İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

TELEKOMÜNİKASYON YÖNETİM AĞININ GÜVENLİĞİ

YÜKSEL LİSANS TEZİ
İlknur KOÇOĞLU

Anabilim Dalı : ELEKTRONİK VE HABERLEŞME MÜHENDİSLİĞİ

Programı : TELEKOMÜNİKASYON MÜHENDİSLİĞİ

MAYIS 2004

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

TELEKOMÜNİKASYON YÖNETİM AĞININ GÜVENLİĞİ

YÜKSEK LİSANS TEZİ

İlknur KOÇOĞLU
(504021305)

Tezin Enstitüye Verildiği Tarih : 22 Nisan 2004
Tezin Savunulduğu Tarih : 21 Mayıs 2004

Tez Danışmanı : Prof.Dr. Günsel DURUSOY
Diğer Jüri Üyeleri Doç. Dr. M.Ertuğrul ÇELEBİ
Y.Doç.Dr. Demir ÖNER (M.Ü.)

MAYIS 2004

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

TELEKOMÜNİKASYON YÖNETİM AĞININ GÜVENLİĞİ

YÜKSEK LİSANS TEZİ

İlknur KOÇOĞLU

Anabilim Dalı: Elektronik ve Haberleşme Mühendisliği
Programı: Telekomünikasyon Mühendisliği

Tez Danışmanı: Prof. Dr. Günsel DURUSOY

MAYIS 2004

ÖNSÖZ

Bu çalışmada Telekomünikasyon Yönetim Ağının (TMN) Güvenliği konusu incelenmiştir.

Hazırlık aşamasındaki yardımlardan dolayı tez danışmanım sayın Prof. Dr. Günsel DURUSOY 'a ve beni her zaman destekleyen aileme teşekkür ederim.

Mayıs 2004 İlknur KOÇOĞLU

İÇİNDEKİLER

KISALTMALAR

3DES : Triple DES
AARE : ACSE Association Response
AARQ : ACSE Association Request
ABS : Application-Based Security
ACC : Access Control Certificate
ACI : Access Control Information
ACSE : Association Control Service Element
AE : Application Entity
AH : Authentication Header
AM : Accounting Management
AP : Application Process
API : Application Programming Interface
ASCII : American Standard Code for Information Interchange
ASE : Application Service Element
ATIS : Alliance for Telecommunications Industry Solutions
ATM : Asynchronous Transfer Mode
BER : Basic Encoding Rules
BML : Business Management Layer
CBC : Cipher Block Chaining
CF : Control Function
CM : Configuration Management
CMIP : Common Management Information Protocol
CMIS : Common Management Information Service
CMISE : Common Management Information Service Element
CORBA : Common Object Request Broker Architecture
CUG : Closed User Groups
DCN : Data Communications Network
DEA : Data Encryption Algorithm
DER : Distinguished Encoding Rules
DES : Digital Encryption Standard
DIB : Directory Information Base
DSS : Digital Signature Standard
EB : Electronic Bonding
EBCDIC : Extended Binary Coded Decimal Interchange Code
ECB : Electronic Code Book
EDE : Encrypt-Decrypt-Encrypt
EDI : Electronically Data Interchange
EML : Element Management Layer
EPROM : Electronically Programmable Read Only Memory
FM : Fault Management
FU : Functional Unit
FW : Firewall

GIOP : Generic Internet Inter ORB Protocol
GMT : Greenwich Mean Time
GSS-API : Generic Security Service-Application Programming Interface
GULS : Generic Upper Layers Security
IA : Interactive Agent
IC : Inter exchange Carrier
IEEE : Institute of Electrical and Electronics Engineers
IETF : Internet Engineering Task Force
IOP : Inter ORB Protocol
IP : Internet Protocol
IPsec : Internet Protocol Security
ISO : International Organization for Standardization
ITU-T : International Telecommunication Union-Telecommunication Standardization Sector
IV : Initialization Vector
LAN : Local Area Network
LEC : Local Exchange Carrier
MAC : Message Authentication Code
MAF : Management Application Function
MFA : Management Functional Area
MIB : Management Information Base
MIM : Management Information Model
MIT : Management Information Tree
MO : Managed Object
NE : Network Element
NEF : Network Element Function
NEL : Network Element Layer
NML : Network Management Layer
NR : Non-repudiation
OID : Object Identifier
OMG : Object Management Group
ORB : Object Request Broker
OS : Operations System
OSF : Operations System Function
OSI : Open System Interconnection
PC : Personal Computer
PDU : Protocol Data Unit
PIC : Primary Interchange Carrier
PM : Performance Management
QoP : Quality of Protection
ROSE : Remote Operations Service Element
RSA : Rivest Shamir Adelman
SDU : Service Data Unit
SECIOP : Secure Inter ORB Protocol
SG : Security Gateway
SHA : Secure Hashing Algorithm
SM : Security Management
SMASE : System Management Application Service Element
SML : Service Management Layer
SNMP : Simple Network Management Protocol

SONET	: Synchronous Optical Network
SSL3	: Secure Socket Layer version 3
STASE-ROSE	: Security Transformations Application Service Element-ROSE
TA	: Trouble Administration
TCP	: Transport Control Protocol
TeNoRIOP	: Telecommunication Non-Repudiaton Inter ORB Protocol
TL1	: Transaction Language 1
TLS	: Transport Layer Security
WS	: Work Station

TABLO LİSTESİ

ŞEKİL LİSTESİ

ÖZET

TELEKOMÜNİKASYON YÖNETİM AĞININ GÜVENLİĞİ

Bu çalışmada, öncelikle TMN 'in ne olduğu, tarihçesi, yapısal özellikleri ele alınmıştır ve giriş kontrolü, doğrulama gibi güvenlik servisleri, şifreleme gibi güvenlik mekanizmaları ve alarmlar, tetkikler gibi destek mekanizmaları aracılığıyla temel güvenlik konuları ve TMN güvenliği için kullanılan dört ayrı yöntem incelenmiştir. Bu dört yöntem: OSI tabanlı TMN güvenliği, EDI tabanlı TMN güvenliği, CORBA tabanlı TMN güvenliği ve SNMP tabanlı TMN güvenliği yöntemleridir. Bunlar içerisinde OSI tabanlı TMN güvenliği yöntemi, temel OSI kavramlarını kullandığı için oldukça yaygın bir kullanım alanına sahiptir. Ancak, oldukça karmaşıktır. EDI tabanlı TMN güvenliği yöntemi, basit doküman iletimlerinde kullanılmak üzere tasarlanmıştır. CORBA tabanlı TMN güvenliği yöntemi, dağıtılmış işleme ve yazılım yeteneğine sahiptir. Kullanım kolaylığı, düşük maliyeti ve etkili araçları nedeniyle tercih edilir. CORBA 'da çalışan tüm uygulamalar, CORBA 'yı destekleyen platformlarda da çalışır. Ancak OSI tabanlı güvenlik kadar güçlü değildir. SNMP tabanlı TMN güvenliği yöntemi, oldukça basittir. Uzaktan yönetmeye imkan verir ve internet uygulamaları için idealdir.

TELECOMMUNICATIONS MANAGEMENT NETWORK SECURITY

SUMMARY

In this study, first of all TMN ’s meaning, history, and architectural views are considered and by looking at security services like access control, and authentication, and security mechanisms like encryption, and support mechanisms like alarms, and audit trails basic security subjects and the four methods that are used in TMN security are examined. These four methods are: OSI based TMN security, EDI based TMN security, CORBA based TMN security and SNMP based TMN security. Among these OSI based TMN security method has a widespread use because it is based on basic OSI concept. But, this method is fairly complicated. EDI based TMN security method is designed for simple document exchange applications. CORBA based TMN security method supports distributed processing and software portability and. Along with user-friendly, low-cost, effective toolkits, it may well be used in network management. Any application that runs on CORBA can run on any platform that supports CORBA. But this method is not such powerful as OSI based security. SNMP based TMN security method is fairly simple. It supports remote management and is ideal for internet applications.

1. GİRİŞ

Telekomünikasyon Yönetim Ağı (Telecommunication Management Network – TMN), altında uzanan tüm telekomünikasyon ağını (Telecommunication Network – TN) kontrol eden yapıdır. Bu sebeple; TMN 'i kontrol eden kişi, telekomünikasyon ağını da kontrol etmiş olur. TMN 'in çeşitli saldırılara karşı korunması gereklidir, fakat TMN güvenliği ile ilgili konular ancak son birkaç yıldır konuşulmaya başlanmıştır.

Bu çalışmada öncelikle TMN 'in ne olduğu, tarihçesi, yapısal özellikleri ele alınmış ve giriş kontrolü, doğrulama gibi güvenlik servisleri, şifreleme gibi güvenlik mekanizmaları ve alarmlar, tetkikler gibi destek mekanizmaları aracılığıyla temel güvenlik konuları incelenmiştir. Daha sonra TMN güvenliği ile ilgili stratejiler ayrıntılarıyla incelenmiş ve karşılaştırılmıştır. Sonuç bölümünde ise TMN 'in geleceği ve ne yönde gelişmeler göstereceği ele alınmıştır.

2. TMN (TELECOMMUNICATION MANAGEMENT NETWORK)

2.1 TMN'İN TARİHÇESİ

Öncelikle TMN 'in tarihçesi üzerinde durularak, günümüze kadar konunun gelişimi incelenmiştir. Burada basit bir kronolojiden çok temel gelişme adımları ele alınmıştır.

1984 yılında Bell System şirketinin parçalanması ile, yerel santral işletenler (Local Exchange Carriers – LECs), milyonlarca telefon hattı için gerekli donanımı rekabet eden çok sayıdaki tedarikçiden karşılamak imkanı bulmuşlardır. Daha önceleri tek bir şirketle çalışan bu LEC 'ler, bu yeni fırsatı değerlendirmişlerdir. Fakat farklı tedarikçilerin bağlaşma ve iletim sistemleri ile ortak işaretleşme ve iletim protokolleri kullanabilmesine rağmen; İşletim Sistemleri (Operation Systems - OSs) arasında aynı arabağdaşımlar kullanılmamaktadır. Bu durumda LEC 'ler, farklı tedarikçilerin Ağ Öğelerini (Network Elements – NEs) yönetebilmek için, farklı OS 'lere ihtiyaç duymuşlardır. Farklı sistemlerin bir arada kullanılmasıyla maliyet artmıştır ve bu artış esneklik azaltılarak dengelenmeye çalışılmıştır. Sonuç olarak farklı tedarikçilerin OS 'leri haberleşemez olmuşlardır. Oluşan bu birlikte çalışamazlık bulutunu dağıtmak için TMN ortaya çıkmıştır. [1]

Yeni kurulan Alliance for Telecommunication Industry Solutions (ATIS) çatısı altında TMN grubu çalışmalarına başlamıştır. Grubun temel amacı farklı tedarikçilerin OS 'lerinin farklı NE 'ler, OS 'ler ve İş İstasyonları (Work Stations – WSs) ile uyumlu çalışabilmesini sağlamaktır. Araştırma kriterleri:

- Birlikte çalışabilirlik,
- Ağ yönetiminde fonksiyonellik,
- Yerel uygulamalarda özgürlük,
- Endüstri tarafından kabul görme

olup, her biri aşağıda açıklanmıştır.

Birlikte çalışabilirlik, iki haberleşme sisteminin aynı donanımı (örneğin; işlemci), aynı işletim sistemini (örneğin; Unix veya Windows) veya aynı veri gösterim kuralını (örneğin; ASCII veya EBCDIC) kullanma zorunluluğunun olmamasıdır. Aynı zamanda diğer sistemlerin hangi donanıma, işletim sistemine veya veri gösterimine sahip olduğunun bilinmesine de gerek yoktur.

Ağ yönetiminde fonksiyonellik, ağ yönetimi ile ilgili mesajların TMN tarafından sağlanan arabağdaşımlar vasıtasıyla taşınabilmesidir. Bu sayede, ağ yöneticileri için verimli bir yönetim çalışmasında gerekli olan fonksiyonlar oluşturulabilir.

Yerel uygulamalarda özgürlük, bir sistem diğer TMN sistemleriyle haberleşebildiği sürece, o sistemin yapısı hakkında kısıtlamaların bulunmamasıdır. Bu da üreticilerin tecrübe ve yaratıcılıklarıyla daha iyi sistemler oluşturabilmelerini sağlar.

Endüstri tarafından kabul görme, TMN uygulamalarının farklı üreticiler ve servis sağlayıcılar tarafından benimsenmesidir.

TMN uygulamaları için çözüm olmaya aday üç seçenek vardır:

1) Bellcore tarafından ortaya atılan TL1 (Transaction Language 1). Yönetici - aracı arası mesajları ASCII dizilerinden oluşan formatta oluşturur. Birkaç üretici tarafından kabul görmüştür, ancak genel bir kuruluş tarafından üretilmiş açık bir standart olmadığından geniş çevrelerce kabul görmemiştir. Ayrıca yönetim fonksiyonelliği de açık değildir.

2) İnternet mühendislik çalışma kolu (Internet Engineering Task Force – IETF) tarafından ortaya atılan SNMP (Simple Network Management Protocol). SNMP ’nin dizayn amacı küçük çaptaki basit elemanlı ağları kontrol etmektir.

3) Uluslararası Standart Organizasyonu (International Standardization Organization – ISO) tarafından ortaya atılan OSI (Open System Interconnection). OSI TMN ’in temeli olarak görülebilir. Ancak adaptasyon sürecinde OSI uzun bir yolculuğun ilk adımı olmuştur.

TMN uygulamalarının uluslararası kabul görmesi CCITT tarafından (bugünkü adıyla ITU-T) tanınmasıyla başlar. Yerel standart organizasyonları da TMN çalışmalarına katkıda bulunmuştur.

2.2 TMN'İN YAPISAL ÖZELLİKLERİ

TMN, haberleşme ağları ve servisleri için yönetim fonksiyonlarını sağlar ve diğer TMN ağları, servisler, haberleşme ağları ile kendi ağ öğeleri arasında haberleşmeyi düzenler. TMN bu yönetim işlemini katmanlı yapı ile gerçekleştirilir. TMN kavramı ITU-T tarafından haberleşme yönetim ağı için M.3010 da tanımlanmıştır. TMN, OSI yönetim yapısı ile güçlü bir ilişkiye sahiptir ve internet yönetimi için ilişkili kavramları da tanımlamaktadır. TMN 'in tanımlanmış bazı görevleri şöyledir:

- Ses, video, grafiksel şekil, internet verisi vb. iletimi ile ilgili farklı donanım ve yazılımın, sistem bileşenlerinin uzak mesafe yönetimi.

- İhtiyaç duyulan servislerin sağlanması ve düzenlenmesi için müşteriler ile etkileşimli bir arabağdaşım sağlamak. Burada farklı yetenekteki müşteriler ve son kullanıcılar için arabağdaşım sağlanmalıdır.

- Kaynaklar ve son kullanıcılarla ilişkili problemlerin düzeltilmesi için otomatik bir yapı sağlamak. Burada hatanın kendi başına düzeltilmesi ve doğrulanması işlemleri vardır.

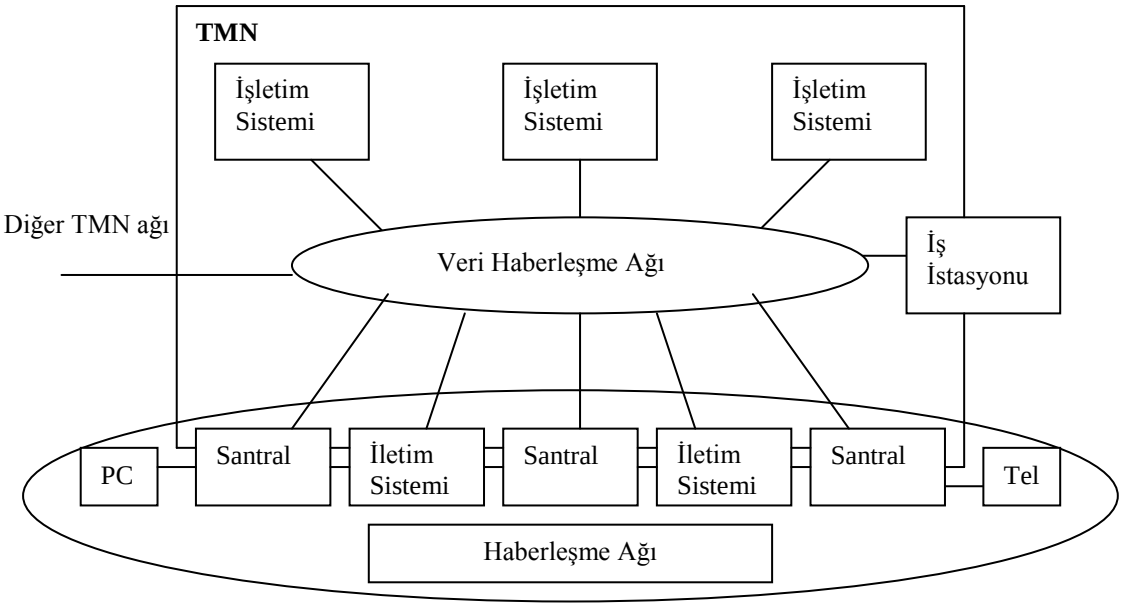
- Yeni donanım ve protokolleri içeren farklı ağlardaki yasal donanımlar ve protokoller için yönetimi ve bağlantıyı sağlamak. Burada güvenli iletişim de söz konusudur.

TMN sadece dijital değil ayrıca analog donanımla da ilişkili yönetim fonksiyonlarından sorumludur. TMN içinde bu ağ öğeleri kaynak olarak adlandırılır ve şunlar olabilir: bir iletim sistemi, bağlaşma sistemi, çoğullayıcı, işaretleşme terminalleri, ön işlemcileri, ana çatı bilgisayarlar, dosya düzenleyiciler vb. TMN uygulama alanları ise çok geniştir. Aşağıda TMN tarafından yönetilebilecek servisler, ağlar ve haberleşme donanımları verilmiştir:

- Ulusal yada özel ağlar {Dar veya geniş bantlı ISDN, ATM, mobil ağlar, özel ses ağları, sanal özel ağlar (Virtual Private Networks - VPN) veya akıllı ağlar (Intelligent Networks - IN) bunlara dahildir}.
- TMN kendisi veya diğer bir TMN ağı parçası.
- İletim terminalleri (Çoğullayıcılar, çapraz bağlantılar, kanal iletim donanımı, SDH vb.)
- Dijital ve analog iletim sistemleri (bakır kablo, fiber, radyo veya uydu ağları.)
- İşletim sistemleri ve onların bileşenleri.
- Dijital ve analog santraller.
- Alan ağları (LAN, WAN, MAN)
- Devre ve paket bağlaşmalı ağlar.
- İşaretleşme terminalleri ve sistemleri .
- PBX , PBX erişim ve kullanıcı terminalleri.
- ISDN kullanıcı terminalleri
- TMN yazılım uygulamaları
- Haberleşme servisleri ilgili yazılımlar. Bağlaşma yazılımı, klasörler, mesaj veri bankaları.
- İlişkili destek birimleri (test modülleri, güç sistemleri, soğutma birimleri, kurulu alarm sistemi vb.)

TMN birçok farklı noktada haberleşme ağı ile arabirimi olan ayrı bir ağdır. TMN ve yönetilen haberleşme ağı arası ilişki Şekil 2.1'de gösterilmiştir. Bu şekle uygun olarak, TMN ve haberleşme ağı arası arabirim noktaları santraller ve iletim sistemleridir. Yönetim için, santral veya iletim sistemi gibi yönetilen ağ öğeleri Veri Haberleşme Ağı (Data Communication Network - DCN) üzerinden bir veya daha

fazla OS ile bağlamıştır. Esas yönetim işlemi komutları OS ’de tanımlanır ve DCN üzerinden uygun mesajın ağ öğesine iletilmesi ile yönetim işlemi gerçekleştirilir. OS ’de tanımlı bir çok fonksiyon vardır. Bu fonksiyonlar operatör tarafından ya da otomatik olarak gerçekleştirilebilir. Buna ek olarak DCN ’ler WS ’lerin ağa erişebilmesini sağlar, bu sayede birden fazla operatör ağ ile ilgili bilgileri görüntüler ve kontrol işlemlerini gerçekleştirebilir.[2]



Şekil 2.1: TMN ve haberleşme ağı arası genel ilişki

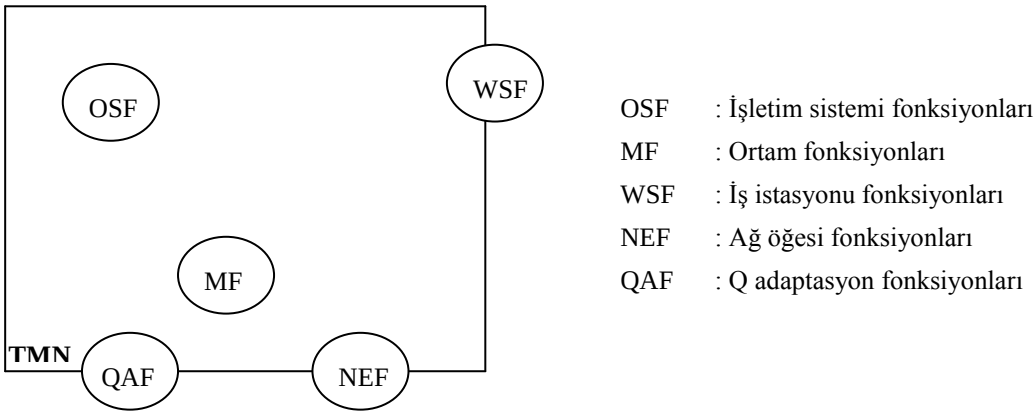
TMN üç ayrı yapısal sistem olarak ele alındığında rahatça anlaşılabilir:

- Fonksiyonel yapı
- Fiziksel yapı
- Haberleşme / bilgi yapısı

Bu üç yapı, TMN ’in neyi, nerede ve nasıl yapacağını anlatır. Bu üç yapı birbirlerinden bağımsız ele alınmalarına rağmen eşzamanlıdır ve bağlantılı olarak çalışırlar.

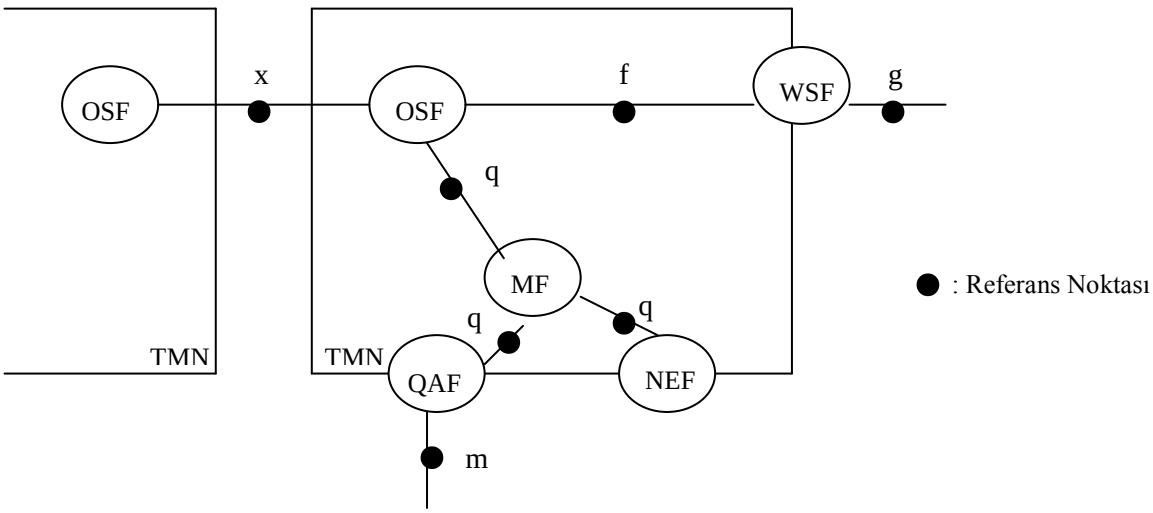
2.2.1 Fonksiyonel Yapı

TMN fonksiyonel yapısı tarafından beş farklı fonksiyon bloğu tanımlanmıştır. Bunların hepsinin her bir TMN yapısı içinde bulunması zorunlu değildir. Diğer taraftan birçok TMN yapısı aynı tipte birçok fonksiyon bloğunu destekler. Şekil 2.2 bu beş fonksiyonel bloğu göstermektedir.[2]



Şekil 2.2: TMN Fonksiyonel Blokları

Bu şekilde iki tip (OSF ve MF) tam olarak TMN içinde tanımlanmışken, diğer üç tipin (WSF, NEF ve QAF) sadece bir kısmı TMN içinde tanımlanmıştır. Bu sebeple TMN 'in kenarında çizilmiştir.



Şekil 2.3: Referans noktalarının yeri

TMN fonksiyonel yapısı fonksiyon blokları arasında referans noktaları tanımlar. Beş farklı referans noktası tanımlanmıştır. Bunların üç tanesi (q, f ve x) tamamen TMN tarafından tanımlanmışken, diğerleri (g ve m) TMN dışında yer alır ve sadece bir kısmı TMN için tanımlanmıştır. Şekil 2.3'te bu referans noktalarının fonksiyon blokları arası yerleri gösterilmiştir. Örneğin; şekilde MF 'ye q arabağdaşımı üzerinden erişilir ve m referans noktası TMN dışından QAF 'ye erişimi sağlar.

Ağ Öğesi Fonksiyonları (NEF - Network Element Functions)

Tipik bir haberleşme ağı, santralleri ve iletim sistemlerini içerir. TMN terminolojisinde santraller ve iletim sistemleri Ağ Öğesi (NE- Network element) olarak adlandırılır. NE tarafından gerçekleştirilen fonksiyonlara ise NEF denir. TMN buna göre;

- Birincil fonksiyonlar (veya haberleşme fonksiyonları), bu fonksiyonlar yönetimin bir parçasıdır ve haberleşme ağı ile kullanıcı arasındaki verinin taşınmasını destekler.

- Yönetim fonksiyonları, bunlar NEF bloğunun bir aracı içinde özel bir görev almasını sağlar. Bu NEF 'in neden TMN 'nin kenarında bulunduğunu açıklar.

İşletim Sistemi Fonksiyonları (OSF - Operations System Functions)

OSF bloğu yönetim işlemini başlatır ve cevapları alır. Yönetici-aracı modeline göre, OSF yönetici tanımlı fonksiyonlar olarak görülebilir. Bir OSF q3 referans noktası üzerinden NEF ile haberleşir.

1988'de tanımlanan M.30'da 3 tane farklı q referans noktası; q1, q2, q3 tanımlanmıştır. q3 referans noktası, yönetim bilgisinin uygulama katmanı yönetim protokolü üzerinden taşınması gerektiğinde kullanılmaktadır. Diğer iki referans noktası ise veri bağı gibi daha düşük seviye yönetim protokolleri katmanı üzerinden yönetim bilgisinin taşınması işleminde kullanılmaktadır. q1 ve q2 referans noktası arası farkları ayırmak imkansızdır bu yüzden bu iki referans noktası qx olarak isimlendirilir. q3 OSI servis ve protokol içeriğine göre tanımlanmıştır.

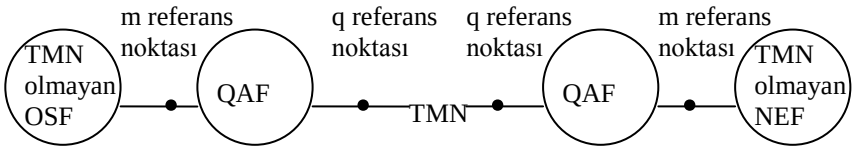
Tek bir TMN içinde (tek bir yönetici tarafından işletilen TMN) bir çok OSF tanımlanabilir. Eğer gerekirse, bu OSF 'ler q3 üzerinden birbirleri ile haberleşebilir. Üstelik farklı TMN yapıları içinde bulunan OSF 'ler de birbirleri ile haberleşebilir. Ama bu durumda haberleşme x referans noktası üzerinden gerçekleştirilir.

İş İstasyonu Fonksiyonları (WSF - Work Station Functions)

WSF, TMN bilgisinin kullanıcının anlayabileceği bir formata dönüştürülmesini sağlar. Yani bir tür grafiksel kullanıcı arabağdaşımı sağlar. Bu işlemde g referans noktası kullanılır ve bu referans noktasının tanımlamaları TMN içeriği dışındadır.

Q Adaptasyon Fonksiyonları (QAF - Q Adaptation Functions)

QAF bloğu, TMN ile TMN tarafından desteklenmeyen varlıklar arası bağlantıyı sağlar. Bir örnek Şekil 2.4'te verilmiştir. TMN olmayan OSF ile TMN olmayan NEF TMN üzerinden bağlanmıştır. Her iki QAF q referans noktası (bunlar TMN referans noktalarıdır) ve m referans noktaları üzerinden işlemlerini gerçekleştirir. m referans noktasının bir kısmı TMN içinde tanımlanmıştır.



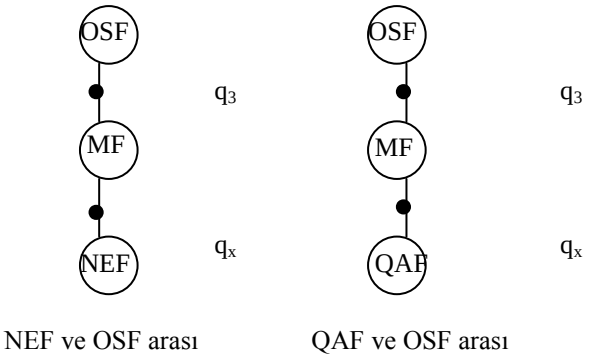
Şekil 2.4: TMN - TMN olmayan bağlantı örneği

Ortam Fonksiyonları (MF - Mediation Function)

MF bloğu TMN içinde bulunmaktadır ve QAF veya NEF 'ler ile OSF 'ler arası bilgi transferi işlemini gerçekleştirir. Şekil 2.5'te gösterildiği gibi, bir MF bloğu bir veya birden fazla NEF veya QAF 'yi OSF 'ye bağlayabilir. Üstelik MF blokları art arda bağlanabilir. MF tipleri:

- OSF 'leri artırmak, örneğin; yönetim bilgisinin depolanması ve filtrelenmesi.

- NEF 'leri artırmak, yönetim bilgisinin yerel sunumundan standart formlara dönüştürülmesi. (Zira her bir NE farklı tedarikçi tarafından üretilmiş olabilir ve farklı tanımlamalara sahip olabilir.)



Şekil 2.5: MF 'nin kullanılması

Fonksiyon Blokları Arası İlişki:

Bütün fonksiyon bloklarının ve referans noktalarının daha iyi anlaşılması için burada bir tablo verilmiştir. (Tablo 2.1)

Tablo 2.1: Referans noktaları ve TMN fonksiyonları arası ilişki

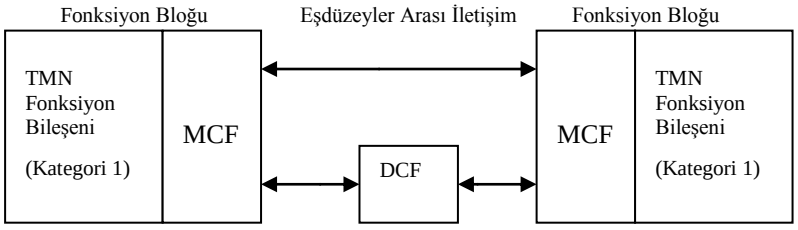
	NEF	OSF	MF	QAF_{q3}	QAF_{qx}	WSF	Non-TMN
NEF		q_3	q_x				
OSF	q_3	x*, q_3	q_3	q_3		f	
MF	q_x	q_3	q_x		q_x	f	
QAF_{q3}		q_3					m
QAF_{qx}			q_x				m
WSF		f	f				g**
Non-TMN				m	m	g**	

m, g = TMN dışı referans noktaları
* =x referans noktası sadece her bir OSF farklı bir TMN içinde ise kullanılır.
** =g referans noktası WSF ile kullanıcı operatör arasında bulunur.

