar arası şifreleme modeli adımlar arası şifreleme modeline göre daha verimli ve güvenli çalışır. Çünkü burada güvenilmesi gereken tek bir birim vardır, o da hedef **kullanıcı ajandır (User Agent - UA)**. Adımlar arası şifrelemede ise hedefe ulaşmak için geçilen tüm adımlara (vekil sunucuların oluşturduğu) güvenmek gerekmektedir. İstemci ile sunucu ajanın arasındaki iletişim şekline göre bu iki yöntemden birisi tercih edilebilir.

- Doğrudan iletişim (istemci ile hedef aygıt ya da RGW arasında) söz konusu ise uçlar arası şifrelemeyi kullanmak daha mantıklıdır. Böylece aktarılan iletiler istemci ile hedef yönetsel alan arasında paylaşılan bir gizli anahtar ile şifrelenir.
- Dolaylı iletişim (hedef yönetsel alan dışındaki bir vekil sunucu üzerinden) söz konusu olduğu durumda ise adımlar arası şifrelemenin kullanılması daha uygun olacaktır. Buna göre iletiler istemci UA ile dış vekil sunucu arasında ortak kullanılan gizli anahtar ile UA tarafından şifrelenip dış vekil sunucuya gönderilir. Burada şifresi çözülüp doğruluğu onaylanan ileti bu sefer de dış vekil ile hedef RGW arasında paylaşılan bir gizli anahtar ile şifrelenip hedef RGW'ye gönderilir. Ve hedef RGW de aynı şekilde şifreyi çözüp iletinin doğruluğunu denetler.

İleti gövdesi her iki şifreleme yönteminde de sorunsuz şifrelenebilmesine rağmen, şifrelenebilen ileti başlık alanları farklılık gösterir. Bunun sebebi ise bazı başlık alanlarının sadece uç birimleri ilgilendirmesine karşın bazı başlık alanlarının ara vekil sunucular tarafından kullanılıyor olmasıdır. Şifrelenemeyen bu alanlardan *To:* başlık alanının içerdiği SIP adresleri aygıt kontrolü bağlamında ilgili ağ aygıtının konumu, türü gibi kritik bilgiler içerebileceğinden bu adreslerin *varlık* bölümlerinin ("@" işaretinin solundaki bölüm) de ayrıca şifrelenmesi güvenliği artıracaktır [31].

SIP kimlik denetimi için, tasarımında esinlenilen HTTP'nin kullandığı mekanizmaları (Temel ve Özet Kimlik Denetimi) benimsemiştir [37]. Bunlardan temel kimlik denetimi modelinde kullanıcı adı ve şifre bilgileri şifrelenmeden, düz yazı şeklinde aktarıldığından, özet kimlik denetiminin (kimlik bilgileri şifrelenerek aktarılır) kullanımı daha güvenlidir.

UAS'ler kendileri ile iletişim kurmak isteyen UAC'lerin kimliklerini sorgulamak için 401 (Yetkisiz) (Çizelge 4.1d.) yanıt iletisini gönderirler. Vekil sunucular hariç kayıt ve yönlendirme sunucuları da 401 kodlu (Çizelge 4.1d.) yanıt iletisini kullanabilirler. Vekil sunucular ise 401 yerine 407 (Vekil Kimlik Denetimi Gerekli) (Çizelge 4.1d.) yanıt iletisini kullanırlar.

Bununla birlikte SIP'de de HTTP'de olduğu gibi Proxy-Authentication, Proxy-Authorization, WWW-Authentication ve Authorization başlık alanları çeşitli iletilere kimlik denetimi işlemleri için eklenmektedir.

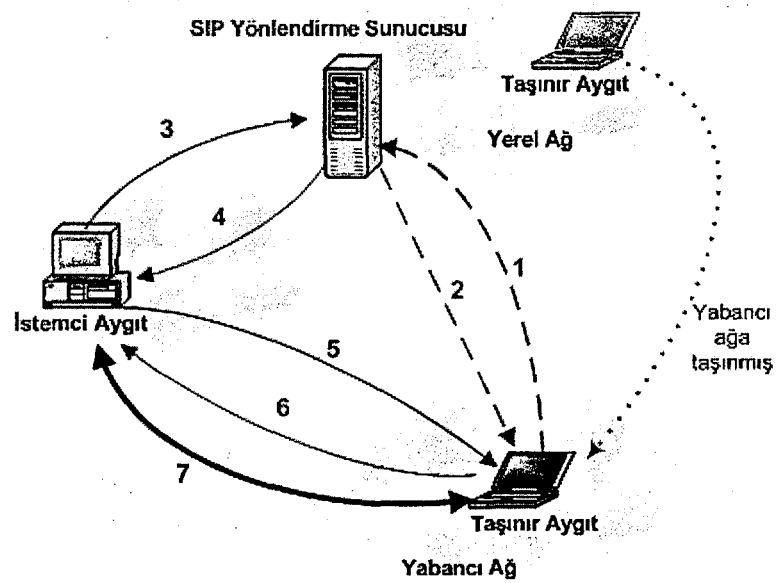
SIP'de kimlik denetimi iki bölüm olarak incelenebilir:

1. *Kullanıcılar arası kimlik denetimi:* Bir UAS kendisine gelen bir istek iletisini yanıtlamadan önce, bu iletiyi gönderen UAC'nin kimliğini sorgulayabilir. İstek iletisinin içerisinde gerekli kimlik bilgileri verilmemişse, bu bilgileri 401 (Çizelge 4.1d.) yanıt iletisi aracılığı ile UAC'den ister. Yanıt iletisi içerisinde WWW-Authentication başlık alanı bulunmak zorundadır. Bu alan içerisinde erişilmek istenen ajanın bağlı olduğu bölge - güvenlik alanı (bir sunucu üzerinden erişilen ağ aygıtları gibi çeşitli kaynakların gruplara ayrılmasıdır. Her alan için erişim politikaları - kimlik denetim modelleri farklı olabilir.) ve bu bölgeye uygulanabilecek kimlik denetim modelleri ve ilgili parametreler belirtilir. Bu parametrelere karşılık kimlik bilgileri (kullanıcı adı, şifre ...gibi) ilgili UAC tarafından bir araya getirilip (kullanıcıdan istenebilir) aynı istek iletisi içerisinde UAS'ye gönderilir. UAS bu bilgilere göre UAC'nin ilgili bölgeye erişim hakkı olup olmadığını denetler. Sonuç olumlu ise, isteği yerine getirir.
2. *Vekil sunucu ile kullanıcı arası kimlik denetimi:* Bir UAC bir vekil sunucuya istek iletisi gönderdiğinde, iletinin içerisinde Proxy-Authorization başlık alanında gerekli kimlik bilgileri mevcut değilse vekil sunucu ilgili UAC için kimlik denetimi gerçekleştirmek isteyebilir. Bunun için UAC'ye 407 (Çizelge 4.1d.) kodlu yanıt iletisini gönderir. Bu iletinin içerisine WWW-Authentication yerine Proxy-Authenticate başlık alanı yerleştirilir. Kullanıcılar arası kimlik denetiminde olduğu gibi UAC gerekli kimlik bilgilerini bir araya getirip yeniden oluşturduğu istek iletisi içerisinde vekil sunucuya gönderir. Vekil sunucu bu bilgilere göre iletinin hedef vekil sunucuya/UAC'ye iletimine karar verir.

4.2.5.5. SIP protokolünde aygıt taşınırlık desteği

SIP aygıt taşınırlığını iki aşamada desteklemektedir:

1. *Görüşme öncesi taşınırlık (Pre-call mobility)* [38]: Bir taşınır aygıtın kendi yönetsel alanı dışında iken herhangi bir iletişim kurmadan önce edindiği her yeni ağ adresine karşılık kendi yönetsel alanında bulunan kayıt sunucusuna yeniden kaydolması durumudur. REGISTER iletisini kullanarak yeni konum bilgisini yerel kayıt (yönlendirme) sunucusuna bildirir. Şekil 4.30.'da görüldüğü gibi, bir taşınır aygıt çalışmaya başladığı yabancı ağda yeni ağ adresi elde ettiği zaman bu durumu REGISTER iletisi ile kendi yerel kayıt sunucusuna (şekilde aynı zamanda yönlendirme sunucusu) bildirir. Bu işlemden sonra bir ağ aygıtı ilgili taşınır aygıtla görüşme isteminde bulunursa, istek iletisini doğal olarak taşınır aygıtın yerel kayıt sunucusuna gönderecektir. Yerel yönlendirme sunucusu taşınır aygıtın yeni konumunu 302 (Çizelge 4.1d.) kodlu yanıt iletisinin içerisinde istemci aygıta gönderir.

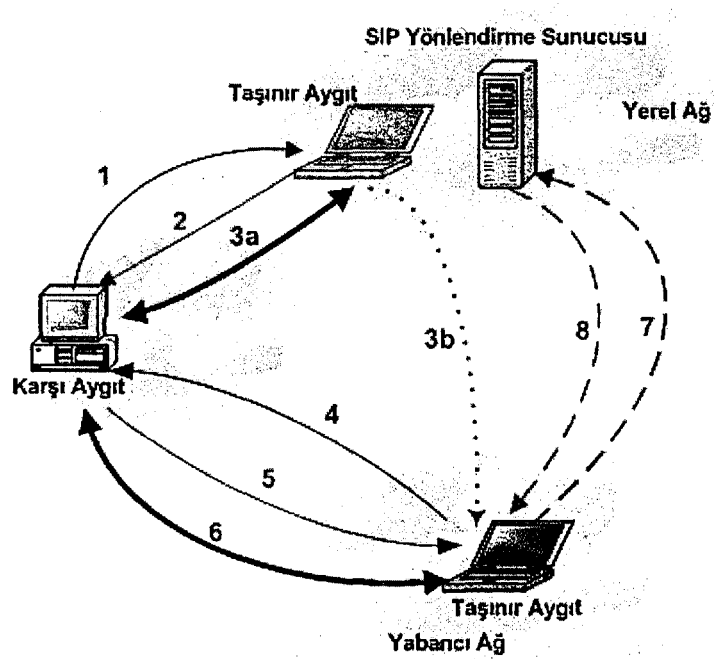


1 : SIP REGISTER iletisi
2, 6 : SIP OK
3, 5 : SIP INVITE iletisi
4 : SIP 302 Geçici olarak taşınmış
7 : Veri

Şekil 4.30. SIP görüşme öncesi konum değiştirme

İstemci aygıt aynı istek iletisinin hedef adres bilgisini bu adres değeri ile güncelleyip ilgili adrese gönderir. İletiyi alan taşınır aygıt Şekil 4.30.'da isteğe olumlu yanıt vermiş ve aralarında veri alışverişi başlamıştır.

2. *Görüşme ortası taşınırlık (Mid-call mobility)* [38]: Bir taşınır ağ aygıtı mevcut bir görüşme devam ederken yabancı bir ağa girip yeni bir ağ adresi (IP ağlarda IP adresi) elde ettiğini anladığında görüşmenin devam edebilmesi için bu durumu iletişim kurduğu aygıta bildirmesi gerekir. Bunu da bir INVITE iletisi oluşturarak gerçekleştirir. Oturum tanımları içerisine yeni ağ adresi yazılarak oluşturulan bu istek iletisi Şekil 4.31.'de de görüldüğü gibi ilgili ağ aygıtına gönderilir. Bu iletiyi alan ağ aygıtı görüşmeyi bu adrese göre sürdürür. Tabi burada gönderilen INVITE iletisinin kimlik denetiminin yapılması da şarttır.



1, 4 : SIP INVITE iletisi
2, 5, 8 : SIP OK
3a, 6 : Veri
3b : Yabancı ağa taşınma
7 : SIP REGISTER iletisi

Şekil 4.31. SIP görüşme ortası konum değiştirme

Çünkü bir davetsiz misafir karşı ağ aygıtına konum değişikliğini bildiren bir istek iletisi göndererek iletişimin yönünü izinsiz değiştirip veri akışını dinleyebilir. Taşınır ağ aygıtı iletişim kurduğu ağ aygıtını bilgilendirdiği gibi kendi yerel yönlendirme sunucusunu da bilgilendirmek zorundadır. Bunun için de görüşme öncesi taşınırlık durumunda izlenen adımları izlemelidir. Yani ilgili yerel yönlendirme sunucusuna bir REGISTER istek iletisi göndererek yeni ağ adresi ile yeniden kaydolmalıdır. Böylece yönlendirme sunucusu kendisine istek iletisi gönderen aygıtları bu yeni konuma yönlendirebilecektir.

4.2.5.6. SIP protokolünde IPv6 desteği

SIP protokolü genel tanımına göre IPv6 adresleri ile de sorunsuz çalışacak şekilde düzenlenmiştir [33]. Buna göre SIP adreslerinde bulunan "@" işaretinden sonra IPv4 adresi ya da alan adı dışında IPv6 adresi de kullanılabilmektedir.

4.2.5.7. SIP protokolü esnek ileti yapısına sahip olmalı

SIP yeni gereksinimleri yeni eklentiler ve düzenlemeler ile kolayca karşılayabilmek için oldukça esnek bir ileti yapısına sahip olarak tasarlanmıştır. Böylece protokolün genel özelliklerine ve işleyişine zarar vermeden kolayca yeni uzantılar tanımlanabilmektedir. Bu yapı sayesinde protokol uyarlayıcıları kendi amaçlarına yönelik olarak yeni başlık alanları tasarlayıp kullanabilirler. Aynı iletişim yolu üzerinde bulunan mevcut SIP elemanları (SIP vekil sunucular gibi) anlamadıkları bu yeni başlık alanlarını göz ardı ederek işletimlerini sorunsuz sürdürebilirler.

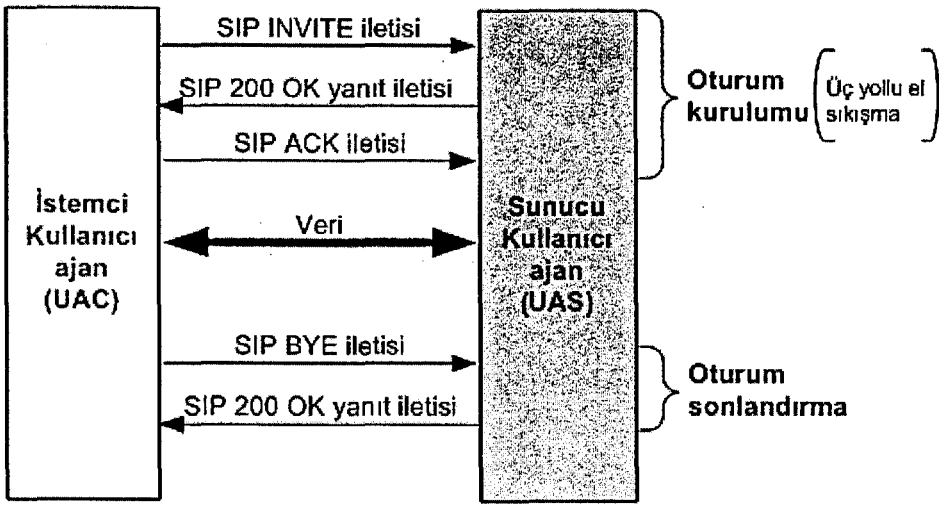
SIP iletileri oldukça esnek veri alanlarına da sahiptirler. Bu alan için bir veri türü kısıtlaması yoktur. Burada aktarılacak veri bir oturum tanımını içerebileceği gibi (SDP aracılığı ile olabilir), örneğin resim formatında (JPEG gibi), anlık mesaj şeklinde ya da herhangi bir MIME (Internet üzerinde elektronik posta aracılığı ile aktarılacak olan bir dosyanın türünü belirlemeye yarayan standart) türünde de olabilir. Aygıt kontrolü bağlamında DMP adı verilen yeni bir MIME türü tanımlanmıştır. XML tabanlı olan ve tasarımı devam eden bu yeni veri alanı türü sayesinde ağ

aygıtlarına, aygıt özelliklerinden bağımsız şekilde komut gönderilebilmekte ve aynı şekilde yanıt alınabilmektedir [39].

4.2.5.8. SIP protokolünün yapısı karmaşık olmamalı

SIP oldukça basit yapıda tasarlanmış metin tabanlı bir uygulama katmanı protokolüdür. Metin tabanlı olması sayesinde aktarılan iletiler kolay oluşturulmakta ve kolay ayrıştırılmaktadır. Ayrıca iletilerin getirdiği ek işlem ve iletişim yükü de bu sayede düşük olmaktadır. Protokolün yapısının basit olmasını sağlayan en önemli özelliklerden birisi de genel tanımında sadece 6 temel ileti yöntemi türünün bulunmasıdır [32]. Bu sayede protokolün kullanım ve gerçekleştirim karmaşıklığı düşmektedir. Bu temel iletilerin yanında genelde ağ aygıt kontrolüne yönelik olarak yeni (DO, MESSAGE, SUBSCRIBE, NOTIFY ...gibi) ileti yöntemleri tanımlanmış olsa da, bu durum protokolün yapısını karmaşıklaştırmamakta, aksine aygıt kontrolü konusunda kolaylık sağlanmaktadır. Bununla birlikte özellikle DO yöntem tanımı ile birlikte ortaya çıkan ve halen geliştirilmekte olan XML tabanlı bir veri tanımlama protokolü sayesinde (DMP) iletilerin veri alanları metin tabanlı olarak daha sade ve aygıttan bağımsız şekilde oluşturulabilecektir. Bu da veri alanlarının oluşturulması ve işlenmesi için gereken ek iş yükünü daha da azaltacaktır.

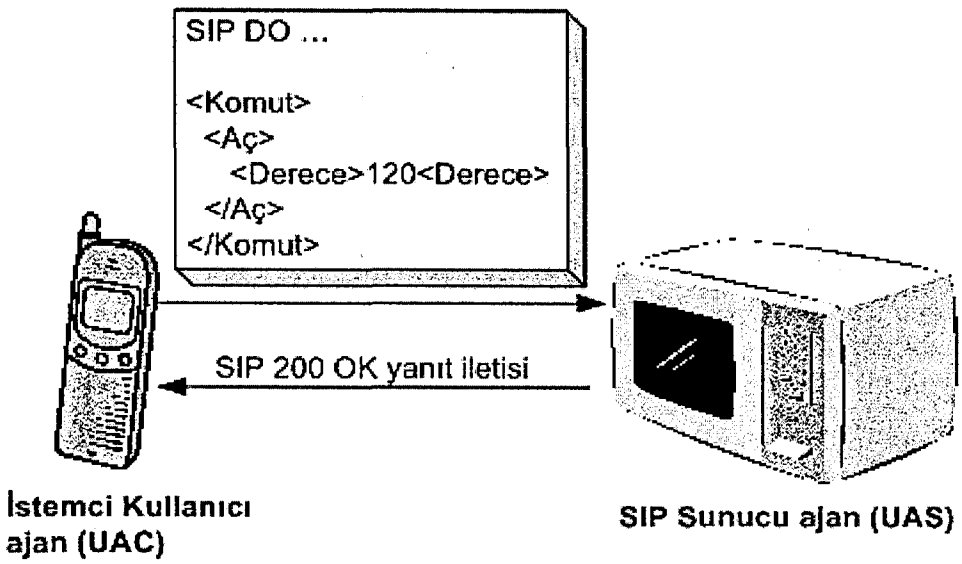
Bununla birlikte SIP iletişimi oldukça sade bir mantık çerçevesinde gerçekleşmektedir. Bir SIP UA başka bir SIP UA ile oturum başlatmak istediğinde bu işlemi **üç yollu el sıkışma (three-way handshake)** mantığı ile kolayca gerçekleştirebilmektedir. Buna göre bir "INVITE" istek iletisi ile oturum isteğinde bulunulduktan sonra karşıdan "200 OK" (Çizelge 4.1b.) olumlu yanıt iletisi beklenir. Bu iletiye karşılık gönderilen "ACK" iletisi ise yanıtın alındığını bildirir (Şekil 4.32.). Aynı şekilde oturum sonlandırma işlemi de oldukça basittir. Karşı SIP ajanına gönderilen bir "BYE" istek iletisine karşılık "200 OK" (Çizelge 4.1b.) yanıt iletisi sonucunda oturuma son verilir.



Şekil 4.32. SIP oturum başlatma ve sonlandırma örneği

Aygıt kontrolü bağlamında, aygıtlara kolayca erişip, bazı işlemleri yaptırabilmek için tasarlanmış olan "DO" iletisi oturum açma ve sonlandırma işlemlerine gerek duymadan kullanıldığından yeterlilikleri ve işlem kapasiteleri düşük olan ağ aygıtları tarafından bile rahatlıkla kullanılabilmektedir. Veri alanında tanımlanmış metin tabanlı komut ve parametre değerlerine göre ilgili aygıttan belli bir işlemi yapması istenir. Aygıt bu bilgileri kullanarak gerekli işlemi yerine getirip, gerekiyorsa sonuçları veri alanı içerisinde "200 OK" (Çizelge 4.1b.) iletisi ile istemciye gönderir. Şekil 4.33.'deki örnekte görüldüğü gibi ağa bağlı bir mikrodalga fırından SIP "DO" iletisi kullanılarak 120 derece sıcaklıkta çalışmaya başlaması istenmektedir.

İleti içerisinde "..." işaretinden sonra gelen bölüm veri alanıdır ve halen geliştirilmekte olan DMP protokolünü örneklemek amacı ile XML tabanlı oluşturulmuştur.



Şekil 4.33. SIP oturum başlatma ve sonlandırma örneği

4.2.6. SIP protokolünde işlevsel gereksinimler

Bu bölümde SIP 3. bölümde belirlenen işlevsel gereksinimlere göre incelenmektedir.

4.2.6.1. SIP protokolünde aygıt denetimi

SIP'i aygıt denetiminde kullanabilmek için "DO" olarak adlandırılan bir yöntem tanımı yapılmıştır [31]. Buna göre oturum başlatmaya gerek duymadan uzaktaki ağ aygıtlarına komutlar göndererek istenilen işlemleri yerine getirmesi sağlanabilmektedir.