Burada sütunlar üzerindeki fonksiyon blokları yönetim bilgisini sol taraftaki fonksiyon bloğuna, bunların kesişim noktasındaki referans noktası üzerinden iletir. Boş olan yerler ise bilgi taşınmasının direkt olarak yapılmadığını gösterir.

Fonksiyon bloğu ve referans noktaları yanında TMN fonksiyonel yapısı içinde bazı ek yapılar tanımlanmıştır. Bunlar; TMN haberleşme yapıları ve TMN fonksiyon bileşenleridir.

M.3010'a uygun olarak, TMN 'in Veri Haberleşme Fonksiyonları (Data Communication Function - DCF) veri iletimi için fonksiyon blokları tarafından kullanılmaktadır. DCF, OSI referans modelinde 1-3 katmanlar arası fonksiyonları sağlar. Her bir TMN fonksiyon bloğu, fonksiyon bileşenlerinden oluşur. Tanımlanan fonksiyon bileşenleri; 1) Yönetim Uygulama Fonksiyonu, 2) Yönetim Bilgi Temeli, 3) Bilgi Dönüşüm Fonksiyonu, 4) Operatör Adaptasyonu, 5) Sunum Fonksiyonu, 6) Mesaj Haberleşme Fonksiyonu (Message Communication Function - MCF). Bu fonksiyonel bileşenler iki kategoride incelenebilir. Burada ilk beş bileşen ilk grupta yer alır ve gerçek yönetim işlemlerini gerçekleştirir. Son bileşen (MCF) ise ikinci kategoride ele alınır. Bu bileşen bütün fonksiyon bloklarına birleşiktir ve bu fonksiyon bloklarının yönetim bilgisini değiş tokuşu esnasında daha alt seviye fonksiyonları sağlar. Yani MCF fonksiyon bloklarının DCF 'lere bağlanması için gerekli protokol yığınlarını sağlar. Birçok durumda MCF OSI referans modeli 4-7 içinde bulunan noktadan-noktaya fonksiyonları sağlar. M.3010'da fonksiyon blokları, fonksiyon bileşenleri ve DCF arası ilişkiyi tanımlar. Burada bu ilişki Şekil 2.6'da verilmiştir.[2]



Şekil 2.6: DCF 'nin kullanılması

2.2.2 Fiziksel Yapı

Fonksiyonel yapının yanında, TMN ’de fiziksel yapı da tanımlanmıştır. Burada tanımlanan TMN fonksiyonlarının fiziksel donanım içinde gerçeklenmesi esastır. Böylece TMN ’de fiziksel yapı, fonksiyonel yapının bir alt katmanıdır. Gerçekte fiziksel yapı fonksiyon bloklarının ve referans noktalarının nasıl uygulanacağını tanımlar. Fonksiyon bloğu birçok fonksiyonel bileşen içerir ve bir yapı bloğu birçok fonksiyon bloğunu tanımlayabilir.

Yapı Blokları

TMN fiziksel yapısı aşağıdaki yapı bloklarını tanımlar;

- Ağ Öğesi (Network Element - NE),
- Arabulucu Cihaz (Mediation Device - MF),
- Q Uyarlayıcı (Q Adapter - QA),
- İşletim Sistemi (Operation System - OS),
- İş İstasyonu (Work Station - WS),
- Veri Haberleşme Ağı (Data Communication Network - DCN)

1) Ağ Öğesi: Tek bir cihaz, bir grup cihaz veya bir haberleşme donanımının parçası olabilir. NEF fonksiyonları burada gerçekleştirilir. Q arabağdaşımına sahiptir, seçmeli olarak da f ve x arabağdaşımları bulunabilir. TMN ortamına erişim ise Q uyarlayıcı ile gerçekleştirilir.

2) Arabulucu Cihaz: Bir ağ geçidi olarak çalışır. Protokol dönüşümü, mesaj dönüşümü, sinyal dönüşümü, adres çevirme ve yönlendirme fonksiyonlarını gerçekleştirir. Ayrıca, bilgi işleme fonksiyonlarına sahip olmalıdır. Örneğin; veri işleme, veri depolama veya veri filtreleme.

3) Q uyarlayıcı: TMN dışı veriyi TMN veri formatına çevirme ya da tam tersi işlemi gerçekleştirir. Örneğin; TMN dışı bir veri TL1 mesajı olabilir. Bu TL1 mesajları DCN ’e iletilir. Ters yönde ise işlem tam tersine gerçekleştirilir.

4) İşletim Sistemi: Haberleşme ağının işletimi, yönetimi, bakımı ve hazırlanması için gerekli işlemleri destekler. Ağ içinde bir yönetici olarak işlem yapar.

5) İş İstasyonu: Yönetim bilgisine, operatör veya kullanıcıların erişebilmesi için bir kapı veya giriş olarak kullanılır. Bir Unix tabanlı iş istasyonu veya bir Windows NT işletim sistemli PC olabilir. Burada yönetim bilgisinin kullanıcıların anlayabileceği bir formata çevrilmesi gereklidir. Bu bilgi MF veya OS tarafından yollanan bir bilgi veya yerel olarak üretilen bir rapor (alarm) olabilir.

6) Veri Haberleşme ağı: NE-WS, WS-OS, OS-NE ve OS-OS arası yönetim bilgisinin taşınması ve yönlendirilmesi işlemini gerçekleştirir. OSI katmanlı modelinin ilk 3 katmanına ait fonksiyonları gerçekler. DCN ’de veri formatında hiçbir değişiklik yapılmaz. Bu yapıya bir örnek olarak tekrarlayıcılar ele alınabilir. OSI katmanlı modelinde sadece fiziksel katmana ilişkin fonksiyonları gerçekleştirir.

Yapı blokları her zaman tanımlandığı fonksiyon bloğu ile aynı ismi alır. Tek bir yapı bloğu içinde aynı veya farklı tipte birçok fonksiyon bloğunu tamamlayabilir. Örneğin; OS birçok OSF ’yi gerçekleştirirken aynı zamanda OSF, MF ve bir WSF ’yi gerçekleştirebilir. Yapı bloğu birçok fonksiyon bloğunu gerçekleştirmesine rağmen zorunlu olduğu fonksiyon bloğunun ismine göre isimlendirilir. Tablo 2.2’de hangi yapı bloğunun hangi fonksiyon bloğunu gerçekleştirdiği gösterilmiştir.

Tablo 2.2: Yapı bloğu - fonksiyon bloğu ilişkisi

	NEF	MF	QAF	OSF	WSF
NE	M	O	O	O	O^
MD		M	O	O	O
QA			M		
OS		O	O	M	O
WS					M
DCN					

M = Zorunlu
O = Seçimli
O^ = Sadece OSF veya MF varsa mevcut.

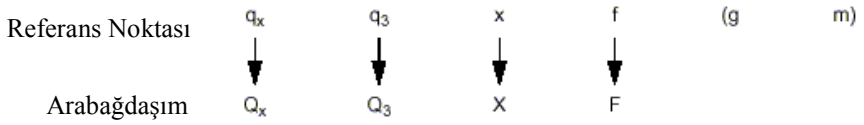
Burada diğerlerinden farklı olan DCN yapı bloğudur. Diğerlerinden farklı olarak, bu yapı bloğu hiçbir TMN fonksiyonel bloğunu gerçekleştirmemektedir. Gerçekte, DCN diğer yapı blokları tarafında yönetim bilgisinin iletilmesinde kullanılır. DCN ’in görevi bir taşıma ağı olmaktır. İlk bakışta TMN tarafından tanımlanan bir yapısal bloğun hiçbir fonksiyonel bloğu gerçekleştirmediğini düşünmek zordur. Fakat bunun

sebebi ilk TMN tanımlamalarında DCF diye ayrı bir fonksiyon bloğunun tanımlanmış olmasıdır. Fakat 1990'da bu fonksiyon bloğu çıkarılmıştır, bu yüzden sanki bu blok hiçbir fonksiyon bloğunu gerçekleştirmiyor gibi görünmektedir.

Arabağdaşımlar

Arabağdaşımlar TMN referans noktalarının fiziksel olarak gerçeklenmesi gibi düşünülebilir. Referans noktaları alt katmandaki servisler ile karşılaştırılırken, arabağdaşımlar bu servisleri gerçekleştiren protokol paketleri ile karşılaştırılabilir. Birçok durumda referans noktaları ve arabağdaşımlar bire bir eşleştirilebilir. Fakat aşağıdaki referans noktaları için bir arabağdaşım yoktur:

- Tek bir yapı bloğu içinde gerçekleştirilen fonksiyon blokları arası,
- TMN dışı tanımlamalar (g ve m). Bu referans noktalarının gerçekleştirilmesi TMN yapısının dışındadır. Arabağdaşımların isimlendirilmesi ilişkili referans noktasına göre yapılır ve Şekil 2.7'de verilmiştir.



Şekil 2.7: TMN arabağdaşımları

Q Arabağdaşımı: q referans noktasında tanımlanmış arabağdaşımdır. Q3 veya Qx arabağdaşımı olabilir. Q3 arabağdaşımı, OSI 'nin yedi katmanının hepsinde kullanılır. Qx arabağdaşımı ise OSI 'nin bütün katmanlarının kullanılmasının gerekli olmadığı koşullarda kullanılır. Zira OSI 'nin yedi katmanın da içerilmesi demek çok fazla bir ek yük demektir. Bu fazlalıktan kurtulmak için Qx kullanılır.

X Arabağdaşımı: X arabağdaşımı, farklı TMN içindeki OS 'leri birbirine bağlama veya bir TMN sistemini, TMN benzeri arabağdaşımlar ile diğer yönetim sistemlerine bağlamak için kullanılır.

F Arabağdaşımı: OSF ve MF iş istasyonlarını DCN üzerinden birbirine bağlamak için kullanılır. Kullanıcılar TMN verisine F arabağdaşımı üzerinden ulaşır.

M Arabağdaşımı: TMN dışında tanımlanmıştır. m referans noktasına karşı düşürülebilir. QAF ile TMN dışı yönetim varlıklarını veya TMN standartlarına uygun olmayan varlıkları sisteme bağlamak için kullanılır.

G arabağdaşımı: g referans noktasına karşı düşürülebilir. TMN 'in bir parçası değildir. G arabağdaşımı iş istasyonunda operatör erişim arabağdaşımı sağlar. Operatörler TMN bilgisine bu arabağdaşım üzerinden erişir.

2.2.3 Haberleşme / Bilgi Yapısı

TMN bilgi yapısı, nesne tabanlı bir yaklaşım kullanır ve OSI 'nin yönetim bilgi modeline dayanır. Bu modele uygun olarak, yönetilen nesnenin görevi yöneticiden gelen işlemleri gerçekleştirmek için uygun özellikleri ve davranışları gerçekleştirmek ve bunun sonucunda bir bildirim mesajını yöneticiye veya yöneticilere göndermektir. OSI referans modelinde bu yönetilen nesnelere aracı ismi verilmişti. TMN 'de de aynı aracı-yönetici yapısı kullanılır, zira OSI 'deki bilgi yapısı aynen TMN 'de de kullanılır. Verinin taşınması Protokol Veri Birimi (Protocol Data Unit - PDU) ile yapılır. Bu taşınan veri bir komut, bir cevap, bir bildirim ya da bir olay raporu olabilir.

OSI referans modelinin amacı farklı yapılardaki sistemlerin standart arabağdaşımlar yardımıyla birlikte çalışabilmesidir. Birlikte çalışabilirlik ve esnekliğin sağlanabilmesi için ISO modeli katmanlar kullanır.

Fiziksel Katman: İki düğüm arasında veri alışverişi için kullanılan gerçek ortamı (elektrik veya optik sinyaller) tanımlar. Örnekleri; T1 elektrik bağlantıları ve OC-3 SONET optik bağlantılardır. Fiziksel katman güvenliği bağın her ucundaki donanım şifreleyiciler ile sağlanır. Bu donanım şifreleyiciler için herhangi bir standart bulunmadığından ve farklı tedarikçiler tarafından üretilen şifreleyiciler birlikte çalışamadığından, bu katmanın güvenliği incelenmemiştir.

Veri Bağı Katmanı: Veri bağı katmanı, tek bir bağ üzerindeki iletim, çerçeveleme ve hata kontrolünden sorumludur. Örnekleri; bir LAN alanındaki Ethernet veya Token Ring veya daha büyük ağlardaki LAP-B ve LAP-D protokolleridir. Veri bağı katmanı güvenliği tüm ağın tek bir bağdan oluştuğu LAN alanlarında tanımlıdır.

TMN uygulamaları çok daha geniş bir alan gerektirdiğinden, bu katmanın güvenliği incelenmemiştir.

Ağ Katmanı: Ağ katmanı, ağ üzerindeki veri iletimini sağlar ve altındaki yapıdan bağımsızdır. Ağ içerisindeki her düğümde yönlendirme, mesajın iletileceği bir sonraki bağı seçme ve o bağ üzerinde veri bağı katmanının servislerini kullanmadan sorumludur. Örnekleri; X.25 ve IP 'dir. Ağ katmanı güvenliği Kapalı Kullanıcı Grupları (X.25 için) ve Ateş Duvarları (IP için) ile sağlanır. Bu yöntemler Güvenlik bölümünde incelenmiştir. Ağ katmanı ayrıca taşıdığı bilgileri Ağ Katmanı Güvenlik Protokolü (OSI için) ve IPsec (IP için) ile korur. Ancak bu yöntemler uç sistemler arasındaki tüm veri akışını korur. TMN uygulamaları daha ayrıntılı koruma gerektirdiğinden bu yöntemler kullanılmaz.

İletim Katmanı: Ağ katmanı her paketin yerine ulaşmasını sağlar ancak paketleri farklı yollarla gönderebilir (değişen trafik yüklerine bağlı olarak). Bu yüzden paketler hedefe gönderildikleri sıralamayla ulaşmayabilir. İletim katmanının bir görevi bu paketleri sıralamaktır. Daha genel olarak, iletim katmanı ağ boyunca uç sistemler arasında güvenli veri transferi yapar. Ayrıca bir sistemdeki farklı uygulamaları tek bir bağlantıda birleştirir. İletim katmanı verinin iletim sırasında bozulmamasını ve gönderildiği sırayla hedefe ulaşmasını sağlar. Örneği; İletim Kontrol Protokolü (Transport Control Protocol – TCP)'dir. IP ile birlikte TCP/IP protokol seti bugünkü internetin de temelidir. İletim katmanı güvenliği doğrulama, bütünsellik ve veri gizliliği gibi işlemleri gerektirir ve bu işlemler Güvenlik bölümünde incelenmiştir.

Üst üç katman kendilerini bağlayan ağdan bağımsız olarak haberleşme fonksiyonlarını yerine getirir.

Oturum Katmanı: Veri iletebileceği boş aralıkları belirler. Çoğu TMN uygulamasında veri karşılıklı eş zamanlı iletilir. Bu yüzden sunum katmanının fazla işi yoktur. Bu katman herhangi bir güvenlik servisini desteklemez.

Sunum Katmanı: TMN ve OSI modeli farklı veri gösterimlerine (örneğin; ASCII ve EBCDIC) sahip sistemler arasındaki iletişimi destekler. Sunum katmanı bu yerel gösterimleri ortak bir gösterime çevirir. Ayrıca sistemlerin bağlantı boyunca hangi veriyi değiştirecekleri konusunda anlaşmalarını sağlar. Bu görevleri içerik yönetimi

ve sözdizimi uyumu fonksiyonları ile sağlar. **İçerik yönetimi** ile hangi verinin değiştirileceği ve hangi gösterimin kullanılacağı seçilir. TMN 'de kullanılan iletim sözdizimi üçlü alanlardan (tip, uzunluk, değer) oluşur. Örneğin müşteri adı gönderilecekse;

- Tip: yazdırılabilir karakter dizisi
- Uzunluk: 21 oktet
- Değer: müşteri adını 21 karakter olarak kodlanmış hali kullanılır.

Sözdizimi uyumu ile kullanılacak iletim sözdizimine ve mesaj kodlama yöntemine karar verilir. Sunum katmanı uygulamadaki veriyi alarak ağda iletilebilecek ve karşı tarafın anlayacağı biçime getirir.

Ayrı Kodlama Kuralları: En sık kullanılan kodlama yöntemi Temel Kodlama Kurallarıdır (Basic Encoding Rules – BER). Bu kural kodlayacağı kümedeki elemanların sıralamasıyla ilgilenmez. Sıralama için de dijital imza eklenebilir. Ancak imzanın eklendiği andaki sıralamanın bozulmaması gereklidir. Bu yüklerden kurtulmak için BER 'e bir takım kurallar eklenmiştir. Sonuçta Ayrı Kodlama Kuralları (Distinguished Encoding Rules – DER) ortaya çıkar. Güvenlik dönüşümleri uygulanan PDU 'larda daha ziyade DER kullanılır.

Uygulama Katmanı: Uygulama katmanı direk olarak kullanıcı uygulamalarına arabağdaşımlık yapar. Bir sistemdeki uygulama ile diğer sistemdeki arasında bağlantı kurar. Uygulamalara sağladığı bazı servisler:

- Bağlantı kurulması ve çözülmesi
- Bağlantı kurulması sırasında eş öğe doğrulaması
- Mesaj korunması
- Çok sayıdaki cevabın ilgili sorular ile eşleştirilmesi
- Rehbere erişim
- Uzak dosya iletimi ve yönetimi

Uygulama katmanını fonksiyonelliği için çeşitli Uygulama Servis Öğelerine (Application Service Elements – ASEs) bölmek pratiktir. ASE 'ler kendi içlerinde yazılım modülleridir. İki sistem arasındaki bağlantıda her ASE farklı fonksiyonlar gerçekleştirir. Kullanıcılarına servisler sağlar ve uzaktaki eş ASE 'ler ile kendine özgü bir protokol kullanarak haberleşir.

Kullanıcılarına sağladığı servisler servis temel öğeleri olarak adlandırılır. Uzaktaki eş ASE 'ye bir PDU gönderir. Alıcı ASE kullanıcısına uyarı gönderir. Bu uyarı ilk ASE kullanıcısının gönderdiği veriyi içerir. Eğer kullanılan protokol cevap vermeyi gerektiriyorsa, Alıcı taraftaki kullanıcı kendi ASE 'sine cevap vermesi için istekte bulunur. ASE karşı tarafa bir PDU gönderir. PDU 'yu alan ilk ASE kullanıcısına bir doğrulama gönderir.

Bir bağlantıda birden çok ASE kullanılabilir. Bu ASE 'lerin yönetimi Kontrol Fonksiyonu (Control Function – CF) ile sağlanır. CF, ASE'lerin birbirleriyle nasıl iletişim kuracağını ve hangi sırayla kullanılacaklarını belirler. Uzaktaki eş sistemle bağlantı kurmaz, yerel bir fonksiyondur.

İki ASE arasındaki iletişimde çeşitli seçenekler olabilir. Fakat birlikte çalışabilirlik için kullanılacak seçenekler üzerinde anlaşılmalıdır. CF, bir uygulamayı destekleyen ASE 'ler ve ASE 'lerin haberleşme protokolleri birlikte bir Uygulama Öğesini (Application Entity – AE) oluşturur.

Bir AE ve ASE üzerinden değiştirilen işleme verisi birlikte bir Uygulama İşlemini (Application Process – AP) oluşturur. Bir AP birkaç AE 'ye sahip olabilir, fakat genellikle tek bir AE kullanılır. Böyle durumlarda bağlantı kurulduğunda AE 'yi tanıtmak gerekli değildir.

AE 'nin amacı bazı uygulamalar için gerekli bilgileri iletmektir. Örneğin AE içindeki ASE 'lerden biri ağdaki sorunlarla ilgili veri toplayabilir. Bu verinin analizi ile gerçek neden bulunabilir. Fakat genellikle bu işlem için daha aşağı katmandaki bir yazılım kullanılabilir. Uygulama Programlama Arabağdaşımı (Application Programming Interface – API) kullanarak diğer katmanları kullanan uygulamalar, verinin nasıl değiştirildiğiyle ilgilenmezler. Bu yüzden ağ yönetiminin yedi katmanın üzerinde yer aldığı kabul edilir.[1]

TMN ’de kullanılan ASE ’ler:

Bağlantı Kontrol Servis Öğesi (Association Control Service Element – ACSE)

Uygulamalar arasındaki bağlantıların kurulması ve çözülmesini sağlar. İki tarafın kendilerini tanıtmasını ve kullanacakları ASE setini belirlemesini sağlar. Bu tanıtma arayan ve aranan AP ve AE ’nin adlarının ve tercihe bağlı olarak bilgilerinin değiştirilmesi ile yapılabilir. ACSE ayrıca bağlantı verisini değiştirmeyi sağlayan bir fonksiyonel üniteye de sahip olabilir.

Uzak Operasyonlar Servis Öğesi (Remote Operations Service Element – ROSE)

Karmaşık bir sorgulamanın pek çok cevabı olabilir. Örneğin son bir ayda iletim tablosundaki bilgileri güncelleyen kullanıcıların adları istendiğinde pek çok isim cevap olarak gelebilir. Bu cevaplar gelirken, isteği yapan öğe farklı sorgulamalara geçmiş olabilir. Bu nedenle gelen cevaplar çakışabilir. ROSE bu karışıklığı gidermek için ortaya çıkmıştır.

ROSE her isteğe bir istek kimliği ekler ve ilgili cevaplar da aynı istek kimliğini taşır. ROSE kullanıcısı bu kimliği kullanarak istek ve cevapları eşleştirir.

ROSE için Güvenlik Dönüşümleri Uygulama Servis Öğesi (Security Transformations Application Service Element for ROSE – STASE–ROSE)

ROSE PDU ’larının güvenliğini sağlar. ROSE PDU ’ları üzerinde bir takım güvenlik dönüşümleri yaparak bütünsellik, gizlilik ve inkar etmeme gibi güvenlikleri sağlar. Ayrıca güvenlik parametrelerinin kararlaştırılması sırasında eş öğeleri doğrular.

X.500 Rehberi Kullanıcı Aracısı

Uygulamaların X.500 rehberine, iletişim yapacakları sistemler hakkında veri almak için (örneğin; isim ve adres bilgileri) erişimini sağlar.

Ortak Yönetim Bilgi Servis Öğesi (Common Management Information Service Element – CMISE)

İletim servislerinin ağ yönetimi için kullanılmalarını sağlar. Karmaşık bir NE ’nin (örneğin bir merkez ofisi) yönetilmesinde iki sistemin birlikte çalışması için NE ’yi,

elemanlarını ve her elemanında hangi yönetim işlemlerinin gerçekleştiğini ortak olarak görmeleri gerekir. Bir Yönetim Bilgi Modeli (Management Information Model – MIM) Yönetilen Nesne (Managed Object – MO) kümelerinden oluşur. MO kaynağı tanımlayan (örneğin; seri numarası, tarih, durum), uyarılar oluşturan (örneğin; yetkisiz kişiler değerleri değiştirmeye kalkıştığında güvenlik alarmı üretme) ve görevleri yerine getiren (örneğin; iletim hattını 2 kHz'lik taşıyıcı ile test edip sonuçları raporlama) bir yapıdır.

Her MO kendi içinde başka MO 'lar içerir. Bu yapı ağaç yapısına benzer. Yönetilen sistem kökte olmak üzere değişik görevlere sahip MO 'lar ağacın dallarını oluşturur. Bu ağaca Yönetim Bilgi Ağacı (Management Information Tree – MIT) denir. Bir MIT çeşitli kural kümeleri üzerine oluşturulabilir.

CMISE diğer ASE 'ler gibi servis ve protokol tanımına sahiptir. Bunlar Ortak Yönetim Bilgi Servisi (Common Management Information Service – CMIS) ve Ortak Yönetim Bilgi Protokolüdür (Common Management Information Protocol – CMIP). CMISE servisleri istek, cevap ve bildirilerden oluşur. Her servis karşılık gelen CMIP PDU 'larından oluşur. Her istek-cevap servisi bir CMIP istek PDU 'su ve sıfır veya daha fazla cevap PDU 'sundan oluşur. CMIP istek PDU 'larından oluşan CMISE istekleri:

- Oluştur
- Sil
- Ayarla (yaz)
- Al (oku)
- İptal et
- Uygula

CMIP istek ve bildiri doğrulama PDU 'larını gönderenler yönetici, CMIP cevap ve bildiri PDU 'larını gönderenler aracı olarak adlandırılır. Bir yönetici sistem sadece yönetici PDU 'larını gönderir ve aracı PDU 'larını alır. Bir aracı sistem ise sadece aracı PDU 'larını gönderir ve yönetici PDU 'larını alır.

CMISE bağlantı kurulumu için ACSE ’yi, veri değişimlerinde ROSE ’u kullanır.

Sistem Yönetim Uygulama Servis Öğesi (System Management Application Service Element – SMASE)

Yönetilen sistemin kontrol edilmesi için servis sunar. Bu işlem için CMISE servislerini kullanır. SMASE çeşitli yönetim fonksiyonlarından oluşur. Her yönetim fonksiyonu da bir veya daha fazla Fonksiyonel Birim (Functional Unit – FU) içerir. Belli bir arabağdaşım için gerekli olmayan FU ’lar desteklenmeyebilir. Bağlantı kurulumu esnasında hangi FU ’ların kullanılacağını kararlaştırmak için ACSE kullanılır.

Elektronik Doküman Değişimi (Electronic Document Interchange – EDI)

CMISE ağ yönetimi için gerekli tüm fonksiyonları gerçekleştirmesine rağmen, basit bir uygulama için fazla gelişmiştir. Örneğin; TMN ağı servis sağlayıcısı ile X arabağdaşımı üzerinden haberleşen ve sadece servis emirleri ileten küçük çaptaki bir kullanıcı için bir doküman değişimi protokolü yeterlidir. EDI bu amaçla kullanılır. EDI her dokümanın yapısını belirler ve birkaç dokümanı büyük bir tanenin içinde taşıyabilir.

EDI tabanlı uygulama bir mesaj göndermek istediğinde, önce EDI çeviriciye gönderir. Burada EDI standardına ve anlaşılan şekle getirilen mesaj alıcıya elektronik posta yoluyla veya direk olarak iletim katmanından gönderilir. Çoğu EDI uygulaması iletim için TCP/IP kullanır.

EDI mesajları için elektronik mail kullanmak çoğu ağ yönetimi uygulaması için oldukça yavaştır. EDI mesajlarının direk TCP/IP üzerinden gönderilebilmeleri için EDI çeviricinin, TCP bağlantılarını gerektiğinde kuran ve koparan bir yapıya gereksinimi vardır. Bu servisler Etkileşimli Aracı (Interactive Agent – IA) ile sağlanır. IA akış kontrolü için kullanılır. Alıcının taşma ve hata durumunu veya geçici olarak akışı durdurma isteğini iletir. IA ayrıca EDI mesajlarının güvenlik servislerini de sağlar.

Ortak Nesne İstek Aracı Yapısı (Common Object Request Broker Architecture – CORBA)

CORBA dağıtılmış işleme ve yazılım yeteneğini destekler. CORBA ağ yönetimindeki kullanım kolaylığı, düşük maliyeti ve etkili araçları sayesinde yaygınlaşmıştır.

CORBA bir örnekle anlatılırsa; bir kullanıcı büyük boyuttaki bir dosyayı sıralamak ister ve bir sıralama programı çağırır. Çoğu durumda bu işlem aynı bilgisayarda gerçekleşir. Bazı durumlarda ise bu sıralamayı yapacak program kullanıcı bilgisayarında yoktur veya dosya orada sıralanabilmek için çok büyüktür. Bu durumda kullanıcı bu işlemi yapacak başka bir bilgisayar bulur ve dosyayı ve gerekli veriyi sunucuya iletir. CORBA bu durumda devreye girer ve kullanıcının bu işlemle uğraşmasına gerek kalmaz. Dosya CORBA tarafından başka bir bilgisayarda sıralansa bile kullanıcı kendi bilgisayarında sıralandığını sanar.

CORBA bu işlemler için iki API bir de arabağdaşım protokolü kullanır. Kullanıcı tarafındaki API gerektiğinde çağırılan bir yapıdır. Dosya sıralayan kod yerine, bu işi yapacak programı aramaya yarayan kod bulunur. Daha sonra İç Nesne İstek Aracı Protokolü (Inter Object Request Broker {ORB} Protocol – IOP) kullanarak dosyayı sunucuya iletir. Sunucu tarafındaki API bir iskelettir. Belirtilen parametrelerle sıralamayı yapacak sistemi bulur, sonucu bekler ve müşteri ORB 'a iletir. CORBA ağ yönetiminden ziyade dağıtılmış işleme için dizayn edildiğinden CMIP 'daki tüm özelliklere sahip değildir.

CORBA 'nın OSI 'de bulunmayan en büyük avantajı yazılım taşınabilirliğidir. Bu sebeple CORBA 'da çalışan tüm uygulamalar, CORBA 'yı destekleyen platformlarda da çalışır. Fakat CORBA birlikte çalışabilirlik konusunda daha karmaşıktır. Her üretici, CORBA yapısına dayanan ve CORBA API 'lerini destekleyen, kendi ürün grubunu üretebilir ancak IOP markalı olmalıdır. Bu dezavantajı ortadan kaldırmak için Nesne Yönetim Grubu (Object Management Group – OMG) Genel IOP 'u (Generic IOP – GIOP) belirlemiştir. Altında yer alan iletim protokolünden bağımsız olduğu için genel bir yapıya sahiptir.

CORBA iletişimlerini korumak için güvenlik dönüşümleri gerektiğinde GIOP mesajlarında uygulanır. Korunmuş GIOP mesajlarının bir protokol yardımıyla taşınması gereklidir. Bu protokole Güvenli IOP (Secure IOP – SECIOP) adı verilir.

İletim katmanı güvenliği Güvenli Priz Katmanı versiyon 3 (Secure Socket Layer version 3 – SSL3) kullanan CORBA ile sağlanır.

Basit Ağ Yönetim Protokolü (Simple Network Management Protocol – SNMP)

İnternet yaygınlaşmaya başladıktan sonra onun yayılmış parçalarının kontrolü için bir yönetim protokolü gerekli olmaya başlamıştır. CMIP o zamanlar çok yeni bir program olmasına rağmen bu kullanım için oldukça karmaşık bir yapıya sahiptir. Boşluğu doldurmak için SNMP ortaya atılmıştır.

Yöneticiden aracıya giden üç çeşit PDU vardır: isteği al, sonraki isteği al, isteği ayarla. Aracıdan yöneticiye giden iki çeşit PDU vardır: isteği al ve kapan.

Her PDU hangi SNMP versiyonunun kullanıldığını içeren ifadeyle başlar. Daha sonra kullanıcı girişi ifadesi yer alır. Bir etiket de PDU ’nun çeşidini belirtir.

SNMPv2 yönetim bilgi yapısında, protokol operasyonlarında ve güvenlikte yenilikler getirmiştir.