"DO" yöntemi ile oluşturulan bir istek iletisinin veri alanı SDP dahil herhangi bir türde veri taşıyabilir. Bu sayede belli bir aygıt türüne yönelik olarak yeni MIME türleri kolayca tanımlanıp kullanılabilir. Bununla birlikte, veri alanı oluşturmak üzere XML tabanlı yeni bir MIME türü tanımı da yapılmıştır [34]. Halen üzerinde çalışılmakta olan ve DMP (Device Messaging Protocol) olarak adlandırılan bu yeni türün aygıt denetimini daha da kolaylaştıracağı düşünülmektedir. Çünkü XML'in ortamdan bağımsızlık özelliği kullanılarak farklı sınıflardaki aygıtlara özgü veri alanları oluşturulabilecektir. Gönderilen "DO" istek iletilerine karşılık ağ aygıtlarının

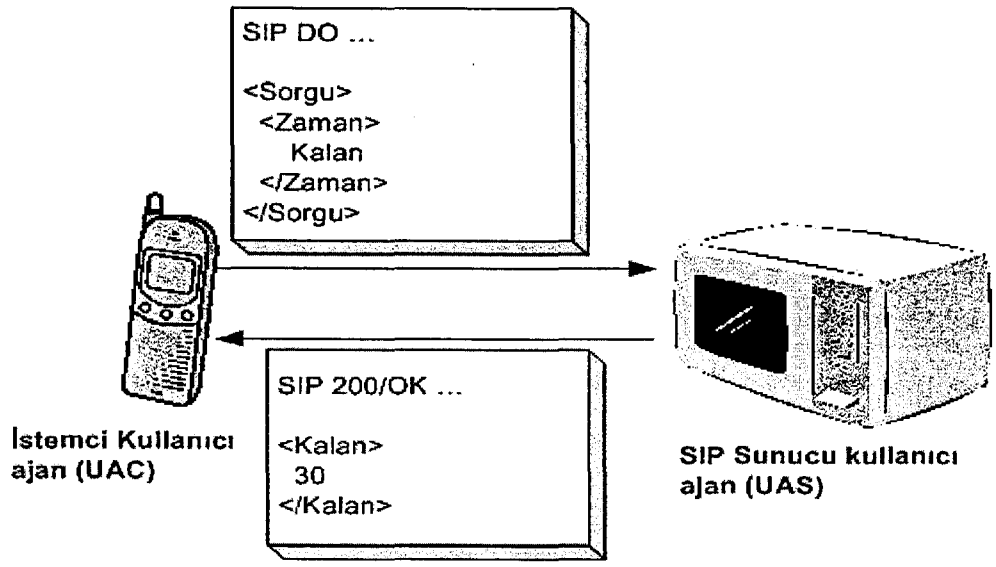
vereceği yanıtlar SIP'in standart yanıt mekanizmasını kullanarak sağlanacaktır. Şekil 4.33.'de SIP ile aygıt denetimine bir örnek verilmektedir.

SIP'in standart tanımında yer alan "INVITE" yöntemi de gerektiğinde aygıt denetimi amacı ile kullanılabilir. "INVITE" yöntemi bir oturum başlatmak için kullanıldığından ağ aygıtları arasında bağlantı gerektirecek uzun süreli işlemler için kullanılabilir. Örneğin bir ağ kamerasından uzaktaki bir kişisel bilgisayara görüntü aktarması istenecekse, bir "INVITE" iletisi ile görüntü akışı için bir oturum oluşturulabilir. Ayrıca görüntü iletimi ve kalitesi ile ilgili belirli parametre değerleri SDP türünde oluşturulup kameraya gönderilebilir. Görüntü akışı sonlandırılmak istendiğinde ise Şekil 4.32.'de de görüldüğü gibi SIP'in standart oturum sonlandırma mekanizması kullanılarak kamera ile bağlantı kesilir.

4.2.6.2. SIP protokolünde durum denetimi

"DO" yöntemi kullanılarak ağ aygıtlarının anlık durumları ile ilgili bilgi sahibi olunabilir [31]. "DO" iletilerinin veri alanları oldukça esnek yapıda olduğundan Şekil 4.34.'dekine benzer bir formatta istek iletisi oluşturularak ağ aygıtından çalışma durumu ya da sahip olduğu parametre değerleri hakkında bilgi istenebilir. Bu bilgi SIP'in standart yanıt mekanizması ile istemciye iletilir.

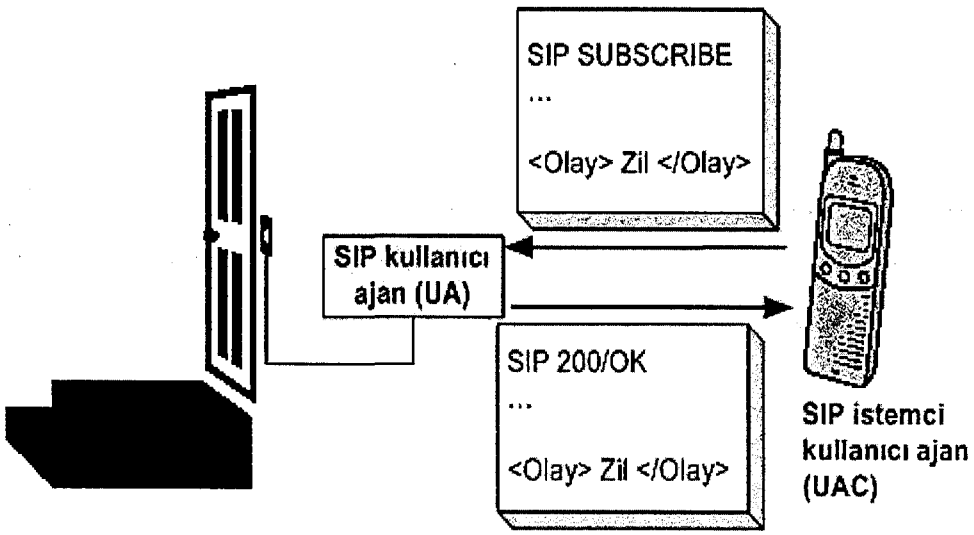
"DO" yöntemi dışında, SIP'in standart tanımında yer alan "OPTIONS" yöntemi ile de bir ağ aygıtının genel kapasitesi ve yetenekleri hakkında bilgi talebinde bulunulabilir. Bu iletiye karşılık bir SIP "200 OK" (Çizelge 4.1b.) yanıt iletisi oluşturularak aygıt bilgileri istemciye gönderilir. Bilgiler yanıt iletisi içerisindeki bazı başlık alanları içerisinde ("*Supported:*" başlık alanı gibi) ve veri alanı (XML tabanlı olabilir) içerisinde tutulurlar.



Şekil 4.34. SIP durum denetimi örneği

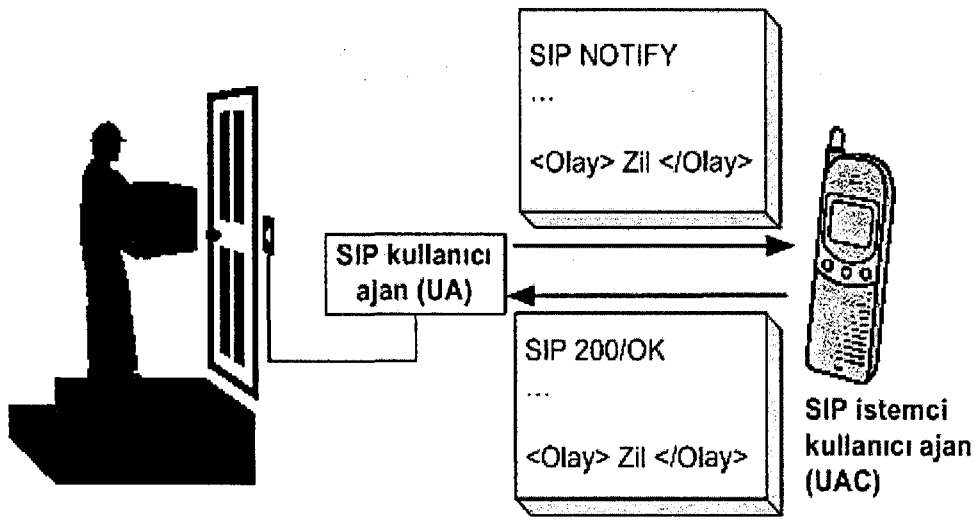
4.2.6.3. SIP protokolünde olay denetimi

SIP, *"Session Initiation Protocol (SIP) – Specific Event Notification"* isimli uzantı sayesinde olay bildirimi işlemlerini de destekler hale gelmiştir [32]. Ağ aygıt kontrolünün çok önemli bir parçası olan olay denetimi "SUBSCRIBE" ve "NOTIFY" ileti yöntemleri sayesinde gerçekleştirilmeye başlanmıştır. SUBSCRIBE iletileri aracılığı ile aygıtlar ya da kullanıcılar bir ağ aygıtında meydana gelebilecek belirli olaylara karşı bildirim almak amacı ile o aygıta üyelik isteminde bulunurlar. Buna karşılık ağ aygıtı ilgili olay meydana geldiği zaman bir "NOTIFY" iletisi oluşturup istemci SIP ajanına göndererek, olayın gerçekleştiğini (gerekirse sonuç bilgileri ile birlikte) bildirmiş olur. Buna örnek olarak Şekil 4.35a'da kullanıcı cep telefonundaki SIP UA aracılığı ile evindeki sokak kapısının çalması olayının kendisine bildirilmesini kapı ziline bağlı SIP UA'dan istemektedir. Bunun için bir "SUBSCRIBE" iletisi hazırlayarak ilgili UA'ya göndermektedir. Kapı zilinin bağlı olduğu UA bu olay üyeliğini kaydedip istemci ajana SIP "200 OK" (Çizelge 4.1b.) yanıt iletisini gönderir.



Şekil 4.35a. SIP olay üyeliği başvurusu örneği

Herhangi bir anda kapı zili çaldığında, Şekil 4.35b.'de görüldüğü gibi bu olay bildirimine üye yaptığı SIP istemci kullanıcı ajanına (cep telefonu) "NOTIFY" iletisi ile olayı haber verir.

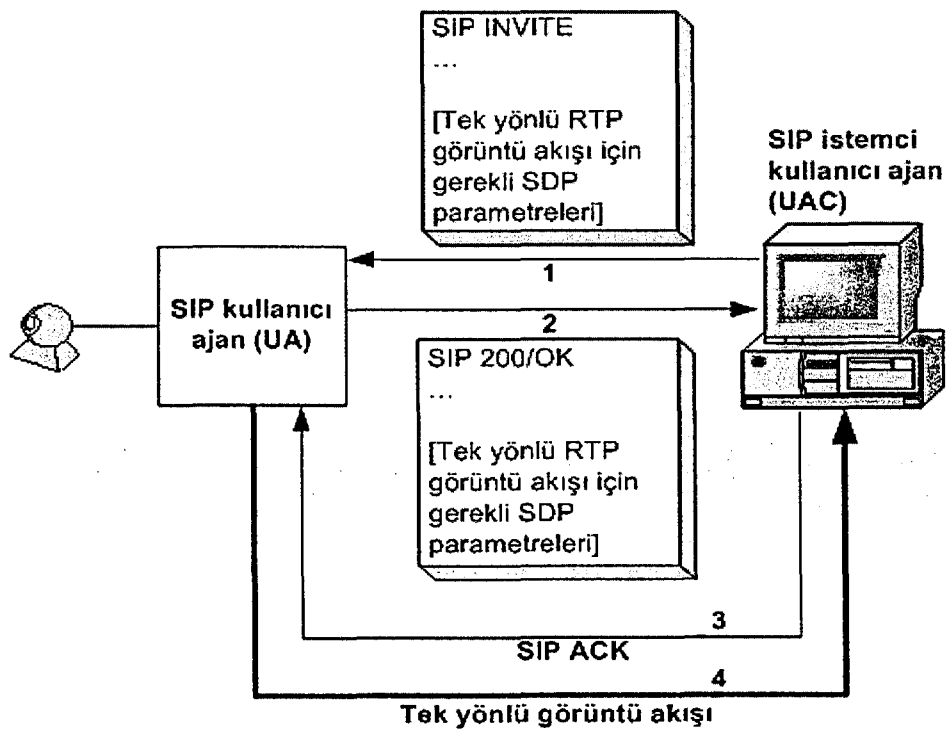


Şekil 4.35b. SIP olay bildirim örneği

4.2.6.4. SIP protokolünde oturum denetimi

SIP ağ aygıtları arasında sürekliliği olan, yani oturum gerektiren işlemlerin gerçekleştirilmesini "INVITE" yöntemi ile sağlar. Sürekliliği olan işlemler genelde veri akışı (ses, görüntü ...vb) ile ilgili olduğundan veri akışı için gerekli parametre değerleri "INVITE" iletisinin veri alanına (genelde SDP kullanılarak) yerleştirilerek sunucu ağ aygıtına gönderilir. Bu ileti ile oturum açmaya davet edilen ağ aygıtı gönderilen parametre değerlerini inceleyip uygun olmayanları kendisine göre değiştirip ya da tamamını kabul edip istemci aygıta bir yanıt iletisi ile gönderebilir. İki taraf oturum parametrelerinde anlaşırlarsa veri akışı genelde **RTP (Real-Time Transport Protocol)** protokolü kullanılarak başlatılır ve bu işlem oturumu bitirmek için kullanılan "BYE" iletileri yardımı ile sonlandırılır.

Oturum denetimine örnek olarak Şekil 4.36.'daki senaryo verilebilir. Buna göre, uzaktaki bir kişisel bilgisayardan evinin salonundaki güvenlik kamerasına erişip, ondan görüntü akışı yapmasını isteyen bir kullanıcı bu isteği bir "INVITE" iletisi ile kameranın kullanıcı ajanına bildirir. UA kendisine gelen parametrelerin uygunluklarını değerlendirir.



Şekil 4.36. SIP oturum başlatma örneği

Eğer veri akışı için bir problem yoksa UA istemciye SIP "200 OK" (Çizelge 4.1b.) iletisi ile kabul ettiğini bildirir. İstemci ise yanıtın kendisine ulaştığını ve akışın başlamasında sakınca olmadığını bildirmek için bir "ACK" iletisi oluşturup kameranın kullanıcı ajanına gönderir. Böylece oturum açılmış olur ve üzerinde anlaşılan RTP parametreleri çerçevesinde tek yönlü görüntü akışı başlatılır.

İstemci bir süre sonra görüntü akışının yeterli olduğunu düşünerek oturumu, dolayısıyla görüntü akışını sonlandırmak isteyebilir. Bunu da bir "BYE" iletisi oluşturarak kameranın kullanıcı ajanına bildirir. Buna karşılık istemciye gönderilen SIP "200 OK" yanıt iletisi ile oturum sonlandırılır.

4.3. SMTP (Simple Mail Transfer Protocol)

Bu bölümde SMTP'nin tarihçesi, yapısı ve işleyişi hakkında bilgiler verilecektir. Daha sonra da bu protokolün tanımlanmış olan temel ve işlevsel gereksinimlere uygunluğu incelenecektir.

4.3.1. SMTP'nin tarihçesi ve yapısı

SMTP posta iletilerini (elektronik posta) Internet ortamında taşımak amacı ile kullanılan bir protokoldür. Uygulama katmanında çalışan SMTP iki sistem arasında ileti aktarımı için güvenilir bir aktarım ortamını tercih eder. Bu ortamı sağlayan aktarım katmanı protokollerinden SMTP gerçekleştirimlerinde en çok kullanılanı TCP'dir. İletişim TCP'nin 25. portunu kullanarak gerçekleştirilir.

İlk olarak 1982 yılında IETF tarafından yayınlanan [40] ve standart (STD 10) haline gelen SMTP temelde bilgisayarlar arası metin tabanlı ileti (posta) aktarımı için tasarlanmış bir protokoldü. Internet posta sisteminin en önemli parçası olan SMTP protokolü ilk tanımlandığı dönemde bazı kısıtlamalar içermekteydi. Örneğin SMTP iletilerinin gövde bölümü sadece 7 ikil ASCII kodlardan oluşmalıydı [41]. Zamanın iletişim imkanlarına ve ihtiyaçlarına göre oldukça anlamlı olan bu kısıtlama, günümüzde iletişim ve çoklu ortam teknolojilerinin gelişmesi ile birlikte büyük bir engel halini almıştır. Bu kısıtlama yüzünden ASCII metinleri dışında örneğin farklı dillerdeki metinleri veya farklı çoklu ortam verilerini aktarmak gerektiğinde içeriğin önce ASCII metin türüne dönüştürülmesi gerekmektedir. Bu işlemin tersi de alıcı

tarafta gerçekleştirilmek durumundadır. Bunun yanında alıcı tarafın veri içeriği ve türü hakkında bilgi sahibi olabilmesi için SMTP iletilerinin sahip olduğu başlık alanlarına ek olarak yeni bazı alanların tanımlı olması gerekmektedir. RFC 2045 ve RFC 2046'da tanımlanmış olan bu alanlar **MIME** [42] adıyla Internet ileti yapısının tanımlandığı RFC 822'ye uzantı olarak eklendi.

Aktarılan veri içeriği dışında zaman içerisinde SMTP'nin kimlik denetimi, güvenlik gibi önemli alanlarda eksikliklerinin olduğu görüldü. Eskisini değiştirip statik bir yeni sürüm yaratmak yerine **ESMTP (Extended SMTP)** adıyla, uzantıların kolayca oluşturulup temel protokol yapısına entegre olabilmesini sağlayacak bir mekanizmanın hazırlanması yoluna gidilmiştir. Son hali RFC 1869'da *"SMTP Service Extensions"* adıyla yayınlanan bu mekanizma sayesinde yeni uzantılar belli adımlar izlenerek tutarlı bir şekilde tanımlanabilmektedir.

SMTP'nin kullanıldığı posta iletim sistemi üç temel bileşene sahiptir:

- ***Posta Aktarıcı Ajan (MTA – Mail Transport Agent)***: Elektronik posta iletilerinin göndericiden alıcıya teslim edilmesinden sorumlu uygulamalardır. Kendilerine gelen iletilerin alıcı adreslerini incelerler. Yerel adrese sahip alıcılar için iletiyi saklarlar, yabancı adrese sahip alıcılara ulaşması için ise iletiyi yol üzerinde bulunan bir sonraki MTA'ya yönlendirirler. Bu işlemleri yapabilmek için ise iletinin başlık alanlarını düzenlemek ve hatta yeni başlık alanları ekleme hakkına sahiptirler.
- ***Posta Kullanıcı Ajan (MUA – Mail User Agent)***: Kullanıcıların elektronik posta iletileri hazırlaması ve okuması için tasarlanan uygulamalardır. MTA ile kullanıcı arasında arayüz sağlarlar. Böylece kullanıcı ileti oluşturduğunda bu iletinin MTA tarafından gönderilmesini, MDA tarafından alınmış olan iletilerin de kullanıcı tarafından okunabilmesini sağlarlar.
- ***Posta Teslimat Ajanı (MDA – Mail Delivery Agent)***: Bağlı olduğu MTA'dan kullanıcı için posta alımını gerçekleştiren uygulamadır. Eğer bir MUA ve bağlı olduğu MTA aynı bilgisayarda çalışıyorsa, MDA'ya gerek kalmaz. Ayrıca SMTP ile ilgili tanımlarda genel olarak bir MUA hem MDA

hem de MUA'nın işlevlerini içeren bir SMTP bileşeni olarak düşünülmektedir.

Elektronik posta iletilerinin göndericisinin ve alıcısının belirlenmesinde e-posta adreslerinden yararlanılır. Bu adresler aşağıda görüldüğü gibi kullanıcı ve alan bilgilerini taşıyan karakter katarlarından oluşur:

<kullanıcı>@<alan>

Burada kullanıcı bölümü ilgili kullanıcıya verilmiş olan kullanıcı adını, alan bölümü ise kullanıcının tanımlı olduğu SMTP sunucusunun alan adını (Alan adı ya da IP adresi) temsil etmektedir (Örn: oguner@anadolu.edu.tr).

4.3.2. SMTP komutları

SMTP'nin bir elektronik posta iletisinin aktarımını gerçekleştirmek üzere gerekli işlem adımlarını yerine getirebilmek için kullandığı temel komutlar söz dizimleri ve tanımları ile aşağıda sıralanmıştır (Komutların yazılışında her türlü büyük/küçük harf kombinasyonu kullanılabilir. Örn: Helo, HeLo ...gibi) :

HELO *gönderen_makine_adı*

SMTP iletişimi bu komut aracılığı ile başlatılır. SMTP sunucusu (MTA) ile bağlantı kurmuş olan bir istemci makine bu komutu kullanarak sahip olduğu makine adıyla (Örn: posta.ceng.anadolu.edu.tr) kendisini sunucuya tanıtır. Sunucu da vereceği olumlu yanıt iletisi ile kendisini istemciye tanıtmış olur.

EHLO *gönderen_makine_adı*

HELO ile aynı işlevi gören bir başka komuttur. Bu komutun kullanılmasının amacı sunucuya ESMTP protokolünün kullanılmak istendiğinin istemci tarafından bildirilmek istenmesidir. Sunucu bu komuta karşılık olarak eğer ESMTP'yi destekliyorsa kullanımına izin verdiği komutların listesini istemciye gönderir.

MAIL From:*<gönderen_e-posta_adresi>*

Elektronik posta iletisini başlatmak için kullanılır. *gönderen_e-posta-adresi* parametre değeri sonuçta oluşacak posta iletisinin *"From:"* başlık alanında tutulacaktır.

RCPT To:<*alıcı_e-posta_adresi*>

Bu komut ile alıcının adresi belirlenir. Bu ileti ardı ardına birçok kere kullanılarak posta iletisinin birden fazla alıcıya ulaşması sağlanabilir.

SIZE=*bayt_sayısı*

Bu komut alıcı sisteme aktarılacak olan iletinin bayt cinsinden boyutunu bildirmek amacı ile kullanılır. Bu işlem atlanırsa, e-posta okuyucuları ve aktarıcı ajanlar iletinin uzunluğunu iletiyi sonlandırmak için kullanılan "." içeren satır ve başlık alanları ile gövde metnini ayırmaya yarayan boş satır gibi belirteçler yardımıyla belirlemeye çalışırlar.

DATA

Bu komut kendisinden sonra bir veri akışının başlayacağını bildirir. Bu veri akışı tek başına "." içeren bir satırın gönderilmesi ile sonlandırılır.

QUIT

Quit komutu SMTP iletişimi için kurulan bir TCP/IP bağlantısını sonlandırır. E-posta iletimini daha etkin hale getirebilmek için tek bir bağlantıda birden fazla e-posta iletisi aktarılabilir. Aktarılmak istenen her bir posta iletisi için bir başka *"MAIL"* komutu kullanılır.

VRFY *kullanıcı_adı*

Bu komut SMTP sunucusundan belirtilen e-posta kullanıcı adına sahip bir kullanıcının olup olmadığını öğrenmek amacı ile kullanılır. SMTP sunucu yanıt iletisinde ilgili kullanıcının oturum açma adını göndererek ilgili kullanıcı adını onayladığını belirtir. Bu özellik güvenlik boşluğu yarattığı için devre dışı bırakılabilir.

EXPN *liste_adı*

Bu komut SMTP sunucusuna *"liste_adı"* argümanının değerinin bir posta listesi adı olup olmadığını sormak amacı ile kullanılır. Eğer böyle bir liste mevcutsa, sunucu listenin tüm üyeliklerini isimleri ve e-posta adresleri ile birlikte istemciye listeler.

HELP

Sunucu tarafından kullanımı desteklenen SMTP komutları hakkında bilgi almak için kullanılan komuttur.

NOOP

Bu komut kendisinden önce kullanılmış komutları ya da sürmekte olan işlemleri etkilemez. Sadece bağlantının sürdüğünü ve karşı birimin çalıştığını anlayabilmek için kullanılır. Bu komuta karşı birim bir olumlu yanıt iletisi ile karşılık verir.

SMTP'nin yukarıdaki komutlara karşı oluşturabileceği standart yanıt ileti kodları ve tanımları Çizelge 4.2'de sıralanmıştır.

Çizelge 4.2. SMTP standart yanıt ileti kodları ve tanımları

Yanıt Kodları	**Tanımları**
211	Sistem durum ya da sistem yardım yanıtı
214	Yardım iletisi
220	(alan) Servis hazır
221	(alan) Servis aktarım kanalını kapatıyor
250	İstenen posta işlemi tamam, gerçekleştirildi.
251	Kullanıcı yerel değil; iletilecek
354	Posta girişini başlat; (CRLF).(CRLF) ile bitir.
421	(alan) Servis uygun değil
450	İstenen posta işlemi gerçekleşmedi: posta kutusu kullanım dışı
451	İstenen işlem durduruldu: İşlem sırasında yerel hata
452	İstenen işlem gerçekleşmedi: yetersiz sistem belleği
500	Söz dizim hatası, komut tanımlanamadı
501	Parametre ya da argümanlarda söz dizim hatası
502	Komut uygulanamadı
503	Yanlış komut sırası
504	Komut parametresi uygulanamadı
550	İstenen işlem gerçekleştirilemedi: Posta kutusu kullanım dışı
551	Kullanıcı yerel değil; lütfen "iletim-yolu" 'nu deneyin
552	İstenen posta işlemi durduruldu: bellek taşması
553	İstenen işlem gerçekleştirilemedi: posta kutusu ismine izin verilmedi
554	Hareket başarısız

4.3.3. SMTP ileti formatı

Göndericiden alıcıya aktarılan elektronik posta iletisi Şekil 4.37'de görüldüğü gibi bir **zarf (envelope)** bir de **içerik (contents)** bölümünden oluşur.

Zarf bölümü iletinin aktarımı sırasında kullanılan ya da üretilen alıcının ve aktarıcının e-posta adresleri gibi aktarım için gerekli bilgileri içerir. Bu bölüm ileti aktarım ajanları (MTA) tarafından iletiyi yönlendirmek amacı ile kullanılır.

İçerik bölümü ise **başlık** ve **gövde** bölümlerinden oluşur.

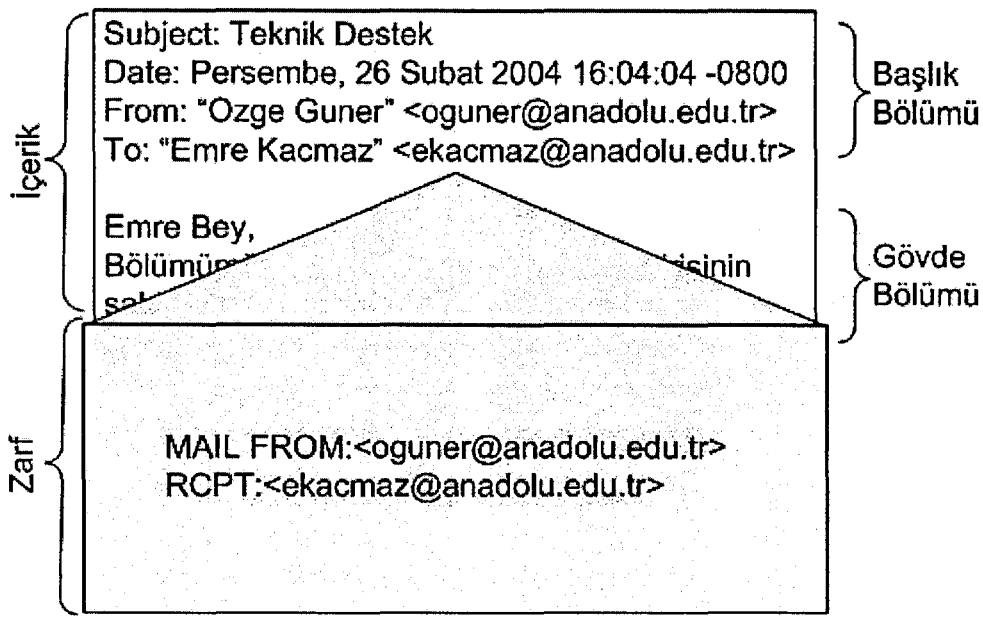
- ***Başlık*** bölümü iletiyi alan MTA'ların hedefine ulaştırabilmek için nasıl yönlendirmeleri gerektiği ve iletinin alıcısı olan MUA'nın iletiyi nasıl okuması gerektiği hakkında denetim bilgileri içerir. Bu bilgiler ileti içerisinde genelde tek satır uzunluğunda alan/değer çiftleri şeklinde tutulur. Başlık alanlarının en önemlileri şunlardır:

  Received: - İletinin geçtiği makinelerin her biri ile ilgili bilgi tutar. Bu bilgiler kullanılarak göndericiye yönelik olarak iletinin aktarımı ile ilgili durum bilgisi iletilebilir.

  From: - Elektronik postayı yazan kullanıcının e-posta adresini tutar.

  To:, Cc:, Bcc: - Alıcıları belirler. Bu alandaki bilgiler alıcılar ya da posta programları tarafından okunan bilgiler içerirler. Bu bilgiler içerisinde uygun olmayan e-posta adresleri de bulunabilir.

  Message-ID: - Mevcut iletiyi ayırt etmek için kullanılan kimlik kodudur. Bu kod "*Reply-To*" alanında da kullanılarak farklı yollardan alıcısına ulaşan aynı iletinin diğer kopyaları göz ardı edilmiş olur.



Şekil 4.37. SMTP posta ileti yapısı

Reply-To: - Bir iletiye karşılık üretilen yanıt iletisinin *"From"* alanındaki değil başka bir kullanıcıya gönderilmesi gerektiğini belirtir.

In-Reply-To: - Orijinal ileti ile ona gönderilen yanıtların eş güdümünü sağlar. Yani yanıt iletilerinin hangi posta iletisine karşı gönderildiğini belirler.

Followup-To: - *"Reply-To"* alanı ile aynı işleve sahip olan fakat grup yanıtları için kullanılan başlık alanıdır.

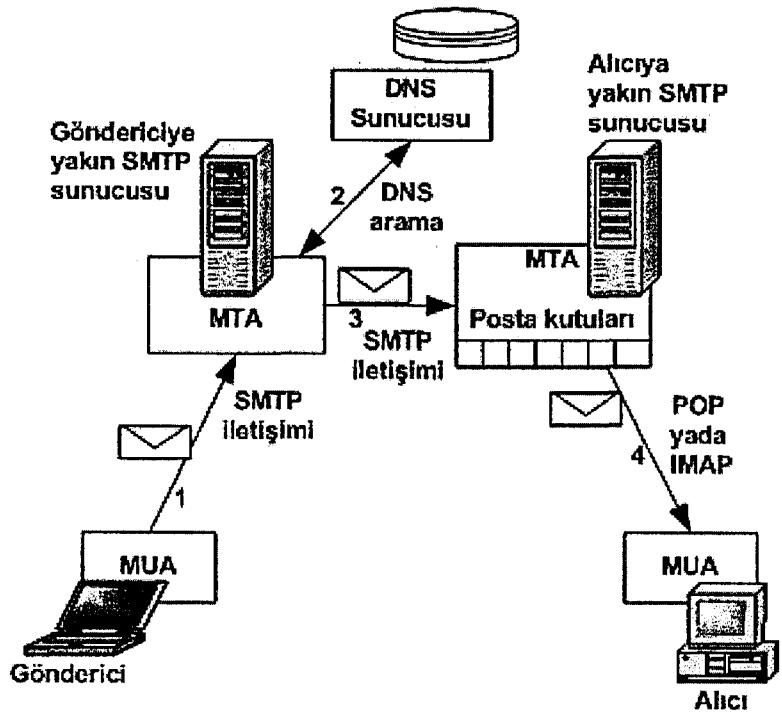
References: - *"In-Reply-To"* ile aynı işleve sahip olup esasen Usenet haber gruplarına atılan iletiler arasında referansları belirtmek için kullanılan başlık alanıdır.

- ***Gövde*** bölümü ise düz (ASCII) metin başta olmak üzere çeşitli çoklu ortam verilerini (resim, ses, video ...gibi) ve/veya dosya eklerini de içerebilir.

4.3.4. SMTP'nin çalışma mantığı

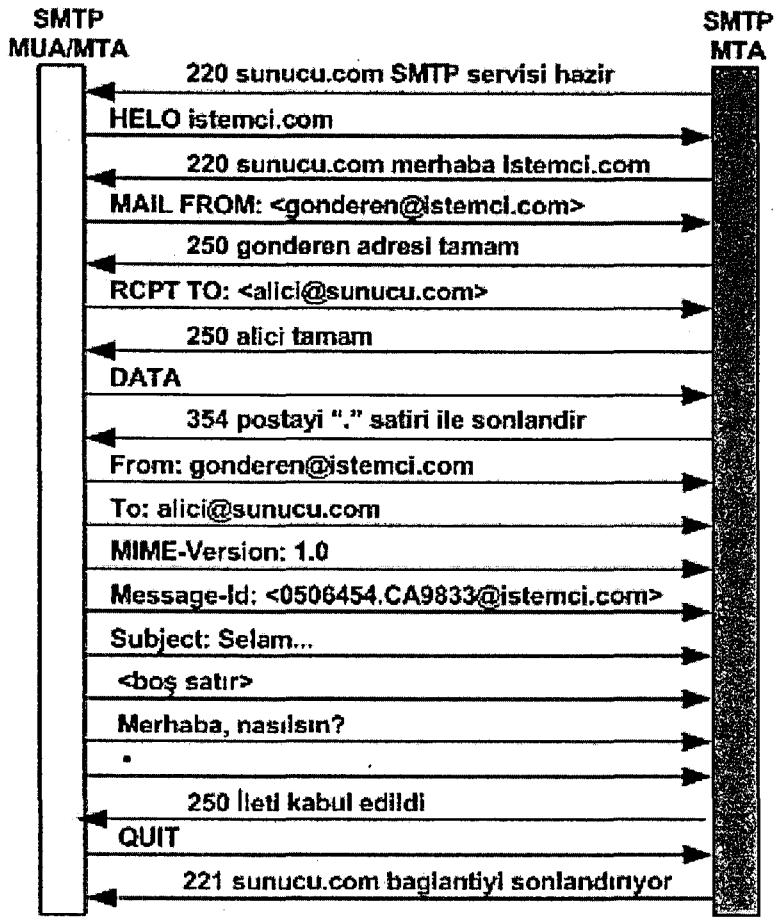
SMTP'nin aktarım protokolü olarak kullanıldığı elektronik posta sistemlerinde iletişim **biriktir ve gönder (store-and-forward)** mantığına göre sağlanır. Yani posta

iletisinin tamamı elde edilmeden bir sonraki birime gönderilmeye başlanmaz. Şekil 4.38'de görüldüğü gibi, bir kullanıcı gönderici bilgisayardaki MUA'yı kullanarak başka bir kullanıcıya elektronik posta iletisi göndermek istemektedir. Bunun için ilk önce kendisine en yakın MTA ile TCP bağlantısı kurar (1). Bu bağlantı üzerinden MTA ile SMTP komutları ile iletişim kurarak posta iletisinin aktarım hazırlığını (gönderen ve alıcı adreslerinin belirlenmesi, kimlik denetimi, güvenlik işlemleri...gibi) gerçekleştiren MUA, posta iletisini MTA'ya gönderdikten sonra bağlantıyı "QUIT" komutu ile sonlandırır. Posta iletisi MTA tarafından alıcıya yakın (alıcının posta kutusuna sahip) olan MTA'ya gönderilmek üzere **aktarım alanı (spool area)** olarak adlandırılan özel bir bellek alanında tutulur. Bu sırada MTA alıcının e-posta adresinin bir parçası olan alan adını DNS sunucudan sorgular (2) (DNS sunucu ayrı bir aygıt olabileceği gibi MTA ile birlikte aynı aygıtta da çalışıyor olabilir). DNS sunucunun posta sunucuları için tuttuğu **MX (Mail Exchange)** kayıtlarından ilgili alan adına karşılık gelen DNS adresi elde edilir. MTA bulunan DNS adresine sahip diğer MTA'ya (alıcıya yakın MTA) TCP bağlantısı kurar (3). Yine SMTP komutları aracılığı ile posta hareketine başlamak için gerekli hazırlıklar yapıldıktan sonra posta iletisi aktarılır. Bu işlemden sonra iki MTA arasındaki TCP bağlantısının sonlandırılması ile birlikte SMTP'nin de görevi sona erer. Posta iletisini alan MTA iletinin alıcı adresini kontrol eder ve iletiyi sahibinin posta kutusunda (her kullanıcı için tutulan bellek alanında) saklar. SMTP'nin Şekil 4.38'deki son adımda (4) kullanılmayışının sebebi alıcı adresinin sahibi olan kullanıcının konumunun sabit olmayışı nedeniyle SMTP tasarımında gönderici denetiminde aktarım mantığının benimsenmiş olmasıdır. Bu yüzden kullanıcı kendisine gelen elektronik posta iletilerini okuyabilmek için başka bir protokol yardımı ile iletileri posta kutusunun bulunduğu MTA'dan kendi bilgisayarına taşıması gerekmektedir. Şekilde bu iş için **POP (Post Office Protocol)** [43] ya da **IMAP (Internet Mail Access Protocol)** [44] önerilmiştir.



Şekil 4.38. SMTP ile elektronik posta iletimi

Şekil 4.38'de görülen tüm bağlantılar (4. adım dışında) üzerinde gerçekleşecek SMTP iletişim işlemleri birbirinin aynıdır. Bu yüzden Şekil 4.39'da bu adımlardan bir tanesi (3. adım) üzerinde gerçekleşen SMTP iletişimi örneklenmektedir. Bu adımda göndericinin yakın olduğu MTA alıcının yakın olduğu MTA'ya bağlantı kurup elindeki posta iletisini göndermek durumundadır. Buna göre iki MTA arasında kurulan TCP bağlantısı üzerinden Şekil 4.39'daki gibi bir diyaloğun gerçekleşmesi ile posta iletisinin aktarımı sağlanmış olur.



Şekil 4.39. Ajanlar arası SMTP iletişim örneği

4.3.5. SMTP protokolünde temel gereksinimler

Bu bölümde SMTP 3. bölümde tanımlanan temel gereksinimlere göre incelenmektedir.

4.3.5.1. SMTP protokolünde geniş alan iletişim desteği

SMTP protokolü aktarım sistemlerinden bağımsız tasarlanmış bir uygulama katmanı protokolüdür. Ancak aktarım için kullanılacak olan protokolün güvenilir iletişim sağlaması şarttır [42]. Aktarım protokolü olarak TCP tercih edilse de, **NCP (Network Control Program)** aktarım servisi [45], **NITS (Network Independent Transport Service)** [48] ya da X.25 aktarım servisi [46] de kullanılabilmektedir [40].

SMTP protokolünün en önemli özelliklerinden bir tanesi de posta olarak adlandırılan iletilerini farklı ağlar arasında nakledebilmesidir. Genellikle SMTP **posta nakli (mail relaying)** olarak adlandırılan bu işlem sayesinde bir posta iletisi aynı ağ üzerindeki bir hedef birime iletilebileceği gibi, farklı ağ ortamında bulunan bir birime de nakil ve ağ geçidi işlemleri aracılığı ile kolayca ulaştırılabilmektedir. Bunun için posta iletisinin kaynak birimden hedef birime ulaşana kadar çeşitli **nakil** (kendisine gelen bir iletiyi bir MUA gibi çalışıp hedefe yönlendirme) ya da **ağ geçidi** (kendisine gelen iletileri SMTP'den farklı bir protokol ile başka bir konuma aktarma) görevi olan birimler üzerinden geçmesi gerekmektedir. Burada hedefe ulaşmak için gidilen yol üzerinde bir sonraki adımı belirleyebilmek için ise DNS'in **Mail Exchange** (MX) olarak adlandırılan mekanizması kullanılır [47,49].

SMTP protokolü, aktarımdan bağımsız (güvenilir olması şartı ile) çalışabilmesi ve posta nakli mekanizması ile iletilerinin farklı ağ ortamları üzerinde adım adım iletilebilmesi sayesinde geniş alan iletişim desteği sağlayan bir uygulama katmanı protokolüdür.

4.3.5.2. SMTP protokolünde güvenilir iletişim desteği

SMTP protokolünün en önemli kısıtlarından birisi sadece güvenilir aktarım protokollerini desteklemesidir. Ancak bu durum iletişim güvenilirliği söz konusu olduğunda avantaj haline gelir. Kullanım izni verdiği aktarım protokolleri SMTP denetim ve posta iletilerinin aktarım güvenilirliğini kendiliğinden sağladıklarından SMTP'nin bunun için ek işlev ve mekanizmalara sahip olması gerekmez. Bu da dolaylı olarak protokol karmaşıklığını ve işlem yükünü azaltır.

Aktarım protokolü güvenilir olsa bile SMTP hareketlerinin çeşitli nedenlerden (SMTP sunucusunun çökmesi, fiziksel bağlantı problemleri ...gibi) dolayı özellikle veri aktarım aşamasında kesilmeye uğrama ihtimali bulunmaktadır. Her defasında işlemlere baştan başlamak gerektiğinden ağ kaynakları da zamanla gereksiz yere harcanmış olur. Bunu önlemek için, "*SMTP Service Extension for Checkpoint/Restart*" isimli uzantıdan yararlanılabilir [50]. Buna göre istemci birim ileti aktarımı için bir hareket tanımı yapar ve sunucu birim bu tanımı ve bağlantı

kesilene kadar aktarılan verileri saklar. Kesilen bağlantıdan sonra istemci tarafından açılan yeni oturumda ise sunucu bu bilgileri kullanarak hareket içeriğini hazırlar ve istemcinin iletime nerden başlanması gerektiğini sorar. Böylece istemcinin istediği yerden iletim sürdürülür.

4.3.5.3. SMTP protokolünde ağ aygıtları konumdan bağımsız şekilde tek (unique) adreslenebilmeli ve bir ağ adresinde birden fazla mantıksal aygıt bulunabilmeli

SMTP'nin standart tanımında posta iletilerinin aktarılabilmesi için kullanıcıların sahip olması gereken posta adresleri kullanıcı ve alan bilgilerini taşıyan karakter katarlarından oluşur:

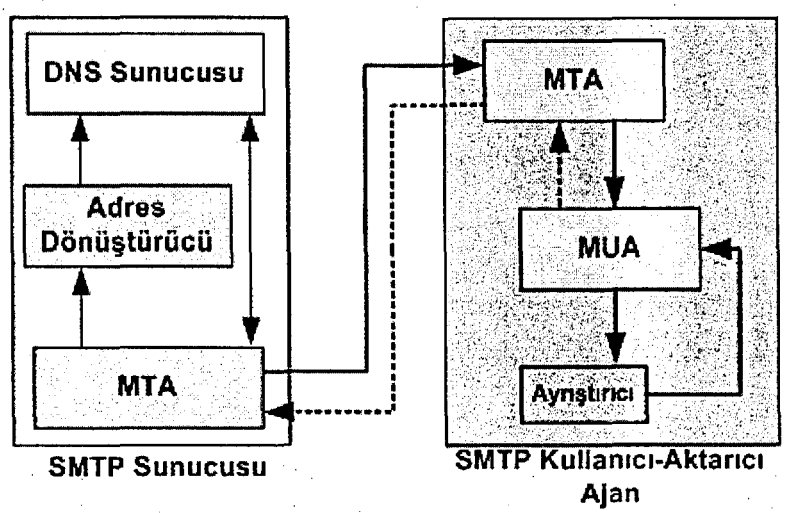
<kullanıcı>@<alan>

Burada kullanıcı bölümü ilgili kullanıcıya verilmiş olan kullanıcı adını, alan bölümü ise kullanıcının tanımlı olduğu SMTP sunucusunun alan adını (Alan adı ya da IP adresi) temsil etmektedir.

Aygıt kontrolü bağlamında bakıldığında e-posta adresinin kullanıcı bölümünün bir ağ aygıtına bulunduğu yönetsel alanda verilmiş olan isim değerini taşıyacağı düşünülebilir. Şekil 4.40.'da görülen tasarıma göre bir SMTP sunucusu temel olarak bir DNS sunucusu, bir adres dönüştürücü ve posta iletilerini nakletmek için kullanılan bir MTA modülünden oluşmalıdır (ağ geçidi olarak kullanılacaksa buna yönelik işlevleri de desteklemelidir). Ağa bağlanabilen aygıtlar ve aygıt denetleyicileri (ağa doğrudan bağlanamayan aygıtlar için kullanılan aygıtlar) içerisinde çalışması gereken SMTP uygulaması ise standart MUA değil, MTA ve MUA işlevlerini birlikte içeren **kullanıcı-aktarıcı ajan** olarak adlandırılan yeni bir birim olmalıdır. SMTP kullanıcı-aktarıcı ajan bu sayede gerektiğinde bir SMTP aktarıcı ajan (MTA) gibi posta iletisi aktarabilecek, gerektiğinde de bir SMTP kullanıcı ajan (MUA) gibi ileti alıp işleyebilecektir. Ancak MUA burada standart işlemleri yerine (posta iletisini kullanıcıya gösterme gibi) posta iletisini ayrıştırıp, içerdiği bilgilere göre aygıt kontrolü kapsamındaki işlemleri yerine getirmek ya da denetlemekle görevlidir.