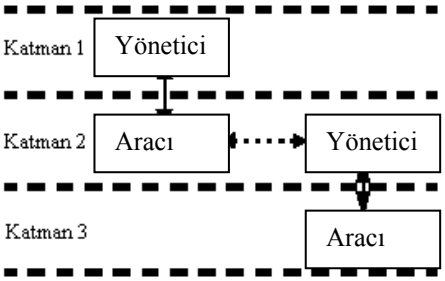
SNMPv3, SNMPv1’in yetersizliğini ve SNMPv2’nin getirdiği yükü ortadan kaldırmak için ortaya çıkmıştır. SNMP Güvenliği konusu Güvenlik bölümünde incelenmiştir.

2.2.4 Lojik Katmanlanmış Yapı (Logical Layered Architecture - LLA)

TMN ’de de insanlar arasındaki yönetim işlemindeki hiyerarşik yapıya benzer bir yaklaşım kullanılmıştır. Bu hiyerarşik yapı yönetim katmanları ile tanımlanmıştır ve bu katmanları tanımlayan yapıya LLA denir. Yönetim katmanları kavramı TMN ’in en önemli kavramı olmuştur.

Burada amaç katmanlı yapı ile yönetimin mevcut karmaşıklığını azaltmaktır. Burada bir katmanlama yapısı Şekil 2.8’de verilmiştir. Katman 1 ve katman 2 arasındaki sınırda katman 2’nin yönetim işlemi katman 1’e verilmiştir. Katman 1’deki yönetim işlemleri için katman 2’deki yönetim bilgisinin tamamına ihtiyaç yoktur. Katman

2’deki aracı katman 1’de ihtiyaç duyulan yönetim bilgisini üst katmana iletir. Bu prensip diğer katmanlar için de aynıdır. Yani katman 3 için de katman 2 aynı anlamı ifade eder.[2]

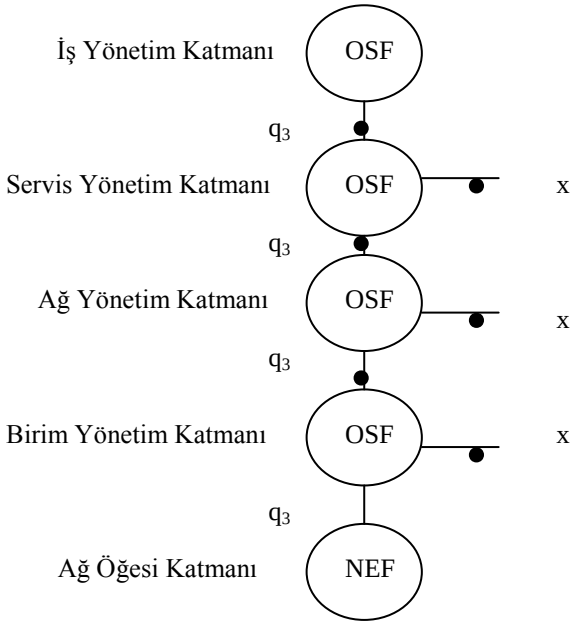


Şekil 2.8 : LLA Yapısı

TMN içinde aşağıdaki katmanlar tanımlanmıştır:

- Ağ öğesi katmanı (Network Element Layer – NEL)
- Birim yönetim katmanı (Element Management Layer – EML)
- Ağ yönetim katmanı (Network Management Layer – NML)
- Servis yönetim katmanı (Service Management Layer – SML)
- İş yönetim katmanı (Business Management Layer - BML)

Bu katmanlar referans noktaları ile beraber Şekil 2.9’da verilmiştir.



Şekil 2.9: TMN 'in katmanlı yapısı

Birim Yönetim Katmanı

Özel bir ağ öğesinin fonksiyonları OSF tarafından birim yönetim katmanında yönetilir. Bu katman üretici tanımlı yönetim fonksiyonlarını kontrol eder ve üstteki ağ yönetim katmanından bu fonksiyonları gizler. Birim yönetim katmanında gerçekleşen fonksiyonlara örnekler: donanım hatlarının sezilmesi, güç tüketimi kontrolü, donanım sıcaklığının ölçülmesi, CPU zamanı, bellek boşluğu, kuyruk uzunluğu gibi kaynakların kontrolü, istatistiksel verinin toplanması vb. birim yönetim katmanı içinde OSF ve NEF aynı veya farklı donanımda gerçeklenebilir.

Ağ Yönetim Katmanı

Birim yönetim katmanının sorumluluğu tek bir donanım parçası içinde NEF işlemlerinin gerçekleştirilmesini yönetmek iken, ağ yönetim katmanının sorumluluğu birçok yönetim parçası arası etkileşimle ilişkili yönetim fonksiyonlarıdır. Ağ yönetim katmanında ağ öğelerinin içsel yapısı ile ilgilenilmez; yani bir yönlendiricinin bellek boşluğu, bağlaşma donanımının sıcaklığı vb. Bu özellikler bu katmanda direkt olarak yönetilmez. Bu katmanda gerçeklenen bazı fonksiyonlar şunlardır: tüm ağ görünümünün oluşturulması, QoS ihtiyaçlarının sağlanması için ağ üzerinden bir atanmış yol sağlanması, yönlendirme tablolarının

değiştirilmesi, link kullanımının izlenmesi, ağ performansının optimizasyonu, arızaların sezilmesi vb.

Ağ yönetimindeki OSF ’ler birim yönetim katmanındaki OSF ’ler tarafından sağlanan üreticiye bağlı olmayan yönetim bilgisini kullanır. Bu ilişki içinde ağ yönetim katmanındaki OSF ’ler bir yönetici rolünü oynarken EML ’ler içindeki OSF ’ler aracı rolündedir.

Servis Yönetim Katmanı

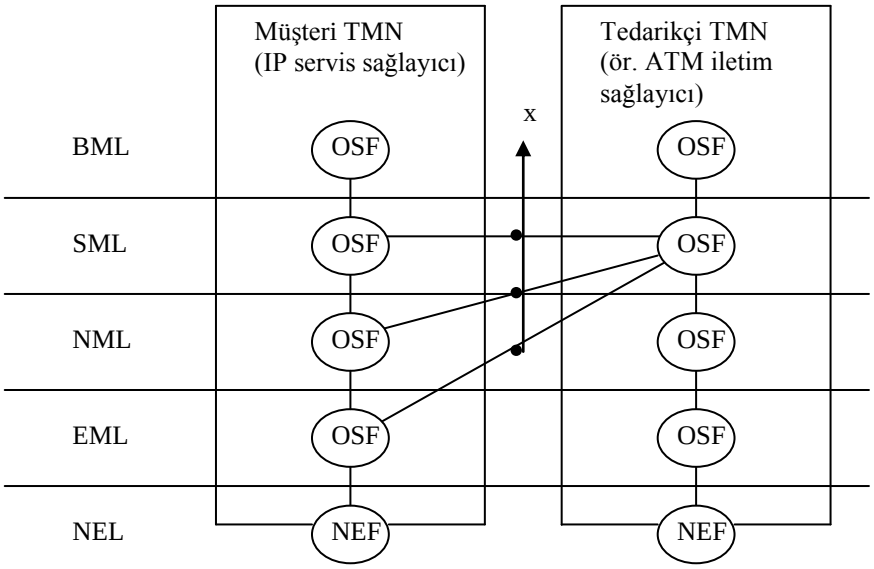
Servis yönetim katmanı haberleşme ağının kullanıcıları tarafından direkt olarak gözlenen beklentilerin yönetimi ile ilişkilidir. Bu kullanıcılar, son kullanıcılar ya da servis sağlayıcılar olabilir. Servis yönetim katmanı, ağ yönetim katmanı tarafından sağlanan yönetim bilgisini işler, fakat ağın iç yapısını göremez. Yönlendiriciler, bağlaşma elemanları, linkler vb. servis yönetim katmanında direkt olarak yönetilemezler. Bu katmanda gerçekleştirilen fonksiyonlara örnekler: QoS yönetimi (gecikme, kayıp vb.), ücretlendirme, kullanıcı ekleme/çıkarma, adres atama, grup adreslerinin yönetimi vb.

Servis yönetim katmanı örnekleri:

1) Ağlar arası (operatörler arası) yönetim için yönetim bilgisinin iki operatör arasında iletilmesi söz konusudur. Ticari kaygılar ve güvenlik kaygıları sebebiyle bu iki operatörden her biri kendi içsel ağ yapısını diğer operatörden gizlemeye çalışır. Sadece gerçekten ihtiyaç duyulan yönetim bilgisi (örneğin; QoS parametreleri) iletilir.

2) İkinci durumda noktadan noktaya iletim servisi sağlayan bir operatörün diğer bir operatöre ait ağ öğelerine bağlanmak için bir ağ kullandığı durum ele alınmıştır. Tipik bir örnek IP servis sağlayıcılarının, diğer operatörlere ait IP yönlendiricilere bağlanmak için ATM (SDH / DWDM) linkleri kullanmalarıdır. Bu durum Şekil 2.10’da gösterilmiştir. Bu şekilde IP servis sağlayıcıdan ATM servis sağlayıcıya üç adet x referans noktası tanımlanmıştır. ATM servis sağlayıcı tarafında, bütün referans noktaları servis yönetim katmanına bağlanmıştır. Böylece IP servis sağlayıcının ATM ağının içsel yapısını izlemesine ve değişiklik yapmasına izin verilmez. Sadece yüksek seviye yönetim

bilgisine, QoS yapısı gibi, erişimine izin verilir. IP servis sağlayıcı için ATM linki onun IP ağında tek bir ağ öğesi olarak görülür, bu birim yönetim katmanı referans noktasını oluşturur. Eğer IP servis sağlayıcıya ATM linki etrafında alternatif yönlendiriciler seçme seçeneği sağlanırsa, ağ yönetim aşamasında da bir referans noktasının IP servis sağlayıcı için tanımlanması gerekir. Sonuç olarak ATM linkinin performansı IP ağının tümünün QoS değerini etkiler. Bunun içinde servis yönetim aşamasında da bir referans noktası olmalıdır.



Şekil 2.10: Servis yönetimi örneği

İş Yönetim Katmanı

İş yönetim katmanı tüm ağın işletilmesinden sorumludur. Bu katman geniş bir alanı kapsar. Haberleşme yönetimi bu katmanın bir parçasıdır. İş yönetimi, bir amacın gerçekleştirilmesi değil bu amaca yönelik kurulumun oluşturulması olarak görülebilir. Bu yüzden iş yönetim katmanı diğer TMN yönetim katmanlarından farklı olarak işlemsel yönetimden çok stratejik ve taktiksel yönetimle ilişkilidir.

2.2.5 Yönetim Uygulama Fonksiyonları (Management Application Functions - MAFs)

TMN yapısında çok sayıda MAF tanımlı olmasına rağmen, genel olarak gerçekleştirdikleri işlemlere göre beş ana Yönetim Fonksiyon Alanında (Management Functional Areas MFAs) toplanabilirler:

- Konfigürasyon Yönetimi (Configuration Management – CM)
- Hata Yönetimi (Fault Management – FM)
- Performans Yönetimi (Performance Management – PM)
- Finans Yönetimi (Accounting Management – AM)
- Güvenlik Yönetimi (Security Management – SM)

Tablo 2.3'te bu beş MFA ile TMN 'in beş katmanı arasındaki ilişki gösterilmiştir.

Tablo 2.3: TMN MFA 'ları ve katmanları

Katmanlar	**Yönetim Fonksiyon Alanları**				
	Konfigürasyon Yönetimi	**Hata Yönetimi**	**Performans Yönetimi**	**Finans Yönetimi**	**Güvenlik Yönetimi**
BML	Yeni servise karar verme	Tamir önceliklerine karar verme	Servis parametrelerine karar verme	Servis fiyatlarına karar verme	Güvenlik politikasını hazırlama
SML	Servis isteklerini işleme	Kullanıcı sorun raporlarını işleme	Servis performansını izleme	Kullanıcı ile fiyat anlaşması yapma	Sertifikasyon yollarını yönetme
NML	Ağ noktalarına servisi kopyalama	Sebep analizi yapma	Ağ performansını izleme	Kullanım verisini toplama ve değerlendirme	İç şifre dağıtımı
EML	Ağ elemanlarını servisi desteklemek için konfigüre etme	NE testlerini yapma	Performans verisinin raporlanması üzerine NE 'lere talimat verme	Kullanım verisinin toplanması üzerine NE 'lere talimat verme	Ağ öğeleri üzerinde güvenlik tetkikleri yapma
NEL	EML konfigürasyon isteklerini cevaplama	EML test isteklerini cevaplama	EML Performans verisi raporlama isteklerini cevaplama	EML kullanım verisi toplama isteklerini cevaplama	EML güvenlik tetkiki değişiklikleri ni cevaplama

İki sınıflandırma arasındaki farklılıklar ve bütünleyici özellikler birkaç basit örnekle görülebilir: Yeni bir servis önerme kararı BML CM 'nin parçasıdır. Benzer şekilde, performans parametrelerini ve servis fiyatını belirleme BML PM ile BML AM arasındadır. Bu sınıflandırma ile yeni servislerin ihtiyaçlarının belirlenmesi kolaylaşır.[1]

2.3 DİĞER YÖNETİM YAKLAŞIMLARI İLE TMN İLİŞKİSİ

M.3010'da internet ve SNMP yönetim protokolü arasında bir referans noktası tanımlanmamıştır. TMN ile SNMP arası ilişkiler vardır. Ayrıca TMN internet yönetimi ile ilgili bir çok kavramı da içermektedir. Bunlar burada ele alınmamıştır.

TMN ile OSI arasında da güçlü bir ilişki vardır. Fonksiyon blokları ve referans noktaları ile tanımlanan TMN fonksiyonel yapısı OSI içeriği ile açıklanabilir. Örneğin; fonksiyon bloklarını fonksiyonel bileşenler içermektedir (sunum fonksiyonları), bunlar OSI protokol varlıkları gibi düşünülebilir. Veya fonksiyon blokları arası tanımlanan referans noktaları OSI terminolojisinde alt katmandaki servis sağlayıcılar olarak düşünülebilir.

3. GÜVENLİK

Her sistem bazı potansiyel tehdit veya tehlikelerle karşı karşıyadır. Sistemin önemi arttıkça, bu tehditlerle gelen riskler de artar. Sistem bir haberleşme ağına bağlandığı zaman, potansiyel tehditler de artar. Binlerce sistemin haberleşme ağlarıyla bağlandığı durumlarda, dışarıdan gelebilecek istenmeyen kullanıcıların hedefi haline gelinebilir. Bunların yanı sıra, TMN 'in başka zayıf yanları da vardır.

Bu bölümde potansiyel tehditlerin tanımları incelenmiştir. Daha sonra bu tehditlerin yol açtıkları riskler ele alınmıştır. Çoğu tehdit ve riskler oldukça genel olup, çoğu sistem ve ağda ortaya çıkabilir. Fakat bu tehditler TMN 'de güçlü bir şekilde ortaya çıkar. Bu tehditleri karşılayacak güvenlik servisleri de mevcuttur. Burada çoğunlukla TMN tarafından kullanılan güvenlik algoritmaları ele alınmıştır.

3.1 GENEL TEHDİTLER VE TMN 'İN ZAYIF NOKTALARI

3.1.1 Potansiyel Güvenlik Saldırıları

İyi bir güvenlik analizi öncelikle potansiyel saldırıları belirlemekle başlar. Her güvenlik saldırısı sınıfı şu özelliklerine göre karakterize edilebilir:

- Sisteme giriş metodu
- Başarılı bir giriş sonucu karşılaşılan risk
- Sınıfın büyüklüğü – kabaca o kategoride kaç kişi bulunduğu
- Sınıfın bir üyesinin saldırı yapma olasılığı

Potansiyel sisteme giriş sınıfları, saldırıların oluşabilme yeteneği ile, sonucunda ortaya çıkan ciddi risklerle ve gerektirdikleri önlemlerle karakterize edilebilir. Örneğin; ciddi suçlar işlemeye ne yeteneği, ne de eğilimi olan küçük bir topluluk için

de güçlü güvenlik yatırımları yapmak gereksizdir. Ancak çoğu durumda sınıfları belirlemek oldukça zordur.

TMN ’e karşı oluşacak potansiyel güvenlik saldırılarında şu yöntemler izlenebilir:

- Dışarıdan teknolojik olmayan metotlarla saldırı
- Dışarıdan teknolojik metotlarla saldırı
- Bir kullanıcının sistemine saldırı
- Kullanıcının sisteminin bir suç için kullanılması
- İçeriden gelen bir saldırı

Bir sistemi olası saldırılar karşısında tümüyle korumak imkansız olsa da, gerekli önlemleri alarak ve tedbirli davranarak istenen güvenlik seviyesi sağlanabilir. Yukarıda listelenen olası saldırılar karşısında alınacak önlemler belirlenmelidir. Ayrıca, listedeki suç işleme oranı da aşağıya indikçe azalmaktadır. Bu yüzden her kurum kendine uygun güvenlik ve risk seviyesi belirlemeli ve uygun adımları hazırlamalıdır.[1]

3.1.2 Potansiyel Güvenlik Tehditleri

TMN’in karşılaştığı olası güvenlik tehditleri:

- İzinsiz giriş
- Gizlice dinleme
- Yerine geçme
- Veri değiştirme
- İnkar etme
- Mesajları yeniden alma, yeniden yönlendirme, yanlış yönlendirme ve silme
- Ağı taşırma

İzinsiz Giriş:

Kullanılması için yetki verilmemiş bir kaynağa erişilmesi ve kaynaktan veri alınması olayıdır. İzinsiz giriş iki türlü olabilir: bir kaynağa gizlice erişilebilir (ör: hedef sisteme bir modem yardımıyla girmek) veya sahte yetkiler kullanılarak sistem aldatılabilir. İlk durumda güvenlik önlemleri yetersizdir, ikinci durumda ise yetkili kullanıcıların bilgileri iyi saklanamamaktadır. Her iki yöntemle de sisteme yıkıcı zararlar verilebilir: gizli bilgiler elde edilebilir, silinebilir veya değiştirilebilir.

Gizlice Dinleme:

Haberleşme kanalının kullanıcıların haberleri olmadan kullanılması ve kanaldan bilgilerin alınması olayıdır. Gizli verilere ulaşılabilir.

Yerine Geçme:

Yetkili kullanıcının taklit edildiği ve kaynakların kullanıldığı durumdur. Kullanıcının kaynak adresi, kullanıcı adı ve şifresi veya imzası kullanılabilir. Bu aldatma sonucu, gizli bilgilere ulaşılıp değiştirilebilir.

Veri Değiştirme:

Bir kaynağa yetkisiz bir şekilde erişerek oradaki verileri değiştirmek ve yeni veriler eklemektir. Bu durumun tehlikeli sonuçları vardır. Örneğin; güvenlik ile ilgili ayarların değiştirilmesi sonucu sisteme istendiği zaman girilmesi kolaylaşabilir.

İnkar Etme:

Bir kullanıcının hizmet aldığını inkar etmesidir. Kullanıcı ile fatura servisleri arasında anlaşmazlığa yol açabilir.

Mesajları Yeniden Alma, Yeniden Yönlendirme, Yanlış Yönlendirme ve Silme:

Mesajları yeniden alma, geçerli mesajların kopyalanarak yeniden kullanılması sonucu kaynaklara erişilmesidir. Kullanma hakkının bulunmadığı servislere erişim için kullanılabilir. Mesajları yeniden yönlendirme, mesajın yolunu değiştirerek gidiş gelişleri esnasında kopyalanabilmesini sağlamaktır. Mesajları yanlış yönlendirme,

yanlış alıcılara mesaj gönderilmesini sağlamaktır. Mesajları silme ise herhangi bir alıcıya ulaşamamasına yol açar.

Ağı Taşırma:

Ağa yanlış ve ilgisiz mesajlar gönderilerek ağın taşırılmasıdır. Sahte istekler ile hedef sistemin kaynakları boş yere harcanmış olur ve gerçek istekler karşılanamaz hale gelir.

3.1.3 Potansiyel Güvenlik Riskleri

Burada güvenlik tehditlerinin muhtemel sonuçları ele alınmıştır. Bu riskler şu şekilde gruplanabilir:

- Veri çalma
- Kaynakların yetkisiz olarak kullanımı
- Servis çalma
- Servisi engelleme

Veri Çalma:

Bir düğüm üzerinden veya iki düğüm arasındaki haberleşme kanalından gizli bilgilere ulaşılması sonucu veri çalma gerçekleşebilir. İzinsiz giriş, yerine geçme ve gizlice dinleme tehditlerinin sonucu olarak oluşabilir.

Kaynakların Yetkisiz Olarak Kullanımı:

Kullanmak için yetkili olunmayan kaynakların kullanılmasıdır. İzinsiz giriş, yerine geçme, veri değiştirme, mesajları yeniden alma, yeniden yönlendirme ve yanlış yönlendirme tehditlerinin sonucunda oluşabilir.

Servis Çalma:

Servislerin yetkisiz olarak ve fatura işlemleri yapılmadan kullanılmasıdır. İzinsiz giriş, yerine geçme, veri değiştirme, inkar etme, mesajları yeniden alma, yeniden yönlendirme ve yanlış yönlendirme tehditlerinin sonucunda oluşabilir.

Servisi Engelleme:

Bir kaynağın beklenilen çalışmasını yapmasının engellenmesidir. İzinsiz giriş, yerine geçme, kaynak yönetim verilerinin değiştirilmesi, mesajları yeniden yönlendirme, yanlış yönlendirme, silme ve ağı taşırma tehditlerinin sonucunda oluşabilir.

Servis engelleme riski tek bir kullanıcıyı veya tüm ağı hedef alabilir.

Tek Kullanıcılı Servisi Engelleme: Tek bir kullanıcının kaynaklara erişimi iki yola engellenebilir. İlk olarak; kullanıcıya gelen ve kullanıcı tarafından gönderilen gizli bilgiler engellenerek hedefine ulaşmaması sağlanabilir. İkinci olarak da; güvenlik yönetim bilgileri değiştirilerek kaynağın yetkili kullanıcının erişimine izin vermemesi sağlanabilir.

Ağ Çapında Servisi Engelleme: Haberleşme kanalındaki tüm geçerli trafiğin durdurulmasıdır. Gereksiz mesajlarla ağı taşırma yöntemiyle gerçekleştirilebilir. Ağın taşması durumunda ise yanlış mesajlar da iletilerek kaynağın bunlara cevap vermesi sağlanabilir.

Tablo 3.1'de özet bir güvenlik tehditleri listesi ve olası sonuçları verilmiştir.

Tablo 3.1: Güvenlik tehditleri ve riskleri özeti

Güvenlik Tehditleri	Güvenlik Riskleri			
	Veri Çalma	Kaynakların Yetkisiz Olarak Kullanımı	Servis Çalma	Servisi Engelleme
İzinsiz Giriş	√	√	√	√
Yerine Geçme	√	√	√	√
Gizlice Dinleme	√			
Veri Değiştirme		√	√	√
Mesajları Yeniden Alma		√	√	
Mesajları Yeniden Yönlendirme, Yanlış Yönlendirme ve Silme		√	√	√
İnkar Etme			√	
Ağı Taşırma				√

3.1.4 Güvenlik Risklerinin TMN Üzerindeki Etkisi

Yukarıda bahsedilen riskler TMN 'in her beş MFA 'sı üzerinde de büyük etkilere sahiptir. Bu etkileri gözlemlemek için her MFA 'dan bir grup MAF seçilerek olası risklere karşı test edilir.

Konfigürasyon Yönetimi – Tedarik Etme

Tedarik etme MFA 'sı servislerin seçilip etkinleştirildiği süreçleri tanımlar. Süreçteki pek çok noktadan TMN 'e saldırılabilir. Bu noktalara örnekler:

- Veri çalma – Ağ üzerinden veya kullanıcının istemiş olduğu servislerden gizli bilgilere ulaşılabilir.

- Servis çalma – TMN yapısı ile oynanarak veya servis istekleri inkar edilerek ödeme yapılmadan servisler kullanılabilir.

- Servisi engelleme – Yanlış istekler gönderilerek TMN 'in servis isteklerine cevap vermesi engellenebilir veya doğru bilgilerin hedeflerine ulaşması engellenebilir.

Performans İzleme

Ağın gerçek zamanlı performansını izlenmesidir. Bu bilgi, trafiğin son halini göstermesi bakımından, TMN için oldukça değerlidir. Ağ performansının ayarlanması için kullanılır. Güvenlik açısından zayıf noktaları:

- Veri çalma – Hangi kaynakların kullanımda olduğu veya bu kaynakların kapasiteleri gibi ağ performansı ile ilgili verilere ulaşılabilir.

- Kaynakları yetkisiz olarak kullanımı – OS 'e bildirilen trafik raporları değiştirilebilir, geciktirilebilir veya engellenebilir. Bu durumda ağ yanlış trafik durumlarına tepki göstermiş olur.

Hata Yönetimi – Sorun Yönetimi

Sorun yönetimi, ağdaki hatalı durumların OS ’lere iletildiği durumlara karşı düşer. Diğer MAF ’larda olduğu gibi, sorun yönetimi mesajları hedef alınarak ağa zarar verilebilir. Örnekler:

- Veri çalma – Ağdaki hangi elemanlar servis kesilmesine uğramış, bu kesilmenin tamir boyutları ne veya hangi kaynaklar tamir amaçlı olarak ayrılmış gibi ağdaki sorunların bilgilerine erişilebilir.

- Servisi engelleme – Ağ problemlerini içeren mesajlar kesilebilir, değiştirilebilir veya engellenebilir. Bu durumda ağın gerçek sorunlar karşısında tepki vermemesi veya yanlış sorunlar karşısında kaynaklarını kullanması sağlanabilir.

Finans Yönetimi

Kullanımın Ölçülmesi: Kullanıcının ağı kullanma süresini ölçülmesidir. Fatura işlemleri için temel oluşturur. Bu servise yapılan saldırılar servis sağlayıcının gelirini veya kullanıcı bilgilerini tehlikeye atar. Örnekler:

- Veri çalma – Kullanım süresi, hangi servislerin kullanıldığı veya kullanım tarihçesi gibi bilgilere ulaşılabilir.

- Servis çalma – Kaynaklardaki kullanım bilgisini çalınması, değiştirilmesi veya engellenmesi gibi durumlar sonucunda uygun OS ’e fatura bilgisinin iletilmesi engellenir.

Tarifelendirme / Fiyatlandırma: Ağdaki servislerin fiyatlandırılması ile ilgilenir. Bu servise yapılan bir saldırı servis sağlayıcının gelirini etkiler:

- Veri çalma – Servislerin fiyatlandırılması ile ilgili bilgilere ulaşılabilir.

- Servis çalma – OS ’ler arasındaki tarife bilgileri değiştirilebilir veya engellenebilir.

3.2 TMN 'E ÖZGÜ TEHDİTLER

3.2.1 TMN 'in Genel Zayıf Noktaları

TMN, gelişmiş bir ağ yönetimi için yeni olanaklar sunarken; yeni zayıf noktalarla da karşı karşıya bırakabilir.

- TMN'in ortaya çıkmasıyla birlikte ağ yönetiminin otomatikleşmesi ve tek merkeze toplanması sonucunda, NE 'lerin **uzaktan yönetim**ine başlanmıştır. Bu durumda sahadaki kontroller azalmış ve saldırılar için yeni olanaklar doğmuştur.

- Standart arabağdaşımlara yönelim sonucunda, TMN dışındaki **dış öğeler**le etkileşimler kolaylaşmıştır.

- TMN 'den önce ağ yönetimi iletişimi için kullanılan protokollerin bilinmesi gerekiyordu ve bu protokoller patentliydi. TMN ile birlikte **standart arabağdaşımlar**ın kullanılmaya başlamasıyla, protokoller hakkında bilgi almak da kolaylaşmıştır.

- TMN 'den önce hedef alınan sistemin işletim sisteminin bilinmesi gerekiyordu. Açık sistem arabağdaşımları ile birlikte hedefte kullanılan işletim sisteminin önemi kalmamıştır.

- Açık sistem arabağdaşımlarında kullanılan ortak **transfer sözdizimi** sayesinde kablodan akan bitler anlam kazanmıştır.

Bu gibi durumlarda TMN 'in avantajları kolaylıkla dezavantaja dönüşebilir.

3.3 GÜVENLİK SERVİSLERİ

Çeşitli güvenlik servisleri TMN 'i daha önceki bölümlerde anlatılan saldırılar karşısında korur. Bu güvenlik servisleri değişik saldırılara karşı ve değişik tekniklerle çalışırlar. Bazı servisler bağımsız çalışırken, bazıları birbirleriyle çakışan özelliklere sahiptir. Bu servislerin masrafları da göz önüne alınarak, TMN güvenlik mimarı doğru servisleri seçmelidir.[1]

3.3.1 Bağlantı Giriş Kontrolü

Bağlantı giriş kontrolü savunmanın ilk hattıdır. Yetkili olmayan kullanıcıların sisteme girişinin engellenmesini sağlar. Yetkili olmayan kullanıcılardan gelen tüm mesajların ve bağlantı isteklerinin yerine ulaşmalarını engeller. Hedef sistemi bu tür mesajlarla uğraşmaktan, ağı da bu tür mesajlar taşımaktan kurtarır.

Bağlantı giriş kontrolü, gelen mesajlardaki kaynak adresi ve hedef adresini yetkili kaynak/hedef adres çiftleriyle karşılaştırır. Bu işlem genellikle Kapalı Kullanıcı Gruplarında (Closed User Groups – CUG) X.25 iletim ağları üzerinde ve ateş duvarlarında (Fire Walls – FW) TCP/IP iletim ağları üzerinde gerçekleştirilir.

Teknik açıdan yeterli olmayan saldırıların büyük bölümünden bu yöntemle korunulabilir. Avantajı basit olmasıdır: tek veya birkaç sistem kullanılarak büyük sayıdaki sistemler korunabilir.

3.3.2 Eş Öğeleri Doğrulama

Eş öğeleri doğrulama, uygulamadan uygulamaya güvenli girişi sağlar. Her grubun birbirlerini şifreleme ile korunmuş santraller üzerinden doğrulaması esastır. Bağlantı giriş kontrolünü aşan saldırılar için ikinci bir hat oluşturur.

Eş öğeleri doğrulama ile isteği yapanın, istek mesajında belirtilenle aynı olup olmadığı doğrulanır.

3.3.3 Veri Orijini Doğrulama

Yetkili kullanıcı başarılı bir şekilde sisteme girdikten sonra iki taraf arasındaki bağlantıya mesajlar yerleştirilmek suretiyle de saldırı yapılabilir. Veri orijini doğrulama ile bu tür mesajların gönderildikleri nokta belirlenebilir. Veri orijini doğrulama ağdaki tüm mesajlara uygulanabileceği gibi, sadece seçilen bazı mesajlara da uygulanabilir.

3.3.4 Bütünsellik

Veri orijini doğrulama ile mesajın doğru kullanıcı tarafından gönderildiği garanti altına alınır ancak bu kaynağı doğru mesajların ağ içerisinde dolaşırken

değiştirilmesi karşısında güvence verilemez. Bütünsellik servisi ile bu tür değiştirilmiş mesajlar fark edilir ve kullanıcılara mesajı geri almaları için uyarı gönderilir. Ancak mesajları geri alınması bütünsellik servisinin kapsamı dışındadır. Bütünsellik servisi birkaç değişik şekilde olabilir:

- **Seçilen Alanda Bütünsellik:** Bazı durumlarda tüm mesajlar arasından sadece seçilmiş kritik bir bölümde bütünsellik servisine gerek duyulabilir. Ancak çoğu durumda tüm mesajın bütünselliğini korumak da eş zorluktadır. Fakat tüm mesajı korumak her zaman pratik olmayabilir.

- **Tüm Mesajlarda Bütünsellik:** Tüm mesajların değişikliğe uğrayıp uğramadığının kontrol edilmesidir. Genel olarak, değişikliğe uğramış mesajdan doğru mesajın çıkarılmasını sağlayamaz.

- **Oturumda Bütünsellik:** Mesajların değiştirildiği anlaşılsa bile, geçerli mesajların silinmesi, sırasının değiştirilmesi, yanlış hedefe yönlendirilmesi gibi saldırılarda da bulunulabilir. Oturumda bütünsellik ile bu saldırılar sezilmeye çalışılır.

3.3.5 Gizlilik

Şimdiye kadar tartışılan güvenlik servisleri aktif saldırılar için hazırlanmışlardı. Ancak ağdaki mesajlara herhangi bir müdahalede bulunmadan, sadece okumak suretiyle gizli bilgilere ulaşılabilir. Bu tür pasif saldırılar karşısında gizlilik servisi uygulanır. Gizlilik servisi birkaç değişik şekilde olabilir:

- **Seçilen Alanda Gizlilik:** Mesajların seçilen kısımlarını korur. Bir mesajın sadece birkaç bölümünde (kullanıcı adı, adresi, telefonu gibi) koruma gerektiğinde; bu bölümler şifrelenir ve iletim ağı ek yükten korunur.

- **Tüm Mesajlarda Gizlilik:** Mesajların tümünü korur.

- **Trafik Akış Gizliliği:** Bazı durumlarda mesajların iki nokta arasında belli bir yol üzerinde olduğu, mesaj sayısı ve uzunluğu bilinebilir. Trafik akış gizliliği bu tür verileri saklamak için uygulanır. Bu durum genellikle askeri uygulamalarda önemlidir. Pahalı bir yöntem olduğu için tercih edilmez.

3.3.6 İnkar Etmeme

İnkar etmeme servisi, bir kullanıcının gönderip aldığı mesajları inkar etmesini engeller. Çoğunlukla yasal koruma sağlar.

3.3.7 Giriş Kontrolü

Ağ seviyesindeki giriş kontrolü, sadece yetkili kullanıcıların TMN öğelerini kullanmasını sağlar. Ayrıca yetkili kullanıcıların da bazı kaynaklara sınırlı ulaşımını sağlar. Giriş kontrolü, gelen mesajları saate, güne veya istek mesajının başlangıç noktasına göre kabul eder veya reddeder.

3.3.8 Güvenlik Alarmı

Herhangi bir güvenlik kuralı çiğnendiğinde güvenlik alarmı oluşturulmasıdır. Alarmın tipine, önemine, gün veya saatine göre gönderileceği noktalar belirlenebilir. Alarmların gönderildiği noktalar tarafından alındı bilgisi gelmelidir. Eğer belli bir süre sonunda bu bilgi gelmiyorsa, alarm alternatif noktalara gönderilir.

Sisteme zarar verildiği durumlarda ise alarm gönderilemez veya yanlış alarmlar gönderilir. Bu yüzden şüpheli durumlarda alarm göndermek sistemin korunması için gereklidir.

Bir sistemdeki ciddi durumlar için “panik kipi” oluşturulabilir ve gönderilen alarmlardan alındı bilgisi beklenmez. Sistem tüm faaliyetleri askıya alır ve sadece güvenlik alarmlarını iletir.

3.3.9 Güvenlik Tetkiki

Bazı saldırılar oldukça görünür şekildedir: ağa zarar verir, fatura verisini yok eder. Bazı saldırılar ise verilere ulaşma amacı güder ve gizlice yapılır, bazı ufak değişiklikler yapılarak herhangi bir ödeme yapılmadan kullanılan bant genişliği arttırılabilir veya fatura kayıtlarındaki değerleri düşürebilir. Bu saldırıların fark edilmesi zordur. Fakat kaynaklar üzerinde uzun süreli analizler yapılırsa (örneğin; birkaç ay boyunca) etkiler ortaya çıkarılabilir. Güvenlik tetkiki, bu tür güvenlik ile ilgili verilerin kayıtlarının alınmasıdır. Düşük seviyeli bir saldırı şüphesi olduğu durumlarda güvenlik tetkiki yapılabilir.

Güvenlik tetkiki ile tüm güvenlik alarmlarını ve bu alarmlara neden olan olayları kaydeder. Ayrıca ağa giriş isteği veya belirli verilere erişme girişimleri gibi tek başına zararsız ama bir araya geldiklerinde tehlikeli sonuçlar doğurabilecek durumlar da kaydedilir.

Tablo 3.2'de güvenlik tehditleri ve bu tehditler karşısında uygulanabilecek güvenlik servisleri verilmiştir. Bu tabloda güvenlik alarmı ve tetkikinin yer almama sebebi; bu servislerin tehditleri önleme değil de sezme amacıyla kullanılmalarıdır. Tablo 3.3'te ise güvenlik riskleri ve servisleri incelenmiştir.

Tablo 3.2: Güvenlik servisleri ve güvenlik tehditleri

	Güvenlik Tehditleri						
Güvenlik Servisleri	İzinsiz Giriş	Gizlice Dinleme	Yerine Geçme	Veri Değiştirme	İnkar Etme	Yeniden Alma, Yeniden Yönlendirme ve Silme	Ağı Taşırma
Giriş Kontrolü	√			√			√
Eş Öğeleri Doğrulama	√		√				
Veri Orijini Doğrulama	√		√				
Bütünsellik*			√	√		√	
Gizlilik		√	√				
İnkar Etmeme					√		

*Bütünsellik servisi saldırıları sezer ancak engellemez.

Tablo 3.3: Güvenlik servisleri ve güvenlik riskleri

	Güvenlik Risleri			
Güvenlik Servisleri	Veri Çalma	Kaynakların Yetkisiz Olarak Kullanımı	Servis Çalma	Servisi Engelleme
Giriş Kontrolü	√	√	√	√
Eş Öğeleri Doğrulama		√	√	
Veri Orijini Doğrulama		√	√	
Bütünsellik*	√	√	√	√
Gizlilik	√	√		√
İnkar Etmeme			√	

*Bütünsellik servisi saldırıları sezer ancak engellemez.

4. TEMEL GÜVENLİK MEKANİZMALARI

Güvenlik mekanizmaları, güvenlik servislerini destekleyen çok çeşitli algoritmalar ve protokollerden oluşur. Burada bahsedilen mekanizmalardan bazıları tek bir güvenlik servisini desteklerken, bazıları birkaç tane servisi birlikte destekler.

4.1 KIYMA

Kıyma genellikle uzun bir M metninin alınarak belli uzunluktaki (örneğin; 16 oktet) bir m mesaj bölümüne dönüştürülmesidir. Kıyma fonksiyonu aşağıdaki dört koşulu sağladığı takdirde, genellikle güvenli ve tek yönlü bir fonksiyon olarak kabul edilir.

- Bit dizisindeki herhangi bir değişiklikte (örneğin; bit ekleme, silme veya değiştirme) mesajın sadece bir kısmı değil, tüm mesaj bölümü değişir. (Bu **güvenlik** özelliğinin bir kısmıdır.)

- Kıyma fonksiyonu, mesaj ve mesaj bölümü bilinse bile, aynı mesaj bölümünü verecek bir mesaj oluşturmak pratik açıdan oldukça güçtür. (Bu **ikinci görüntülemeye karşı direnç** özelliğidir.)

- Aynı kıyma değerine sahip iki mesaj bulmak hesaplama açısından mantıklı değildir. (Bu özellik **çakışma önleyici** olarak bilinir.)

- Kıyma fonksiyonu bilinse bile, mesaj bölümünden orijinal mesajı elde etmek imkansızdır. (Bu **tek yönlülük** özelliğidir.)

Son yıllarda bazı kıyma algoritmaları popülerlik kazanmaya başlamıştır: MD2, MD4, MD5, SHA (Secure Hashing Algorithm – Güvenli Kıyma Algoritması) ve MDC-2. Bunlardan MD2 oldukça yavaştır, MD4'ün bazı güvenlik açıkları vardır, MD5 tercih edilebilir. SHA, MD5 ile benzerlikler gösterir. [1]

Hiçbir kıyma algoritmasının güvenli olduğu ispatlanmamıştır. Örneğin; farklı bit dizilerinden aynı mesaj bölümünü oluşturacak bir yol bulunabilir. Bazı kıyma

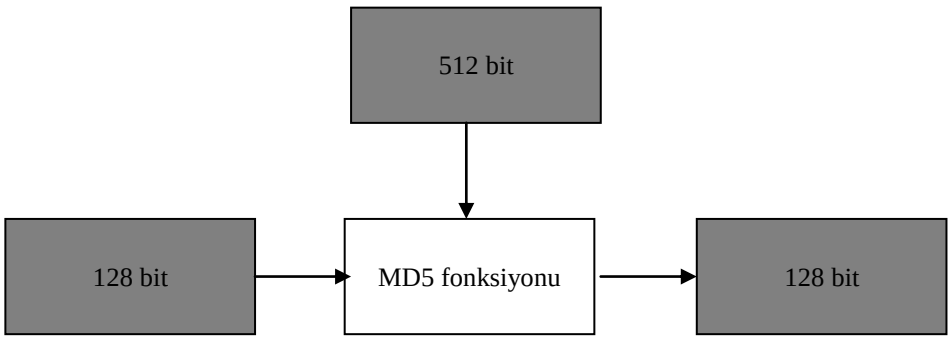
algoritmaları da güvensiz olarak bilinir. Örneğin; bazı kıyma algoritmaları aynı mesaj bölümü ile sonuçlanacak iki farklı mesaj üretebilir. Diğer kıyma algoritmaları güvensiz olarak bilinmez.

MD5 128 bitlik bölüm oluşturur ve çakışmalardan bağımsız değildir. Bu nedenle bazı uygulamalar için uygun olmadığı düşünülse de, ağ yönetiminde kullanılmasına engel değildir. MD5, ağ yönetim mesajlarının bütünselliğinin sağlanmasında kullanılır. İki farklı bit dizisinin aynı MD5 mesaj bölümüne dönüştüğü görülebilir ancak bunlardan herhangi biri geçerli bir ağ yönetim mesajı olamaz. SHA daha yeni bir kıyma algoritmasıdır ve 160 bitlik bölüm oluşturur. Ağ yönetimi güvenliğinde kullanılan en önemli kıyma algoritmaları da MD5 ve SHA 'dır.

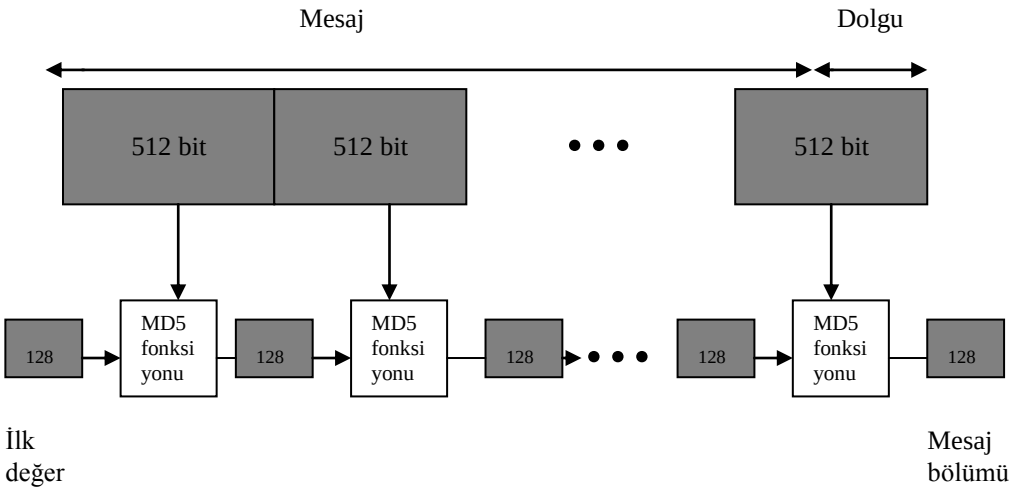
Kıyma tek başına bir güvenlik servisi olarak kullanılmaktan çok, dijital imzalar ve anahtarlı kıyma ile birlikte kullanılır. MD5 yapısı en güvenli kıyma algoritması olarak bilinir. MD5, 512 veya katlarındaki bit dizisine rastgele uzunlukta başlangıç mesajı ekleyerek başlar. Tek bir 1 bitinden sonra 512'nin katından 64 bit eksik olacak şekilde 0 biti ekler. Sonra ilk mesajın uzunluğunun 64 bitlik gösterimini ekler.MD5'in özü iki bit dizisi (biri 512 bit diğeri 128 bit) üzerinde çalışan ve lineer olmayan bir fonksiyondur. MD5'in sonucu 128 bitlik bir dizidir. MD5 fonksiyonu sırasıyla 512 bitlik dolgulu kısım üzerinde çalışır. İlk 512 bitlik blok için şu 128 bitlik diziyi (onaltılık gösterimde) ikinci giriş olarak kullanır:

$$\text{İlk değer} = 0123456789abcdeffedcba9876543210 \qquad (4.1)$$

Sonraki her biri giriş olarak, bir önceki 128 biti kullanır. Son çıkış MD5 tarafından üretilen mesaj bölümüdür. Tek bir blok üzerindeki MD5 işlemi Şekil 4.1'de gösterilmiştir. Tüm mesaj üzerindeki MD5 işlemi ise Şekil 4.2'de gösterilmiştir.



Şekil 4.1: 512 bitlik blok için MD5 yapısı



Şekil 4.2: Tüm mesaj üzerindeki MD5 operasyonu

4.1.1 Anahtarlı Kıyma

İletişimde bulunan her iki taraf tarafından bilinen gizli bir anahtarı orijinal mesaja ekleyerek ve mesajı şifrelenmiş veya şifrelenmemiş olarak kıyma fonksiyonuyla hesaplanmış mesaj bölümüyle birlikte göndererek oluşturulur. Eş öğelerin karşılaştırılmalarını sağlar. Her kısım kendi belirteci, alıcının belirteci ve özel bir zaman işaretinden oluşur. Bit dizisine gizli anahtarı ekleyerek, anahtarlı kıyma değerini hesaplar ve sonucu kıyar. Daha sonra orijinal metin mesaj bölümü ile birlikte iletilir. Alıcı taraf, metne paylaşılmış anahtarı uygulayarak aynı kıyma değerini elde eder ve göndericiye doğrulama bilgisi gönderir. Eğer gönderici tarafından gönderilen ile hesaplanan mesaj bölümü aynı ise gönderenin kimliği doğrulanır. Bu tip doğrulamalar cevaba karşı dirençlidir çünkü, her doğrulama farklı bir zaman işaretine sahiptir. Ayrıca yanlış yönlendirme ve yeniden yönlendirmelere karşı da dirençlidir çünkü, hedeflenen alıcının adresi de kıyma ile korunmuş bit dizisi içerisindedir.

Anahtarlı kıyma ayrıca veri orijininin doğrulanmasında da kullanılır. Gönderici her mesaja bir kimlik belirleyici ekleyerek koruma sağlar. Cevap alımını önlemek için de, her mesaj farklı bir zaman işareti kullanır.

Anahtarlı kıyma ayrıca bütünsellik de sağlar. Gönderici metne bir anahtar ekler ve sonuçtaki bit dizisinin mesaj bölümünü hesaplar. Eğer mesaj gönderim sırasında değiştirilmiş ise, alıcı tarafından hesaplanan anahtarlı kıyma değeri ile alınan mesaj

bölümü birbirlerine uymazlar. Mesajın yeniden gönderilmesini önlemek için tekrar etmeyen belli bir dizi orijinal metinde bulunmalıdır. Bu dizi belli bir zaman işareti, bir sıra numarası, tekrar etmeyen rastgele bir numara veya bunların bir karışımı olabilir. Zaman işareti kullanılması, gecikmelerin de belirlenmesini sağlar. Zaman işareti veya sıra numarası kullanılması, mesajın yeniden sıralandığının belirlenmesini sağlar. Sıra numarası kullanılması, mesajın silindiğinin belirlenmesini sağlar. Aynı anahtarı paylaşan diğer kısımlara yapılan yeniden yönlendirme ve yanlış yönlendirme saldırılarının önlenmesi için, orijinal mesajda alıcının kimliği de bulunmalıdır.

Popüler kıyma fonksiyonlarını anahtarlı kıyma ile kullanmanın bazı zayıf noktaları vardır. Bazı durumlarda ilk metne ve onun karşılığı olan mesaj bölümüne ulaşılabilir ve metne eklemeler yapılarak bu eklemelerin alıcı tarafından kabul edilecek mesaj bölümleri hesaplanabilir. Bu durumu engellemek için çift kıyma yapılabilir. Çift kıymada öncelikle, tek kıymada olduğu gibi, metne bir anahtar eklenir ve mesaj bölümü hesaplanır. Daha sonra, hesaplanan mesaj bölümüne aynı veya farklı bir anahtar eklenir ve aynı veya farklı bir kıyma fonksiyonu kullanılarak kıyılır. Farklı kıyma fonksiyonları kullanmak daha güvenli bir uygulamadır.

HMAC sıklıkla kullanılan bir çift kıyma prosedürüdür. IP ağlar üzerinde MD5 ile doğrulama yapmak amacı ile belirlenmiştir. 64 oktet uzunluğundaki K anahtarını kullanır. Eğer paylaşılmış anahtar 64 oktetten küçükse, 64'e tamamlayacak kadar 0 oktetleri eklenir. HMAC ayrıca iki tane daha dolgu sabiti kullanır:

ipad = 0 okteti x 36 64 kere tekrar (4.2)

opad = 0 okteti x 5C 64 kere tekrar (4.3)

Bir mesaj metni için HMAC-MD5 şöyle tanımlanır:

MD5(K XOR opad, MD5(k XOR ipad, "metin")) (4.4)

XOR bit başına exclusive-OR anlamındadır.

4.1.2 S-anahtarı

S-anahtarı, eş öğeleri doğrulama ve şifre doğrulamayı kolaylaştırmak ve aynı zamanda yeniden yönlendirme saldırılarından korumak amacıyla Bellcore tarafından geliştirilmiştir.

Tek kıymanın en önemli dezavantajı kullanılan şifrenin öğrenilmesidir. S-anahtarı ile orijinal şifre N kere (örneğin; 1000 kere) kıyılır. Sadece son kıymanın sonucu kullanıcıyı doğrulayacak sisteme iletilir ve alıcıda lokal olarak saklanır. Kullanıcı orijinal şifresini ve onu kaç kere kullandığını gösteren I sayacını tutar. Her oturum için şifre N-I-1 kere kıyılır ve sonuç doğrulama değeri olarak kullanılır. Alıcı bu değeri bir kere kıyar ve sonucu o kullanıcı için lokal olarak saklanmış değerle karşılaştırır. Eğer kıyılmış doğrulama değeri (N-I kere kıyılmış orijinal şifre) ile lokal olarak saklanmış değer aynı ise kullanıcı doğrulanmış olur ve yeni doğrulama değeri (N-I-1 kere kıyılmış orijinal şifre) o kullanıcı için lokal olarak saklanmış değerin yerini alır. Aynı orijinal şifre bu yüzden N-1 kere kullanılabilir.

S-anahtarına dayalı doğrulama, zaman işaretleri ve sıra numaralarına gerek kalmadan yeniden yönlendirme, silme ve yeniden sıralandırmayı önleyerek veri orijinin doğrulanmasını sağlar. Ancak bütünsellik sağlamaz.

4.2 ŞİFRELEME

Şifreleme veriyi sadece yetkili kullanıcının anlamasını sağlayacak şekilde kodlamaktır. Kullanıcı kodlanmış veriye ulaşmak ve onu eski haline getirmek için bir anahtara ihtiyaç duyar. Şifrelemenin iki çeşidi vardır:

- Simetrik şifreleme
- Asimetrik şifreleme

Simetrik şifrelemede kodlama ve kod çözme için aynı anahtar kullanılırken, asimetrik şifrelemede iki farklı anahtar kullanılır.

4.2.1 Simetrik Anahtarlı Şifreleme

Simetrik anahtarlı şifreleme en eski ve sıklıkla kullanılan şifreleme yöntemidir. Bu yöntemde her iki taraf aynı şifreyi paylaşırlar ve şifreleme ve şifre çözmeyi gerçekleştirirler.

Simetrik anahtarlı şifrelemeye karşı yapılabilecek bir saldırı tüm olası anahtarları deneyerek mesajın şifresini çözmeye çalışmaktır. İyi bir şifreleme algoritması kullanılması ile bu tip bir saldırı dışındaki saldırılar başarısız kılınabilir. Algoritmaların gücü (gizli anahtarı elde etmek için ortalama kaç tane işlem yapılması gerektiği) anahtarın boyutu ile birlikte üstel olarak artar.

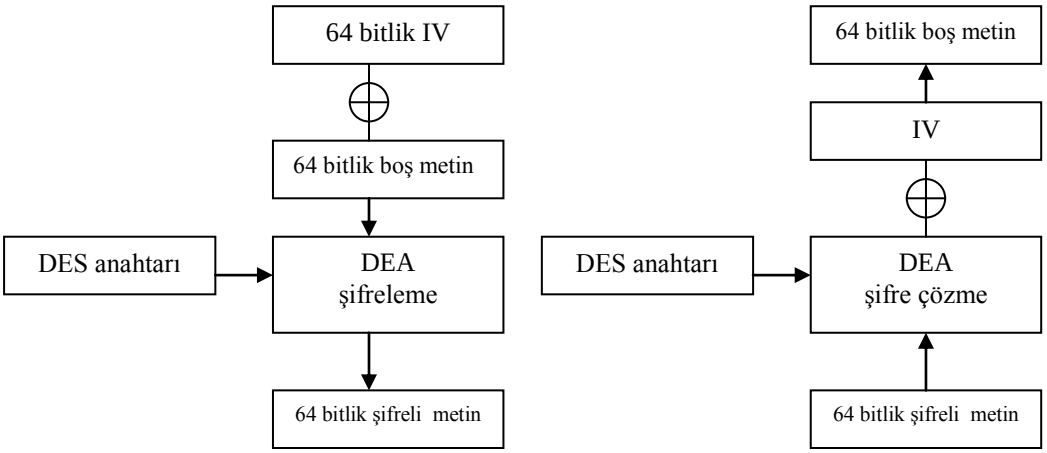
TMN 'deki simetrik şifreleme için Veri Şifreleme Standardı (Data Encryption Standard – DES) kullanılır. DES 20 yıldan beri kullanılmasına rağmen belirlenmiş başarılı bir saldırı yoktur. DES 64 bit anahtar kullanır, bunlarda 8'i parite bitidir bu yüzden 56 rastgele bitten oluşur. DES algoritması, DES şifreli bir mesajı çözmek ve anahtarı bulmak için, ortalama olarak 2^{55} kere çalıştırılmalıdır. Yeterince işlem gücü ile (örneğin; binlerce PC) DES şifreli bir mesaj birkaç günde hatta saatte çözülebilir. Bu yüzden DES anahtarı en az günde bir kez değiştirilmelidir. Ayrıca, şifreleme anahtarları listesi gibi hassas bilgiler için DES şifreleme yapılmamalıdır.

DES Veri Şifreleme Algoritmasına (Data Encryption Algorithm – DEA) dayanır. DEA iki giriş üzerinde çalışır:

- Gizli bir 64 bitlik DES anahtarı
- Gizli koruma isteyen 64 bitlik boş metin bloğu

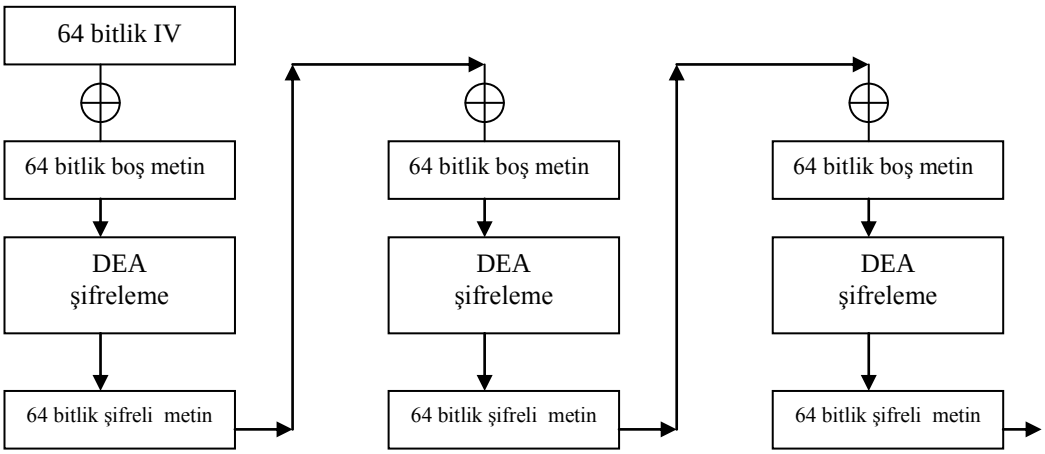
Sonucunda 64 bitlik şifreli metin ortaya çıkar. Ters işlemde ise şifreli metin ve anahtar kullanılarak orijinal metin geri elde edilir.

Aynı metni aynı anahtar ile şifrelemek aynı sonucu oluşturur. Bu yüzden aynı şifreli metinleri gözlemlemek bile yapı hakkında bir fikir verir. Bu saldırıdan korunmak için metinler önce rastgele 64 bitlik bir Başlatma Vektörü (Initialization Vector – IV) ile exclusive-OR işlemine tabii tutulur. Alıcı IV 'yi bilmelidir. Genellikle şifrelenmeden ve şifreli metin ile birlikte gönderilir. DES 'in bu kullanımı Şekil 4.3'te gösterilmiştir ve Elektronik Kod Kitabı (Electronic Code Book – ECB) adını alır.

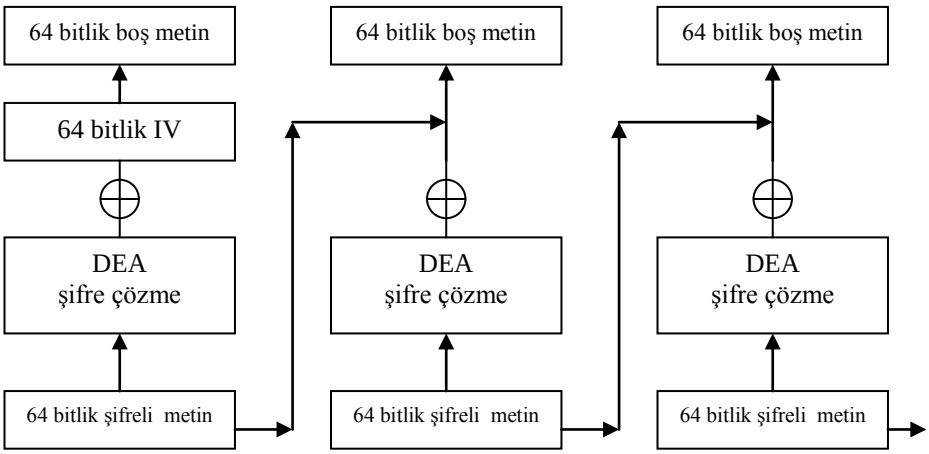


Şekil 4.3: ECB kipinde DES şifreleme ve şifre çözme

ECB 'nin bir dezavantajı büyük mesajlarda her 64 bitlik blokla birlikte IV 'yi de birlikte gönderme zorunluluğudur. Bu yük, bir blokta oluşan şifreli metni diğeri için IV olarak kullanan Şifre Blok Zincirleme (Cipher Block Chaining – CBC) ile aşılabilir. Sadece ilk blok, Şekil 4.4'te gösterildiği gibi, ayrı bir IV 'ye ihtiyaç duyar. ECB ve CBC 'nin dışında iki tane daha DES çalışma kipi vardır. Fakat CBC kipi uzun mesajlar için en uygun şifreleme yöntemidir ve TMN 'de kullanılır. Şekil 4.5 CBC kipinde DES şifre çözmeyi göstermektedir.



Şekil 4.4: CBC kipinde DES şifreleme



Şekil 4.5: CBC kipinde DES şifre çözme

Dolgulama

DES algoritmasında giriş, 8 oktetin tam katları uzunluktaki metin ve şifreleme anahtarından oluşur. Eğer metin 8 oktetin tam katı değilse dolgu oktetleri eklenir.

İlk TMN uygulamalarında metnin son okteti hangi oktetlerin veri içerdiğini gösteriyordu. Son veri okteti ile kaç veri okteti bulunduğunu gösteren metnin son okteti arasındaki oktetler rastgele idi. Daha sonraları bu rastgele oktetlerin de metnin son oktetiyle aynı olmasına karar verildi.

IV Seçimi

Genellikle IV 64 bitlik rastgele bir dizidir. Fakat bu durum TMN 'in ilk uygulamalarında bir zayıf nokta oldu.

Bu uygulamaların doğrulama prosedürlerinde CBC kipinde DES şifrelenmiş bir zaman işareti kullanılması ve metinle birlikte rastgele bir IV kullanılması belirtilmiştir. DES şifreleme anahtarının da 4 günde bir değiştirilmesi önerilmiştir. Bu doğrulamaların kopyaları alınarak, kopyanın alındığı zaman ve şifreleme anahtarı belirlenebilir. Ertesi gün aynı saatte bu doğrulama uygun bir IV ile birlikte yeniden yönlendirilebilir. IV 'nin ilk 8 okteti zamanı belirtir: YYYYMMDD (yıl için 4 oktet, ay için 2 oktet, gün için 2 oktet). İkinci gün için: YYYYMMDD′ olmak üzere;

$$IV' = (IV \text{ XOR } YYYYMMDD) \text{ XOR } YYYYMMDD' \quad (4.5)$$

Buradan da;

$$IV' \text{ XOR } YYYYMMDD' = IV \text{ XOR } YYYYMMDD \quad (4.6)$$

Böylece anahtar değişmediği sürece ilk günün şifresi çözülebilir. DES anahtarı her 24 saatte bir değiştirilmelidir.[1]

Hata Yayılımı

CBC kipinde DES 'in bir avantajı şifreli metinde oluşan iletim hatalarının, şifresi çözülmüş mesaj üzerinde sınırlı bir etkisinin olmasıdır. n. şifreli metin bloğundaki bir veya birkaç bitin bozulmuş olduğu düşünülürse, Şekil 4.4'ten de görüleceği gibi karşılık düşen n. şifresi çözülmüş metin de karmaşık olarak elde edilir. Ayrıca (n+1). blok da yanlış biti alır. Diğer bloklar bu durumdan etkilenmezler.

Fakat bit silme veya eklemelerin etkisi daha ciddidir. Çünkü geriden gelen tüm bloklar etkilenir. Bu yüzden güvenliğin yanında bütünselliğin de ele alınması gereklidir.

Üçlü DES

DES anahtarı, DES algoritması 2^{56} kere çalıştırılarak çözülebilir. Bu çoğu TMN uygulaması için yeterli olurken, şifreleme anahtarları listesi gibi hassas veriler için uygun değildir. Daha fazla güvenlik için iki farklı anahtar kullanılarak DES iki kere uygulanabilir. Fakat sonuç hala güvenli değildir.