Yukarıda bahsedilen ve Şekil 4.40.'da bileşenleri gösterilen SMTP varlıklarının çalıştığı bir yönetsel alanda SMTP kullanıcı-aktarıcıları ağ geçidi SMTP sunucusu

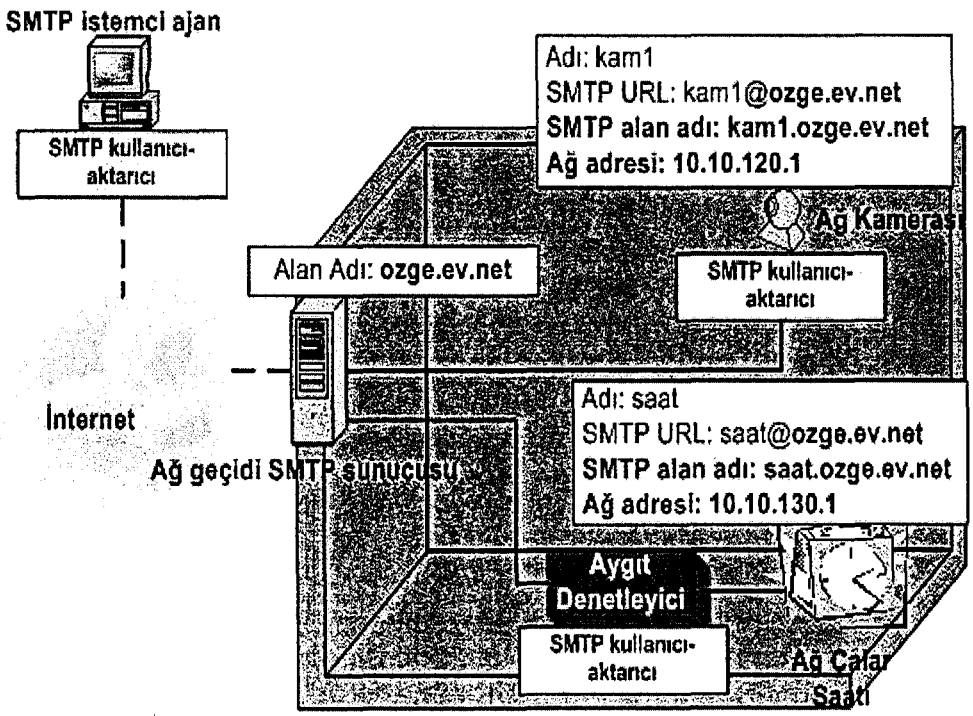
tarafından aslında birer SMTP sunucusu olarak görülürler. Yani her birinin yönettiği birer alan varmış gibi davranırlar. Alan dışından bakıldığında ise bu aygıtların kendilerine ait posta adresleri bulunmaktadır. Örneğin alan adı "*ozge.ev.net*" olan ağ geçidi SMTP sunucusuna "*kamera1@ozge.ev.net*" alıcı adresli bir posta iletisi geldiğinde sunucu adres dönüştürücüyü kullanarak aygıtın posta adresini "*kamera1.ozge.ev.net*" alan adına dönüştürür ve bu değeri DNS sunucusunda arar. Bu isme karşılık gelen ağ adresine iletiyi gönderir. Böylece iletiyi bir başka SMTP sunucusuna göndererek bir nakil sunucu gibi çalışmış olur. Halbuki sunucunun gönderdiği adreste bir ağ aygıtı bulunmaktadır. İletiyi alan aygıtın içinde çalışan SMTP MTA birimi olacaktır. MTA iletinin sahibi içinde çalıştığı ağ aygıtı olduğundan birlikte çalıştığı MUA uygulamasının kullanabilmesi için iletiyi posta kutusuna atar. MUA bir şekilde (periyodik tarama yaparak ya da işletim sistemi yardımı ile) posta kutusuna ileti geldiğini anlayarak iletiyi ayrıştırıcı uygulamaya gönderir. Ayrışan başlık alanları ve gövde bölümünden edindiği bilgileri kullanarak gerekli işlemleri gerçekleştirir.



Şekil 4.40. Tasarlanan SMTP sunucusu ve SMTP kullanıcı-aktarıcı yapısı

Ağ geçidi olarak da çalışması düşünülen ve ilgili yönetsel alandan sorumlu SMTP sunucusu sahip olduğu ya da yönettiği DNS sunucusu (Şekil 4.40.) aracılığı ile ağ aygıtlarının sahip olduğu alan adları ve bu değerlere karşılık gelen ağ (IP)

adreslerini saklayabilir. Böylece SMTP sunucusuna yönetsel alan dışından gelen bir posta iletisinin sahibi, iletinin içerisinde bulunan alıcı e-posta adresi kullanılarak belirlenir. DNS sunucusunun sakladığı kayıtlar içerisinden alıcı adresindeki alan adının bulunduğu kayıt bulunur ve posta iletisi buradan elde edilen ağ adresine gönderilir. Şekil 4.41.'de de görüldüğü gibi bir SMTP istemci kullanıcı-aktarıcı ajan *özge.ev.net* alanı içerisindeki ağ kamerasına posta iletisi göndermek istediğinde iletiyi *kam1@ozge.ev.net* adresine gönderir. Bu adreste bulunan SMTP sunucusu adres dönüştürme işlemi ile kameranın alan adını DNS sunucusunda arar ve iletiyi bulduğu ağ adresine gönderir. Böylece ileti hedefine ulaştırılmış olur.



Şekil 4.41. SMTP adresleri ve SMTP sunucu/ağ geçidi

4.3.5.4. SMTP protokolünde güvenlik, kimlik denetimi ve gizlilik desteği

SMTP posta iletilerinin aktarımında iki aşama söz konusudur:

- İstemci ajandan yerel SMTP sunucusuna (MTA) iletinin aktarımı,
- SMTP sunucusunun, hedef posta adresinden sorumlu SMTP sunucusuna posta iletisini nakletmesi.

Aygıt kontrolü bağlamında ise bölüm 4.3.5.3.'deki tasarıma göre iletişim aslında iki MTA arasında gerçekleşmektedir. Bu yüzden güvenliğin uçlar arasında ve aracı MTA'lar arasında sağlanması gerekir.

SMTP uçlar arası (gönderici-alıcı arası) aktarım mantığına sahip olmadığından aktarım seviyesinde yani adımlar arası güvenlik anlayışı benimsenmiştir. Bu türde güvenlik ve kimlik denetimi yöntemlerini tanımlayan çeşitli SMTP uzantıları bulunmaktadır [51,52]. Bu yöntemlerin genel amacı posta iletisinin aktarımı sırasında görev yapan her bir SMTP varlık (MUA-MTA ya da MTA-MTA arası) çifti arasında aktarım güvenliğini sağlamaktır. Bu işlem genelde varlık çiftleri arasında kurulacak diyalog neticesinde güvenlik ve kimlik denetimi konusunda varılan uzlaşmalar sonucunda edinilen ortak güvenlik parametrelerine göre gerçekleştirilmektedir.

Göndericiden-alıcıya aktarımda posta iletilerinin güvenliğini ve bütünlüğünü sağlayabilmek için sayısal imza ve şifreleme hizmetleri veren yöntemlerden yararlanılabilir (Örn: PGP, S/MIME) [53,54]. Bunlardan **S/MIME (Secure/Multipurpose Internet Mail Extensions)** posta iletilerinin gövde bölümünde aktarılan çeşitli MIME türlerindeki verilerin şifrelenmesi ve sayısal imza ile imzalanması işlemlerini gerçekleştirmektedir. Bu sayede kimlik denetimi, ileti bütünlüğü, gönderenin kabulü (sayısal imzalar ile), gizlilik ve veri güvenliği (şifreleme ile) hizmetlerini yerine getirebilmektedir. Bu yönteme ek olarak aygıt kontrolü bağlamında güvenlik düzeyini arttırmak amacı ile SMTP ileti yapısına yeni başlık alanları eklenebilir. Örneğin "*X-user*" ve "*X-passwd*" olarak adlandırılan iki başlık alanı, istemci SMTP biriminin sahip olduğu (otomatik olarak ya da kullanıcıdan istenen) kullanıcı adı ve şifre değerlerini içerir. Bu başlık alanları "*SMTP Service Extension for Secure SMTP over Transport Layer Security*" [51] ya da "*SMTP Service Extension for Authentication*" [52] isimli uzantılarda da kullanılan şifreleme yöntemleri yardımı ile şifrelenmiş bir şekilde alıcı birime ulaştırılabilir. Alıcı birim (kullanıcı-aktarıcı) ya da bağlı olduğu SMTP ağ geçidi/sunucusu bu değerleri inceleyerek istemcinin hem kimliğini hem de bu kimlikteki bir istemcinin istenen işlemi (iletinin gövde bölümünde aktarılan komut) ilgili ağ aygıtına yaptırmaya yetkisi olup olmadığını denetlemiş olur.

Aynı şekilde ilgili ağ aygıtından gönderilen yanıt posta iletilerinin içerisine aygıtın kullanıcı ismi ve şifre bilgileri yazılarak istemcinin iletinin doğru yerden gelip gelmediğini tam olarak anlayabilmesi sağlanır. Böylece karşılıklı kimlik denetimi daha sağlam bir şekilde gerçekleştirilmiş olur.

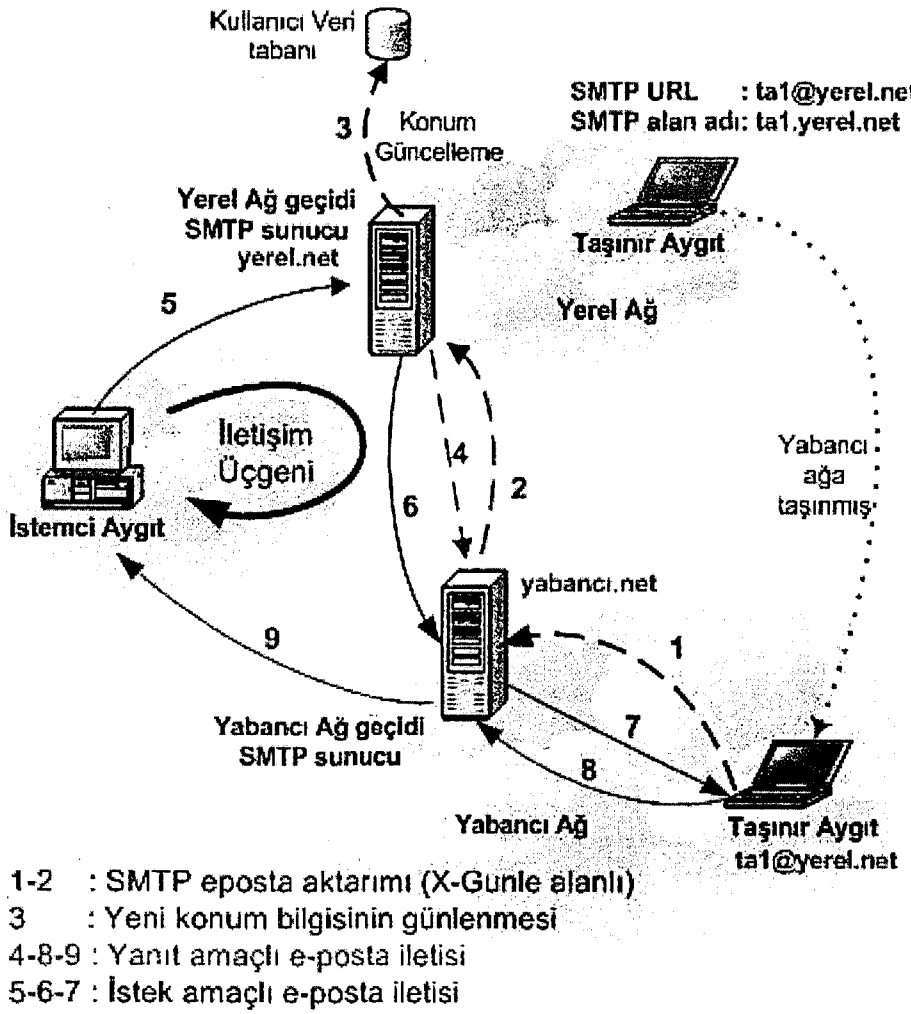
4.3.5.5. SMTP protokolünde aygıt taşınırlık desteği

SMTP iletişimi posta iletilerinin istemciden, bağlı olduğu sunucuya, oradan da alıcının bağlı olduğu sunucuya kadar olan aktarma işlemlerini (sunucudan alıcıya aktarım işlemini gerçekleştirmez) kapsadığından sadece kullanıcıların taşınırlığından söz edilebilmektedir. Bu sayede kullanıcılar posta kutularının bulunduğu sunucuya konumdan bağımsız olarak bağlanarak e-posta gönderme ve gelen e-postaları okuma imkanına sahiptirler. Ancak aygıt kontrolü kapsamında bakıldığında aygıt taşınırlığını sağlamaya yönelik olarak SMTP'ye bazı eklentilerin yapılması gerekir.

İlk olarak konum güncelleme işlemini gerçekleştirmek üzere yeni bir başlık alanından yararlanılabilir. Örneğin "*X-Gunle*" olarak adlandırılabilecek olan bu alan, kullanıcı-aktarıcı ajan uygulaması çalıştıran bir ağ aygıtının yeni bir yönetsel alan içerisinde çalışmaya başladığında kendisine yeni ağ adresini sağlayan SMTP sunucusunun adresini (DNS adı ya da IP adresi) asıl bağlı olduğu SMTP sunucusuna bildirmesi için kullanılacaktır. Buna göre, ağ aygıtı yeni ağ adresini edinir edinmez kendisine bu ağ adresini sağlayan SMTP sunucusu ile TCP bağlantısı kurar. Sahip olduğu yeni ağ adresini "*X-Gunle*" başlık alanına yerleştirdiği bir e-posta iletisini alıcısı kendi yerel SMTP adresi olacak şekilde gönderir. Yabancı sunucu iletiyi aldığında iletinin zarfındaki gönderici adresini "*tal@yerel.net*"den "*tal.yerel.net@yabanci.net*"e değiştirir ve bu eşleme bilgisini kendi içerisinde saklar. İleti iki sunucu arasında kurulan TCP bağlantısı ile yerel sunucuya ulaştığında yerel sunucu alıcı adresi olarak verilen ve halen kendi yönetsel alanında bulunduğunu düşündüğü aygıta (alan adı bölüm 4.3.5.3.'de anlatılan yöntemle oluşturulan kullanıcı aktarıcı ajana) TCP bağlantısı kurmaya çalışacaktır. İlgili aygıtla bağlantı kurmayı başaramayan sunucu iletiyi incelemek durumundadır. Aygıt ile aralarındaki güvenlik ölçütlerine göre iletiyi inceleyen sunucu iletide "*X-Gunle*" başlık alanının varlığını

tespit ettiğinde iletiye üzerinden geçtiği her ara sunucu tarafından eklenen "*Received*" başlık alanlarından ilkini alıp, önce buradaki alan adını gönderen adresinin içerdiği alan adıyla karşılaştıracak, aynı ise bu alan adına sahip sunucunun aygıtın o an bulunduğu yabancı ağın sorumlusu olduğunu düşünerek yönettiği aygıtlar için tuttuğu veritabanını bu değerle günleyecektir. Bundan sonra ilgili aygıt için kendisine gelecek e-posta iletilerini bu yeni sunucuya yönlendirecektir. Bu yönlendirme işlemini yaparken iletilerin zarf bölümündeki alıcı adresini yine "*tal.yerel.net@yabanci.net*" olarak değiştirmek durumundadır.

Şekil 4.42'de örneklendiği gibi bir taşınır aygıt yabancı bir ağa taşındığında bu durumu kendi sunucusuna bildirmek için yabancı SMTP sunucusuna e-posta gönderme isteminde bulunuyor. Yabancı sunucu yerel sunucu ile bağlantı kurarak istenen e-posta iletisini aktarıyor. İletinin gönderici ve alıcı adresleri "*tal@yerel.net*" olmak durumunda. Yerel sunucu alıcının adresini "*tal.yerel.net*" alan adına dönüştürüp bu adın karşılığı olan IP adresini DNS sunucusundan aratıyor. Bu adrese TCP bağlantısı kurmak istese de bu mümkün olamayacağından, sunucu iletinin içeriğini inceliyor. "*X-Gunle*" başlık alanı ile karşılaşacağından iletiye eklenen ilk "*Received*" başlık alanında bulunan alan adını kullanarak DNS sunucusunu ve kullanıcı veritabanını güncelliyor. DNS sunucusunda ilgili aygıtın alan adına karşılık IP adresinin bulunduğu kaydı silip, iletileri yönlendirmek amacı ile yabancı sunucu ile aygıtın alan adlarının eşleştirildiği bir MX kaydını ekliyor. Sunucu artık bu kaydı kullanarak ileride ilgili aygıt için kendisine gelen iletileri yabancı sunucuya yönlendirebilecektir. Böylece iki aygıt arasında gelişecek bir iletişim üçgen şeklinde bir yapı oluşturacaktır (Şekil 4.42.).



Şekil 4.42. SMTP aygıt taşınırlığı tasarım örneği

İletişim gecikmesini azaltmak için ağ aygıtının yer güncellemesi için iki sunucu arasında kurulan TCP bağlantısı sürekli açık tutulmalıdır. Bu da birbirlerine periyodik olarak göndereceklerı SMTP "*NOOP*" iletileri ile sağlanabilir.

SMTP TCP gibi bağlantılı protokollere dayalı çalıştığından iletişim sırasında gerçekleşecek bir konum değişikliği (ağ adres değişikliği) arada kurulan bağlantının ve aktarım işleminin yarıda kesilmesine neden olacaktır. Ancak kesintiye uğrayan bir ileti aktarımına baştan başlamayı önlemek amacı ile tanımlanmış olan "*SMTP Service Extension for Checkpoint/Restart*" [50] isimli uzantı sayesinde sunucu ile yeni bağlantı kurulduğunda yarım kalan ileti aktarımı kaldığı yerden devam

ettirilebilmektedir. Fakat bu uzantı ağ aygıt kontrolü bağlamında iki aygıt arasında sürekli ve gerçek zamanlı veri akış oturumları için yeterli değildir. Çünkü kopan bağlantının yeniden sağlanması ve veri akışının tekrar başlatılması için geçen süre bağlantısız iletişim ile sağlanan iletişime göre daha uzun sürecektir.

4.3.5.6. SMTP protokolünde IPv6 desteği

SMTP'nin adresleme mantığı gereği SMTP sunucuları ileti aktarımı için gerekli hedef SMTP sunucu konum bilgilerini DNS sunucularını kullanarak elde ederler. Bu yüzden SMTP sunucularının IPv6 adresleri ile iletişim kurabilmesi için DNS sunucuların IPv6 desteğinin olması gerekir. Bir DNS sunucusunun *ornek.net* isimli alanda çalışan *mx1* adlı SMTP sunucusu için tuttuğu kayıtlar aşağıda görülmektedir:

ornek.net	*IN MX 1 mx1.ornek.net*
mx1.ornek.net	*IN A 193.140.24.3*
mx1.ornek.net	*IN AAAA 3eef:420:ffff::1*

Birinci satırdan, *ornek.net* alanındaki bir adrese (Örn: ozge@ornek.net) gönderilecek olan bir ileti *mx1.ornek.net* makinesine gönderilmesi gerektiği anlaşılmaktadır. İkinci satırda *mx1.ornek.net* makinesinin IPv4 adresi, üçüncü satırda ise IPv6 adresi bulunmaktadır. DNS sunucu üçüncü satırda örneği verilmiş olan türde kayıtlar tutarak IPv6 desteği sağlamış olur.

Bunun yanında SMTP adreslerinin alan bölümüne alan adı ya da IPv4 adresi yazılabilmektedir. Bu bölümün azami boyu 255 karakter olduğundan IPv6 adreslerinin kullanımı da mümkündür. Aynı şekilde SMTP komutlarının bir kısmında kullanılan adres bilgilerinin (Örn: RCPT TO: ozge@ornek.net) azami boyutu 512 karakter olduğundan, bu bilgilerin IPv6 adreslerle oluşturulmasında (Örn: RCPT TO: oze@*3eef:420:ffff::1*) da bir sakınca bulunmamaktadır.

4.3.5.7. SMTP protokolü esnek ileti yapısına sahip olmalı

SMTP posta iletilerinin sahip olduğu başlık ve gövde bölümleri oldukça esnek tanımlanmıştır [42]. Yeni başlık alanları ya da SMTP komutlarının tanımlanması kolaylaştırılmıştır. Uzantılar IANA tarafından tescil edilebileceği gibi, sadece yerel

kullanıma yönelik olarak da tanımlanabilmektedirler. Bu tür başlık alanı ya da komut isimleri "X" ile başlatılarak kayıtlı olanlardan ayırt edilmiş olur.

Bununla birlikte posta iletilerinin gövde bölümü de çeşitli MIME türlerinde (SDP, XML ...gibi) veri taşıyabilmektedir.

Aygıt kontrolü kapsamında, farklı türdeki aygıtların işleyişine yönelik olarak kullanılması düşünülen denetim işlemleri yeni SMTP komutları ya da ileti başlıkları şeklinde kolayca tasarlanabilir. Örneğin "*X-Komut*" olarak adlandırılan bir başlık aktarılacak olan komutu tanımlamak için kullanılabilir. O zaman "*X-Komut: Sor*" şeklindeki bir başlık posta iletisinin alıcısına örneğin sorgulama isteğinin geldiğini ifade edebilir. Bundan sonra alıcı posta iletisinin örneğin gövde bölümünde gönderilmiş olan parametrelere yönelik sorgu sonuçlarını kendisine gelen iletinin gövde bölümüne ekleyerek istemciye geri gönderebilir. Ancak burada iletilerin istek mi yanıt mı olduğunu belirleyebilmek için bir başka başlık alanına ihtiyaç duyulabilir. Örneğin "*X-Tür*" olarak adlandırılan bu alan posta iletisi içerisinde "*X-Tür: İstek*" ya da "X-Tür: *Yanıt*" şeklinde oluşturularak iletinin türü de belirtilmiş olur.

Ayrıca eğer bir istek yerine getirilemiyorsa ilgili aygıtın sebebi açıklamaya yönelik bir yanıt iletisi oluşturması gerekir. Bunun için ise "*X-Hata*" olarak adlandırılabilecek bir başka yeni başlık alanından yararlanılabilir. Bu alan hata ile ilgili yanıt kodunu içerir ve yerine getirilemeyen isteğin bulunduğu istek iletisinin içerisine yerleştirilir. Ayrıca iletinin "*X-Tür*" alanının içeriği "*X-Tür: Yanıt*" şeklinde değiştirilerek istemciye gönderilir. Bu sayede istemci isteğin neden yerine getirilemediğini anlamış olur. Burada kullanılacak olan hata kodları da genel ya da aygıta özel olarak belirlenebilir. Olumlu bir yanıt iletisi içerisinde bu alanın karşılığı boş bırakılabilir.

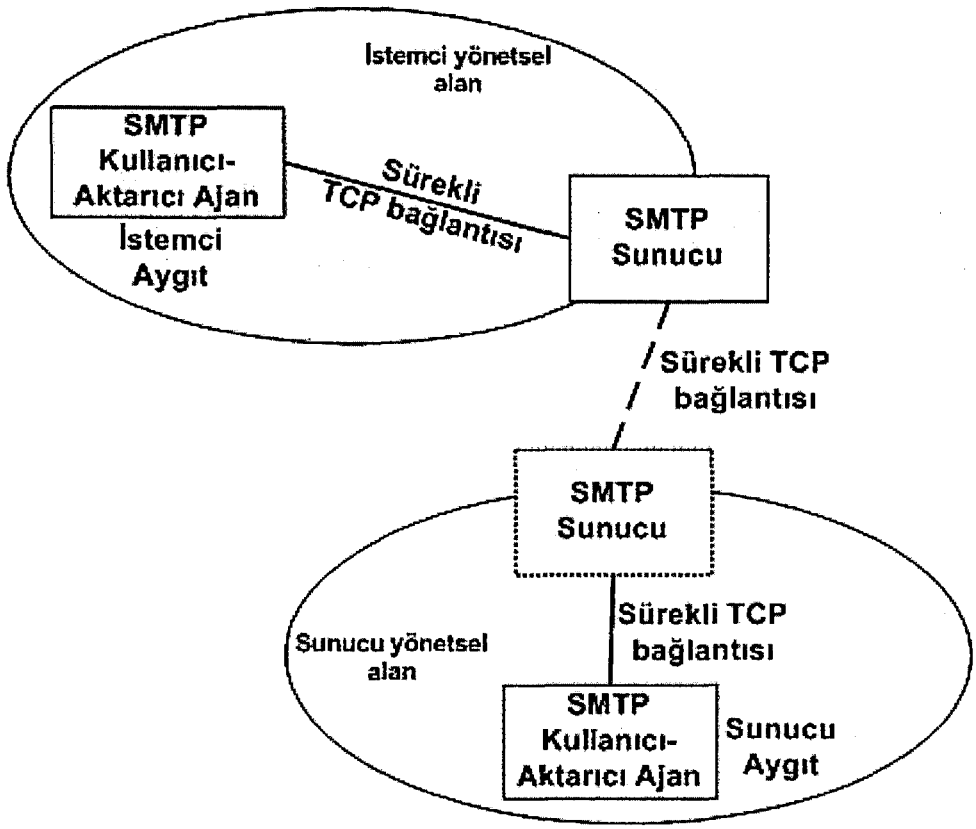
4.3.5.8. SMTP protokolünün yapısı karmaşık olmamalı

SMTP protokolü oldukça basit yapılı bir protokoldür. Bunun sebeplerinden birisi metin tabanlı bir protokol olmasıdır. Bununla birlikte sunucu ile istemciler arasında oturum başlatmak ve ileti aktarmak için kullanılan komutların bir düzine kadar olması da protokol gerçekleştirimlerinin basit yapıda olmasını sağlayan bir

etmendir. Bununla birlikte ileti aktarım ortamı olarak TCP gibi güvenilir aktarım protokollerinin seçilmiş olması iletilerin güvenilirliği ile ilgili işlemlerin (akış denetimi işlemlerinin) seçilen aktarım protokolü tarafından otomatik olarak gerçekleştirilmesini sağlayacağından SMTP protokol gerçekleştirimlerinin ek güvenilirlik mekanizmalarına sahip olmasına da gerek yoktur.

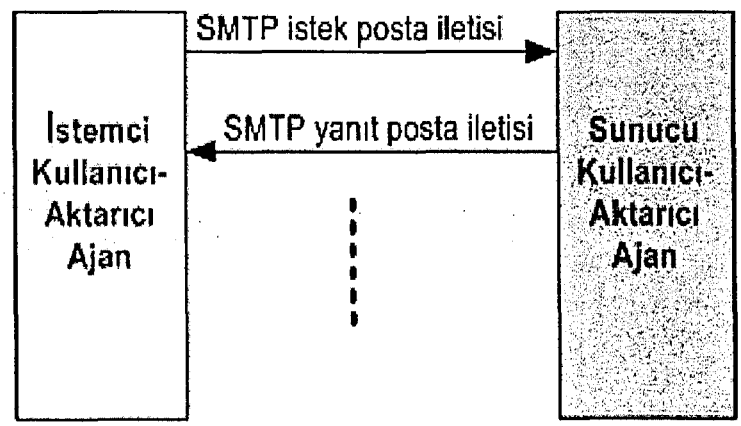
SMTP aygıt kontrolünde kullanılmak istendiğinde esnek ileti yapısından kolayca yeni başlık alanları tanımlayarak faydalanılabilmektedir. Ancak aygıtlar arası iletişim için komut ve yanıt bilgilerini taşıması düşünülen bu başlık alanları ileti yapısını ve iletiyi ayrıştırma işiyle görevli protokol bölümünün işlem karmaşıklığını bir miktar arttırabilir.

Ayrıca aygıt kontrolünde iletişim klasik kullanıcı – aktarıcı ajanlar arasında değil, bölüm 4.3.5.3.'de tasarlanan ve kullanıcı-aktarıcı olarak adlandırılan karma ajanlar ve sunucu ajanlar arasında gerçekleştirilecektir. Bu durumda hem kullanıcı hem de aktarıcı özelliklerine sahip olan ajanlar, standart SMTP kullanıcı ajanlarına göre bir miktar daha karmaşık yapıya sahip olacaklardır. Ancak bu durum aygıtlar arası iletişimi oldukça basitleştirmektedir. Bu sayede SMTP'nin klasik mantığı yerine uçlar arası iletişim gerçekleştirilmiş olacaktır. Yani iletişim istemci aygıtta çalışan SMTP sunucusu ile hedef aygıtta çalışan SMTP sunucusu arasında gerçekleşmiş olacaktır. Bu iletişimi biraz daha kolaylaştırmak amacı ile Şekil 4.43'de de görüldüğü gibi, ileti aktarımı için kurulacak olan TCP bağlantıları sürekli açık tutularak her ileti aktarımında bağlantı işlemlerinin getireceği ek iş yükü ve kaynak kullanımı da önlenmiş olacaktır.



Şekil 4.43. Aygıtlar arası SMTP iletişimi için sürekli açık tutulması gereken TCP bağlantıları

Şekil 4.44.'de de görüldüğü gibi aygıt kontrolü kapsamında SMTP iletişimi türü (istek ya da yanıt) başlık alanları ile belirlenecek olan posta iletilerinin karşılıklı aktarımı şeklinde gerçekleşecektir.



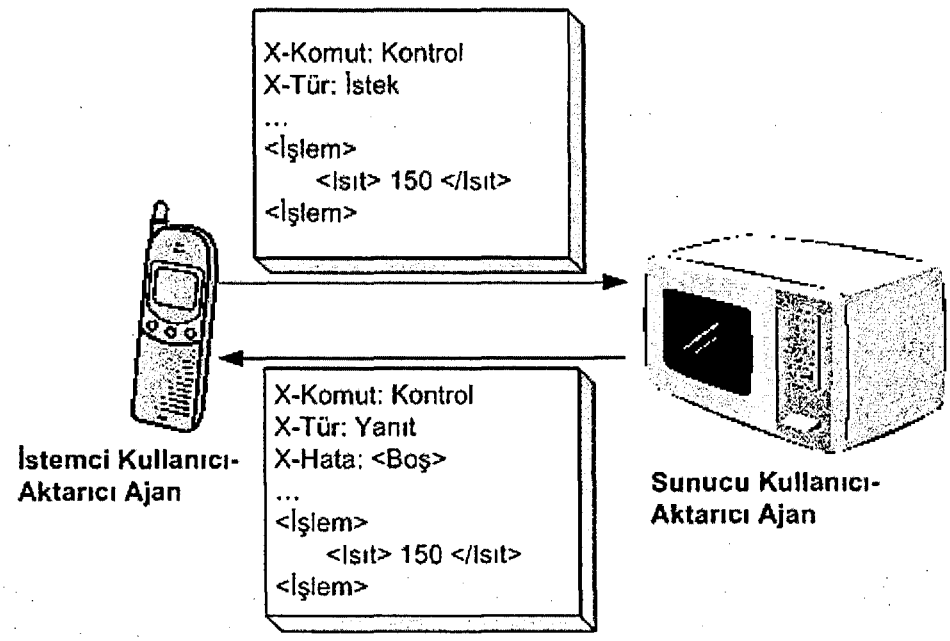
Şekil 4.44. Aygıt kontrolü kapsamında SMTP ajanları arası iletişim

4.3.6. SMTP protokolünde işlevsel gereksinimler

Bu bölümde SMTP 3. bölümde tanımlanan işlevsel gereksinimlere göre incelenmektedir.

4.3.6.1. SMTP protokolünde aygıt denetimi

SMTP ile aygıt denetim işlevini yerine getirebilmek için hedef aygıta yapması gereken işin tanımının başlık alanları halinde ya da ileti gövdesinde bir MIME türü kullanılarak bildirilmesi gerekir. Bunun için bölüm 4.3.5.7.'deki tasarımda örnek olarak verilen "*X-Komut*" başlık alanının gövde bölümüne örneğin "*Kontrol*" değeri yazılarak iletinin bir denetim komutu taşıdığı belirtilebilir. *"Kontrol"* anahtar kelimesini gören alıcı, iletinin gövde bölümünü ayrıştırarak tam olarak istenen işin tanımına ulaşabilir. Bu tanım daha önce de bahsedildiği gibi herhangi bir MIME türünde olabilir (SDP, XML ...gibi).



Şekil 4.45. SMTP aygıt denetim örneği

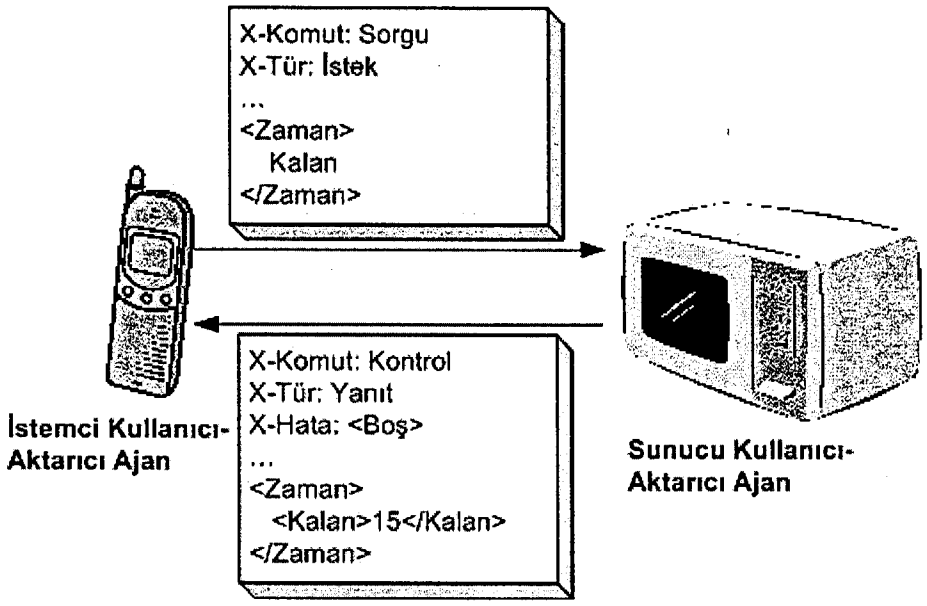
Şekil 4.45.'de görüldüğü gibi istemci birim mikrodalga fırına 150 derecede çalışmaya başlaması için bir istek iletisi göndermektedir. Bu iletinin gövde bölümü

gösterim amaçlı olarak XML şeklinde gerçekleştirilmiştir. Bu istek iletisine karşılık fırın olumlu ya da olumsuz bir posta iletisi göndermek durumundadır. Örnekte istek yerine getirildiğinden "*X-Hata*" başlık alanının gövde bölümü boş bırakılmıştır.

4.3.6.2. SMTP protokolünde durum denetimi

Durum denetimi bölüm 4.3.5.7.'de tanımlanan "*X-Komut*" başlık alanının gövde bölümünde durum denetimi işlemlerini ifade etmesi için "*Sorgu*" anahtar kelimesi kullanılarak gerçekleştirilebilir. Bu sayede hedef aygıt istek iletisinin bir durum denetleme işlemi içerdiğini anlar. İstemci birim hedef aygıtla ilgili sorgulamak istediği durum parametrelerini posta iletisinin gövde bölümüne yerleştirir. Hedef aygıt ise bu parametrelerden yanıtlayabileceklerinin (aygıtın parametreye gerçekten sahip olmasına, istemcinin yetkilerine ...vb göre) değerlerini kendisine gelen istek iletisinin içerisine yerleştirir. Karşılığını gönderemeyeceği parametre alanlarını boş bırakabilir ancak hiçbir parametreye yanıt veremiyorsa o zaman "*X-Hata*" başlık alanını ilgili hata kodu ile doldurmak zorundadır.

Şekil 4.46.'da görüldüğü gibi hedef aygıta (mikrodalga fırın) aktarılan istek posta iletisinin "*X-Komut*" başlık alanına "*Sorgu*" anahtar kelimesi yerleştirilmiştir. Burada da iletinin gövde bölümü sadece gösterim amacıyla XML kullanılarak oluşturulmuştur. İletiyi alan fırın, geride kalan pişirme süresini yine XML formatında geri göndermektedir. Eğer istek iletisi içerisinde gönderilen sorguyu hedef aygıt desteklemiyorsa, bu durumu belirtecek bir hata kodunu "*X-Hata*" başlık alanına yerleştirmesi gerekir.

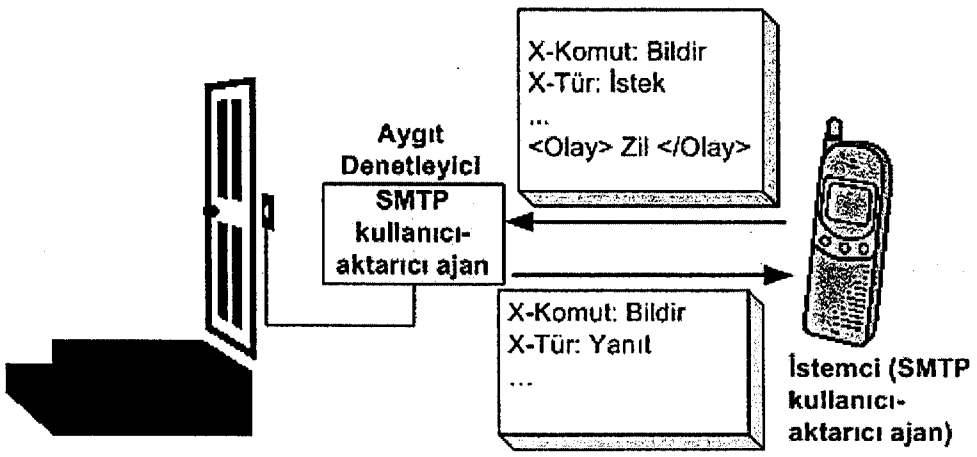


Şekil 4.46. SMTP durum denetim örneği

4.3.6.3. SMTP protokolünde olay denetimi

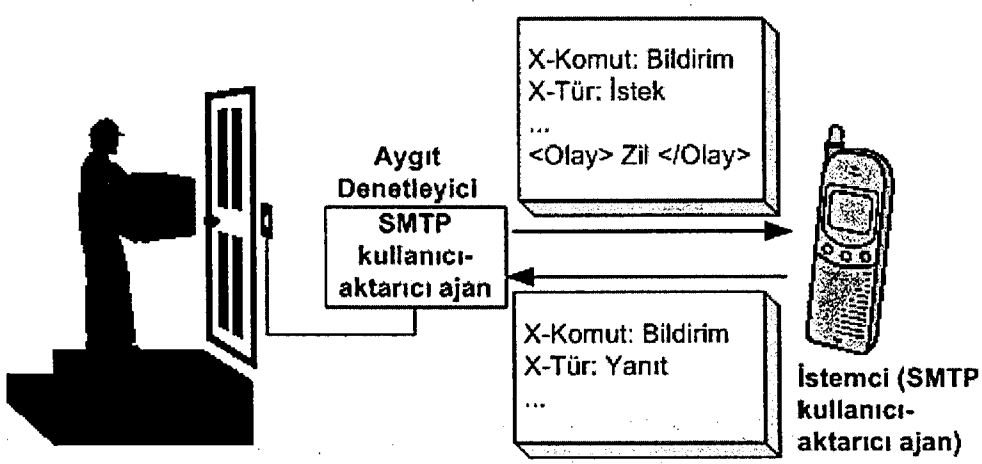
Olay denetimi bölüm 4.3.5.7.'de tanımlanan "*X-Komut*" başlık alanının gövde bölümünde "*Bildir*" anahtar kelimesi kullanılarak aktarılan iletinin bir olaya yönelik bildirim isteği içermesi ile sağlanabilir. Bununla birlikte olay ile ilgili tanımlamalar posta iletisinin gövde bölümüne eklenir. Bu istek iletisini alan aygıt eğer tanımlanan olayı destekliyorsa bu bildirim isteğini kaydetmek durumundadır. Kayıt işleminin hatasız gerçekleştiğinin bildirimi ise gelen istek posta iletisinin gövde bölümü çıkarılmış hali istemciye geri gönderilerek sağlanabilir. Eğer kayıt işleminin gerçekleşmesini engelleyecek bir durum varsa, bu durum ile ilgili hata kodu, gönderilecek yanıt iletisindeki *X-Hata* başlık alanına yerleştirilerek istemciye bildirilir.

Şekil 4.47a.'da bir kullanıcı evindeki kapı zilinin çalma olayının kendisine bildirilmesi için kapı zilinin bağlı olduğu aygıt denetleyicisine bildirim isteminde bulunmaktadır. Aygıt denetleyici içerisinde çalışan SMTP kullanıcı-aktarıcı ajan ileti içeriğinin uygun olup olmadığını inceler. Herhangi bir problem yoksa bildirim isteğini kaydeder ve istemciye kayıt işleminin sonucunu şekilde görüldüğü gibi bir posta iletisi ile aktarır.



Şekil 4.47a. SMTP olay bildirim isteminde bulunma örneği

Olay meydana geldiğinde (kapı zili çaldığında) Şekil 4.47b.'de görüldüğü gibi zile bağlı aygıt denetleyici olay bildiriminde bulunmak üzere istemci aygıta bir posta iletisi gönderir. Bu iletinin içeriği şekilde görüldüğü gibi *"X-Komut: Bildirim"* başlık değerine sahiptir. Bu iletinin alındığını bildirmek üzere istemci kendisine gelen iletiyi gövde bölümünü çıkararak aygıt denetleyiciye geri gönderir.



Şekil 4.47b. SMTP olay bildirim örneği

4.3.6.4. SMTP protokolünde oturum denetimi

Aygıtlar arasında sürekliliği olan veri akış işlemlerinin oturumlarını denetlemek amacı ile SMTP bünyesinde bölüm 4.3.5.7.'de tanımlanan "*X-Komut*" başlık alanı

kullanılarak yeni bir komut tanımlanabilir. İki aygıt arasında bir oturum başlatmak amacı ile istek iletisinin "*X-Komut*" başlık alanına "*Oturum Başla*" anahtar kelimeleri yerleştirilebilir. Bunun yanında başlatılmak istenen veri akış oturumunun türü ve parametre değerleri iletinin gövde bölümüne yerleştirilir. Bu istek iletisini alan ağ aygıtı açılmak istenen oturumun türünü ve parametre değerlerini inceler. Parametre değerlerinden uygunsuz olanlar değiştirilerek istemciye aynı istek iletisi içerisinde gönderilir. Bunun için gelen istek iletisinin "*X-Tür*" başlık alanına "*Yanıt*" anahtar kelimesi yerleştirilerek gönderilecek iletinin bir yanıt iletisi olduğu belirtilmiş olur. İstemci yanıt iletisinin kendisine ulaştığını ve eğer varsa parametre değişikliklerini kabul ettiğini bildirmek amacı ile gelen yanıt iletisini gövde bölümü çıkarılmış halde oturum isteminde bulunduğu aygıta gönderir ve bundan sonra üzerinde uzlaşılan parametre değerleri kullanılarak bir oturum başlatabilir. Bu tür veri akışlarında kullanılan bir aktarım protokolü (Örn: RTP) aracılığı ile, Şekil 4.48'de de görüldüğü gibi, belirlenmiş olan parametre değerlerine dayalı olarak iki aygıt arasında iletişim başlatılır.

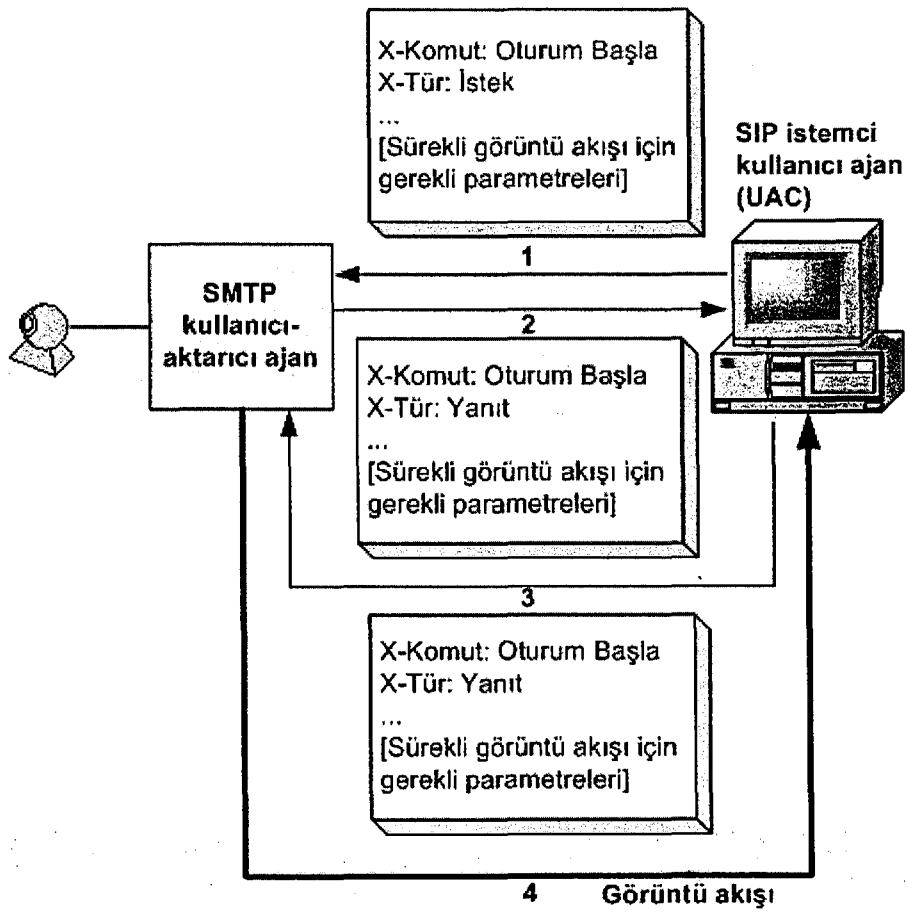
Veri akışı sırasında oturumun türünde ya da parametre değerlerinde değişiklik yapılması gerekirse, bu işlemi gerçekleştirmek için "*X-Komut*" başlık alanında "*Oturum Değiştir*" anahtar sözcükleri bulunan bir posta iletisi oluşturulabilir. İlgili oturumu belirtmek için ise bir belirteç değerinden yaralanılabilir. Bu değer ise oturum istemi için oluşturulan "*Oturum Başla*" iletilerine eklenecek örneğin "*X-OturumNo*" adlı bir başlık alanı içerisinde karşı aygıta aktarılabilir. Belirteç değerinin aygıtlar arasında tek olması için aşağıdaki gibi bir yapı kullanılabilir:

sayı.istemci_aygıt_SMTP_adresi (Örneğin: 12345.kaml@ozge.ev.net)

Oturumun türünü ya da parametre değerlerini değiştirmek için oluşturulacak olan iletinin içerisine "*X-OturumNo*" alanına ilgili oturumun oturum belirteci yerleştirilir. Gövde bölümü de değişiklik gerektiren tür ya da parametre değerleri ile doldurulduktan sonra ileti diğer aygıta gönderilir. Bu iletiyi alan aygıt istenen değişiklikler kendisi için uygunsa almış olduğu istek iletisini aynı şekilde yanıt iletisi olarak geri gönderir, uygun değilse de istek iletisini gövdesi çıkarılmış ve "*X-Hata*" alanına ilgili probleme yönelik hata kodu yerleştirilmiş halde geri gönderir. Oturum

değişiklik isteminde bulunan aygıt olumlu yanıt aldığı taktirde mevcut veri akışını durdurur ve yeni değerlerle yeni bir bağlantı üzerinde veri akışını başlatır.