Bir blok metin P k1 ve k2 anahtarları kullanılarak şifrelenir ve sonuçta C şifreli metni ortaya çıkar:

$$E_{k1}(E_{k2}(P)) = C \quad (4.7)$$

Her iki tarafın k1 ile şifresini çözersek:

$$D_{k1}(E_{k1}(E_{k2}(P))) = D_{k1}(C) \quad (4.8)$$

Sonuç olarak:

$$E_{k2}(P) = D_{k1}(C) \quad (4.9)$$

P'nin bilindiği düşünülürse, ki birçok mesaj bir zaman işareti, gönderenin belirteci veya alıcının belirteci ile başlamaktadır, P 'nin tüm olası k2 anahtarları ile şifrelenmesi, C 'nin tüm olası k1 anahtarları ile şifresinin çözülmesi ve sonuçtaki listelerin karşılaştırılması yeterlidir.

Yüksek bir güvenlik için DES algoritması üç kere çalıştırılmalıdır. 3DES bilinen saldırılara karşı dirençlidir. 3DES genellikle Şifrele-Çöz-Şifrele (Encrypt-Decrypt-Encrypt - EDE) kipinde çalışır. Eğer üç anahtar da aynıysa EDE kipinde 3DES tek DES 'e dönüşür.

Dijital Mühürler

Daha önce de değinildiği gibi anahtarlı kıyma veri bütünselliğini sağlar. Diğer bir metot da şifrelenmiş kıyma veya dijital mühürlerin kullanılmasıdır. Bir mesajın dijital mührü, o mesaja herhangi bir gizli anahtar eklemeden kıyılması ve elde edilen mesaj bölümünün DES gibi bir simetrik anahtarlı şifreleme algoritması ile şifrelenmesi sonucu elde edilir. Anahtarlı kıymanın bir avantajı güçlü şifreleme işlemlerine gereksinim duymamasıdır.

4.2.2 Asimetrik Şifreleme

Asimetrik şifrelemede şifreleme için bir anahtar kullanılır ve şifre çözmek için ilk anahtarla yakından ilişkili başka bir anahtar kullanılır. Kullanıcı bir çift anahtar üreterek başlar. Bu çifte özel anahtar denir. Kullanıcı özel anahtardan eşi olan kamusal anahtarı oluşturur. Kullanıcı bu kamusal anahtarı herkese vermeye özgürdür, buradan özel anahtar çıkarılamaz. Anahtarlardan birisi şifreleme için kullanıldığında, diğeri şifre çözmek için kullanılır.

Bir kişinin kamusal anahtarını bilen herkes bu kamusal anahtarla şifreleyerek o kişiye gizli bir mesaj gönderebilir. Ve sadece özel anahtarı bilen alıcı bu mesajların şifresini çözebilir. Bu yüzden özel anahtar kimseye verilmemelidir.

Kamusal anahtarlı şifreleme için kullanılan bazı yöntemler: RSA (Rivest-Shamir-Adelman), El Gamal (Sadece dijital imzalar için kullanılır), Eliptik eğri şifreleme sistemi, Rabin.

RSA en eski ve en sık kullanılan kamusal anahtarlı şifreleme algoritmasıdır. RSA özel anahtarı iki büyük asal sayıdan (daha doğrusu iki büyük asal sayının bir fonksiyonundan) oluşur, buna karşılık gelen kamusal anahtar da bu iki asal sayının çarpımıdır. RSA prosedürü Şekil 4.6'da gösterilmiştir. RSA 'nın avantajı büyük sayılarla işlem yapma zorluğudur. Ancak gelişen matematik ve bilgisayar teknolojisi ile bu durum avantaj olmaktan çıkmıştır. Bu yüzden beklenen kullanım süresi boyunca çözülemeyecek kadar uzun özel anahtarlar kullanılmalıdır. Günümüzde 512 bitlik anahtarlar güvenlidir. Ancak yeterli işlem gücüyle birkaç ayda çözülebilirler. 786 bitlik anahtarlar ise çoğu uygulama için yeterlidir. Kamusal anahtarları sertifikalama gibi kritik uygulamalarda 1024 bitlik anahtarlar kullanılabilir. Önümüzdeki yıllarda güvenliğin sağlanması için bu anahtarlar daha da büyüyecektir.

RSA Prosedürü	Örnek
Kullanıcı bir özel ve kamusal anahtar çifti oluşturur	
Kullanıcı iki asal sayı seçer p ve q.	p = 7, q = 17
n = pxq	n = 7 x 17 = 119
Kullanıcı (p-1)(q-1) ile asal e sayısını üretir	e = 5, (7-1)(17-1) = 96 ile asal
Kullanıcı dxe = 1 mod ((p-1)(q-1)) olacak şekilde bir d sayısı üretir.	d = ((p-1)(q-1)(e-1) + 1)/e = 77 dxe = 385 = 4 x 96 + 1
Kullanıcının kamusal anahtarı (n,e)	(119 , 5)
Kullanıcının özel anahtarı (n,d)	(119 , 77)
Kullanıcıya kendi kamusal anahtarı ile şifrelenmiş bir mesaj gelir.	
Mesajın ikili gösterimi bir m tamsayısı olarak okunur (m<n)	m = 19 (<119)
Şifrelenmiş mesaj $s = m^e \bmod(n)$	$19^5/119 = 20807$ ve kalan = 66
Kullanıcıya s mesajı gönderilir.	s = 66
Kullanıcı özel anahtarı ile şifreyi çözer.	
$m = s^d \bmod (n)$	$66^{77}/119 = 1237...$ kalan m = 19
Euler – Fermat teoremi: p ve q iki asal sayı ve m pxq ile asal olmak üzere: $m^{(p-1)(q-1)} = 1 \bmod (pxq)$	

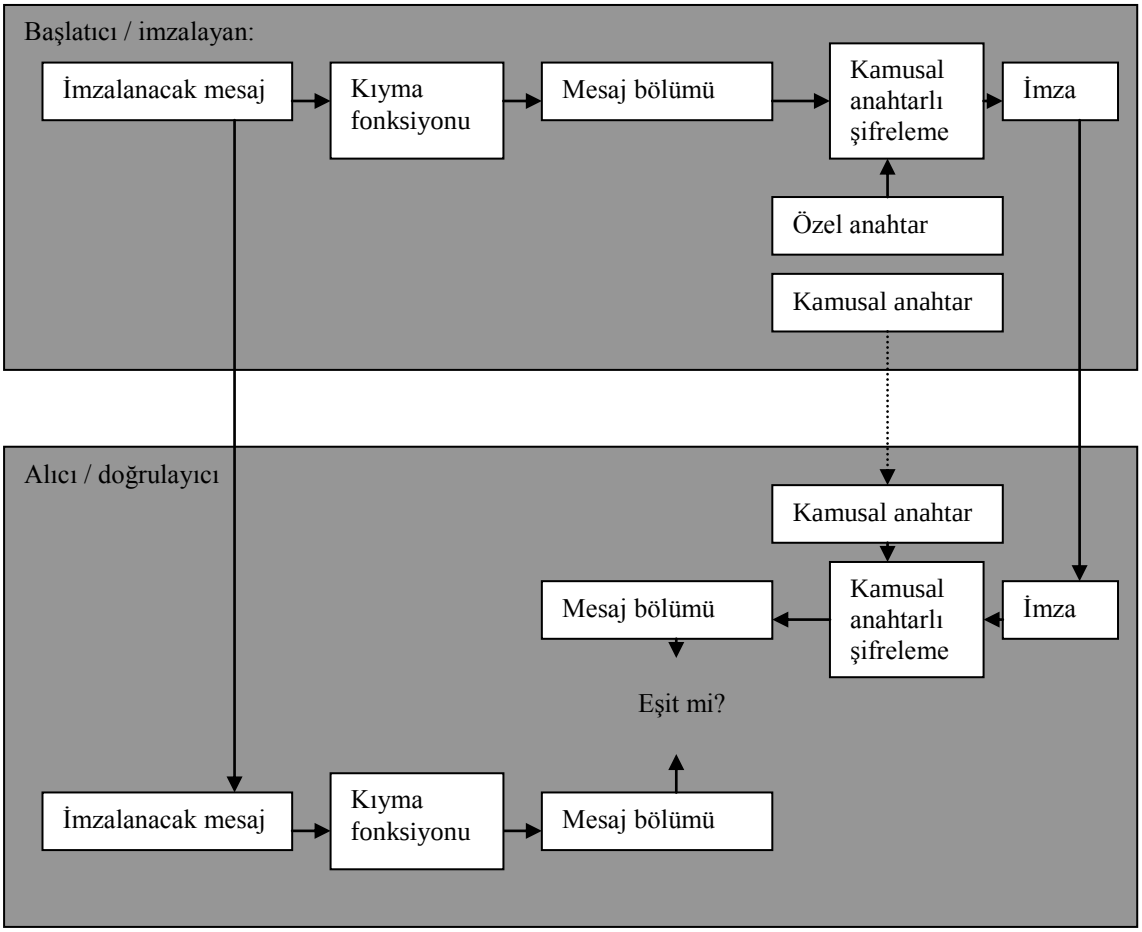
Şekil 4.6: RSA prosedürü örneği

4.3 DİJİTAL İMZALAR

Önemli iş iletimlerinde karşılıklı bağlama için imzalanmış kontratlar kullanılır. Eğer taraflardan biri anlaşmaya uymazsa diğer taraf tazminat için mahkemeye başvurabilir. Elektronik ticaret geliştikçe, kağıttaki imzaların yerini elektronik eşdeğerlerinin alması gerekmektedir. Genellikle kamusal anahtarlı olarak şifrelenmiş dijital imzalar kullanılmaktadır.

Herhangi bir uzunluktaki mesajın dijital imzası, genellikle orijinal mesaj ve gizli bir özel anahtarla elde edilebilen sabit bit uzunluğundaki dizidir. Kamusal anahtarı bilenler tarafından doğru imza olup olmadığı kontrol edilebilir.

Şekil 4.7'de dijital imza uygulamaları için bir örnek gösterilmiştir.



Şekil 4.7: İmzalama ve doğrulama

1. Mesaj M öncelikle kıyma fonksiyonundan geçerek m mesaj bölümüne dönüşür. Verilmiş bir dijital imza standardı için tüm kullanıcılar aynı kıyma fonksiyonunu kullanır.

2. Başlatıcı taraf m mesajını kendi özel anahtarı ile şifreler. Sonuçta M mesajının s imzası oluşur.

3. İmzalanmış mesaj alındıktan sonra (M ve s birlikte), alıcı mesaj bölümü m 'yi 1. adımdaki gibi hesaplar. Ayrıca kamusal anahtar ile imzanın şifresini çözer. Eğer sonuçlar aynı ise göndericinin gerçekten düşünülen kişi olduğu ve mesajın yolda değiştirilmediği anlaşılır.

Dijital imzalar veri orijini doğrulaması ve bütünsellik yanında inkar etmeyi de önler. Mesajı gönderen daha sonra bu gönderimi inkar ederse, kamusal anahtar ile doğrulama yapılarak imzalama yapan kişi ortaya çıkarılabilir.

Ayrıca alıcı mesaja kendi adını ve bir zaman işareti ekleyip imzalayarak geri gönderebilir ve iletimin gerçekleştirildiğini kanıtlar. Eğer alıcı bu bilgiyi göndermiyorsa, bu alıcı ile olan iletişim askıya alınabilir. Çoğu TMN uygulamasında bu yöntem uygulanır.

En sık kullanılan dijital imza algoritması RSA kamusal anahtarlı şifrelemedir. Diğer bir algoritma da Dijital İmza Standardıdır (Digital Signature Standard – DSS). DSS algoritması RSA 'dan farklı olarak şifreleme yapamaz.

İnkar etmeme ve güvenliğin birlikte sağlanması üç yöntemle gerçekleştirilebilir:

1. Mesajı şifreleyip daha sonra şifrelenmiş mesajdan dijital imzayı hesaplamak.

2. Mesajın dijital imzasını hesaplayıp daha sonra mesajı şifrelemek ve şifreli mesaj ile dijital imzayı birlikte iletmek.

3. Mesajın dijital imzasını hesaplayıp daha sonra mesajı dijital imza ile birlikte şifrelemek.

Birinci seçeneğin dezavantajı şifreleme ve dijital imza farklı yerlerde yapılıyorsa, dijital imzayı hesaplayan taraf mesajın doğruluğundan emin olamaz. Ayrıca dijital imzanın üçüncü bir kişi tarafından doğrulanması gerekiyorsa, bu kişi şifreleme

anahtarını da bilmelidir. Bu anahtar başka yerlerde de kullanılmış olabilir ve bilinmesi halinde diğer mesajların da güvenliği tehlike altına girer. Üçüncü seçeneğin dijital imzanın da şifrelenmesi gibi bir yükü vardır. Bu durumda ikinci seçenek öne çıkar.[1]

4.4 SERTİFİKALAR

Kamusal anahtarlı şifrelemenin bir avantajı gizli herhangi bir bilginin paylaşılma zorunluluğunun olmamasıdır. Sadece kamusal anahtar paylaşılır. Bu paylaşım da rehberi yayınlama yoluyla ve güvenli olarak yapılmalıdır. Rehberdeki herhangi bir kamusal anahtarın değiştirilmediğinden emin olunması için kamusal anahtar sertifikalama kullanılır.

Kamusal anahtar sertifikalama ile bir kullanıcının kamusal anahtarı ile o kullanıcının belirteci birbirlerine bağlanırlar. Bu kayıt Sertifikasyon Uzmanı (Certification Authority – CA) tarafından imzalanır. Sertifikanın geçerliliğini doğrulamak için CA ’nın kamusal anahtarı bilinmelidir.

ITU-T X.509 versiyon 3’e göre sertifikaların içermesi gereken bilgiler şunlardır:[1]

- Versiyon (varsayılan versiyon 1’dir)
- Seri numarası
- Sertifikayı imzalamak için kullanılan imzalama algoritması
- CA ’nın adı
- Sertifikanın geçerlilik süresi
- Sertifika düzenlenecek konu adı
- Kamusal anahtar ve bu anahtara karşılık gelen algoritma
- CA ’nın belirteci (sadece versiyon 2 veya 3’te)
- Sertifikalanacak konunun belirteci (sadece versiyon 2 veya 3’te)

- Eklemeler
- Önceki alanların CA tarafından şifrelenmiş dijital imzaları

TMN 'deki her öğe, TMN 'in CA 'sının kamusal anahtarını bilmelidir. Diğer tüm kamusal anahtarlar yerel CA tarafından sertifikalanmalıdır.

4.5 GİRİŞ KONTROL MEKANİZMALARI

Giriş kontrolü oldukça karmaşık olabilir. Bir yönetim sistemine yüzlerce kullanıcı ve uygulama giriş yapar. Bazı kullanıcıların sadece belli yerlere ve belli saatlerde giriş izni vardır. Her potansiyel kullanıcı için bu detayları saklama işi koruma işinden daha kompleks hale gelebilir. Bu işi kolaylaştırmak için bazı mekanizmalar kullanılabilir.

Tüm giriş kontrol mekanizmalarının temeli, giriş kontrolü karar fonksiyonudur. Bu fonksiyon sistem kaynaklarına ulaşmak üzere gelen tüm mesajları inceler ve istekleri kabul veya reddeder. Giriş kontrolü karar fonksiyonu Giriş Kontrol Bilgisi (Access Control Information – ACI) yardımıyla karar verir. Dört çeşit ACI vardır:

- Giriş kontrolü karar fonksiyonu tarafından tutulan **kurallar** kümesi
- Sistem servis isteğinde yer alan **hedef ACI**
- Başlatıcı tarafından istek mesajına eklenen **başlatıcı ACI**
- Ayrı bir isteği belirten **istek ACI**

4.5.1 Kurallar

İki çeşit kural vardır: **kabul** veya **ret**. Örneğin;

- Güvenlik yöneticisinin güvenlik tetkiklerini okuma isteği kabul edilmeli.
- Güvenlik tetkikini silme isteği reddedilmeli

Bir kural tüm sistemi etkileyebilir (**global**). Veya sistemin sadece bir kısmında işleyebilir (**parça kural**). Bir kural tüm başlatıcılara veya sadece başlatıcıların bir kısmına (**varsayılan kural**) uygulanabilir.

Tek bir isteğe birden fazla kural uygulanabilir. Birbirleriyle çelişebilecek kurallar karşısında şu öncelik sıralaması uygulanır:

- Global kurallar parça kurallardan önce gelir.
- Ret kuralları kabul kurallarından önce gelir.
- Başlatıcıya özgü kurallar varsayılan kurallardan önce gelir.

Bir kural haftanın belli günlerinde veya günün belli saatlerinde uygulanabilir. Örneğin; önleyici bakım yapan sistemin çalışmasına, olumsuz etkileri azaltmak için, gece geç saatlerde izin verilebilir.

4.5.2 Başlatıcı ACI

Başlatıcı tarafından belirlenen ACI üç gruba ayrılır:

- Doğrulanmış belirteç başlatıcı
- Yazarı bilinmeyen doğrulanmış başlatıcı
- Yazarı bilinmeyen doğrulanmamış başlatıcı

Doğrulanmış Belirteç Başlatıcı

Giriş kontrol kararı başlatıcını doğrulama belirtecine bağlıdır. Bazı durumlarda bu belirteç bağlantının kurulduğu anda belirlenir ve bağlantı süresince geçerli olur. Daha güvenli bir uygulama her istek mesajı için orijin doğrulaması yapmaktır. Bu senaryo için hedef sistemin başlatıcıyı tanımasını gerektirir. Başlatıcı, hedef sistemin giriş kontrol kurallarının bir veya birkaçında bulunabilir. Bu yöntemin dezavantajı çok sayıda potansiyel başlatıcı bulunması ve giriş kontrolü için pek çok verinin şifresinin çözülmesidir.

Yazarı Bilinmeyen Doğrulanmış Başlatıcı

Bazı durumlarda başlatıcı hedef sistem tarafından tanınmaz ancak yetkili kaynak tarafından verilmiş bazı giriş üstünlüklerine sahiptir. Bu durumda başlatıcı bir Giriş Kontrol Sertifikasına (Access Control Certificate – ACC) sahiptir. Bir ACC genellikle şu bilgileri içerir:

- Yetkiyi veren
- Düzenlenme tarihi
- Geçerlilik süresi
- Kullanıcı
- Yetki verilen hedefler
- Önceki bölümler için yetkiyi verenin dijital imzası

Yazarı Bilinmeyen Doğrulanmamış Başlatıcı

Bu sistem TMN tarafından kullanılmaz.

4.5.3 İstek ACI

İstek ACI ’nin iki amacı vardır:

- İsteğin düzgün bir şekilde oluşturulması
- Giriş kurallarını destekleme

İyi oluşturulmuş bir istek gerekli tüm veriyi (örneğin; ACC ’ler) sağlar. Bu gerekli veriyi istek mesajına bağlamak için, tüm mesaj (ACC ’ler ve başlatıcının kamusal anahtar sertifikası da dahil olmak üzere) Mesaj Doğrulama Kodu (Message Authentication Code – MAC) ile korunur. Yeniden yönlendirmeyi önlemek üzere zaman işareti gibi bir dizi de içerebilir.

Korunmuş bir kaynağa erişim haftanın günü, günün saati veya isteğin yeniliği gibi zamanla veya isteğin yapıldığı yerle ilgili bir sınırlamaya tabii olabilir. Bu giriş

kontrol kurallarını destekleyen bir istek ACI, bir zaman işareti ve başlatıcının ağ adresini içerir.

4.5.4 Hedef ACI

Sıklıkla kullanılan giriş kontrol mekanizmalarına bağlı olarak üç tip hedef ACI vardır:

- Giriş kontrol listeleri
- Yetenekler
- Etiketler

Giriş Kontrol Listeleri

Bir giriş kontrol listesi (Access Control List – ACL) bir kaynağı kullanmaya yetkili başlatıcıların listesidir. ACL ’deki bir başlatıcı ayrı bir kişi, sistem veya uygulama olabilir. Ayrıca bir grubu da tanımlayabilir. Kullanılan tipik doğrulama eklenen ACC ’lerdir.

Yetenekler

Yetenekler bir grup üyeliğinden çok ACC ’de verilen haklardır. Hedef ACI her hedef için kimlerin ACC verme yetkisi olduğunu belirler.

Güvenlik Etiketleri

Giriş kontrol için kullanılan en kolay yöntemdir. Dokümanları güvenilir, gizli, çok gizli olarak üçe ayırır. Her hedef ve başlatıcı için bir güvenlik etiketi (çoğunlukla bir katsayı) tanımlanmalıdır. Eğer başlatıcının güvenlik etiketi hedef ile aynı veya hedeften yüksekse giriş sağlanır.

4.6 DIFFIE – HELLMAN ANAHTAR DEĞİŞTİRME

Diffie – Hellman anahtar değiştirme güvenli olmayan bir kanal üzerinden iki tarafın gizli bir anahtarı (simetrik anahtarlı şifreleme veya anahtarlı kıyma için) paylaşmasıdır. Önce iki taraf büyük bir asal sayı n ve mod n üreteci g sayısı üzerinde anlaşırlar (1'den n-1'e kadar her b sayısı için $g^a = b(\text{mod } n)$) koşulunu sağlayan bir a sayısı vardır). n ve g sayıları gizli olmak zorunda değildir. Algoritma şu adımlardan oluşur:

1. Birinci kullanıcı büyük bir x tamsayısını yollar $X = g^x(\text{mod } n)$.
2. İkinci kullanıcı büyük bir y tamsayısını yollar $Y = g^y(\text{mod } n)$.
3. Birinci kullanıcı $k = Y^x(\text{mod } n)$ sayısını hesaplar.
4. İkinci kullanıcı $k' = X^y(\text{mod } n)$ sayısını hesaplar.

$k = k' = g^{x.y}(\text{mod } n)$ paylaşılmış anahtardır. Kanal dinlenerek g, n, X ve Y elde edilebilir ancak x ve y hesaplanmadan k hesaplanamaz. Bu işlem de uzun sürelidir.

4.6.1 Geçici Diffie – Hellman Anahtar Değiştirme

Diffie – Hellman anahtar değiştirme ile gizli anahtar elde edilemese bile iletime müdahale edilerek tarafların aldığı bilgiler değiştirilebilir. Bu saldırıdan korunmak için mesajlar DSS ile doğrulanabilir. Gerektiğinde kullanıcılar kamusal anahtarlarını paylaşabilirler.

4.6.2 Sertifikalı Diffie – Hellman Parametreleri

Eğer iki kullanıcı aynı g ve n değerini kullanan bir gruba aitse Diffie – Hellman anahtar değiştirme kolaylaşır. Her kullanıcı rastgele, büyük bir x tamsayısı seçerek $X = g^x(\text{mod } n)$ parametresini hesaplar ve kendi X parametresi için sertifika alır. Anahtar değişimi için iki kullanıcı sertifikalı parametrelerini birbirlerine gönderirler. Bu parametrelerden gizli anahtar hesaplanır.[1]

4.7 DOĞRULAMA PROTOKOLLERİ

Doğrulama protokolleri güvenlik protokolleridir ve iletim protokollerinden farklıdır. Güvenlik protokolleri verini tipini, güvenlik özelliklerini belirler, içeriğiyle ilgilenmez.

İyi bir doğrulama protokolü güvenli olmayan bir ağ üzerindeki iki kullanıcının birbirlerinin kimliğinden emin olmalarının sağlar. Güvenli olmayan ağda mesajlar görüntülenebilir, silinebilir, geciktirilebilir, değiştirilebilir, yeniden gönderilebilir, yeniden sıralanabilir veya yeniden yönlendirilebilir. Doğrulama protokolleri bu uygulamalarla diğer kullanıcının taklit edilmesini engellerler.

Doğrulama iki taraf arasında olabilir veya üçüncü tarafa bırakılabilir. İkinci durumda üçüncü tarafın da doğrulaması gerektiğinden ve mesaj sayısı artacağından TMN güvenliği için sıklıkla kullanılmaz.

Direk doğrulama protokolleri ikiye ayrılır:

- Hedef – cevap doğrulama
- Bildirim doğrulama

4.7.1 Hedef – Cevap Doğrulama

Bir örnekle açıklarsak:

- Bir kullanıcı diğerine güvenli bir iletim yapmak istiyor. Önce bağlantı kurma isteği gönderir. Daha önce doğrulama amacıyla hiç kullanmadığı bir bit dizisi üretir (bu bir zaman işareti, rastgele bir sayı, bir sıra numarası veya bunların bir karışımı olabilir).

- Karşı taraf bu istek mesajını kendi özel anahtarı ile şifreler ve geri gönderir. Bu mesaja doğrulama amacıyla kendi ürettiği bit dizisini de ekleyebilir.

- İlk taraf cevabın şifresini karşı tarafın kamusal anahtarı ile çözer ve ilk gönderdiği mesaj ile karşılaştırır. Daha sonra cevabı kendi özel anahtarıyla şifreleyerek gönderir.

- Karşı taraf mesajın şifresini ilk tarafın kamusal anahtarı ile çözer ve ilk gelen mesajla karşılaştırır.

Burada üç mesaj değiştirilmiştir. Bu bir hedef – cevap doğrulama için minimum sayıda mesajdır. Ek mesajlar ile güvenlik arttırılabilir.

Bu yöntemin web uygulamalarında ideal olmasının sebebi karşı tarafın kimliği hakkında bilgi gerektirmemesidir. Sadece kamusal anahtarın bilinmesi gerekmektedir. Taraflar birbirlerine kamusal anahtar sertifikalarını gönderebilir veya bir rehberden alabilirler.

4.7.2 Bildirim Doğrulama

Her ne kadar hedef – cevap doğrulama popüler olsa de OSI gibi bağlantı kurulumu için iki mesaj değişimi isteyen ortamlarda çalışamaz. Aynı güvenlik düzeyi iki mesajla da sağlanabilir. Mesaj sayısını bir azaltmanın bedeli ek olarak gelen durum bilgileridir. Bir örnekle açıklarsak:

- Bir kullanıcı diğerine güvenli bir iletim yapmak istiyor. Önce bağlantı kurma isteği gönderir. Daha önce doğrulama amacıyla hiç kullanmadığı bir bit dizisi üretir (bu bir zaman işareti, rastgele bir sayı, bir sıra numarası veya bunların bir karışımı olabilir). Bu mesaja karşı tarafın belirtecini de ekler ve kendi özel anahtarı ile imzalar. İmza ile metni birlikte gönderir.

- Karşı taraf mesajı aldığında imza ile metni karşılaştırır. Cevap mesajı olarak hiç kullanmadığı bir bit dizisi üretir (bu bir zaman işareti, rastgele bir sayı, bir sıra numarası veya bunların bir karışımı olabilir). Bu mesaja ilk kullanıcının belirtecini de ekler ve kendi özel anahtarı ile imzalar. İmza ile metni birlikte gönderir.

- İlk kullanıcı mesajı alır ve imza ile metni karşılaştırır.

Bu protokolün önemli bir zayıf yanı vardır: İlk kullanıcının mesajı kopyalanarak daha sonra karşı tarafa gönderilebilir. Bunu engellemek için taraflar bir liste halinde daha önce hangi kullanıcıdan hangi bit dizilerini aldıklarını tutmalıdırlar. Eğer iki kullanıcı sıklıkla haberleşiyorsa bu liste büyür, veya kullanıcılardan biri pek çok kişi

ile haberleşiyorsa çok sayıda liste oluşur. Her bağlantı isteğinde önce listedeki mesajlarla karşılaştırılması gereklidir.

Üretilen her rastgele dizide bir bitme süresi olmalıdır. Bu durumda o bitme süresi sonuna kadar saklanan mesajlar daha sonra listeden silinebilirler.

Bu çözüm pek çok TMN uygulamasında daha da kolay olarak kullanılmaktadır: kullanıcı mesajın başlangıç tarihini rastgele dizi olarak kullanır ve bu da sistem saatine eşdeğerdir. Karşı taraf da gecikmelerden mesajın kopyalanıp gönderildiğini anlayabilir.

Bu kolay yöntemde bile durum bilgisi içerilmelidir. Eğer kopyalanan mesaj yeterince çabuk gönderilirse gecikme dikkat çekmeyebilir. Taraflar o kullanıcıdan aldıkları son mesajın zaman işaretini saklayarak, daha sonraki mesajların bu işaretten büyük bir işarete sahip olmalarını bekleyebilir. Kabul edilen her mesajdan sonra da son zaman işareti güncellenir.

Fakat eğer ağdan kaynaklanan yöntemlerle bir mesaj diğerine göre daha geç geliyorsa (örneğin; farklı yollar kullanıyorlarsa) işler karışır. Bu durumda ağ gecikmesi için bir süre tanımlanarak bu süreden daha büyük gecikmelere dikkat edilebilir.

Daha da kötüsü kullanıcını saati bozulabilir ve tüm mesajlara aynı zaman sabitini ekleyebilir veya saat yeniden başlatılabilir ve mesajlara daha eski bir zaman sabiti eklenebilir. Bu durumda mesajlar kabul edilmez. Burada izlenecek yol sürekli artan zaman sabitleri kullanmaktır. Bunu oluşturmak için dört çeşit zaman vardır:

1. GMT astronomik olarak doğru saattir.

2. Sistem Saati (System Clock – SC) sistem tarafından üretilir.

3. Sanal Zaman (Virtual Time – VT) sadece güvenlik uygulamaları tarafından kullanılır (diğer sistem elemanları da kullanılabilir).

4. Dış Zaman (External Time – ET) gelen bir PDU ’dan alınan zamandır.

VT her giden PDU için üretilen ve her gelen PDU 'dan alınan zaman işaretini okur. Her okumada VT değeri güncellenir ve kalıcı bellekte saklanır. Güncelleme prosedürü:

$$\text{eğer VT} < \text{SC ise VT} = \text{SC}, \tag{4.10a}$$

$$\text{değilse VT} = \text{VT} + 1 \text{ tick} \tag{4.10b}$$

burada tick VT 'nin artabildiği en küçük miktardır (tipik olarak 10 ms). Eğer SC durursa VT kullanımına göre artmaya devam eder. SC, GMT 'ye göre güncellendiğinde ise gerçek zamanı yakalar.

Bildirim doğrulama, hedef – cevap doğrulamanın yanında mesajların yeniden sıralanmasını ve gecikmesini de tespit eder. Eğer sıra numarası eklenirse mesaj silinmesini de tespit edebilir.

4.8 GÜVENLİK SERVİSLERİ VE GÜVENLİK MEKANİZMALARI

Tablo 4.1'de güvenlik servisleri ve bunları destekleyen güvenlik mekanizmaları verilmiştir.