Mevcut bir oturumu sonlandırmak için ise "*X-Komut*" başlık alanı "*Oturum Bitir*" değerine sahip bir posta iletisi oluşturulabilir. Bu ileti de ilgili oturumun belirteç değerini içermelidir. Oturumun bitirilmesini isteyen aygıt oluşturduğu istek iletisini karşı aygıta gönderir. Bu iletiyi alan aygıt iletiyi aldığını ve oturumu kapatmaya hazır olduğunu bildirmek üzere kendisine gelen iletiyi aynı şekilde geri gönderir. Bu işlem sonucunda veri akışı sonlandırılabilir.



Şekil 4.48. SMTP ile oturum başlatma örneği

5. SONUÇLAR

Mevcut Internet yapısı içerisinde farklı kullanım amaçları olan SNMP, SMTP ve SIP protokollerinin aygıt kontrolü bağlamında belirlenmiş olan temel ve işlevsel bileşenlere yönelik gereksinimleri ne ölçüde karşıladıkları belirlenmiş ve tanımlanmıştır. Standart protokol tanımında ilgili gereksinimi karşılamaya yönelik bir desteğin bulunamadığı durumlarda ise protokolün mevcut kullanım ortamlarında uyumsuzluk yaratmayacak şekilde bazı uzantı tanımları önerilmiştir.

- **Geniş alan desteği**: Bu konuda SIP diğer iki protokole göre daha uygun özelliklere sahiptir. Sahip olduğu sunucu türleri ve adresleme mantığının yardımı ile Internet üzerinde kendisine ait bir iletişim ağı oluşturabilecek durumdadır. Ayrıca ileti aktarımı için çok çeşitli aktarım protokol ve ortamlarını kullanabilmektedir. Buna karşılık SNMP ve SMTP standart tanımlarında sadece iki temel işlevsel birimden oluşmaktadır (SNMP için yönetici – ajan, SMTP için posta aktarıcı ajan (sunucu) – posta kullanıcı ajan (istemci)). Bu yüzden bazı işlevleri yerine getirmek için bu birimleri amaçları dışında kullanmak gerekmektedir. Hatta aygıt kontrolü işlevini gerçekleştirebilmek için SMTP kullanıcı-aktarıcı ajan isminde yeni bir bileşen tasarlanmış, SMTP sunucusunun işlevsel içeriği de buna göre güncellenmiştir. SMTP protokolünün bir kısıtlaması da iletişim için sadece güvenilir aktarım imkanı veren protokollerden yararlanmasıdır.
- **Güvenilir iletişim desteği:** SNMP'de aktarım için UDP tercih edildiğinden ve UDP bir bağlantısız protokol olduğundan iletilerin kaybolduğunun anlaşılabilmesi için *Request ID* ileti alanından yararlanılır. Ancak tekrar gönderme ile ilgili protokolün tanımladığı ve zorunlu kıldığı bir yöntem bulunmamaktadır. Bu durum yönetici varlıkların inisiyatifine bırakılmıştır. SMTP'de ise iletişim TCP başta olmak üzere güvenilir iletişim sağlayan aktarım protokolleri üzerinden gerçekleştirildiğinden iletilerin alıcıya doğru sırada ulaşıp ulaşmadığını denetleyecek ek akış denetim mekanizmaları içermemektedir (çünkü TCP gibi güvenilir aktarım protokolleri bu tür hizmetleri zaten içermektedirler). Ancak üzerinden iletişim kurulan güvenilir bağlantıların kopması yüzünden yarıda kalan

işlemlerin kaldığı yerden devam edebilmesi için bir uzantı tanımı yapılmıştır [50]. SIP hem bağlantılı hem de bağlantısız iletişim ortamlarında güvenilirliği sağlamak ve güçlendirmek için bazı tanımlar içermektedir. Bu sayede SNMP ve SMTP'ye göre daha güvenilir bir aktarım ortamı sağlamaktadır.

- **Ağ aygıtları konumdan bağımsız şekilde tek (unique) adreslenebilmeli ve bir ağ adresinde birden fazla mantıksal aygıt bulunabilmeli :** Aygıt kontrolü bağlamında düşünüldüğünde SMTP'de aygıtları konumdan bağımsız adresleyebilmek için mevcut kullanıcı adresleme yönteminden doğrudan yararlanılabilir. SIP'de de benzer bir adresleme mantığı kullanıldığından konumdan bağımsız adresler oluşturulabilmektedir. SNMP'de ise varlıklar içerisinde çalışan SNMP motorlarını, dolayısıyla ilgili varlığı ayırt etmek için snmpEngineID belirtecinden yararlanılır. Ancak SNMPv3'ün standart tanımında snmpEngineID belirteci tek bir yönetsel alan içerisinde tek değerlere sahip olabilecek şekilde tanımlanmıştır. Aygıt kontrolü bağlamında bu belirteci kullanabilmek için belirli şartlara uygun şekilde içeriğinin tasarlanması gerekmektedir [9]. Bunun için yapılmış olan tasarım sayesinde SNMP'ye konumdan bağımsız adresleme özelliği kazandırılmıştır. Ancak bu adresleme yapısı SIP ve SMTP'nin kullandığı yapıya göre daha karmaşık ve kullanışlılığı daha düşüktür. Belli bir alan içerisindeki ağ aygıtlarının Internet'e tek bir noktadan bağlantı kurabilmesi için SNMP'de hem ajan hem de yönetici işlevlerine sahip ADY'lerden yararlanılır. Bu birimlerin aynı zamanda ağ geçidi olarak kullanılacağı düşünülmüştür. SIP özel olarak bu işleve yönelik çalışan bir sunucu tanımı içermektedir. Vekil sunucu olarak adlandırılan bu birimler aygıt kontrolü bağlamında mesken ağ geçidi adıyla kullanılırlar. SMTP'de de işlevi gereği ağ geçidi ve Internet bağlantı noktası olarak kullanılmak üzere SIP'deki gibi standart bir birim tanımlanmamıştır. Bu yüzden bu işlevi yerine getirmek üzere yeni bir SMTP sunucu tanımı yapılmıştır. Bu yeni sunucu yeni tanımlanan SMTP kullanıcı-aktarıcı ajan içeren aygıtların Internet ile olan iletişimlerini denetlemektedir.

- **Güvenlik, kimlik denetimi ve gizlilik desteği:** SIP ve SNMPv3'ün standart tanımları içerisinde güvenlik, gizlilik ve kimlik denetimine yönelik çeşitli mekanizma ve yöntemler bulunmaktadır. Ancak SNMPv3 SIP'e göre daha düzenli, modüler ve sağlam bir güvenlik yapısına sahiptir, çünkü bu gereksinimleri karşılayacak yöntemleri bir arada güvenlik modeli olarak tanımlamaktadır (Örn:USM). Aynı zamanda standart tanımında bulunan USM dışında/ile birlikte başka güvenlik modellerinin tanımlanıp kullanılmasına da izin vermektedir. SIP ve SMTP uçlar arası iletişimin yanında adımlar arası iletişime de izin verdiğinden uçlar arası şifreleme işlemi tam olarak sağlanamamaktadır. Çünkü iletinin bazı bölümleri yol üzerindeki aracı birimler tarafından kullanılmakta hatta değiştirilmektedir. Bu güvenlik açığını kapatmak için SIP ve SMTP gerçekleştirimlerinin kimlik denetim işlemleri (alıcı adres alanı şifreleme gibi, TLS [55] , IPsec gibi aktarım katmanı mekanizmaları) gerçekleştirmesi gerekir.
- **Aygıt taşınırlık desteği:** İncelenen protokollerden sadece SIP aygıt taşınırlığını desteklemektedir. Bu gereksinimi karşılayabilmek için bazı komut ve sunucu türleri tanımlamaktadır. SNMP'de aygıt taşınırlığını sağlayabilmek için tasarlanmış olan adres yapısından yararlanılabilir. Ancak SMTP'de işler biraz daha zordur çünkü TCP gibi bağlantılı aktarım ortamları üzerinde çalıştığından konum değiştirme sonucunda yeni konumu belirlemek ve bağlantıyı yeniden kurmak için yapılacak işlemler daha fazladır. Ayrıca taşınırlığı sağlayabilmek için ek başlık alanı ve adresleme mekanizmalarına ihtiyaç vardır.
- **IPV6 desteği:** Her üç protokol de standart tanımlarında hem IPv4 hem de IPv6'ya yönelik tanımlar içermektedirler. Böylece bir işlemin ya da ifadenin her iki protokol sürümü için de karşılığı bulunmaktadır.
- **Esnek ileti yapısına sahip olma:** SIP ve SMTP (e-posta) iletileri yapı olarak birbirine benzer özellikler taşımaktadır. Ayrıca ESMTP tanımı ile birlikte SMTP ve SIP iletilere (başlık alanları, komutlar .. gibi) uzantı tanımlama kolaylığı açısından da birbirine yakın esneklikte iletiler içermektedir. Buna karşılık SNMP ileti yapısı diğerlerine göre oldukça sabit bir özelliğe sahiptir. İşlemler MIB nesne

değerlerine dayandığından iletilerin veri bölümünde (gövde) iletişim kuran her iki birimde tanımlı olan nesneler ve değerleri bulunacaktır. Bu değerler farklı türlerde olabileceğinden (sekizli, tam sayı...gibi) iletim esnasında meydana gelebilecek bir veri hatası işlemin yapılmasını engelleyecektir.

- **Ağ aygıt kontrol protokolünün yapısı karmaşık olmamalı:** SIP ve SNMP ileti çeşidi olarak oldukça kısıtlı ama yeterli bir komut/yöntem kümesine sahipler. SMTP ise aygıt kontrolü bağlamında kullanılabilecek komut ya da ileti kümesine sahip değildir. Bu eksikliği giderebilmek için e-posta iletileri içerdiği ek başlık alanlarının aldığı değerlere göre istek ve yanıt iletisi olarak kullanılmaktadır. Bunun yanında SIP ve SMTP metin (ASCII formatında metin) tabanlı protokoller olmasına karşın SNMP iletilerini sekizli dizisi şeklinde oluşturduğu PDU'lar şeklinde aktarmaktadır. Bu da SIP ve SMTP'ye göre protokolün işlevsel olarak daha karmaşık olmasına neden olmaktadır.
- **Aygıt, durum, oturum ve olay denetimi:** SIP ve kısmen SNMP bu amaçlara yönelik komut/yöntem tanımları içermektedir. Ayrıca SIP oturum denetimi amacı ile tasarlanmış bir protokol olduğundan bu konuda ek tanımlara ihtiyaç duymaz. SNMP MIB nesneleri ve değerleri üzerinde işlemler gerçekleştirdiğinden aygıt kontrolü bağlamında aygıtların özellikleri ve işlevlerini ifade eden nesnelerin tanımlı olduğu varsayılarak, mevcut komut tanımlarından yararlanılmıştır. Bu komutlar ile nesne değerleri üzerinde işlemler yapılarak aygıt, durum, oturum ve olay denetimi işlemleri gerçekleştirilmeye çalışılmıştır. SMTP'nin aygıt kontrolünde kullanılabilecek komut/ileti tanımları bulunmadığından bu işlemler için yine e-posta iletilerinden yararlanılması düşünülmüştür. E-posta iletilerine eklenecek başlık alanlarının alacağı değerlere göre işlem türü belirlenecektir. SMTP TCP protokolünü kullandığından taşınırlık söz konusu olduğunda oturum denetimi verimli bir şekilde sağlanamamaktadır. Ayrıca olay denetiminin de etkin yapılabilmesi için kurulan TCP bağlantılarının sürekli açık tutulması gerekir.

Yukarıdaki karşılaştırmalı değerlendirmeye göre, belirlenmiş gereksinimleri en iyi karşılayan protokolün SIP olduğu görülmektedir. Her ne kadar SIP'in de

güvenlik, iletişim eniyileme gibi konularla ilgili bazı zayıflıkları/eksiklikleri olsa da, bunlar diğer protokollerdeki temel yapı eksiklikleri yanında önemsiz kalmaktadır. Ayrıca SIP protokolü ve zayıflıklarının giderilmesi ile ilgili çalışmalar yoğun bir şekilde sürdürülmekte olduğundan protokolün aygıt kontrolünde en uygun hale kısa süre içerisinde getirileceğine şüphe yoktur.

KAYNAKLAR

1. SCHULZRINNE, H., BLATHERWICK P., TSANG S. ve MOYER, S., *Internet Personal Appliance (IPAC) Control Discussion,* (2001).
2. Networked Appliances Research Web Site – FAQs http://www.argreenhouse.com/iapp/appliances-faq.shtml.
3. MOYER, S., MARPLES, D., ve TSANG, S., *A Protocol for Secure Wide-Area Networked Appliance Communication*, IEEE Communication Magazine, **53** (2001).
4. Hughes Software Systems Web Site – *Requirements for Networked Appliances,* http://www.hssworld.com/hss_mindsystem/ietf/networked_appliances.htm.
5. MOYER, S., MAPLES, D., TSANG, S., ve GHOSH, A., *Service Portability of Networked Appliances*, IEEE Communications Magazine , **116** (2002).
6. MOYER, S., MAPLES, D., ve TSANG, S., *Requirements for Networked Appliances: Wide-Area Access, Control, and Interworking*, Internet Taslağı draft-tsang-appliances-reqs-01 (2000).
7. USKELA, S., Mobility Management In Mobile Internet, Nokia, FINLAND.
8. Cisco Systems, "Internetworking Technology Handbook" - http://www.cisco.com/ univercd/ cc/td/ doc/cisintwk/ito_doc/snmp.htm.
9. ROSE, M., ve MCCLOGHRIE, K., *Structure and Identification of Management Information for TCP/IP-based Internets* , RFC 1155, Internet Engineering Task Force (IETF) (1990).
10. HARRINGTON, D., PRESUHN, R., ve WIJNEN, B., *An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks*, STD 62, Internet Engineering Task Force (IETF) (2002).
11. DOMINGOS, J., ve DAVISON, R., *Implications of SNMPv3 on TMN* , http://www.eurescom.de/~public-web-deliverables/P800-series/P812/final_p812/ main_results/ case_studies/ bt81219v1195 /bt81219v1195-Summary.html.

12. SCHOENWAELDER, J., *Simple Network Management Protocol (SNMP) over Transmission Control Protocol (TCP) Transport Mapping* , RFC 3430, Internet Engineering Task Force (IETF) (2002).
13. SCHOFFSTALL, M., DAVIN, C., FEDOR, M., ve CASE, J., *SNMP over Erhernet*, RFC 1089, Internet Engineering Task Force (IETF) (1989).
14. BOSTOCK, S., *SNMP Over IPX* , RFC 1420, Internet Engineering Task Force (IETF) (1993).
15. PRESUHN, R., CASE, J., MCCLOGHRIE, K., ROSE, M., ve WALDBUSSER, S., *Transport Mappings for the Simple Network Management Protocol (SNMP)*, RFC 3417, Internet Engineering Task Force (IETF) (2002).
16. MINSHALL, G., ve RITTER, M., *SNMP over AppleTalk*, RFC 1419, Internet Engineering Task Force (IETF) (1993).
17. BARNES, J., BROWN, L., ve ROYSTON, R., *Modem Management Information Base (MIB) using SMIv2* , RFC 1696, Internet Engineering Task Force (IETF) (1994).
18. SMITH, R., WRIGHT, F., HASTINGS, T., ZILLES, S., ve GYLLENSKOG, J., *Printer MIB*, RFC 1759, Internet Engineering Task Force (IETF) (1995).
19. CASE, J., *UPS Management Information Base* , RFC 1628, Internet Engineering Task Force (IETF) (1994).
20. The Official Website of SNMP Research International, Inc., *Mid Level Managers*, http://www.snmp.com/products/mlm.html.
21. BLUMENTHAL, U., ve WIJNEN, B., *User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)*, RFC 2574, Internet Engineering Task Force (IETF) (1999).
22. RACE, T. L., *Comparing and Contrasting the Security of SNMPv1 with SNMPv3*, - http://www.cs.utk.edu/~race/594paper.html, (2000).
23. STALLINGS, W., *Security Comes To SNMP: The New SNMPv3 Proposed Internet Standards*, The Internet Protocol Journal, 1-3 (1998).
24. National Institute of Standards and Technology (NIST), *Data Encryption Standard*, FIPS Publication 46-1 (1977).

25. National Institute of Standards and Technology (NIST), *DES Modes of Operation*, FIPS Publication 81 (1980).
26. DAVIS, E., *SNMPv3 - View Access Control Model* , http://www.pobox.com/~ead/misc/vacm.html
27. LAMSAL, P., *Management of the Next Generation IP Core Network*, http://www.tml.hut.fi/Opinnot/Tik-110.551/1999/papers/12ManagementOfIPng Core/ipcore.html #IPv6 %20Core%20Network%20Management, (1999).
28. DANIELE, M., ve SCHOENWAELDER, J., *Textual Conventions for Transport Addresses*, RFC 3419, Internet Engineering Task Force (IETF) (2002).
29. ROSENBERG, J., ve SHOCKEY, R., *The Session Initiation Protocol (SIP): A key component for Internet Telephony, Computer Telephony*, **88**, 8-6 (2000).
30. MOYER, S., MARPLES, D., TSANG, S., ve GHOSH, A. , *Service Portability of Networked Appliances* , IEEE Communication Magazine (2002).
31. MOYER, S., MARPLES, D., TSANG, S., KATZ, J., GURUNG, P., CHENG, T., DUTTA, A., SCHULZRINNE, H., ve ROYCHOWDHURY, A., *Framework Draft for Networked Appliances using the Session Initiation Protocol*, Internet Taslağı, draft-moyer-sip-appliances-do-02 (2001).
32. ROACH, A.B, *Session Initiation Protocol (SIP)-Specific Event Notification*, RFC 3265, Internet Engineering Task Force (IETF) (2002).
33. ROSENBERG, J., SCHULZRINNE, H., CAMARILLO, G., JOHNSTON, A., PETERSON, J., SPARKS, R., HANDLEY, M., ve SCHOOLER, E., *SIP: Session Initiation Protocol*, RFC 3261, Internet Engineering Task Force (IETF) (2002).
34. TSANG, S., MOYER, S., MARPLES, D., SCHULZRINNE, H., ve ROYCHOWDHURY, A., *SIP Extensions for Communicating with Networked Appliances*, Internet Taslağı draft-tsang-sip-appliances-do-00 (2000).
35. GUTTMAN, E., PERKINS, C., VEIZADES, J. ve DAY, M., Service Location Protocol, Version 2, RFC 2608, Internet Engineering Task Force (IETF) (1999).
36. GUTTMAN, E., PERKINS, C., ve KEMP, J., *Service Templates and Service: Schemes*, RFC 2609, Internet Engineering Task Force (IETF) (1999).

37. FRANKS, J., HALLAM-BAKER, P., HOSTETLER, J., LAWRENCE, S., LEACH, P., LUOTONEN, A., ve STEWART, L., *HTTP authentication: Basic and digest access authentication*, RFC 2617, Internet Engineering Task Force (1999).
38. SCHULZRINNE, H., ve WEDLUND, E., *Application-layer mobility using SIP*, IEEE (2000).
39. KHURANA, S., GURUNG, P., ve DUTTA, A., *Device Message Protocol (DMP): An XML based format for Wide Area Communication with Networked Appliances*, Internet Taslağı, draft-khurana-dmp-appliances-00 (2000).
40. POSTEL, J.B., *Simple Mail Transfer Protocol*, RFC 821, Internet Engineering Task Force (1982).
41. KUROSE, J.F., ve ROSS, K.W., *Computer Networking: A Top Down Approach Featuring The Internet*, Addison Wesley, USA (2003).
42. KLENSIN, J., *Simple Mail Transfer Protocol*, RFC 2821, Internet Engineering Task Force (2001).
43. MYERS, J., ve ROSE, M., *Post Office Protocol – Version 3*, RFC 1939, Internet Engineering Task Force (1996).
44. CRISPIN, M., *Internet Message Access Protocol – Version 4rev1*, RFC 2060, Internet Engineering Task Force (1996).
45. MCKENZIE, A., *Host/Host Protocol for the ARPA Network*, NIC 8246, (1972).
46. CCITT, *Recommendation X.25 - Interface Between Data Terminal Equipment (DTE) and Data Circuit-terminating Equipment (DCE) for Terminals Operating in the Packet Mode on Public Data Networks*, CCITT Orange Book, Vol. VIII.2, International Telephone and Telegraph Consultative Committee, Geneva (1976).
47. MOCKAPETRIS, P., *Domain Names - Implementation and Specification*, STD 13, Internet Engineering Task Force (1987).
48. PSS/SG3, *A Network Independent Transport Service*, Study Group 3, The Post Office PSS Users Group (1980).
49. PARTRIDGE, C., *Mail Routing and The Domain System*, RFC 974, Internet Engineering Task Force (1986).