Tablo 4.1: Güvenlik servisleri ve güvenlik mekanizmaları

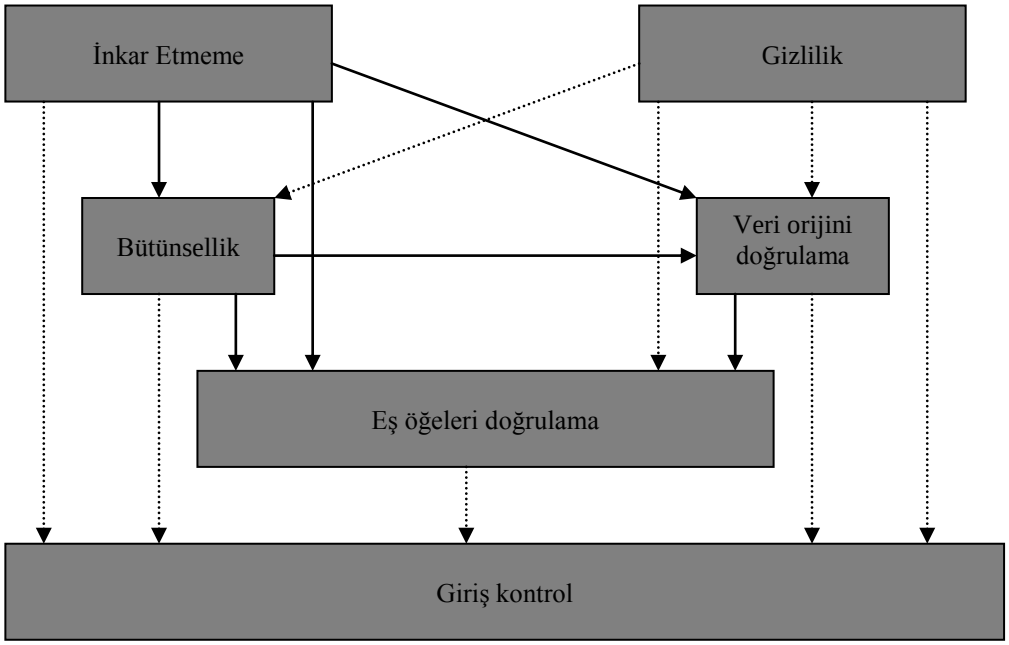
Güvenlik Servisleri	Güvenlik Mekanizmaları					
	Giriş Kontrol Mekanizması	Anahtarlı Kıyma	Simetrik Şifreleme	Asimetrik Şifreleme	Dijital İmzalar	Dijital Sertifikalar
Giriş Kontrolü	√					
Eş Öğeleri Doğrulama		√	√	√	√	√
Veri Orijini Doğrulama		√			√	
Bütünsellik		√			√	
Gizlilik			√	√		
İnkar Etmeme					√	

Tablo 4.2'de ise güvenlik tehditleri ile güvenlik mekanizmaları eşleştirilmiştir.

Tablo 4.2: Güvenlik mekanizmaları ve güvenlik tehditleri

Güvenlik Mekanizmaları	Güvenlik Tehditleri						
	İzinsiz Giriş	Gizlice Dinleme	Yerine Geçme	Veri Değiştirme	İnkar Etme	Yeniden Alma, Yeniden Yönlendirme ve Silme	Ağı Taşırma
Giriş Kontrolü	√			√			√
Anahtarlı Kıyma	√		√	√		√	
Simetrik Şifreleme	√	√	√				
Asimetrik Şifreleme	√	√	√				
Dijital İmzalar	√		√	√	√		
Dijital Sertifikalar	√		√				

Şekil 4.8'de ise güvenlik servislerinin kısmi sıralaması verilmiştir. Şekildeki oklar birbirlerini sağlayan servisleri, kesikli oklar da destekleyen veya uygun olan ancak her zaman sağlamayan servisleri göstermektedir.



Şekil 4.8: Güvenlik servislerinin kısmi sıralaması

5. DESTEK MEKANİZMALARI

Geçen bölümde ele alınan güvenlik mekanizmaları saldırıları önleme amaçlıdır. Ancak tüm saldırılar engellenemez. Bu tür saldırıların gerçekleşme ihtimaline karşılık, bunları tespit etmek ve etkilerini engellemek gereklidir. Güvenlik tetkikleri bu konuda kullanılabilir ancak anında etki etmezler. Güvenlik alarmları akla gelen ilk yöntemdir.

5.1 GÜVENLİK ALARMLARI

Güvenlik ile ilgili bir sorunla karşılaşıldığında (örneğin; kullanım süresi dolmuş bir şifre veya anahtar kullanıldığında veya yetkili olunmayan bir sisteme girilmeye çalışıldığında) yönetilen nesne bir güvenlik alarmı üretir ve yöneticisine iletir. Bu bilgi daha sonra olası saldırı hakkında bilgi edinmek için kullanılır. Bir güvenlik alarmında yer alan bilgiler şunlardır:

- Güvenlik sorunundan etkilenen nesnenin kimliği
- Güvenlik sorununun zamanı
- Olay çeşidi (bütünselliğin bozulması, işlemsel saldırı, fiziksel saldırı, güvenlik servislerine veya mekanizmalarına saldırı veya zaman bölgesine saldırı)
- Güvenlik alarmının nedeni (bu parametrenin olası değerleri olay çeşidine bağlıdır)
- Güvenlik alarmının önem derecesi – orta, kritik, büyük, küçük veya uyarı
- Güvenlik alarmı algılayıcısı – güvenlik sorununu algılayan
- Servisi kullanan – güvenlik alarmına neden olan servisi kullanan
- Servis sağlayıcı – güvenlik alarmına neden olan servisi sağlayan

5.2 GÜVENLİK TETKİK KAYDI

Güvenlik ile ilgili olayların, özellikle de güvenlik alarmlarının, kayıtlarının tutulmasıdır. Bu veri daha sonra olası saldırı hakkında bilgi edinmek için kullanılabilir. Bir güvenlik tetkiki kaydında şu bilgiler bulunabilir:

- Güvenlik sorunundan etkilenen nesnenin kimliği
- Olay çeşidi (servis raporu veya kullanım raporu)
- Güvenlik sorununun zamanı
- Eğer olay çeşidi servis raporu ise, bu raporun nedeni

Bir servis raporu, servis sağlama, inkar etme, iyileştirme gibi olayların raporlarından oluşur. Bir kullanım raporu, güvenlik ile ilgili istatistiksel bilgilerden oluşur.

Güvenlik tetkik kaydında elde edilen bilgiler güvenlik saldırılarını analizlerinin yapılmasında ve güvenlik politikalarının gözden geçirilip gerekli değişikliklerin planlanmasında kullanılabilir. Güvenlik tetkik kaydı ile çeşitli kaynaklardan veri toplanabilir. Böylece ağın hangi bölümlerinin saldırıya uğradığı tespit edilebilir. Güvenlik tetkik kaydı şu andaki saldırıları engelleyemese de gelecektekiler için yardım eder.

5.3 ANAHTAR DAĞITIMI

Bahsedilen tüm güvenlik mekanizmaları güvenli dağıtıma (veri orijini doğrulama, bütünsellik ve özel anahtarların güvenliği) ve gizli verinin yönetilmesine gereksinim duyar. Burada yönetici ile kullanıcı arasındaki anahtar alışverişi ele alınmıştır. Bu sistem Elektronik Bağlamada (Electronic Bonding – EB) –LEC ’ler ile Ara Santral İşletenler (Inter Exchange Carriers – ICs) arasındaki elektronik arabağdaşım– kullanılır.

5.3.1 Anahtar Listeleri

EB ’de tanımlanan anahtar dağıtımı prosedürü az kullanıcılı sistemlerde etkili olurken, kullanıcı sayısı arttığında karmaşıklaşır. Burada sadece servis sağlayıcının TMN ’i ve müşteri arasındaki X arabağdaşımı üzerindeki EB ele alınmıştır.

Servis sağlayıcı müşterisine anahtarların listesini sunar. Listedeki varsayılan anahtar sayısı 1000’dir.

Müşteri belli zaman aralıklarıyla (bir günü geçmeyecek şekilde) anahtarları değiştirir. Değiştirilen anahtarlar tekrar kullanılamazlar. Listenin bir yedeği servis sağlayıcıda ve kulanıcıda bulunur.

Listedeki anahtarların çoğunun kullanılması halinde veya bir yıl geçtiğinde (hangisi önce gerçekleşirse) servis sağlayıcı müşteriye yeni bir liste gönderir. Buradaki anahtar numaraları 1001-2000 arasındadır.

Her yeni liste bir önceki listedeki kullanılmamış üç anahtar kullanılarak üçlü 3DES ile şifrelenir ve gönderilir.

5.3.2 Kamusal Anahtar Dağıtımı

Çeşitli kullanıcıların kamusal anahtarları herkesin ulaşabileceği bir noktaya yerleştirilebilir. Güvenli bir bağlantı kurma isteyen kişi mesajını karşı tarafın kamusal anahtarı ile şifreleyerek gönderir. Şifreleme algoritmalarını karmaşıklığı sebebiyle daha sonra kullanılmak üzere bir simetrik anahtar da gönderilebilir. Bu simetrik anahtar da karşı tarafın kamusal anahtarı ile şifrelenir. Ayrıca veri orijini doğrulaması için kullanıcının özel anahtarı ile imzalanabilir.

Kamusal anahtarlı şifreleme ile gizli bilgilerin paylaşılmasına gerek kalmadan güvenli iletim sağlanır. Kamusal anahtarların tutulduğu rehberde ACL ve ACC’ler de tutulabilir. Bu işlem için X.500 rehberi tanımlanmıştır.

5.4 REHBER

X.500 rehberi kamusal anahtarlı şifrelemeye dayanır. Kullanıcılar arasında paylaşılması gereken bilgiler için kullanılır. Her kullanıcı Rehbere kayıt yapabilir. Diğer bir TMN kullanıcısının kamusal anahtar sertifikası ve ACC 'si rehberden istenebilir.

5.4.1 Otomatik Kayıt

TMN öğelerinin kamusal anahtar sertifikalarının rehbere girilmesi ve güncellenmesi ciddi bir yüktür. Bu yük her öğenin kendisini otomatik olarak kaydetmesi ile azalabilir. TMN rehberi için bir gereklilik Rehber Bilgi Tabanına (Directory Information Base – DIB) otomatik kayıt yapılmasını desteklemesidir. Yanlış kayıtlar yapılmasını önlemek amacıyla, otomatik kayıt yapılan mesajların orijinlerinin doğrulanması ve bütünsellikleri kontrol edilmelidir. Kayıt yaptıran öğe hakkında bir şey bilinmediği için, bu yeni öğe ile gizli bilgilerin paylaşılmasından ziyade kamusal anahtarlama kullanılır. Burada her öğe kendi özel ve kamusal anahtarı ile CA 'nın kamusal anahtarını bilmelidir. Daha az trafiğe neden olur. Ayrıca adımlar halinde uygulanarak esneklik sağlar.

Rehber güvenliği için en basit yöntem TMN 'deki tek güvenlik bölgesidir. Bu yöntemle otomatik kayıt ve rehbere güvenli erişim aşağıdaki gibi olur:

1. Bir öğe ilk kurulduğunda (rehber de dahil olmak üzere), CA 'dan aşağıdaki dört maddeyi kendi güvenlik bölgesine alır:

    - Belirteç
    - CA 'nın kamusal anahtarı
    - Kendi özel ve kamusal anahtarı
    - CA tarafından imzalanmış bir sertifika

    Bu maddeler CA tarafında güncellenebilir. Özel anahtarlar sadece ilgili öğe tarafından okunabilir.

Tüm TMN öğeleri bu dört maddeyi almaya ve güvenli olarak saklamaya programlanmalıdır. Bunun için elektronik olarak programlanabilen okumalı bellek kullanılabilir (Electronically Programmable Read Only Memory – EPROM).

2. Bir öğe rehbere ilk defa kaydedildiğinde, CA 'dan aldığı sertifikayı sunar. Rehberde CA 'nın kamusal anahtarı bulunduğundan sertifika kontrol edilebilir. Buna göre öğe kaydedilir. İlave bilgiler eklenmesi veya bulunanları değiştirilmesi için öğe tarafından imzalanmış mesajlar gönderilebilir.

3. Kayıtlı bir öğe rehber ile güvenli bir iletişim sağlamak isterse imzaladığı mesajları gönderebilir. Burada imzalanan mesaj zaman işareti gibi tekrarlanmayan bitlerden oluşan bir diziye sahip olmalıdır. Rehber de aynı prosedürü izler ve ek olarak kendi sertifikasını ekler.

4. Eğer bir öğe diğerinin kamusal anahtarını isterse, rehber istek yapan öğeye ilk kayıtta aldığı sertifikayı yollar. Tüm öğeler CA 'nın kamusal anahtarını bildiği için de sertifikanın doğruluğu kontrol edilebilir.

5. Eğer bir öğe farklı bilgiler de isterse ve bu bilgilerin bütünlüğünün korunması gerekiyorsa, rehber bilgileri kendi özel anahtarıyla imzalayarak gönderir. Bu durumda rehberin sertifikası ve kamusal anahtarı da gönderilmelidir.

6. Eğer bir öğe farklı bilgiler de isterse ve bu bilgilerin gizliliğinin korunması gerekiyorsa, rehber bilgileri isteği yapanın kamusal anahtarıyla (rehberin DIB 'inde bulunmaktadır) şifreleyerek gönderir. Eğer iletilen mesaj çok büyükse (örneğin; birkaç bin oktet, bu durum rehber servislerinde nadiren görülür) kamusal anahtarlı şifreleme yeterli olmayabilir. Bu durumda rehber simetrik şifreleme kullanabilir (örneğin; DES) ve kullandığı gizli anahtarı alıcının kamusal anahtarıyla şifreleyerek gönderir.

7. 5 ve 6. Maddeler bir araya getirilerek bütünsellik ve gizlilik sağlanabilir.

Bu örnek durum farklı giriş kontrol yöntemleri veya farklı TMN 'lere ait çeşitli güvenlik bölgeleri kullanılarak genelleştirilebilir.

5.4.2 Rehber Giriş Kontrolü

Yukarıdaki basit örnekte rehber giriş kontrol için sadece ACI 'a ihtiyaç duyar. Bu durum eğer aşağıdaki güvenlik politikaları mevcutsa yeterlidir:

- Herhangi bir sertifikalı öğe rehberdeki kayıtları okuyabilir.
- CA rehberden kayıt silebilir.
- Her sertifikalı öğe sadece kendi kayıt bilgilerini değiştirebilir.

Ayrıca hangi öğelerin hangi giriş haklarına sahip olduğunu gösteren ACL 'ler de kullanılabilir.

Güvenlik etiketlerine dayalı bir giriş kontrolde her öğe bir veya daha fazla güvenlik grubuna dahil olabilir. Her güvenlik grubu belirlenmiş özel haklara sahiptir. Bir istek mesajı, bu istek için yeterli haklara sahip bir güvenlik grubundan geldiğini de belirtmelidir. Bunun için başlatıcı, uygun gruba ait olduğunu gösteren ve CA tarafından oluşturulup imzalanmış bir sertifikaya sahip olmalıdır. Eğer birkaç gruba üye olunmuşsa o sayıda sertifikaya sahip olunabilir. Her sertifika öğenin ve bir veya birkaç grubun belirtecinden oluşur. Ayrıca sertifikanın düzenlenme tarihi, geçerli olduğu süre ve diğer bilgileri de içerebilir.

Güvenlik etiket sertifikasının izinsiz olarak yeniden gönderilmesini engellemek amacıyla, özel olarak istek mesajına bağlanabilir. Bu bağlama için güvenlik etiket sertifikası başlatıcı tarafından imzalanmış bir mesajın içinde yer alabilir. Bu mesaj zaman işareti gibi tekrar etmeyen bitlerden oluşan bir diziye sahip olmalıdır. Güvenlik etiket sertifikası istek mesajının yanında da gönderilebilir. Başlatıcı mesajı ve sertifikayı imzalar. İkinci durumda, eğer rehber sertifikanın bir kopyasını saklarsa, yeni isteklerde sertifika sunmaya gerek kalmaz.

Bir ACC, CA tarafından oluşturulup imzalanır. Ait olduğu öğenin belirtecini ve o öğenin sahip olduğu giriş haklarını içerir. Ayrıca sertifikanın düzenlenme tarihi, geçerli olduğu süre ve diğer bilgileri de içerebilir. ACL 'lerin rehberde saklanmasına

rağmen, ACC ’ler ait oldukları öğelerde saklanır. ACC ’nin bir mesaja bağlanması güvenlik etiket sertifikalarında olduğu gibidir.

5.4.3 Çoklu Güvenlik Bölgesi

Eğer bir TMN farklı güvenlik bölgelerine sahipse ve farklı bölgelerdeki öğeler birbirleriyle haberleşmek istiyorsa, sertifkasyon yolları kurulmalıdır. Basit bir sertifikasyon yolunda, bir öğenin CA ’sı o öğeye, diğer bir öğenin geçerli bir CA ve CA ’nın kamusal anahtarı olduğunun sertifikasını verir.

Sertifikasyon yolu bir TMN ’deki rehber ile diğer bölgelerdeki öğeler arasında da güvenli iletişimi sağlar.

5.5 GÜVENLİK PROTOKOLLERİ

Şifreleme ve dijital imzalar gibi güvenlik mekanizmaları tehditlere karşı iyi çözümler olmalarına rağmen bütünsellik konusunda zayıftırlar. Güvenli bir mesajda, alıcı imzanın esas mesajdan önce mi sonra mı geldiğini, imzanın bir veya daha çok tamsayıdan mı yoksa bir oktet dizisinden mi meydana geldiğini, mesajı imzalamak ve şifrelemek için hangi algoritmaların ve anahtarların kullanıldığını bilmelidir. Bu düğümü çözmek için özel protokoller önerilmiştir.

Bu protokollere geçmeden önce anlambilim (semantic) üzerinde durulursa:

> **Haberleşme protokolleri** (örneğin; IP, TCP veya X.25) sözdizimle (syntax) ve veri değişimi için gerekli prosedürlerle ilgilenir.
>
> **Güvenlik protokolleri** (örneğin; Diffie – Hellman anahtar değiştirme, hedef – cevap doğrulama) güvenlik fonksiyonlarını yerine getirmek için iletilen bilgilerin sadece anlambilimiyle ilgilenir (semantic).
>
> **Güvenli haberleşme protokolleri** (örneğin; SSL3, STASE-ROSE) mesajların hem anlambilimiyle hem de sözdizimiyle ilgilenir.[1]

5.5.1 Güvenli Haberleşme Protokollerinin Yapısı

Bir güvenli haberleşme protokolünün iki fazı vardır:

1. Tokalaşma

2. Güvenli transfer

Bunlardan her biri bir çift (potansiyel olarak çakışan) faza sahiptir.

Tokalaşma fazında güvenlik içeriği doğrulanır. Güvenlik içeriği, oturum boyunca hangi mekanizmaların ve algoritmaların kullanılacağını belirler. Ayrıca kullanılacak şifreleme anahtarları gibi güvenlik parametrelerini de belirleyebilir. Genellikle gizli güvenlik anlaşmalarından kaçınılır ve varsayılan içerik otomatik olarak alınır.

Doğrulama üç şekilde yapılabilir:

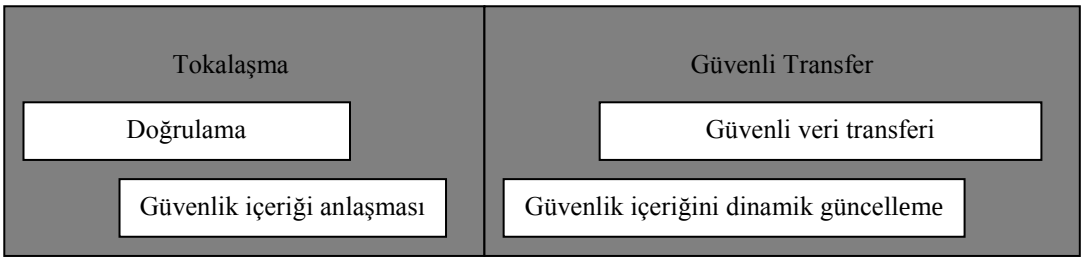
1. **Sadece başlatıcıyı doğrulama** bir hedef sisteme güvenli bir iletim ağı üzerinden pek çok istek varsa kullanılır.

2. **Sadece hedefi doğrulama** başlatıcı kamusal bir hedefe (örneğin; bir Web sitesi) güvenli olmayan bir ağ üzerinden bağlanmaya çalıştığında kullanılır.

3. **İki yönlü eş öğeleri doğrulama** tüm TMN uygulamalarında kullanılır.

Doğrulama mekanizmaları bildirimli (doğrulanan protokol hakkında bilgi gerektiren) veya bildirimsiz (birkaç mesaj değişimi gerektiren) olabilir.

Güvenli transfer fazında iki taraf kararlaştırılan güvenlik seviyesinde veri alışverişi yaparlar. Ayrıca güvenlik içeriğinin otomatik olarak güncellenmesini de sağlar (örneğin; farklı mesajlar için farklı algoritmaların kullanılması, şifreleme anahtarının güncellenmesi). Şekil 5.1'de bir güvenli haberleşme protokolünün hayat döngüsü gösterilmiştir.



Şekil 5.1: Bir güvenli haberleşme protokolünün hayat döngüsü

Yukarıda anlatılan güvenli haberleşme protokolleri bağlantı-yönelimli iletimler için geçerlidir. Bağlantısız iletimlerde (örneğin; datagram iletimi) güvenlik içeriği anlaşması yapılamaz, sadece gönderici doğrulaması desteklenir.

Bir güvenli haberleşme protokolü, herhangi bir haberleşme protokolünde olduğu gibi, iki protokol durum makinesi arasında değiştirilen mesajları belirler. Tipik bir protokol durum makinesi hangi bilgilerin koruma gerektirdiğine veya hangi şifreleme algoritmasının kullanılacağına karar veremez. Bu yüzden uzak eş öğeyle olan protokol arabağdaşımlarından ayrı olarak lokal **yönetimsel arabağdaşımlara** sahip olmalıdır. Yönetim arabağdaşımlarından biri hangi güvenlik parametrelerinin kullanılması gerektiğini söylerken, diğer bir tanesi şüpheli durumlarda alarm verilmesini sağlar.[1]

5.5.2 Katmanlarla Güvenlik

Bir güvenli haberleşme protokolü bir uygulama içinde veya yedi katmanlı OSI 'nin oturum dışındaki herhangi bir katmanında gerçekleştirilebilir. Hangisi tercih edilmelidir?

Bir güvenli haberleşme protokolünü bir uygulama içinde gerçekleştirmek, ağ yönetim uygulamalarıyla yeterince meşgul olan uygulama programcıları için en az çekici olandır. Fakat bu seçim yazılım kullanımını azaltır. İstenen güvenlik servisleri haberleşme protokol yığını tarafından desteklenmediğinde son çare olarak düşünülebilir.

Protokol yığınında aşağılara inildiğinde güvenlik sağlamak için esneklik kaybedilir. N. katmanda N+1 seviyeli bir PDU bir Servis Veri Birimi (Service Data Unit – SDU) olarak alındığında anlaşılmaz bir oktet dizisidir. Bu yüzden sunum katmanının altında seçili alan koruması yapılamaz. Daha da kötüsü iletim katmanı büyük mesajları bölerek, küçük mesajları da birleştirerek belli büyüklükte bloklar oluşturur. Mesaj bazında güvenlik kavramı ortadan kalkar.

Fiziksel tabaka güvenliği ise tüm dizi için aynı güvenliği sağlar ve trafiği arttırır. Fakat bu tip bir güvenlik TMN uygulamaları için yeterli değildir.

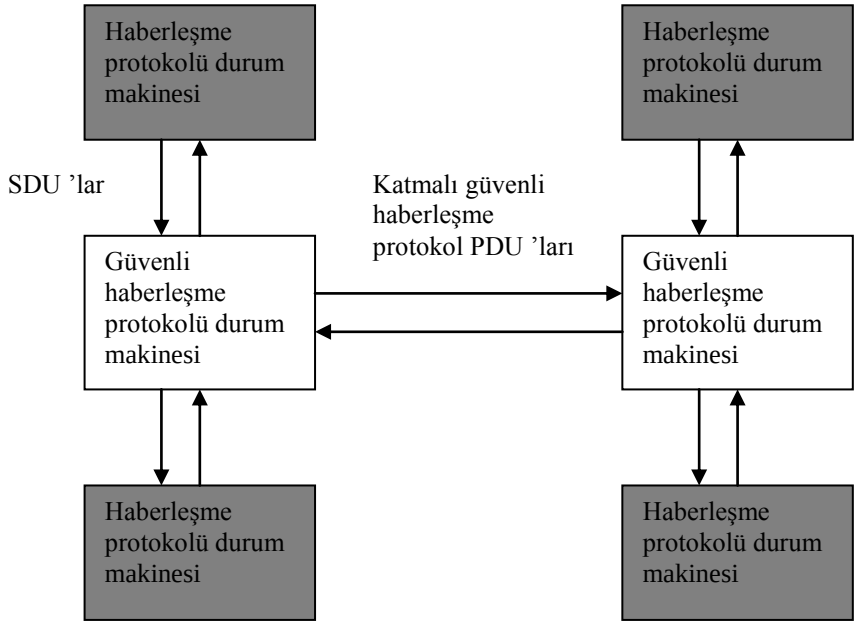
Veri bağı katmanı güvenliği, ayrı hatlardan akan verini korunmasını sağlar. Tüm değişimlerin tek bir paylaşılmış hat üzerinden olduğu LAN güvenliği için idealdir. Fakat TMN çok daha geniş çaptaki veri iletişim hatlarının kullanılmasını gerektirdiğinden, bu güvenlik tipi de TMN için yeterli değildir.

5.6 GSS-API

Bir şifrelemeci için tasarladığı şifreleme algoritmaları kendi seçimidir. Ancak bir sistem mühendisi veya güvenlik yöneticisi, farklı katmanlardaki farklı güvenlik algoritmalarını birlikte çalıştırmak zorundadır. Bu durumda en güzel yöntem bir güvenlik algoritmaları kütüphanesi kurulmasıdır. Böyle bir kütüphane herhangi bir sistemdeki, herhangi bir katmandaki, herhangi bir protokol tarafından kullanılabilir. Kütüphane içinde güvenlik algoritmalarının güncelleştirilmesi yapılabilir. Böylece birlikte çalışabilirlik artar.

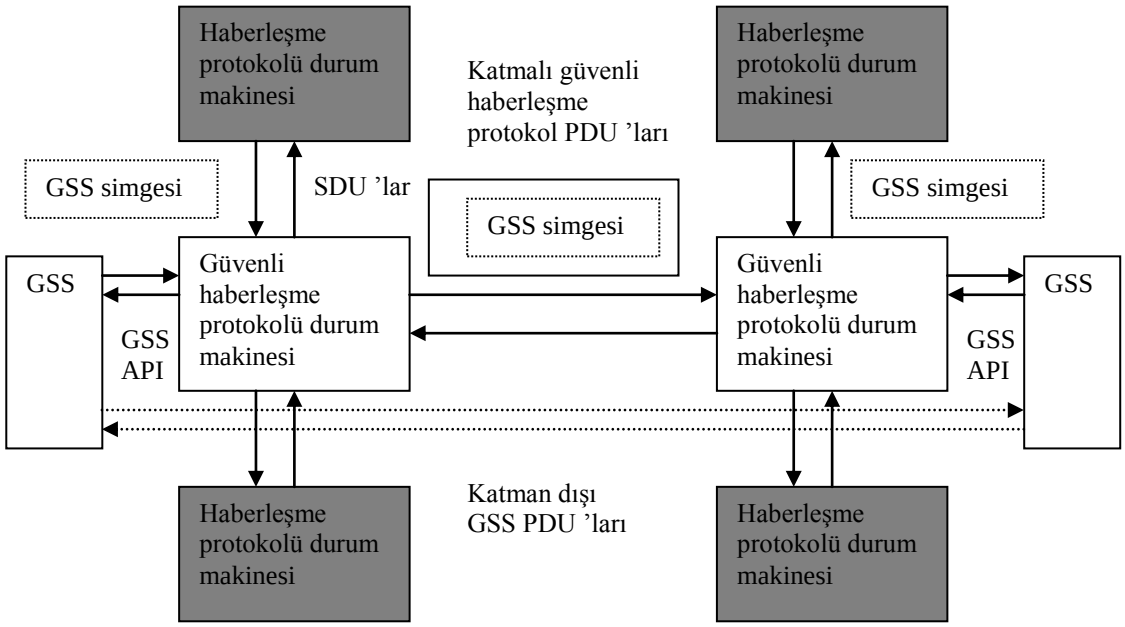
Bu kütüphane için Genel Güvenlik Servis Uygulama Programlama Arabağdaşımı (Generic Security Service Application Programming Interface – GSS-API) standardı kullanılır.

Herhangi bir uygulama, istenilen güvenlik fonksiyonlarının (örneğin; doğrulama belgelerinin oluşturulması) gerçekleştirilmesi için GSS 'i kullanabilir. Bir sistemdeki GSS 'in oluşturduğu belgeleri diğer sistemdeki GSS 'in tanıması için ortak dönüşümlerle belge hazırlamalıdırlar. Ayrı GSS 'ler arasında birlikte çalışabilirlik uygulanabilmesi için GSS-API, API 'nin yanında temel bir GSS içi protokol daha tanımlar. Bu yapı **katman dışı güvenli haberleşme protokolü** olarak adlandırılır. Bu yapıya geçmeden önce Şekil 5.2'de gösterilen geleneksel katmanlı güvenli haberleşme protokolü incelenirse farklılık daha kolay anlaşılır. Burada katmanın alt ve üstündeki haberleşme protokolü durum makineleriyle SDU 'lar, uzak eşdeğeriyle de PDU 'lar değiştirilir.



Şekil 5.2: Katmalı güvenli haberleşme protokolü

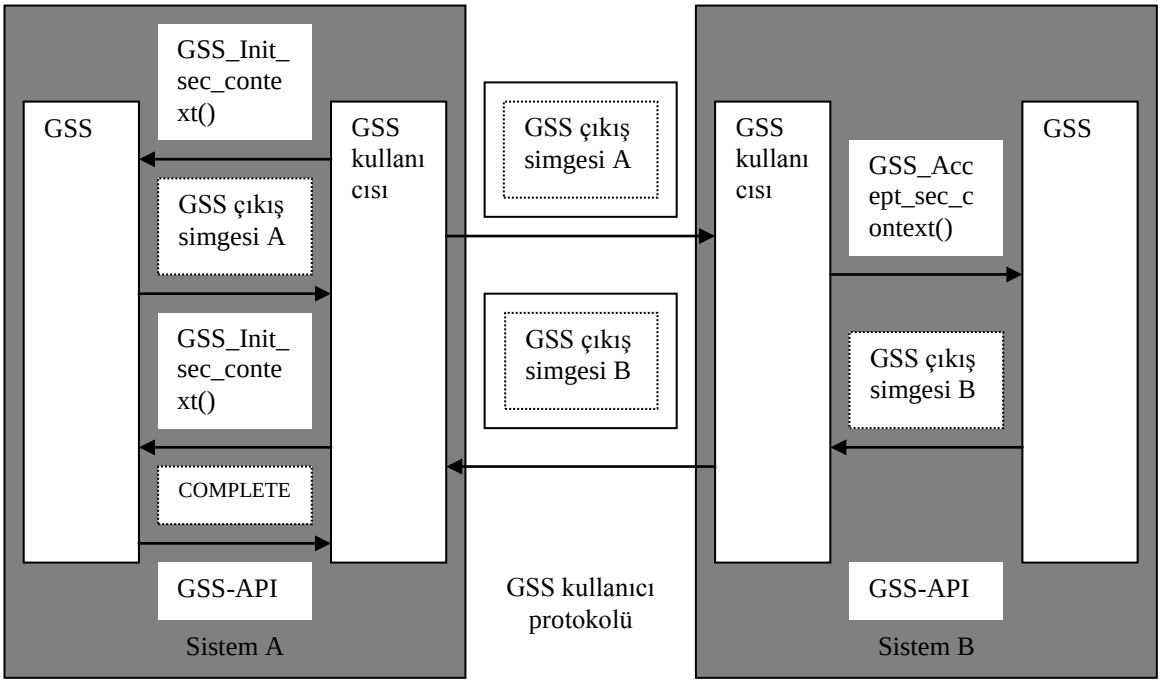
Farklı olarak Şekil 5.3'te, GSS PDU 'ları GSS simgeleri olarak ve GSS servisini kullanan modül PDU 'ları içinde taşınırlar. GSS, GSS-API 'yi kullanarak veri değişimi yapar.



Şekil 5.3: GSS katman dışı güvenli haberleşme protokolü

5.6.1 GSS Tokalaşma

Şekil 5.4'te GSS 'e yapılan çağrılar, cevaplar ve bunlara karşı düşen GSS simgelerini taşıyan ve tokalaşma sırasında değiştirilen PDU 'lar gösterilmiştir.



Şekil 5.4: GSS-API tokalaşma zinciri

1. Sistem A 'daki başlatıcı bir GSS_Init_sec_context() çağrısını kendi lokal GSS 'ine göndererek güvenlik içeriğini başlatır. Bu çağrının parametreleri hedefin adı (sistem B 'deki), güvenlik mekanizması tipi, kullanılacak belgeler, güvenlik simgelerinin geçerlilik süresi veya çeşitli ek karakteristikler olabilir.

2. Bu başlatıcıya cevap hedef sisteme (B) iletilen bir çıkış simgesidir. Burada da ek parametreler içerilebilir, ve çağrının kabul edilip edilmediğini belirtir. Çıkış simgesi bir doğrulayıcı ve güvenlik içeriği taşır.

3. İlk GSS kullanıcısı uzaktaki eş sisteme çıkış simgesi taşıyan bir PDU gönderir.

4. Hedef B ’deki eş sistem, parametrelerinden birinin az önce aldığı çıkış simgesi olduğu bir GSS_Accept_sec_context() çağrısını kendi lokal GSS ’ine gönderir.

5. B ’nin GSS ’i bir alındı çıkış simgesi üretir.

6. B ’deki GSS kullanıcısı bu simgeyi A ’daki eş sisteme gönderir.

7. A ’daki GSS kullanıcısı, parametrelerinden birinin B ’den alınan çıkış simgesi olduğu bir GSS_Init_sec_context() çağrısını kendi lokal GSS ’ine gönderir. Bu çağrı devam gerektiğini ifade eder.

8. A ’nın GSS ’i çıkış simgesini onaylar ve GSS_S_COMPLETE ile tokalaşmanın başarılı olarak tamamlandığını belirtir.

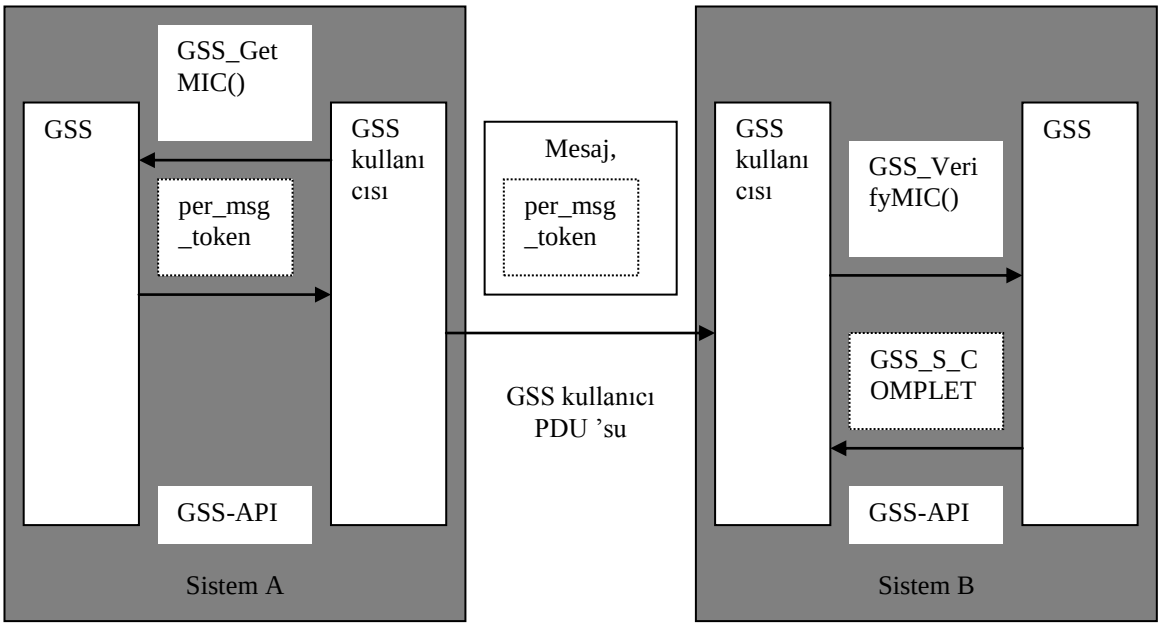
Tokalaşma fazı GSS-API çağrıları, **içerik – seviyesinde çağrılar** adıyla Tablo 5.1’de verilmiştir.

Tablo 5.1: İçerik – seviyesinde çağrılar

GSS_Init_sec_context	Giden güvenlik içeriğini başlatır
GSS_Accept_sec_context	Gelen güvenlik içeriğini kabul eder
GSS_Delete_sec_context	Gerek kalmadığı zaman içeriği temizler
GSS_Process_context_token	İşlem içerikten kontrol simgesi alır
GSS_Context_time	İçeriğin kalan geçerlilik süresini gösterir
GSS_Inquire_context	İçerik hakkındaki bilgileri gösterir
GSS_Wrap_size_limit	GSS_Wrap simgesinin büyüklüğünü belirler
GSS_Export_sec_context	Diğer işlemlere içerik iletir
GSS_Import_sec_context	İletilen içeriği alır

5.6.2 GSS Güvenli Transfer

Şekil 5.5’te güvenli veri transferi gösterilmiştir.



Şekil 5.5: GSS-API güvenli transfer fazı

1. A 'daki GSS kullanıcısı bir GSS_GetMIC() çağrısı üretir. Bu çağrı ile bir mesajın oktet dizisinde içerilen Mesaj Bütünselliği Kodu (Message Integrity Code – MIC) istenir.

2. GSS mesaj MIC 'ını içeren bir per_msg_token gönderir.

3. A 'daki GSS kullanıcısı hedef B 'deki eş sistemine az önce alınan per_msg_token 'ı içeren bir PDU yollar.

4. B 'deki GSS kullanıcısı parametrelerinden birinin alınan per_msg_token olduğu bir GSS_VerifyMIC() oluşturarak kendi GSS 'ine gönderir.

5. B 'nin GSS 'i, per_msg_token 'daki MIC 'ın doğru olduğunu onaylar ve GSS_S_COMPLETE ile çağrıyı bitirir.

Yukarıdaki iki çağrı dışında GSS_Wrap ve GSS_Unwrap da vardır ve bu çağrılar farklı olarak sadece şifreleme ve şifre çözmeyi desteklerler.

Güvenli transfer fazı GSS-API çağrıları, **mesaj başına çağrılar** adıyla Tablo 5.2'de verilmiştir.

Tablo 5.2: Mesaj başına çağrılar

GSS_GetMIC	Bütünselliği kontrol eder, mesajdan ayrı bir simge olarak alınır
GSS_VerifyMIC	Mesajla birlikte gönderilen simgeyle bütünselliği doğrular
GSS_Wrap	Bütünselliği kontrol eder, isteğe bağlı şifreler
GSS_Unwrap	İsteğe bağlı şifre çözer, bütünselliği doğrular

5.6.3 GSS Yönetimsel Arabağdaşımlar

Tokalaşma ve güvenli transfer fazlarındaki çağrılara ek olarak, GSS-API 'de yönetimsel çağrılar da vardır. Bunlar belge yönetim çağrıları ve destek çağrıları olmak üzere ikiye ayrılır.

Belge yönetim çağrıları genellikle tokalaşma fazından önce sistem içinde istenir. Bu istekle GSS güvenli transfer için gerekli belgeleri (örneğin; şifreler ve şifreleme anahtarları) elde eder. Güvenli transfer fazı bittikten sonra, belge yönetim çağrıları geciken belgeler için istenir. GSS-API belge yönetim çağrıları Tablo 5.3'te gösterilmiştir.

Tablo 5.3: Belge yönetim çağrıları

GSS_Acquire_cred	Kullanım için belge ister
GSS_Release_cred	Kullanımdan sonra belgeyi bırakır
GSS_Inquire_cred	Belgeler hakkında bilgileri gösterir
GSS_Add_cred	Belgeleri artan sırayla oluşturur
GSS_Inquire_cred_by_mech	Mekanizma başına belge bilgilerini gösterir

Destek çağrıları çok sayıda yönetim fonksiyonunu sağlar ve Tablo 5.4'te listelenmişlerdir.

Tablo 5.4: Destek çağrıları

GSS_Display_status	Durum kodlarını yazdırılabilir biçime dönüştürür
GSS_Indicate_mechs	Yerel sistem tarafından desteklenen mekanizmaları gösterir
GSS_Compare_name	İki ismi karşılaştırır
GSS_Display_name	İsmi yazdırılabilir biçime dönüştürür
GSS_Import_name	Yazdırılabilir ismi normal biçimine getirir
GSS_Release_name	Normal biçimdeki isim için önceden ayrılmış boş bellek
GSS_Release_buffer	Yazdırılabilir isim için önceden ayrılmış boş bellek
GSS_Release_OID	Nesne Belirleyici (Object Identifier – OID) tarafından belirlenmiş nesne için önceden ayrılmış boş bellek
GSS_Release_OID_set	OID nesne kümesi için önceden ayrılmış boş bellek
GSS_Create_empty_OID_set	Boş bir OID kümesi oluşturur
GSS_Add_OID_set_member	OID kümesine bir öğe ekler
GSS_Test_OID_set_member	OID 'in OID kümesinin üyesi olup olmadığını test eder
GSS_OID_to_str	OID 'i dizi şeklinde gösterir
GSS_Str_to_OID	Diziden OID 'i çıkarır
GSS_Inquire_names_for_mech	Mekanizmalar tarafından desteklenen isimleri gösterir
GSS_Inquire_mechs_for_name	İsim tipini destekleyen mekanizmaları gösterir
GSS_Canonicalize_name	İsmi mekanizma başına çevirir
GSS_Export_name	Mekanizma başına ismi dışarı çıkarır
GSS_Duplicate_name	İsim nesnesini yedekler

5.6.4 Durum Raporlama

Her GSS-API çağrısı iki durum değeri üretir. Ana durum değerleri kullanılan mekanizmadan bağımsız olarak, normal akışı sağlamaya yeterli olan çağrı durumunu gösterir (örneğin; GSS_S_COMPLETE, GSS_S_FAILURE, GSS_S_CONTINUE_ NEEDED). İki çeşit GSS-API ana durum kodu vardır: Tablo 5.5'te gösterilen hata kodları ve Tablo 5.6'da gösterilen bilgilendirici durum kodları.

Tablo 5.5: Hata kodları

GSS_S_BAD_BINDINGS	Kanal bağlama uyumsuzluğu
GSS_S_BAD_MECH	Desteklenmeyen mekanizma isteği
GSS_S_BAD_NAME	Uygun olmayan isim sağlanması
GSS_S_BAD_NAMETYPE	Desteklenmeyen çeşit adı sağlanması
GSS_S_BAD_STATUS	Uygun olmayan durum seçici
GSS_S_BAD_SIG	Uygun olmayan bütünsellik kontrollü simge
GSS_S_CONTEXT_EXPIRED	Zamanaşımına uğramış güvenlik içeriği
GSS_S_CREDENTIALS_ EXPIRED	Zamanaşımına uğramış belge bulunması
GSS_S_DEFECTIVE_CREDENTIAL	Hatalı belge bulunması
GSS_S_DEFECTIVE_TOKEN	Hatalı simge bulunması
GSS_S_FAILURE	Hata, GSS-API seviyesinde tanımlanmamış
GSS_S_NO_CONTEXT	Geçerli güvenlik içeriği belirtilmemesi
GSS_S_NO_CRED	Geçerli belge sağlanmaması
GSS_S_BAD_QOP	Desteklenmeyen Koruma Kalitesi (Quality of Protection – QOP) değeri
GSS_S_UNAUTHORIZED	İzinsiz işlem
GSS_S_UNAVAILABLE	Mümkün olmayan işlem
GSS_S_DUPLICATE_ELEMENT	Kopya belge öğesi isteği
GSS_S_NAME_NOT_MN	Çoklu – mekanizma öğeleri içeren isim

Tablo 5.6: Bilgilendirici durum kodları

GSS_S_COMPLETE	Normal tamamlama
GSS_S_CONTUNIUE_NEEDED	Program devam çağrısı isteği
GSS_S_DUPLICATE_TOKEN	Mesaj başına kopya simge bulunması
GSS_S_OLD_TOKEN	Mesaj başına zamanaşımına uğramış simge bulunması
GSS_S_UNSEQ_TOKEN	Mesaj başına yeniden sıralanmış simge bulunması
GSS_S_GAP_TOKEN	Atlanmış önceki simgelerin bulunması

Küçük durum kodları daha detaylı durum bilgisi verir.[1]

5.7 GULS

OSI ’nin yedi katmanlı modelinin gelişmesiyle birlikte bu katmanlar için çeşitli güvenlik protokolleri de gelişmektedir. Genel Üst Katman Güvenliği (Generic Upper Layers Security – GULS) standardının çıkması ile OSI güvenlik standartları ilerlemiştir. GULS seçili alan koruması sağlar. Güvenli haberleşme protokollerinde bir ilk olarak veri transferine başlamadan önce kullanılacak şifreleme kurallarını belirler.

GULS iki kısımdan oluşur: tokalaşma fonksiyonu uygulama katmanında, güvenli transfer fonksiyonu sunum katmanındadır. Bu düzenleme ile şifreleme/şifre çözme ve güvenli transfer birlikte optimum olarak kullanılabilir.

GULS yayınlandığında kullanımda olan daha az fonksiyonelliğe sahip seçenekler olmasına rağmen karmaşık yapısı nedeniyle tercih edilmemiştir.

5.8 SSL3

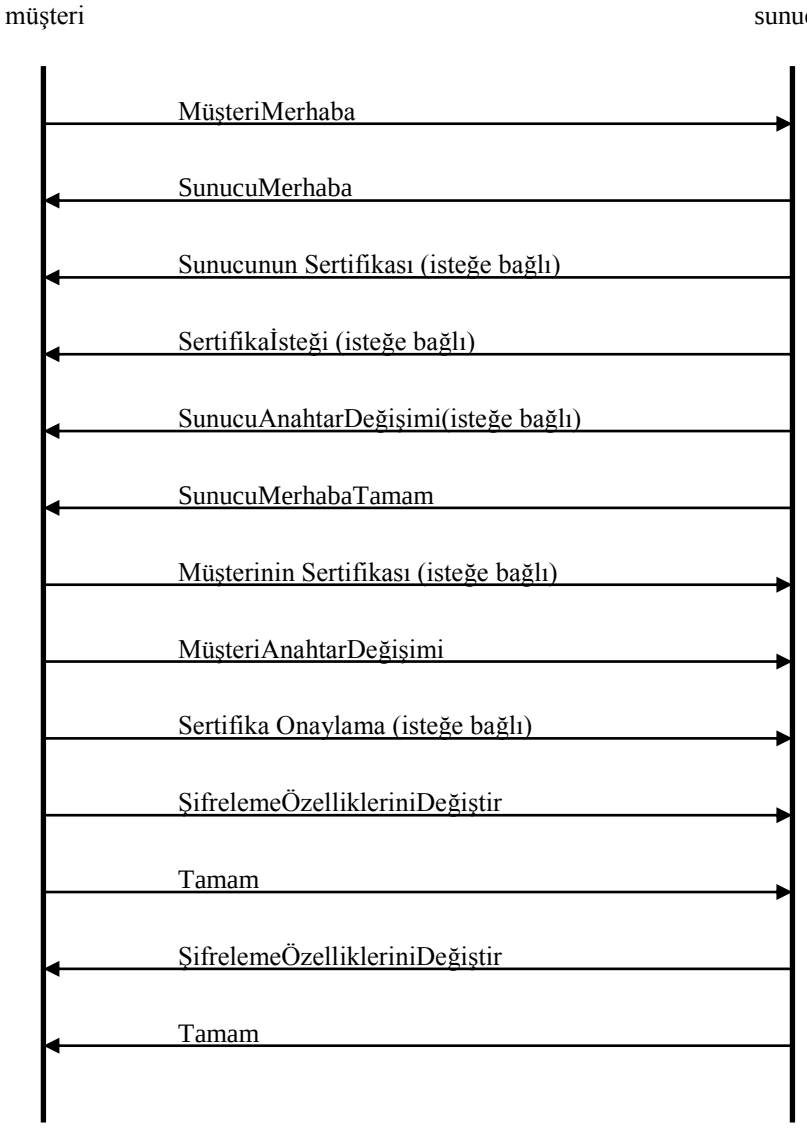
SSL3 Web 'de güvenli dolaşım için tasarlanmıştır. Netscape ile popüler olmuş ve IETF tarafından İletim Katmanı Güvenliği versiyon 1 (Transport Layer Security version 1 – TLS1) adıyla standart haline getirilmiştir. Geniş kullanımı ve kolaylığı nedeniyle internet üzerindeki iletimlerin güvenliği için en uygun adaydır.

SSL3 iki kısımdan oluşur. Birinci kısım bağlantı kurulurken aktiftir, eş öğeleri doğrulamayı ve güvenlik bilgilerinin değişimini sağlar. İkinci kısım haberleşme sırasında akan bitlerin güvenliğini sağlar.

5.8.1 Tokalaşma

SSL3 Web güvenliği için tasarlandığından SSL3 oturumunu başlatan kişi müşteri, isteğe cevap veren sunucu olarak adlandırılır. TMN uygulamalarında ise isteği yapan CMIP yöneticisi cevap veren de CMIP aracısı olabilir.

SSL3 tokalaşma protokolü Şekil 5.6'da gösterilmiştir.



Şekil 5.6: SSL3 tokalaşma protokolü

5.8.2 Güvenli Transfer

Tokalaşma protokolü mesajları da dahil olmak üzere tüm SSL3 mesajları, SSL3 kayıt katmanı üzerinden iletilir. Kayıtlar öncelikle 2^{14} veya daha az oktete parçalanırlar. Aynı tipten kısa mesajlar aynı kayıt altına toplanabilirler. Sonra bu kayıtlar sıkıştırılırlar. Bütünsellik ve güvenliğin sağlanması için bir koruma yöntemi belirlenir.

6. OSI TABANLI TMN PROTOKOLLERİNİN GÜVENLİĞİ

İki öğe güvenli olarak haberleşmek istiyorlarsa, dijital imzalar, doğrulama bilgileri veya kullanılacak şifreleme algoritmaları gibi güvenlik bilgilerini değiş tokuş etmek zorundadırlar. Bu işlem için pek çok TMN protokolü tanımlıdır. Bu bölümde OSI tabanlı TMN protokolleri ele alınmıştır.

6.1 ACSE GÜVENLİĞİ

ACSE tüm OSI uygulama bağlantılarının kurulmasında (özellikle TMN uygulamalarında) kullanılır. ACSE belirleme, doğrulama ve ASE güvenliği gibi temel güvenlik özelliklerini destekler.

6.1.1 Belirleme

Bir OSI bağlantısı iki AP arasında kurulur. AP 'ler sistemler arasındaki etkileşimleri belirleyen ağ yönetim fonksiyonlarını gerçekleştirirler. AP 'ler, AE olarak adlandırılan haberleşme arabağdaşımları aracılığıyla haberleşirler. Her AP, farklı AP 'ler ile bağlantı sağlayan pek çok AE 'ye sahip olabilir. Fakat çoğu durumda her AP bir AE 'ye sahiptir. Her AP bir veya daha fazla ASE 'ye sahiptir ve bunlar diğer AP 'lerdeki ASE 'lerle haberleşirler. Örneğin; iki CMISE ASE 'si CMIP kullanarak haberleşir. CMISE içeren bir AE ayrıca, bağlantı kurulumu için ACSE ve hangi cevapların hangi isteklere istinaden üretildiğini ayırmak için ROSE da içermelidir. Böyle bir AE ayrıca, SMASE 'nin fonksiyonları olan bir veya daha fazla FU da içerir. SMASE güvenlik tetkik yönetimi ve güvenlik alarm yönetimi gibi sistem yönetim fonksiyonlarına sahiptir ve bu sistem yönetim fonksiyonlarının her biri birkaç FU'ya sahiptir.

Bir bağlantıyı başlatmak için iki ACSE mesajı değiştirilir: Uygulama Bağlantı İsteği (Application Association Request – AARQ) ve Uygulama Bağlantı Cevabı (Application Association Response – AARE). AARQ çağrıyı yapan ve hedef

AE'lerin bilgilerini içerir. AARE cevap veren AE'nin bilgilerini içerir (AARQ'da hedeflenenden farklı olabilir). Tüm bu alanlar isteğe bağlıdır.

Güvenli transfer için her iki taraf da kendilerini tanıtmalıdır. Bu işlem için doğrulama kullanılabilir.

6.1.2 Doğrulama

Doğrulama gizli bilgiler (şifre veya şifreleme anahtarı) kullanarak göndericinin kimliğini alıcıya onaylar. Bu onaylamanın şekli kullanılan doğrulama mekanizmasına bağlıdır. Örneğin; bir doğrulama mekanizması basit bir şifre değişimi, bir zaman işareti bulundurulması veya dijital imza olabilir.

Eğer iki öğe kullanılacak doğrulama mekanizmasına karar vermişse bu bilginin ACSE PDU 'larında yer almasına gerek yoktur.

ACSE tercihe bağlı olarak bir doğrulama FU 'suna sahiptir. Bu FU üç bölümden oluşur:

1. ACSE gereklilikler bölümünde hangi ACSE FU 'sunun kullanıldığı belirtilir.

2. Doğrulama mekanizması adı bölümü tercihe bağlıdır ve sadece ACSE gereklilikler bölümü varsa kullanılır.

3. Doğrulama değeri bölümü sadece ACSE gereklilikler bölümü varsa kullanılır. Dört seçenek vardır:

   - Grafik dizisi (şifre değişimleri için uygundur)

   - BIT STRING (genel doğrulamalar için uygundur)

   - EXTERNAL (doğrulayıcı sözdiziminin ACSE dışındaki bir standart tarafından belirlendiği durumlar için uygundur)

   - ANY DEFINED BY (EXTERNAL ile aynı özellikleri sağlar fakat doğrulama mekanizması adı bölümünün bulunması zorunludur)

6.1.3 ASE Güvenliği

Bazı ASE 'ler bağlantı kurulumu sırasında bilgi değişimi gerektirir. Bu sebeple AARQ ve AARE PDU 'ları kullanıcı bilgi bölümü içerirler. Bu bölüm EXTERNAL tipindedir.

6.2 CMISE GÜVENLİĞİ

CMISE ASE tarafından kullanılan CMIP protokolü düzensiz bir güvenlik sağlar. CMIP 'ın yönetim operasyonları PDU 'ları (al, ayarla, oluştur, sil ve etkile) isteğe bağlı birer giriş kontrol bölümü içerebilir. CMIP cevap verir, bildirir ve doğrular fakat güvenlik eksiklikleri vardır.

Giriş kontrol alanı aynı zamanda veri orijini ile ilgili bilgiler de taşır ve yeniden yönlendirme, silme, geciktirme ve yanlış yönlendirme saldırılarının belirlenmesini de sağlar.

6.2.1 Elektronik Bağlama Doğrulaması

Telekom şirketleri arasında X arabağdaşımı üzerinden yapılan ilk CMIP tabanlı uygulama sorun yönetimidir (trouble administration – TA). LEC 'ler ile IC 'ler arasındaki ilk TMN uygulamaları serisi ise EB 'dir.

Tarihçe: IC 'ler müşterilerine servis sağlayabilmek için LEC 'lerden devreleri kiralarlar. Bu devrelerden biri koptuğunda IC 'nin OS 'ine TA için bir sorun raporu gider. OS sorun raporunu takip eder ve çözmek için sorunu LEC 'e iletir. Bu iletim işlemleri işgücü kaybına neden olur. EB TA ile sorun raporları IC OS 'ten LEC OS 'e elektronik olarak gider.

TA EB uygulanmaya başlandığında, avantajları sebebiyle kabul görürken, özellikle LEC tarafındaki yöneticilerde sisteme yabancıların girişini sağlaması nedeniyle korku yaratmıştır. Bu sorunu çözmek için bazı güvenlik uygulamaları gerekmiştir.

Öncelikle tüm taraflar Kapalı Kullanıcı Grupları (Closed User Groups – CUGs) ve güçlü, şifrelenmiş eş öğelerin doğrulanması kullanmanın gerekliliğinde hemfikir olmuşlardır. CUG 'lar üstünde yer aldıkları X.25 tarafından desteklendikleri için

kolaydırlar. Doğrulama için önerilen ilk algoritma Enigma olmuştur. Fakat bu algoritma Alan Turing tarafından 2. Dünya Savaşı sırasında çözülmüştür. RSA kamusal anahtarlı şifreleme alternatif olarak sunulmuştur. Fakat bu algoritma daha önce kullanılmamış olması sebebiyle ilgi görmemiştir ve her iki algoritma da EB güvenliği için uygun kabul edilmemiştir.

Bazı kullanıcılar her EB sonlandırma noktasında donanım şifreleyiciler kullanılmasını önermiştir. Bu uygulama ağ için herhangi bir yük getirmemesine rağmen farklı tedarikçilerden elde edildikleri için birlikte çalışamazlar. Bu yöntem de bu sebeple kabul edilmemiştir.

Daha sonra EB güvenliği için GULS standardı önerilmiştir. Bu yöntem çok güçlü olmasına rağmen çok karmaşık bir yapıya sahiptir. Bununla birlikte EB güvenliği için uygun bir yöntem bulunamamıştır.

Doğrulayıcı: CBC kipinde DES şifrelenmiş bir zaman işareti kullanılarak doğrulama yapılır. DES şifreleme ile 8 bitlik IV şifrelenmiş mesaja eklenir. Rastgele üretilen bu vektör için bir liste oluşturulmalı ve gönderilen IV değerleri kontrol edilmelidir. Bu yöntemin de, IV 'nin kopyalanmasından doğan dezavantajları vardır. Bu dezavantajları önlemek için anahtarın sık değiştirilmesi gereklidir. DES anahtarları listesi için de üçlü DES kullanılarak şifreleme yapılabilir.

Bu sistem TA için tasarlanmış olmasına rağmen, diğer tüm CMIP tabanlı EB uygulamalarında da kullanılabilir.

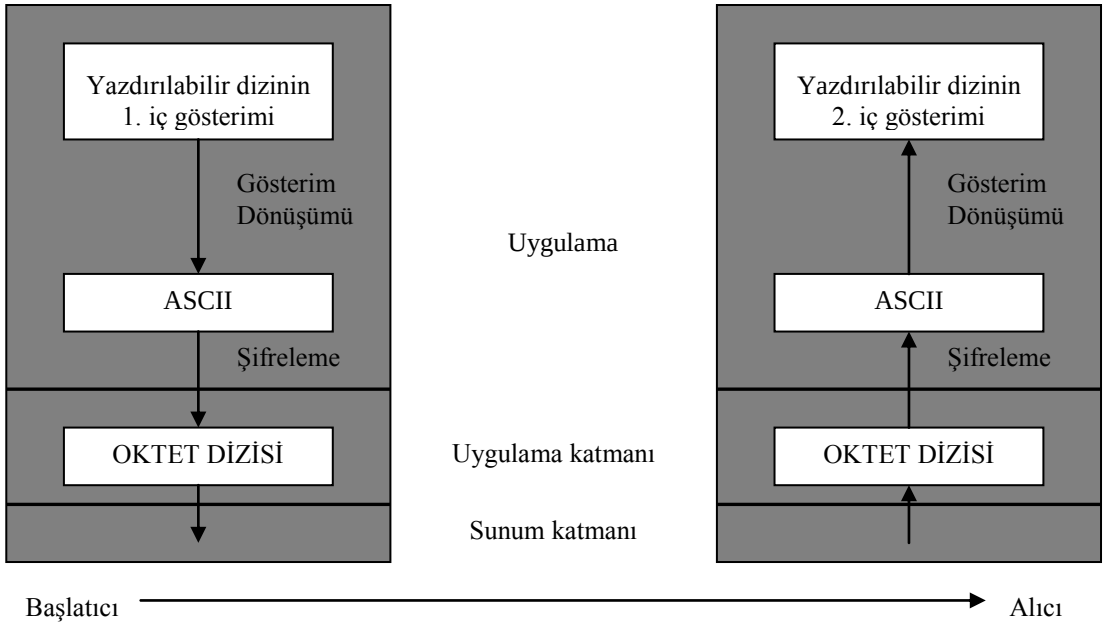
6.2.2 Seçili Alan Doğrulaması

İkinci CMIP tabanlı EB uygulaması Birincil Ara Santral İşletendir (Primary Inter Exchange Carrier – PIC). Uç kullanıcılar kendi PIC 'lerini değiştirdikçe, LEC 'lerini veya yeni IC 'lerini haberdar ederler. Uç kullanıcılardan bu bilgiyi alanlar diğerlerine de haber verirler. PIC kullanımı arttıkça EB otomasyonu için aday hale gelmiştir.

PIC santralleri müşteri adı, adresi ve telefon numarası gibi bilgileri içerebilir. Bunlar gizli bilgilerdir ve yetkili kişiler arasında değiştirilirler. Diğer bir önlem de müşteriye özgü bilgilerin ağda taşınması sırasında şifrelenmesidir.

PIC uygulamaları da TA uygulamaları gibi güvenlik fonksiyonelliğinden yoksundur ve özel bilgilerin korunması için güvenlik gereklidir.

Seçili alanların korunması için Uygulama Tabanlı Güvenlik (Application-Based Security – ABS) geliştirilmiştir. ABS 'de ortak bir veri gösterimi kullanılan alanlarda sadece gerekli bölümler şifrelenir. Bir örnek Şekil 6.1'de gösterilmiştir.



Şekil 6.1: Uygulamada şifreleme

Şekildeki güvenlik gerektiren uygulamada yazdırılabilir biçimdeki dizi ASCII dizisine dönüştürülür. Daha sonra şifrelenerek sunum katmanına iletilir. Burada kodlanır. Alıcı tarafta kodu çözülmüş oktet dizisi uygulama katmanına iletilir ve şifresi çözülerek ASCII dizisi elde edilir. Yazdırılabilir biçim gösterim dönüşümünden sonra elde edilir.

ABS 'nin temel kuralları:

- Her basit alan bit dizisi, oktet veya ASCII karakterleri olarak gösterilebilir.
- Gösterim dönüşümünden sonra şifreleme ile birlikte güvenlik dönüşümleri (örneğin; kıyma, imzalama) de yapılabilir.
- Eğer bir güvenlik dönüşümü yapılmışsa karşı tarafın da kullanılan parametreleri (örneğin; hangi algoritma, hangi anahtar) bilmesi gereklidir.

ABS'nin avantajları ek yük getirmemesi, farklı alanlarda farklı güvenlik dönüşümlerine izin vermesi ve alana özgü güvenlik parametrelerinin değişimini sağlamasıdır. Bu esnekliğe karşın tüm güvenlik yükü uygulamanın üzerindedir.