50. CROCKER, D., FREED, N., ve CARGILLE, A., *SMTP Service Extension for Checkpoint/Restart*, RFC 1845, Internet Engineering Task Force (1995).
51. HOFFMAN, P, *SMTP Service Extension for Secure SMTP over Transport Layer Security*, RFC 3207, Internet Engineering Task Force (2002).
52. MYERS, J., *SMTP Service Extension for Authentication*, RFC 2554, Internet Engineering Task Force (1999).
53. CALLAS, J., DONNERHACKE, L., FINNEY, H. ve THAYER, R., *OpenPGP Message Format*, RFC 2440, Internet Engineering Task Force (1998).
54. RAMSDELL, B., *S/MIME Version 3 Message Specification*, RFC 2633, Internet Engineering Task Force (1999).
55. DIERKS, T. ve ALLEN, C., *The TLS Protocol Version 1.0*, RFC 2246, Internet Engineering Task Force (1999).
56. Sun Microsystems, Inc., *JMF Home Page*, http://java.sun.com/products/java-media/jmf/.
57. Sun Microsystems, Inc., *The Source for Java Technology*, http://java.sun.com
58. Apache Software Foundation, *The Jakarta Site – Apache Tomcat*, http://jakarta.apache.org/tomcat/.

EK-1 SONUÇ KARŞILAŞTIRMA TABLOSU

Gereksinimler \ Protokoller	SNMP	SIP	SMTP
Geniş alan desteği	Destekliyor	Destekliyor	Destekliyor
Güvenilir iletişim desteği	Kısmen destekliyor	Destekliyor	Kısmen destekliyor
Ağ aygıtları konumdan bağımsız şekilde tek (unique) adreslenebilmeli ve bir ağ adresinde birden fazla mantıksal aygıt bulunabilmeli	Desteklemiyor (uzantı tanımı gerekiyor)	Destekliyor	Kısmen destekliyor
Güvenlik, kimlik denetimi ve gizlilik desteği	Yüksek güvenlik ve gizlilik sağlar	Yeterli derecede güvenlik ve gizlilik sağlar	Yeterli derecede güvenlik ve gizlilik sağlar
Aygıt taşınırlık desteği	Desteklemiyor (uzantı tanımı gerekiyor)	Destekliyor	Desteklemiyor (uzantı tanımı gerekiyor)
IPV6 desteği	Destekliyor	Destekliyor	Destekliyor
Esnek ileti yapısına sahip olma	Düşük esneklikte	Oldukça esnek	Oldukça esnek
Ağ aygıt kontrol protokolünün yapısı karmaşık olmamalı	Karmaşık	Basit	Basit
Aygıt, durum, oturum ve olay denetimi	Kısmen destekliyor	Destekliyor	Kısmen destekliyor

EK-2 YEREL UYGULAMA

Uzaktan aygıt denetimi için belirlenmiş ve tanımlanmış olan gereksinimlere bağlı olarak incelenen SNMP, SMTP ve SIP protokollerinden bu iş için en uygununun SIP olduğu görülmüştür.

Bu bölümde SIP'den yararlanılarak geliştirilmiş olan uygulama hakkında bilgiler verilmektedir.

Konu

İnsanlar günlük yaşantılarında yapmaları gereken işleri ileride zamanı geçmeden hatırlayabilmek için unutma riskine karşı bir yerlere kaydetme ihtiyacı duyarlar. Ajanda bu amaçla eskiden beri kullanılan bir araçtır. Kişisel bilgisayar sistemlerinin yaygınlaşması sonucunda bu araç bilgisayar uygulaması olarak karşımıza çıkmaya başlamıştır. Bu uygulamalar defter şeklindeki klasik ajandalardan daha çok işleve, esnekliğe ve kullanışlılığa sahiptirler. Kullanıcı bilgisayarında kurulu olan uygulamayı kullanarak işlerini düzenleyebilir, zamanlayabilir ve zamanı geldiğinde uygulamanın kendisini bilgisayar başında örneğin bir uyarı sesiyle haberdar etmesini isteyebilir.

Aygıt denetimi kapsamında geliştirilmiş olan uygulamanın tasarımında ajanda uygulamasının sadece ajanda işlemlerini gerçekleştirmesi için geliştirilmiş bir ağ aygıtı içerisinde çalışacağı, kullanıcının ise Internet üzerinden bu aygıta erişip kayıt yapıp mevcut kayıtlara erişebilmesi üzerinde durulmuştur.

Çözüm

Ajanda uygulaması bir bölümü ajanda ağ aygıtında sunucu olarak çalışacak, diğer bölümü ise kullanıcı tarafında uzaktan sunucuya erişim için kullanılacak olan bir istemci/sunucu uygulaması olarak geliştirilmiştir. Uygulamayı çalıştıracak olan aygıtın bir ses kayıt/aktarım cihazı olarak da düşünülmesi mümkündür.

Ajanda sunucusuna uzaktan erişimde kullanılacak olan istemci modülün tek başına bir program olarak geliştirilmesi durumunda kullanıcının her gittiği yere bu programı da götürüp kurması gerekecekti. Bu yüzden istemci programın web ara

yüzüne sahip olması gerektiği düşünülmüştür. Programlama dili olarak seçilmiş olan Java'nın applet türü program mantığı bu amaçlara uygun bir ortam sağlamaktadır.

Java appleti olarak geliştirilmiş olan istemci program ile geleneksel uygulama mantığı ile geliştirilmiş olan sunucu program arasındaki aygıt denetim işlemler SIP iletileri kullanılarak gerçekleştirilmiştir. Bunun için SIP'in standart tanımından ve aygıt denetimi için tanımlanmış eklentilerinden yararlanılarak protokolün kısıtlı bir gerçekleştirimi yapılmıştır. Buna göre işlevsel gereksinimlerden aygıt ve oturum denetimlerini gerçekleştirebilmek için "DO" ve "INVITE" iletileri kullanılmıştır. Fakat güvenlik ve kimlik denetimi işlevleri göz ardı edilmiştir. Aktarılan iletileri ayrıştırmak amacı ile bir ayrıştırıcı Java sınıf tanımı bulunmaktadır. Bunun dışında Internet üzerinde ses ve görüntü akışını destekleyen RTP protokolü kullanılarak istemci (UAC) ve sunucu (UAS) arasındaki ses akışlarını gerçekleştirmek üzere bazı sınıf tanımları da yapılmıştır.

Uygulamanın çoklu ortam işlemlerini gerçekleştirebilmesi için Java'nın **JMF (Java Media Framework) API (Application Programming Interface)** [56] olarak adlandırılan çoklu ortam uygulamaları geliştirme ortamından yararlanılmıştır. JMF'nin 2.1.1e sürümü kullanılarak mikrofondan ses yakalama, kaydetme ve uzaktaki bir ortama akışını sağlama işlemleri gerçekleştirilmiştir.

Yazılım Gereksinimleri

Uygulama Java 2 dilinin 1.4.1.02 sürümü kullanılarak Microsoft Windows 2000 ve XP işletim sistemleri üzerinde geliştirilmiştir. Java platformdan bağımsız çalışma imkanı verse de uygulamanın doğru çalışabilmesi için Java VM (virtual machine)'nin ya da **JRE (Java runtime environment)**'nin güncel sürümlerinin istemci ve sunucu aygıtlarda kurulu olması gerekmektedir. Bu uygulamalara Sun Microsystems, Inc.'in web sayfasından [57] erişilebilir.

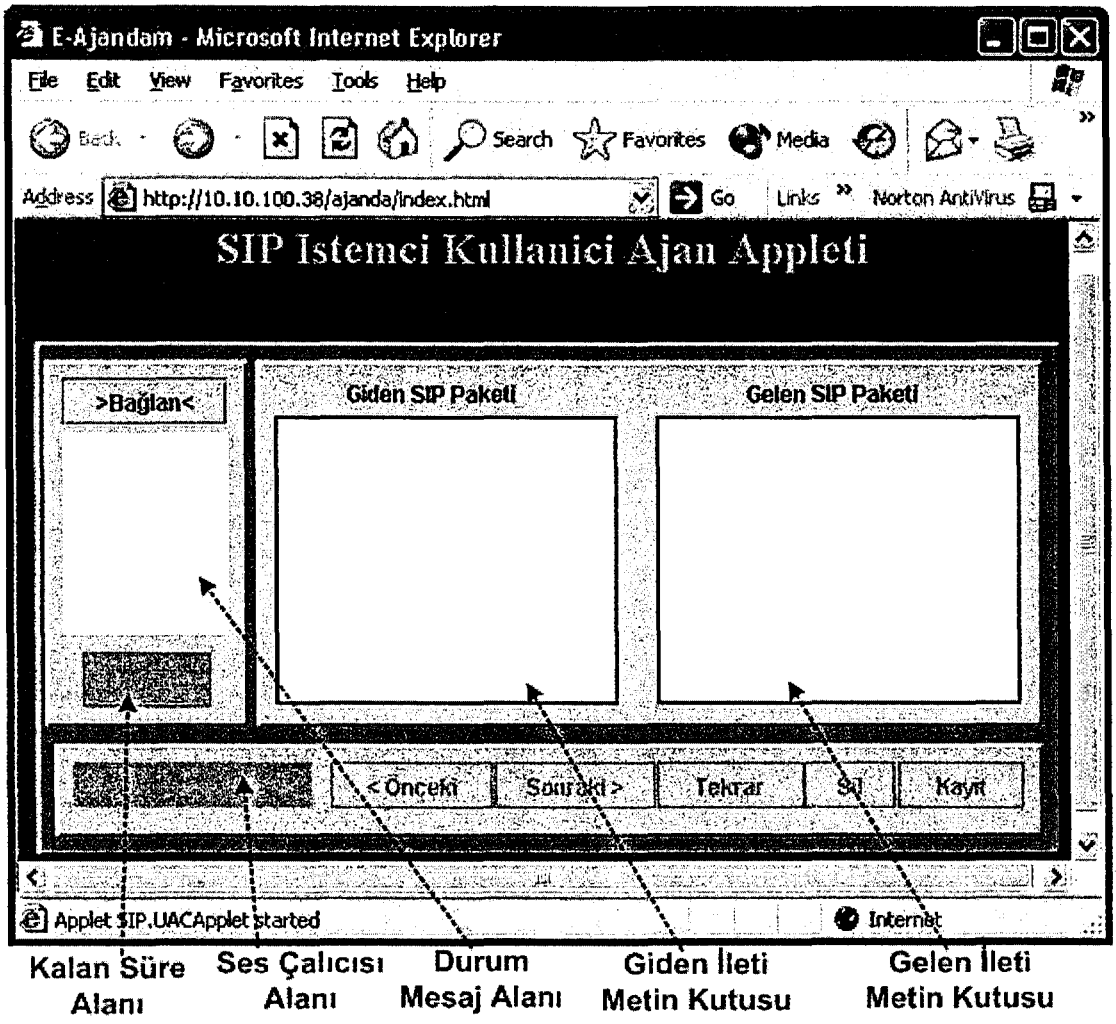
Uygulamanın deneme aşamasında, UAS (sunucu) tarafında tutulacak ve istemcinin web tarayıcısı (Internet Explorer, Netscape ...gibi) aracılığı ile erişebileceği istemci (UAC) ara yüzünün yönetimi Apache Yazılım Lisansı altında geliştirilmekte olan bir web sunucusu olan Tomcat'in [58] 4.1. sürümü kullanılarak

sağlanmıştır. Ancak Java desteği olan diğer web sunucular da bu amaçla kullanılabilir.

İşleyiş

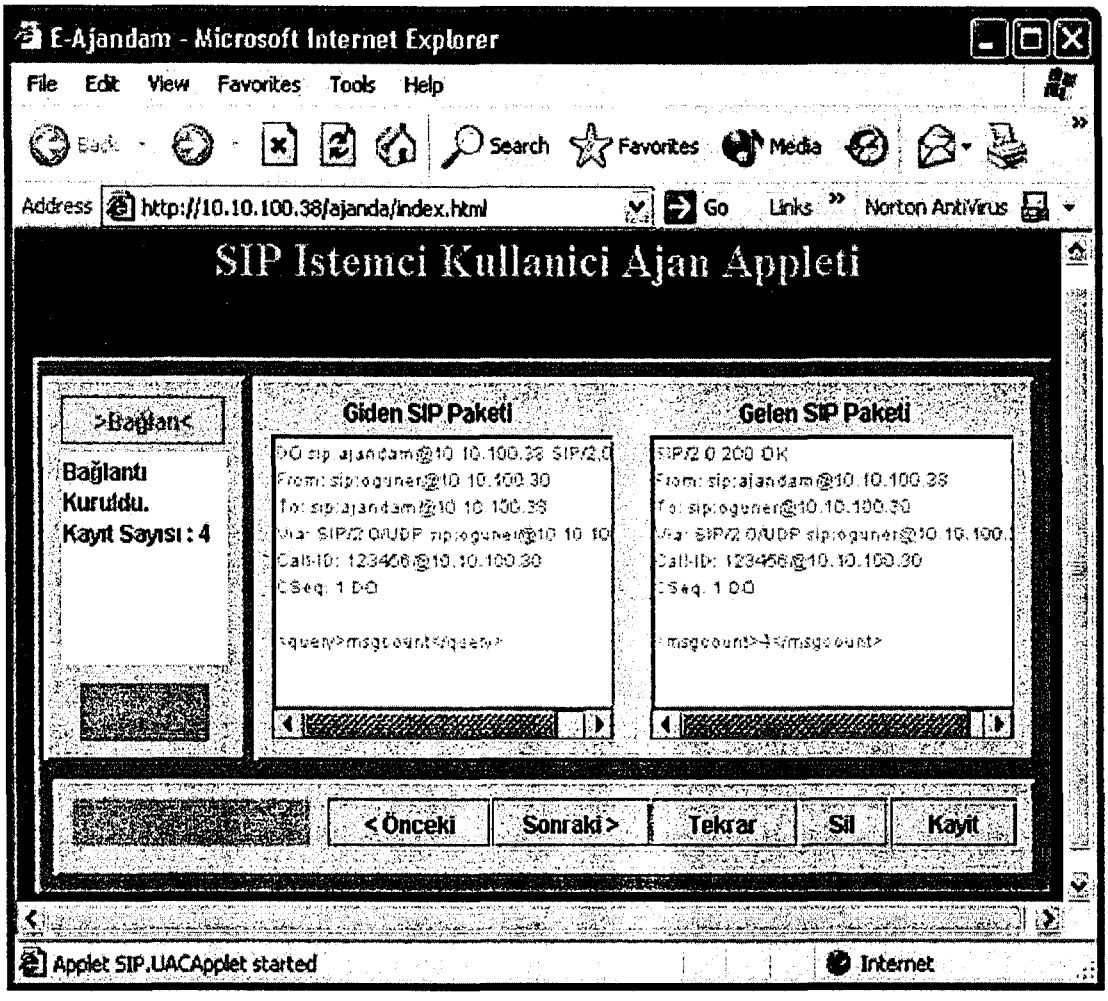
Uygulamayı oluşturan istemci ve sunucu bölümler ayrı ayrı incelenecek olursa:
SIP istemci kullanıcı ajan (UAC):

Ajanda cihazına uzaktan erişimi sağlayan Java appleti kullanıcıya mevcut kayıt sayısını öğrenme, dinleme, istediklerini silme ve yeni kayıtlar ekleme imkanı vermektedir. Şekil 6.1.'de uygulamanın istemci bölümünün Internet Explorer 6.0 tarayıcısı üzerindeki görüntüsü verilmiştir. Sayfa üzerinde çalışan UAC appleti çeşitli düğme ve metin alanları içermektedir. Bunlar sırasıyla:



Şekil 6.1. Ajanda uygulaması, istemci ara yüzü

Bağlan düğmesi: Kullanıcının Ajanda sunucusuna (UAS) bağlantı kurması için kullanılır. Aslında ses akış işlemleri dışında istemci ve sunucu arasındaki tüm SIP ileti aktarımlarında UDP kullanıldığından aktarım katmanı bağlantısından söz edilemez. Burada istemci Şekil 6.2.'de de görüldüğü gibi önce içerisinde *"**msgcount**"* komutu bulunan bir ***SIP DO*** iletisini sunucuya gönderir.



Şekil 6.2. Ajanda uygulaması, sunucu ile bağlantının kurulması

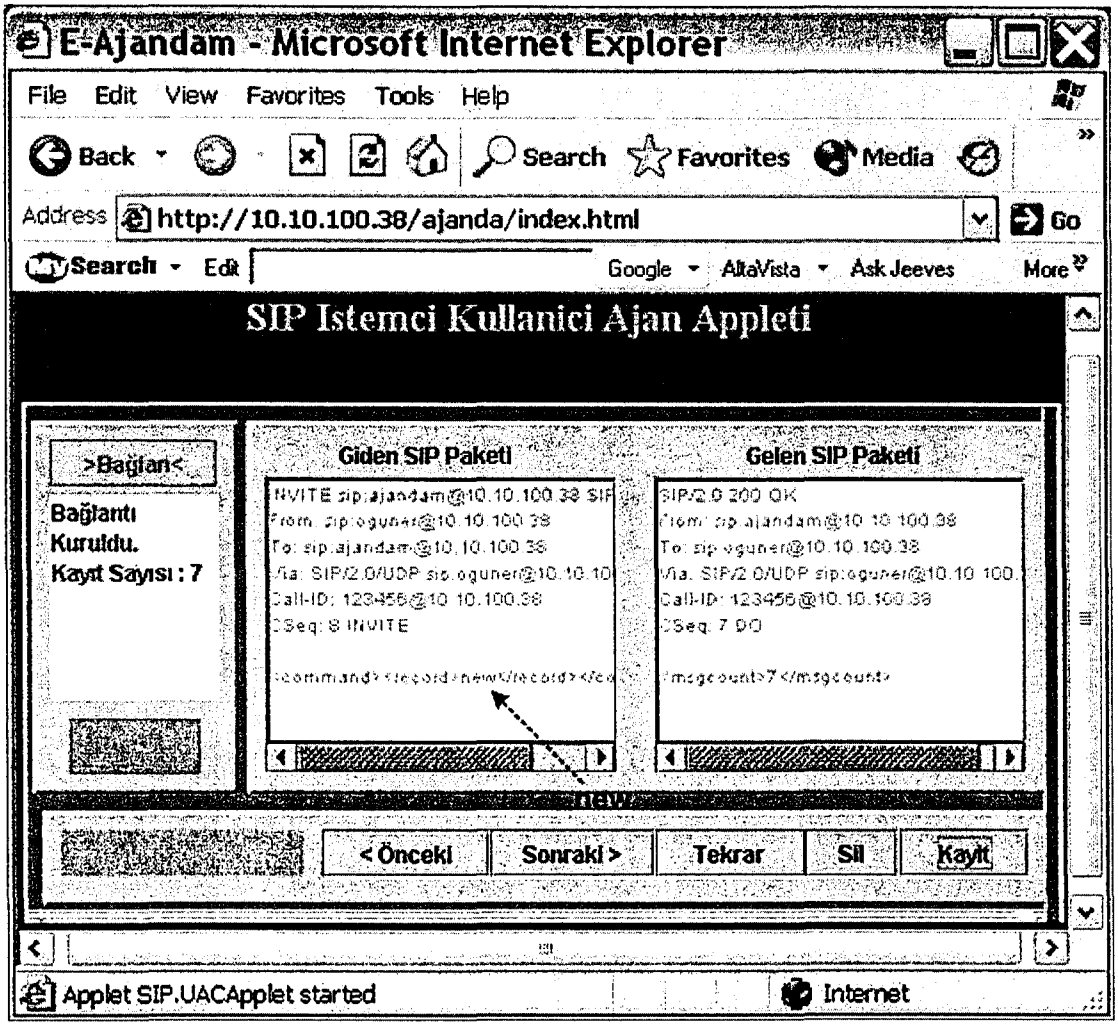
Bu iletiyi alan sunucu istemci ve ileti içeriğini kullanarak istemci hakkında bir sanal oturum kaydı tutar. Bu sayede sunucu hem istemci ile arasında sanal bir bağlantı kurmuş hem de aynı anda birden fazla bağlantıya izin vermemiş olur.

Sunucu kendisine gelen bu ***DO*** iletisine karşılık kayıt işlemlerini yaptıktan sonra sahip olduğu kayıt dosyalarının sayısını *"**msgcount**"* komutuna karşılık istemciye bir **SIP OK** yanıt iletisi içerisinde gönderir. Bu işlemlerden sonra istemci appletin alt bölümündeki düğmeler aktif hale gelir.

Giden SIP paketi metin alanı: Bu alan SIP sunucu ajana gönderilen tüm SIP iletilerinin içeriğini kullanıcıya göstermek amacı ile kullanılmaktadır (Şekil 6.1.).

Gelen SIP paketi metin alanı: Bu alan SIP sunucu ajan tarafından gönderilen tüm SIP yanıt iletilerinin içeriğini kullanıcıya göstermek amacı ile kullanılmaktadır (Şekil 6.1.).

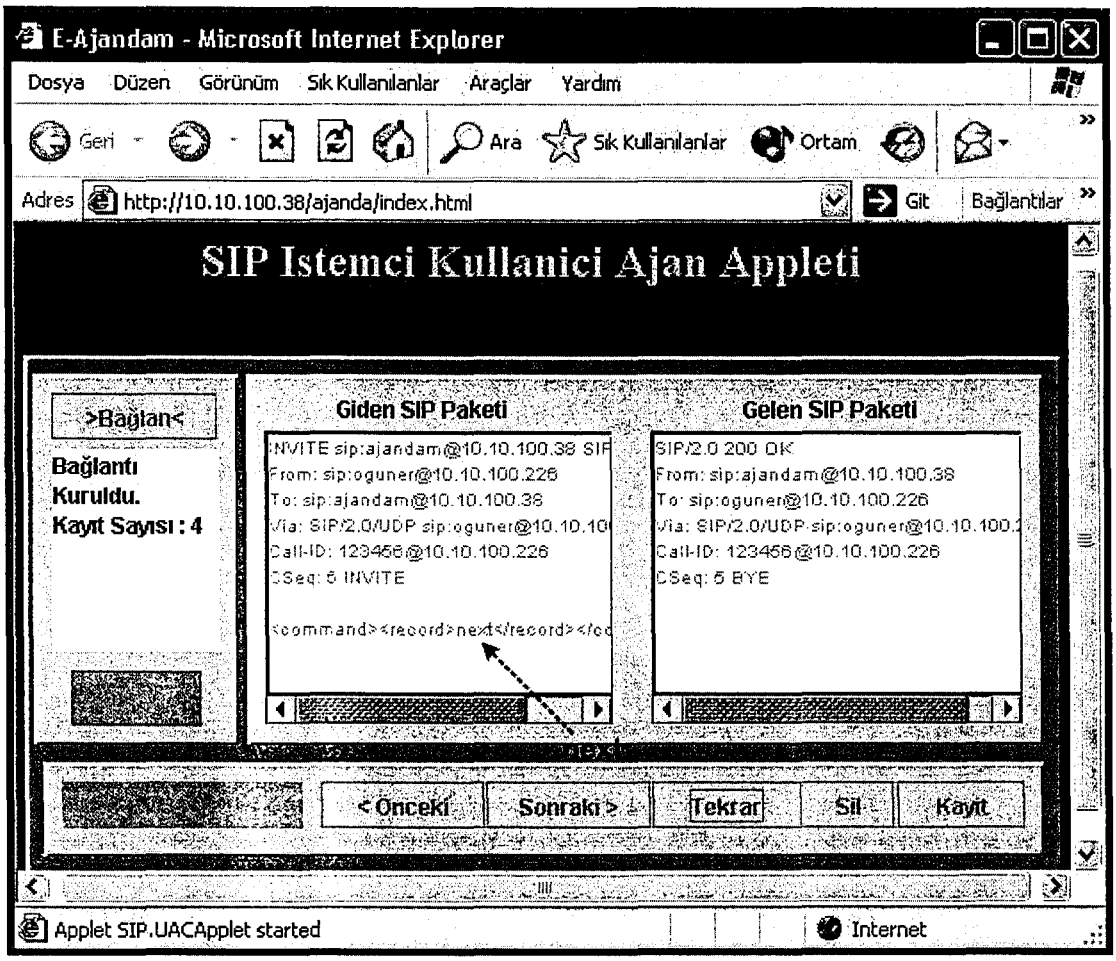
Kalan Süre alanı: Kayıt düğmesine tıklandığında kullanıcıya kayıt işleminde kalan süresinin ondan geriye doğru sayılarak gösterildiği alan (Şekil 6.1. ve 6.3.).



Şekil 6.3. Ajanda uygulaması, Kayıt düğmesine tıklandıkta sonra

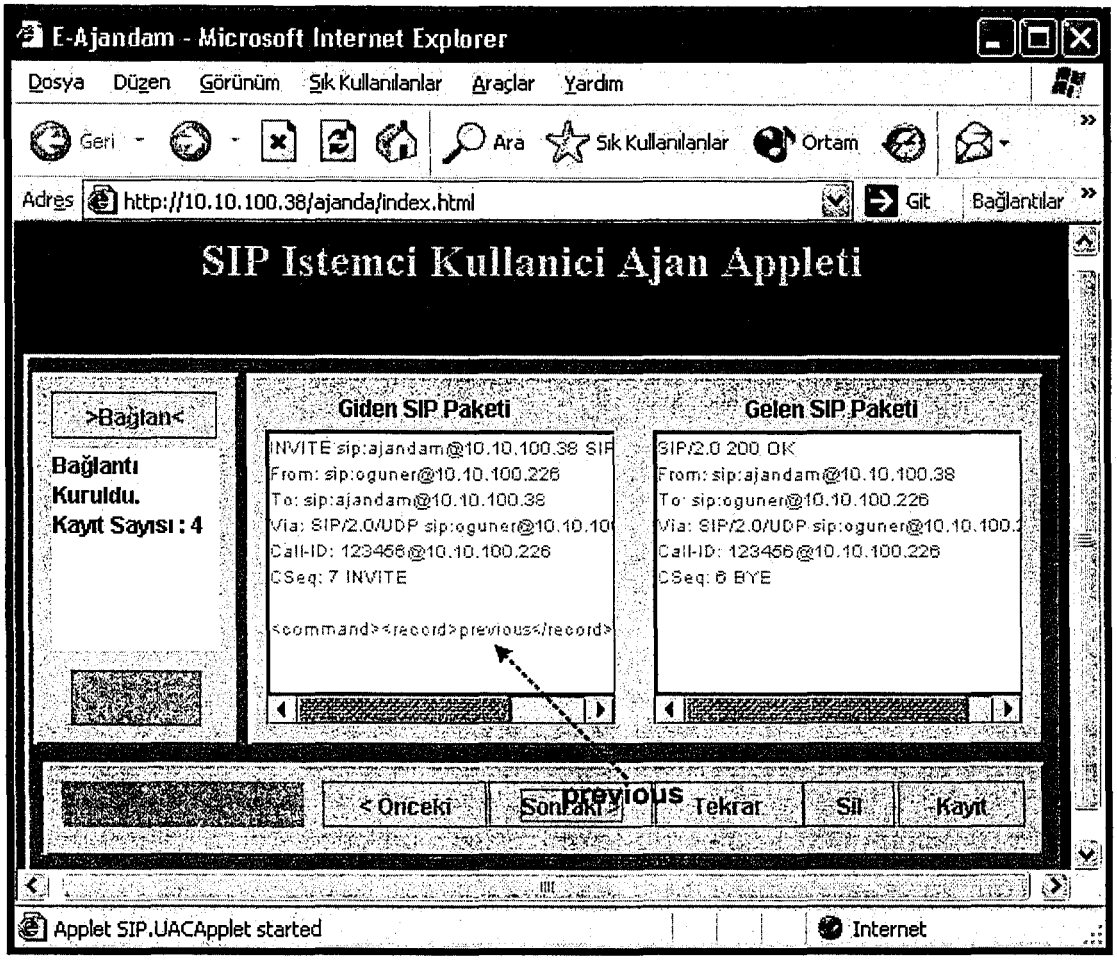
Önceki – Sonraki düğmeleri: Sunucuda tutulan mevcut ses kayıtlarını sıra ile dinlemek için kullanılırlar. *Sonraki* düğmesine tıklandığında sunucuya ***SIP DO*** iletisi içerisinde ***"next"*** komutu (Şekil 6.4.), *Önceki* düğmesine tıklandığında ise ***"previous"*** komutu gönderilir (Şekil 6.5.). Buna karşılık işlemin durumuna yönelik bir yanıt iletisi oluşturulur. İşlem uygulandıysa istemciye ***SIP OK*** iletisi, uygulanamadıysa ***SIP 403 Forbidden*** yanıt iletisi gönderilir. İstemci ise sunucuya ***SIP ACK*** iletisi göndererek yanıt iletisinin kendisine ulaştığını bildirir. (Şekil 6.1.). Bu işlemler sonucunda Ses Çalıcısı alanında (Şekil 6.1.) JMF ses çalıcı nesnesi belirerek ses kaydının kullanıcı tarafından dinlenebilmesi sağlanır.

Kullanıcı sunucuya bağlantı (Bağlan düğmesi ile) kurduktan sonra sunucudaki mevcut kayıtları dinleyebilmek için *Sonraki* düğmesini tıklamak durumundadır.

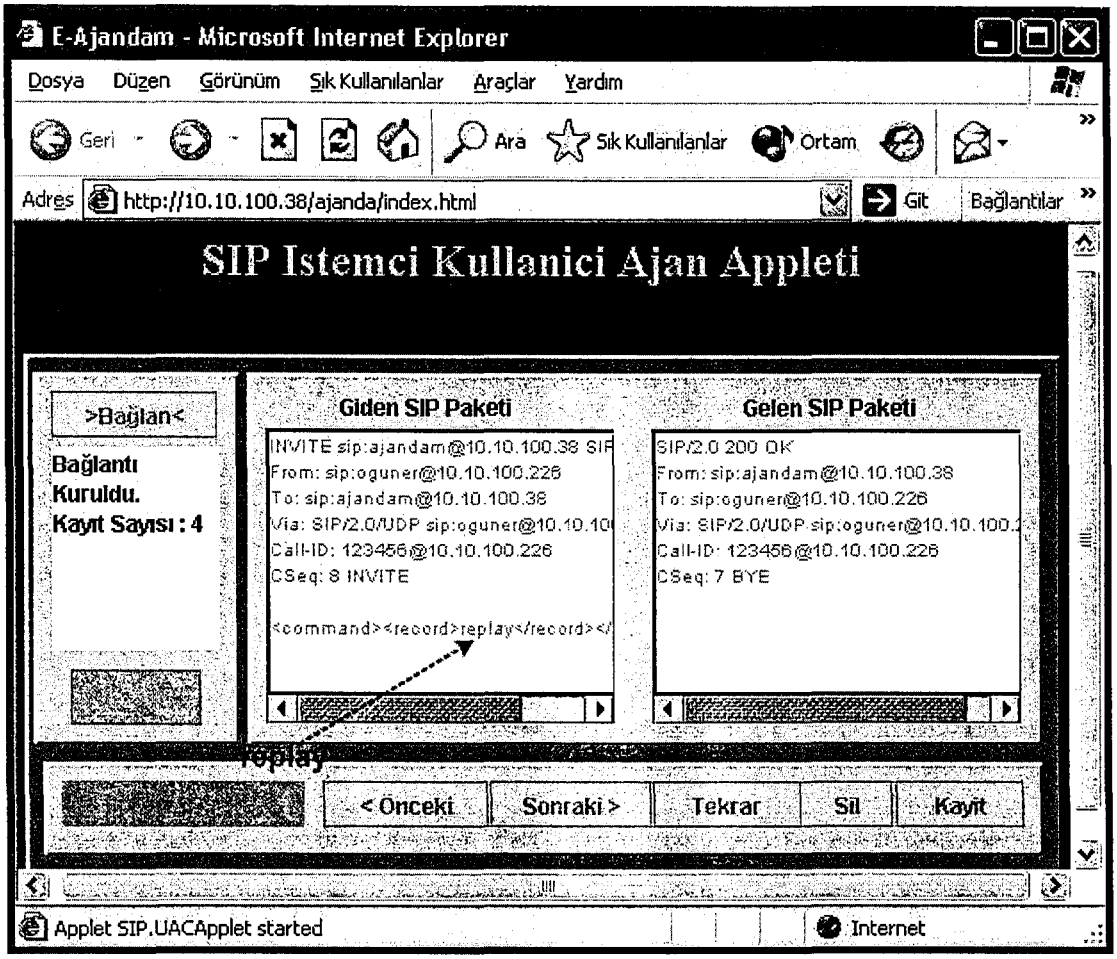


Şekil 6.4. Ajanda uygulaması, Sonraki düğmesine tıklandıkta sonra

Tekrar düğmesi: En son dinlenmiş olan ses kaydını tekrar dinlemek için sunucu ajana istekte bulunmak için kullanılır. Bunun için sunucuya ***SIP DO*** iletisi içerisinde ***"replay"*** komutu gönderilir (Şekil 6.6.). Sonraki işlemler *Önceki* ve *Sonraki* düğmelerini tıklandıktan sonra gerçekleşenlerle aynıdır.

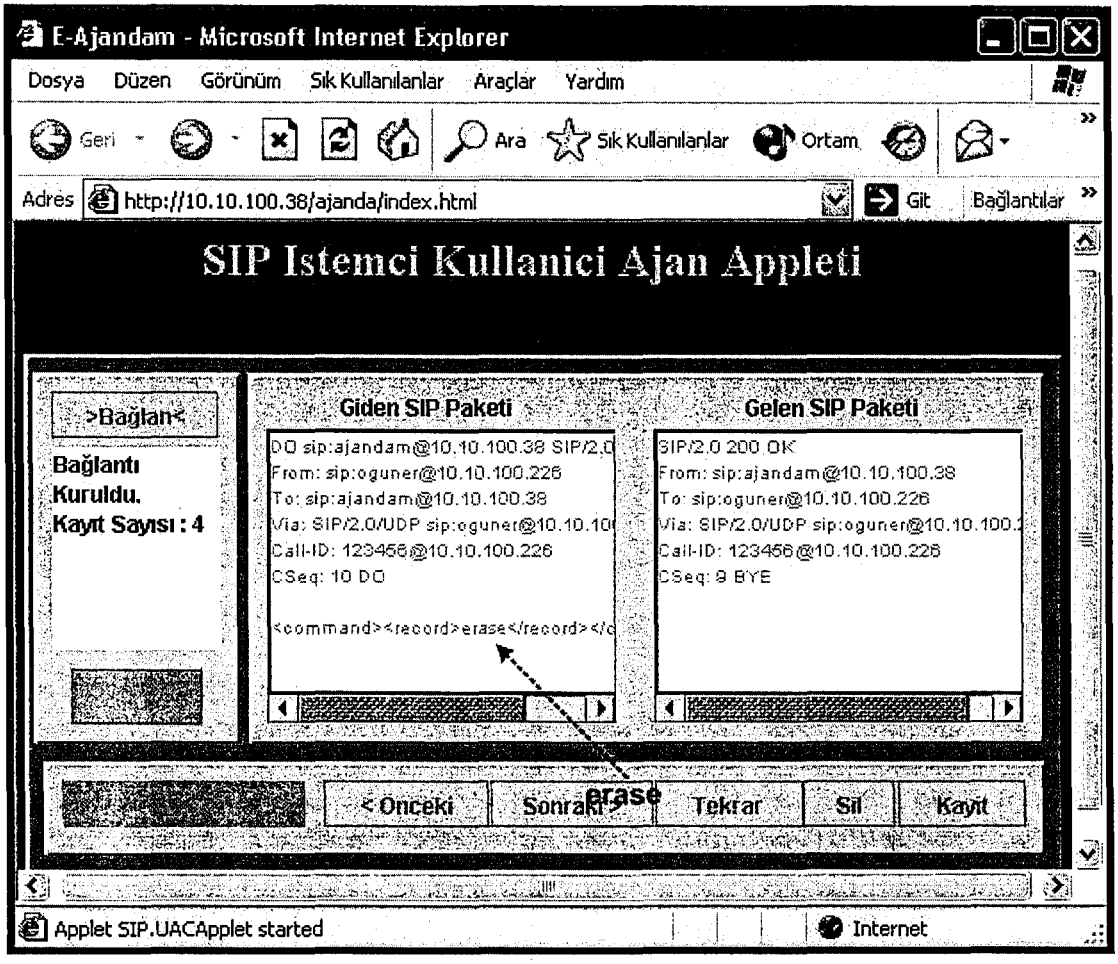


Şekil 6.5. Ajanda uygulaması, Önceki düğmesine tıklandıkta sonra



Şekil 6.6. Ajanda uygulaması, Tekrar düğmesine tıklandıktan sonra

Sil düğmesi: En son dinlenmiş olan kaydın sunucu ajan tarafından silinmesini istemek için kullanılır. Bunun için sunucuya ***SIP DO*** iletisi içerisinde ***"erase"*** komutu gönderilir. Bu komutu alan sunucu en son dinlenmiş olan kaydın dosyasını sabit diskten siler. İşlemin sonucunu bir **SIP OK** yanıt iletisi ile istemciye bildirir (Şekil 6.7.).



Şekil 6.7. Ajanda uygulaması, Sil düğmesine tıklandıktan sonra

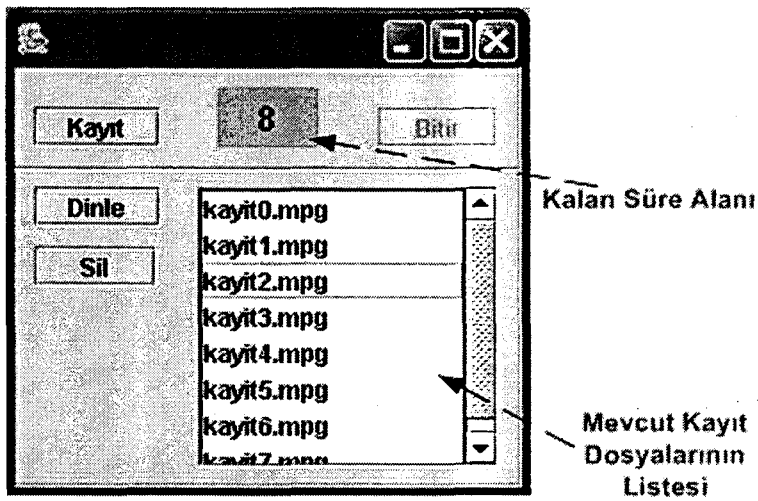
SIP sunucu kullanıcı ajan (UAS):

Sunucu ajanın temel görevi istemci ajandan gelen SIP iletilerini işleyip, geriye yanıt iletileri göndermektir. Bunun yanında kullanıcının sunucuya yerel ortamda da ses kaydı yapabilmesi ve kayıtlar üzerinde işlemler yapabilmesi için basit bir ara yüz oluşturulmuştur (Şekil 6.8.). Sunucunun yarattığı ve üzerinde işlemler yaptığı ses kayıtlarının içerik türü MPEG olarak belirlenmiştir. Kullanıcının en fazla on adet kayıt oluşturabilmesine izin verilmiştir. Kayıtların isimlendirilmesi ise *kayit+sıranumarası.mpg* şeklinde olmaktadır (Şekil 6.8.). Arada silinen kayıtların numaraları daha sonra yapılacak kayıtlarda tekrar kullanılmaktadır.

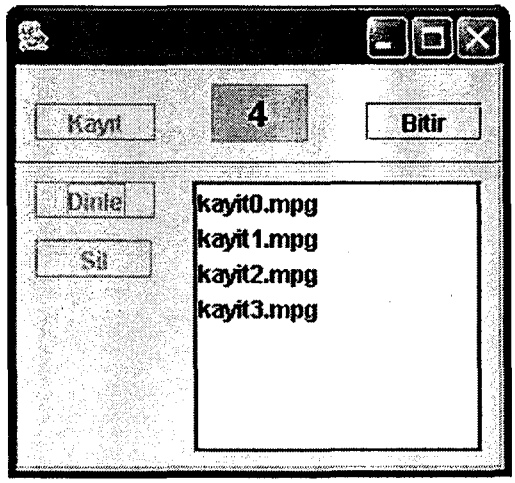
Şekil 6.8.'de görülen alanlar sırası ile:

Kayıt düğmesi: Kullanıcının yerel ortamda ses kaydı yapması için kullanılır (Şekil 6.9.).

Süre alanı: Kayıt düğmesine tıklandığında kullanıcıya, program geliştirilirken test amacı ile belirlenmiş olan 10 saniyelik bir kayıt süresi tanınır. Süre alanı içerisinde ondan geriye doğru sayılarak kullanıcının ne kadar süresi kaldığını görmesi sağlanır (Şekil 6.9.).



Şekil 6.8. Ajanda uygulaması, sunucu ara yüzü

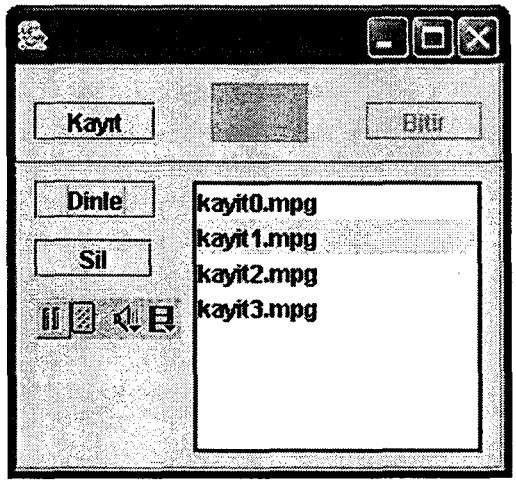


Şekil 6.9. Ajanda uygulaması, Kayıt düğmesi tıklandıktan sonra

Bitir düğmesi: Kullanıcını kendisine tanınan on saniyelik süre bitmeden kayıt işlemini bitirebilmesi için kullanılır (Şekil 6.9.).

Kayıt dosyaları alanı: Bu alanda sunucuda kayıtlı ses dosyaları alt alta kullanıcıya gösterilir.

Dinle düğmesi: Kayıt dosyaları alanında tıklanarak seçilen bir ses kayıt dosyasını dinlenmesi için kullanılır (Şekil 6.10.).



Şekil 6.10. Ajanda uygulaması, Dinle düğmesi tıklandıktan sonra

Sil düğmesi: Kayıt dosyaları alanında tıklanarak seçilen bir ses kayıt dosyasının silinmesi için kullanılır.

EK-3 UYGULAMANIN ALGORİTMASI

İstemci için:

- **Bağlan** düğmesini tıkla ve sunucuya bağlan
- Seçilen düğme,
  - **Sonraki** ise, sunucuya mevcut kayıtlarından,
    a. Bağlantı yeni ise birinci,
    b. Değil ise, bir sonrakini dinleme istek iletisini gönder.
    c. İstenen kayıt mevcutsa gelen ses akışını yürüt.
  - **Önceki** ise, sunucuya mevcut kayıtlardan,
    a. En son dinlenenden bir öncekini dinleme istek iletisini gönder.
    b. İstenen kayıt mevcutsa gelen ses akışını yürüt.
  - **Tekrar** ise, sunucuya mevcut kayıtlardan,
    a. En son dinlenmiş olanı dinleme istek iletisini gönder.
    b. İstenen kayıt mevcutsa gelen ses akışını yürüt.
  - **Sil** ise, sunucuya mevcut kayıtlardan,
    a. En son dinleneni silme istek iletisini gönder.
    b. Silme işlemi sonucunda sunucudan gelen kayıt sayısı bilgisini **Durum Mesaj Alanı** içinde göster.
  - **Kayıt** ise, sunucuya,
    a. Yeni bir kayıt oluşturma istek iletisini gönder.
    b. Yanıt olumlu ise, kullanıcıya kayıt boyunca kalan süreyi **Kalan Süre Alanı** içinde göster.
    c. Kaydı sakla.

Sunucu için:

- İstemciden istek iletisi bekle,
- Gelen istek iletisi,
  - ➢ **next** yöntemini içeriyorsa istemciye mevcut kayıtlarından,
    a. Bağlantı yeni ise birinci,
    d. Değil ise, bir sonrakini dinlemesi için veri akışı şeklinde gönder.
  - ➢ **previous** yöntemini içeriyorsa istemciye mevcut kayıtlarından en son dinlenenden bir öncekini dinlemesi için veri akışı şeklinde gönder.
  - ➢ **replay** yöntemini içeriyorsa istemciye mevcut kayıtlarından en son dinlenmiş olanı dinlemesi için veri akışı şeklinde gönder.
  - ➢ **erase** yöntemini içeriyorsa istemciye mevcut kayıtlarından, en son dinleneni sil.
  - ➢ **record** yöntemini içeriyorsa istemcinin veri akışı şeklinde gönderdiği ses kaydını sakla.
- Seçilen düğme,
  - ➢ **Kayıt** ise,
    a. Kullanıcıya kayıt boyunca kalan süreyi **Kalan Süre Alanı** içinde göster
    b. Kaydı sakla.
  - ➢ **Dinle** ise, **Mevcut Kayıt Dosyaları Listesi**'nden kullanıcının seçtiği kaydı yürüt.
  - ➢ **Sil** ise, **Mevcut Kayıt Dosyaları Listesi**'nden kullanıcının seçtiği kaydı sil.
  - ➢ **Bitir** ise,
    a. Kayıt düğmesi ile başlatılmış olan bir ses kayıt işlemini sonlandır.
    b. Kaydı sakla.