6.3 STASE – ROSE

ABS ve EB tüm PDU 'nun bütünsellik ve inkar etmeme güvenliğini sağlayamaz. STASE – ROSE bu açığı gidermek için ortaya çıkmıştır. ROSE PDU 'larını veya ROSE kullanan herhangi bir ASE 'yi (örneğin; CMISE veya X.500) korur. Fonksiyonelliği dört ayrı alana bölünebilir:

- Tüm ROSE PDU 'larında güvenlik dönüşümleri
- Eş öğeleri doğrulama
- Güvenlik algoritmaları ve parametrelerinin anlaşması
- Güvenlik parametrelerinin dinamik olarak güncellenmesi

6.3.1 ROSE PDU 'larında Güvenlik Dönüşümleri

STASE – ROSE 'un ana fonksiyonelliği sunum katmanı ile ROSE arasındadır. ROSE PDU 'larında kullandığı güvenlik dönüşümleri aşağıdakilerden herhangi biri olabilir:

- Güvenlik dönüşümü uygulamamak
- Simetrik bir algoritmayla gizlilik amacıyla şifrelemek
- Asimetrik bir algoritmayla gizlilik ve inkar etmeme amacıyla şifrelemek
- Bütünsellik amacıyla kıymak
- Bütünsellik amacıyla mühürlemek
- Bütünsellik ve inkar etmeme amacıyla imzalamak

- Gizlilik amacıyla simetrik bir algoritmayla şifrelemek, bütünsellik ve inkar etmeme amacıyla imzalamak

- Gizlilik amacıyla simetrik bir algoritmayla şifrelemek, bütünsellik amacıyla kıymak

- Gizlilik amacıyla simetrik bir algoritmayla şifrelemek, bütünsellik amacıyla mühürlemek

Bu dönüşümlerden birini seçmek uygulamanın gereksinim duyduğu güvenlikle ilgilidir. Tüm ağ yönetim uygulamaları bütünselliğe gereksinim duyar.

6.3.2 Eş Öğeleri Doğrulama

Doğrulamanın kullanılması isteğe bağlıdır. STASE – ROSE sözdizimi için iki seçenek sunar: simetrik ve asimetrik şifreleme. Çoğu TMN uygulamasında asimetrik şifrelenmiş doğrulayıcı kullanılır. Bu doğrulayıcı, gönderici ve alıcının belirteçlerini, bir zaman işaretini, isteğe bağlı olarak alıcının kamusal anahtarı ile şifrelenmiş simetrik şifreleme anahtarını, önceki alanlar için göndericinin özel anahtarı ile şifrelenmiş dijital imzayı, isteğe bağlı olarak göndericinin kamusal anahtar sertifikasını içerir.

6.3.3 Güvenlik Algoritmalarının Anlaşması

İki tarafın güvenli olarak haberleşebilmesi için aynı güvenlik algoritmalarını kullanmaları gereklidir. STASE – ROSE ile taraflar iletişime başlamadan önce kullanılacak varsayılan algoritmalar kümesine karar verebilir. STASE – ROSE bu varsayılanlar listesini oluşturur ve taraflar aksi yönde karar vermedikçe bu liste kullanılır. Bu uygulama kurulumdaki görüşme yükünü azaltır.

STASE – ROSE ’un belirlediği **varsayılan liste:**

- Simetrik anahtarlı şifreleme için belirlenen varsayılan şifreleme algoritması CBC kipinde DES ’tir.

- Eğer 3DES gerekliyse varsayılan yöntem CBC dış geri besleme kipinde EDE ’dir ve üç farklı DES anahtarı kullanılır.

- Eğer IV belirlenmemişse, ilk IV 64 sıfır değerli bit olur, sonraki her IV en son şifrelenen ROSE PDU 'sunun son 8 baytını alır.

- Varsayılan kamusal anahtarlı şifreleme algoritması RSA 'dır.

- Varsayılan kıyma algoritması MD5'tir.

- Varsayılan MAC (anahtarlı kıyma için) HMAC 'tır.

- Varsayılan mühür DES ile şifrelenmiş ROSE PDU 'sunun MD5 ile kıyılmasıdır.

- Varsayılan dijital imza RSA ve kullanıcının özel anahtarı ile şifrelenmiş ROSE PDU 'sunun MD5 ile kıyılmasıdır.

- Eş öğelerin doğrulanması kurulum sırasında yapılır ve ACSE 'deki FU 'lar kullanılır. Burada istek yapan ve cevap veren doğrulama alanları AARQ ve AARE PDU 'larıdır.

- Eğer eş öğeleri doğrulama ve kamusal anahtarlı şifreleme birlikte kullanılacaksa, şu bilgiler içerilmelidir: Gönderici ve alıcını belirteci, bir zaman işareti, isteğe bağlı olarak alıcının kamusal anahtarı ile şifrelenmiş simetrik şifreleme anahtarı, önceki alanlar için göndericinin özel anahtarı ile şifrelenmiş dijital imza, isteğe bağlı olarak göndericinin kamusal anahtar sertifikası.

- Taraflar aksi yönde karar vermedikçe dijital imza için MD5 kıyma ve RSA kamusal anahtarlı şifreleme kullanılır. AARQ ve AARE mesajlarındaki kamusal anahtarla şifrelenmiş simetrik şifreleme anahtarları farklı olabilir.

- Kullanılan zaman işaretleri monoton olarak artar.[1]

Anlaşma: Çoğu TMN uygulamasında STASE – ROSE tarafından belirlenen algoritmalar kullanılabilir. Fakat belirlenmiş bazı algoritmalar bazı uygulamalara uygun olmayabilir. Bu yüzden kullanıcılar kendi aralarında varsayılan listeler üzerinde anlaşmalıdır. Bunun için AARQ 'da bir güvenlik algoritmaları listesi gönderilir ve AARE 'de benzer bir liste (AARQ 'dakilerin alt kümesi olacak şekilde) ile cevap verilir.

STASE – ROSE ile algoritmalar dışında şifreleme parametreleri (örneğin; kamusal anahtarlar, şifreler) üzerinde de anlaşılabilir. Bunlar için varsayılan değerler belirlenmemiştir.

6.3.4 Güvenlik Parametrelerinin Dinamik Olarak Güncellenmesi

Eğer varsayılan uygulamalar seçilmemişse ve aynı güvenlik dönüşümü için birden fazla algoritma üzerinde anlaşılmışsa, belirli bir PDU üzerinde hangi algoritmanın uygulanacağına karar verilmelidir. Aynı zamanda kullanılacak güvenlik parametreleri de belirlenmelidir. TMN uygulamalarında bu güvenlik parametrelerinden sadece simetrik şifreleme anahtarı için dinamik güncelleme gereklidir. Alıcının kamusal anahtarı ile şifrelenmiş ve göndericinin özel anahtarı ile imzalanmış simetrik şifreleme anahtarının gönderilmesi için STASE – ROSE kullanılabilir.

6.3.5 STASE – ROSE Servisleri

Bir ASE 'nin tanımında onun diğer ASE 'ler için sağladığı servisler yer alır. STASE – ROSE için bu oldukça kolaydır: sadece bir kullanıcıya (ROSE), bir servis sağlar (SR-TRANSFER).

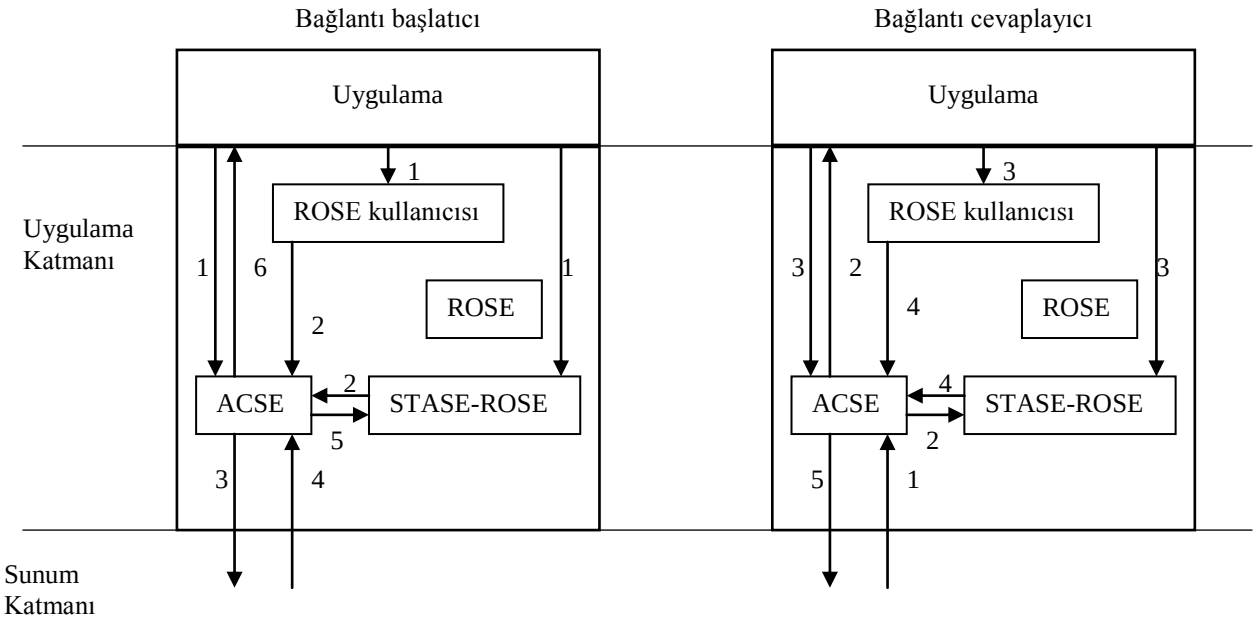
Gönderici taraf SR-TRANSFER isteği göndererek korunması gereken bir ROSE PDU 'su olduğunu belirtir. Alıcı taraf SR-TRANSFER işareti ile uzak bir sistemdeki ROSE 'dan bir PDU aldığını belirtir.

SR-TANSFER için gerekli parametreler: ROSE PDU 'su, Şifreleme-Tipi (kullanılacak güvenlik dönüşümünü belirleyen değerdir), Şifreleme-Parametreleri (güvenlik dönüşümünde kullanılacak parametrelerdir).

6.3.6 ASE 'ler Arası Etkileşimler

Uygulama, ACSE, ROSE, STASE – ROSE, ROSE kullanıcıları ve sunum katmanları arasındaki etkileşimleri tanımlar. İki uygulama arası iletişimin çeşitli adımlarını kapsar. Etkileşimlerin seçimi yerel bir sorundur.

Bağlantı Kurulumu: Şekil 6.2'de bağlantı kurulumu esnasında uygulama, çeşitli ASE 'ler ve sunum sağlayıcı arasındaki etkileşimler gösterilmiştir.



Şekil 6.2: Bağlantı kurulumu sırasındaki etkileşimler

Bağlantı kurulumunu başlatan taraftaki etkileşimler:

1. STASE – ROSE içeren uygulama, ACSE ’ye bağlantı kurulumu için bir A-ASSOCIATE isteği gönderir. Eğer eş öğe doğrulaması isteniyorsa, uygulama doğrulama değerini AARQ PDU ’sunun doğrulama değeri bölümünde taşınmak üzere ACSE ’ye iletir. Aynı anda STASE – ROSE ve ROSE kullanıcısına da haber verilir ve STASE – ROSE ’un gerekli şifreleme parametrelerini iletmesi sağlanır.

2. STASE – ROSE, ACSE ’ye şifreleme parametrelerini sunar. Aynı anda ROSE kullanıcısı, ilgili ASE ’nin bilgilerini ACSE ’ye iletir. Tüm bu bilgiler ACSE ’nin kullanıcı bilgileri alanında taşınır.

3. ACSE, bir uygulama bağlantısı kurmak için sunum sağlayıcıya bir P-CONNECT isteği gönderir. Sunum sağlayıcı bu P-CONNECT isteğini gönderir ve cevap alır (şekilde gösterilmemiştir).

4. Sunum sağlayıcı, aldığı P-CONNECT onayını ACSE ’ye iletir.

5. ACSE, STASE- ROSE ’a yeni bağlantının kurulduğunu haber verir ve eğer varsa şifreleme parametreleri ile ilgili değerleri iletir.

6. ACSE, A-ASSOCIATE onayını uygulamaya iletir ve bağlantının kurulduğunu haber verir ve eğer varsa şifreleme parametreleri ile ilgili değerleri iletir.

Bağlantı kurulumunu cevaplayan taraftaki etkileşimler:

1. Sunum sağlayıcı, uzak servis kullanıcısının bağlantı isteğini içeren P-CONNECT uyarısını ACSE 'ye gönderir.

2. ACSE, uygulamaya göre bir A-ASSOCIATE uyarısı gönderir. Aynı anda STASE – ROSE 'a haber verilir ve varsa önerilen güvenlik bilgileri iletilir. ACSE ayrıca ASE ile ilgili bilgileri ROSE kullanıcılarına iletir.

3. Uygulama, ACSE 'ye bir A-ASSOCIATE cevabı göndererek bağlantıyı kabul veya reddeder. Eğer mesajda şifreleme parametreleri mevcutsa, kabul edilen parametreler STASE – ROSE 'a iletilir. Aynı anda ROSE kullanıcılarının ASE 'lerine de haber verilebilir.

4. STASE – ROSE, kabul edilen şifreleme değerlerinin ACSE 'ye iletir. Aynı anda ROSE kullanıcısı, ilgili ASE 'nin bilgilerini ACSE 'ye iletir.

5. ACSE, P-CONNECT cevabını sunum sağlayıcıya ileterek bağlantıyı kabul veya reddeder.

6.3.7 STASE – ROSE Protokolü

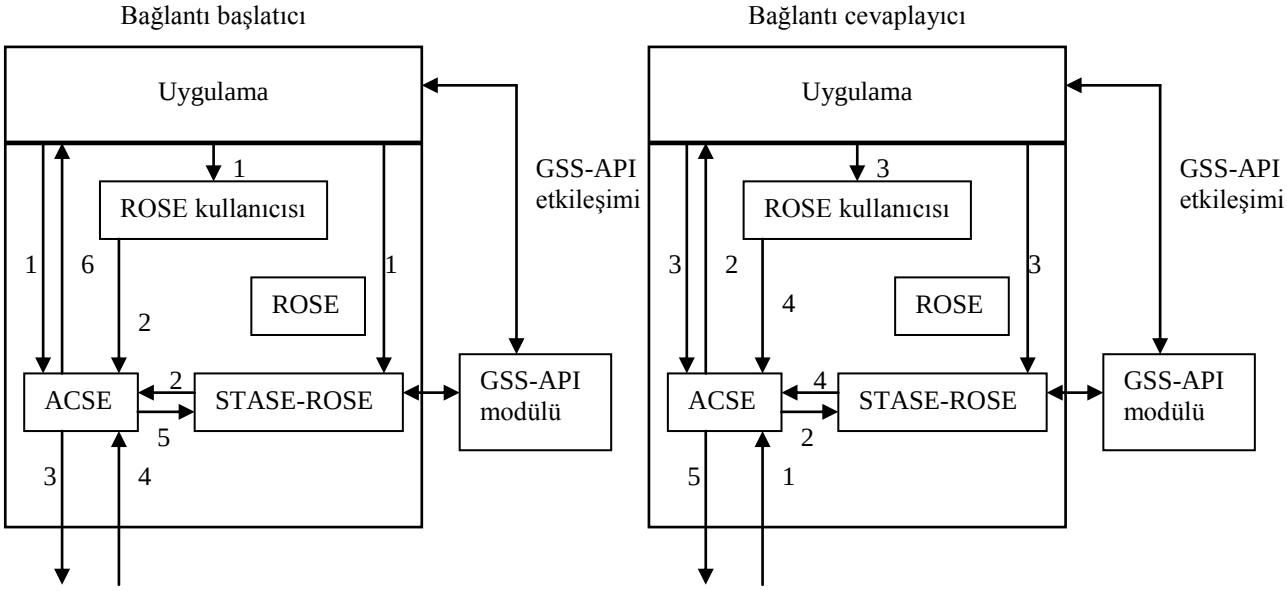
STASE – ROSE servislerinin dağıtılabilmesi için, uzak taraftaki eş öğeyle iyi tanımlanmış bir protokol kullanarak haberleşilmesi gereklidir. STASE – ROSE, ROSE PDU 'larını güvenli olarak taşıyabilmek için sunum katmanının servislerini kullanır.

STASE – ROSE protokol makinesi (SRPM), ROSE ile SR-TRNSFER servisi yardımıyla haberleşir. SPRM, sunum katmanından gelen uyarılar ve ROSE 'dan gelen istekler üzerine çalışır.

6.3.8 STASE – ROSE ile GSS-API 'nin Birlikte Kullanılması

GSS – API haberleşme güvenlik servislerinin entegrasyonunu sağlayan üst katman bir API 'dir. STASE – ROSE için GSS – API kullanılması halinde güvenlik algoritmaları ve mekanizmaları, STASE – ROSE değiştirilmeden iletilebilir.

Şekil 6.3'te bağlantı kurulumu sırasında GSS – API 'nin kullanılması gösterilmiştir. Hem uygulama hem de STASE – ROSE aynı GSS – API modülünü kullanır. Uygulama GSS – API modülünü doğrulama amacıyla, STASE – ROSE ise güvenli veri iletimi amacıyla kullanır.



Şekil 6.3: Bağlantı kurulumu sırasında GSS – API kullanımı

Bağlantı kurulumunu başlatan taraftaki GSS etkileşimleri:

a. Uygulama GSS_aquire_cred() mesajıyla GSS – API modülünden belgeleri ister.

b. Uygulama GSS_init_sec_context() mesajıyla belirlenmiş bir bağlantı cevabıyla birlikte güvenlik içeriğini oluşturur. Uygulama burada kullanılacak güvenlik parametrelerine karar vermelidir. GSS – API modülü uygulamaya bir içerik simgesi gönderir.

c. Uygulama, ACSE ’ye içerik simgesini taşıyan bir A-ASSOCIATE mesajı gönderir. Aynı anda STASE – ROSE ’a da kullanılacak güvenlik parametrelerini iletir.

d. Diğer adımlar GSS – API ’nin bulunmadığı bağlantı kurulumu gibidir.

Bağlantı kurulumunu cevaplayan taraftaki GSS – API etkileşimleri:

a. İlk iki adım GSS – API ’nin bulunmadığı bağlantı kurulumu gibidir.

b. Uygulama A-ASSOCIATE mesajını aldığı zaman GSS_aquire_cred() mesajıyla GSS – API modülünden belgeleri ister. Daha sonra GSS_accept_sec_context() mesajıyla başlatıcı tarafından gönderilen simgeyi alır. GSS – API simgenin geçerliliğini kontrol ederek başlatıcıyı doğrular. Eğer başlatıcı karşılıklı doğrulama istiyorsa, GSS – API ikinci bir simgeyi başlatıcıya geri gönderir.

c. Diğer adımlar GSS – API ’nin bulunmadığı bağlantı kurulumu gibidir.[1]

6.4 Q3 GÜVENLİĞİ

Q3 arabağdaşımı için çeşitli güvenlik servisleri tanımlanmıştır. Ayrıca bağlantı kurulumu sırasında eş öğelerin doğrulanması gereklidir. Bu doğrulama STASE – ROSE ’da kullanılandır ve ACSE ’nin doğrulama FU ’sunda taşınır.

Veri orijini doğrulama için EB ’de kullanılan yöntemler kullanılır ve doğrulama bilgisi CMIP giriş kontrol alanında taşınır. Basitçe gönderen ve alıcının kimlikleri, zaman işareti, alıcının kamusal anahtarı ile şifrelenmiş simetrik şifreleme anahtarı içerilebilir.

STASE – ROSE ile varsayılan yöntemler kullanılarak PDU koruması gerçekleştirilir. Giriş kontrolü için ACL ve ACC yaklaşımları kullanılır. Bağlantı başlatıcılar farklı önceliklere sahip gruplara ayrılabilir. Ayrıca hedef sistemler de farklı haklar veren gruplara ayrılabilir. Bu durumda giriş kontrolünün yönetilmesi sadece gerekli gruplara ekleme ve silme ile gerçekleşebilir.

6.5 X.500

X.500 rehber standardı güvenlik için gerekli imkanlar sağlar. Bu imkanların bir çoğu STASE – ROSE gibi farklı standartlara adapte edilmiştir. Rehberdeki herhangi bir andaki güvenlik seviyesi, içerilen bilginin hassasiyetine bağlıdır. Eğer bilgi gizli değilse değişimi sırasında şifrelenmesine gerek yoktur. Hatta bilgiyi kimin istediğine bile bakılmayabilir. Ancak bu durum istenmeyen kaynaklardan gelen istekler sebebiyle rehberin meşgul edilmesine neden olacağından tercih edilmez. Fakat tüm durumlarda mesajın bütünselliğinin korunması gereklidir.

X.500 rehberi ile bağlantı, rehber bağlama operasyonu ile sağlanır. Güvenli bir TMN rehberinde belgeler bölümü bulunmalıdır. Q3 arabağdaşımı üzerindeki etkileşimler için bu belge bir sertifika olabilir.

Rehber operasyonları PDU 'larının içerdikleri güvenlik parametreleri ile rehberden gelecek cevapların imzalanıp imzalanmayacağına karar verilir.

6.6 X.25

TMN uygulamalarındaki çoğu güvenlik işlemi uygulama katmanında gerçekleştirilir. Burada her uygulamanın ihtiyacı olan güvenlik çeşitli servislerle sağlanır. Fakat yetkili olmayan kişilerin sistem kaynaklarını kullanmaları burada engellenemez. Bağlantı isteği reddedilse bile, bazı sistem kaynakları kullanılmış olur. Bağlantı istekleri sistem üzerinde ciddi bir iş yükü yaratır. Bu saldırıları engellemek için en iyi yer ağ katmanıdır. İdeal olarak ilk düğümde bu istekler durdurulur. Böyle bir güvenlik X.25 ağlarında CUG kullanılarak çalışabilir.

CUG 'un amacı TMN öğelerine izinsiz erişimi engellemektir. Bir sistem bir veya daha fazla CUG 'a üye olabilir ve sadece bu gruplardan en az birine üye olanlar tarafından adreslenebilirler.

Eğer bir TMN öğesi iki veya daha fazla CUG 'a üyeyse, X.25 ağı CUG 'u dışarıya giriş seçimiyle desteklemeli ve çağrı başlangıcını istenen CUG 'a yöneltebilmelidir. CUG öğelerine erişim kontrollü olmasına rağmen, bir CUG öğesi tarafından erişim

daha kolaydır. Bazı sitemler herhangi bir CUG üyesine giriş hakkı tanırken, diğerleri sadece aynı CUG içinde giriş hakkı tanır.

Eğer birden fazla X.25 ağı kullanılacaksa, X.25 ağları arasındaki arabağdaşımlar CUG 'u desteklemelidir.

7. EDI TABANLI TMN GÜVENLİĞİ

EDI mesajları genellikle elektronik posta yardımıyla taşınır. Bu depola ve gönder iletimi TMN 'in gerçek zamanlı uygulamaları için yeterince hızlı değildir. Bu sebeple hızlı iletim için IA kullanılır. IA ayrıca akış kontrolü de sağlar.

TMN uygulamalarında EDI mesajlarının iletimi için sadece TCP/IP kullanılır. TCP 'de güvenlik amacıyla SSL3 kullanıldığından, EDI için de kullanılabilir. SSL3 ile eş öğe doğrulaması, gizlilik ve bütünsellik sağlanabilir ancak inkar etmeme sağlanamaz. Bu servis SSL3 tarafından herhangi bir bölme yapılmadan önce uygulanmalıdır. Böyle bir bölme işleminden önce devreye giren IA inkar etmemeyi sağlar. SSL3'ün standart versiyonu olan TLS1 yaygınlaştığında TMN uygulamalarında SSL3'ün yerini alması beklenmektedir.

7.1 EDI İÇİN TLS1

EDI TMN mesajlarının güvenliği için kullanılan TLS1 şu tanımlara uyar:

- Tüm bağlantılar için kamusal anahtarlı şifrelemeye dayanan eş öğe doğrulaması yapılır.
- Oturumdaki gizli bilgiler alıcının kamusal anahtarı ile şifrelenir.
- Mesaj şifreleme tercihe bağlıdır.
- TLS1 ile bütünsellik için SHA kullanılabilir.
- Eğer gizlilik için TLS1 seçilmişse, CBC kipinde DES kullanılarak simetrik anahtarla şifreleme yapılır.
- Mesaj bütünselliği şifrelenmemiş mesaj üzerinde hesaplanmalıdır.
- Bir varlığın kamusal anahtar büyüklüğü en azından 768 bit olmalıdır.

- CA ’nın kamusal anahtar büyüklüğü en azından 1024 bit olmalıdır.
- Şu şifreleme setleri desteklenmelidir (SSL3 şifreleme setleri kamusal anahtar algoritması, simetrik şifreleme algoritması ve kıyma algoritmasıdır):
    - RSA, NULL (algoritma yok anlamında), SHA (gizlilik istenmediğinde)
    - RSA, DES – CBC, SHA (gizlilik istendiğinde)

SSL3 bir oturumun sonlanmasını ve daha sonra önceden anlaşılan parametrelerle tekrar başlamasını sağlar. TMN X arabağdaşımı üzerindeki EDI tabanlı uygulamalar için yeniden başlayan oturumlar tehdit oluşturmaz.

SSL3 ile HMAC gibi çift kıyma tabanlı bütünsellik sağlanmaz. Tek kıyma ile gerçekleştirilen bütünsellik (burada SHA) bazı zayıf noktalara sahiptir. Bu zayıf noktaların önüne bütünsellik ve şifrelemenin birlikte kullanılmasıyla geçilebilir.

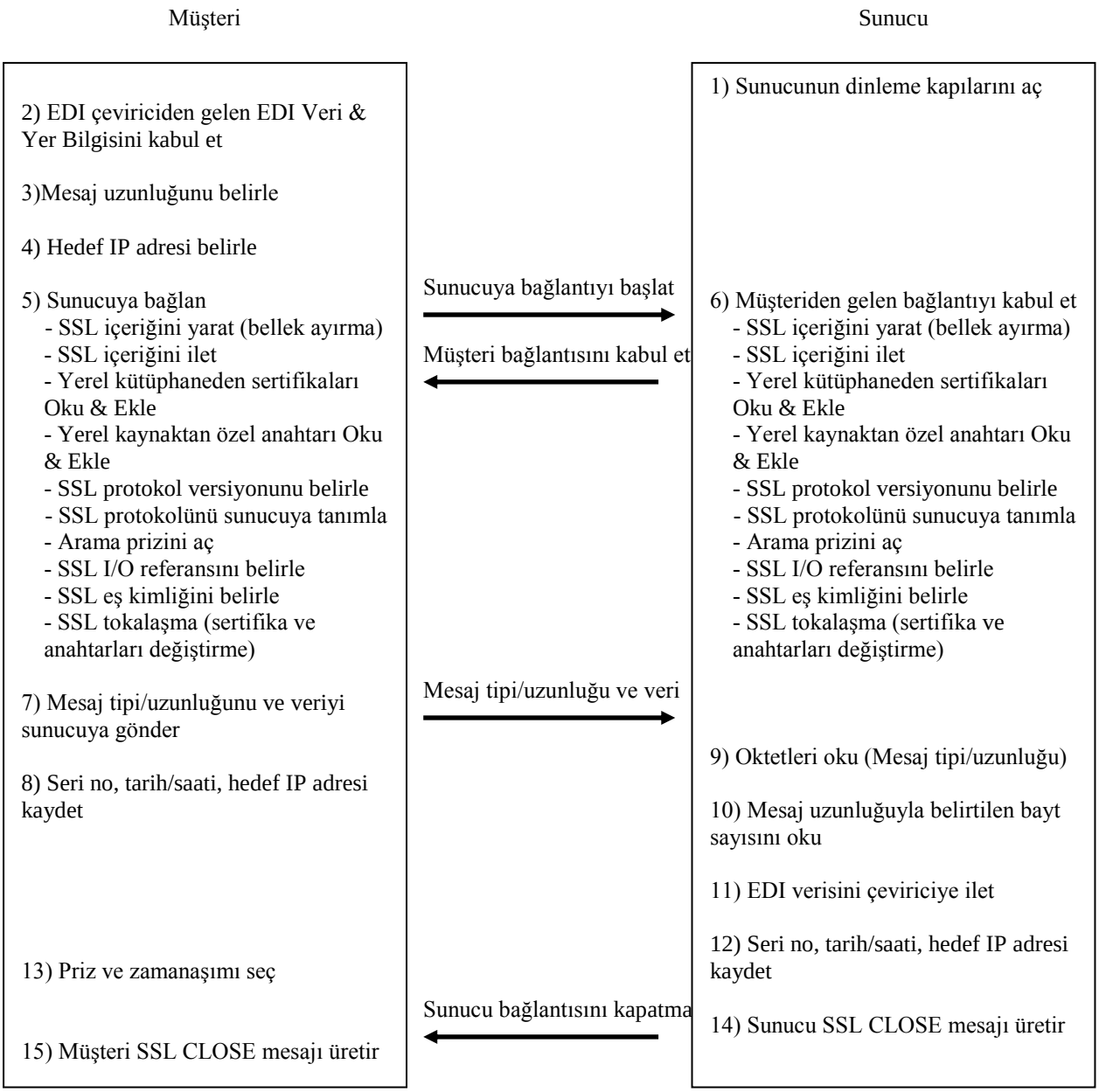
7.2 ETKİLEŞİMLİ ARACI

Şekil 7.1’de IA mesajlarının akışı gösterilmiştir.

7.2.1 Mesaj Biçimi

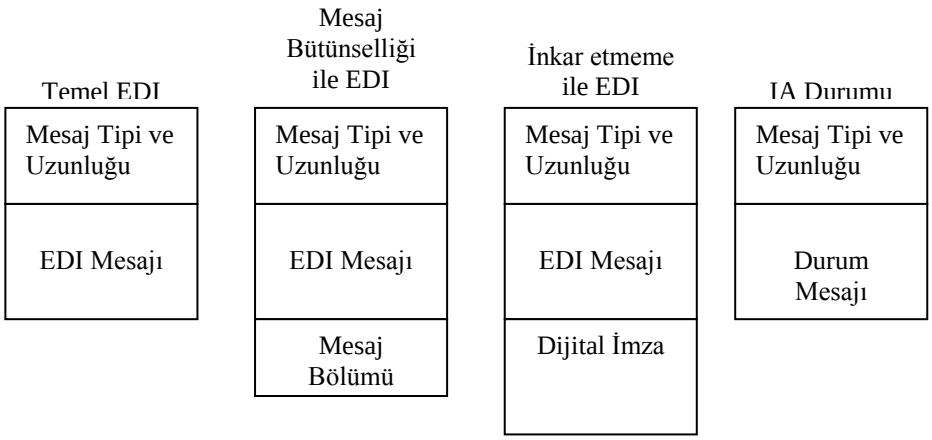
Şirketlerin hem iş hem de güvenlik gerekliliklerini yerine getirebilmek için beş mesaj tipi belirlenmiştir:

- Temel Güvenlik (Mesajlar sadece gizlilik için şifrelenir)
- Bütünsellik desteğiyle Genişlemiş Güvenlik
- İnkar etmeme desteğiyle Genişlemiş Güvenlik (mesaj gizliliği ve bütünselliğini de kapsar)
- Etkileşimli Aracı durumu
- Mesaj alımı (isteğe bağlı)



Şekil 7.1: IA müşteri – sunucu etkileşimi

Bu mesaj tipleri (isteğe bağlı mesaj alımı hariç) Şekil 7.2'de gösterilmiştir.



Şekil 7.2: Zorunlu mesajlar için mesaj biçimi

Temel EDI

Başlık (1)	EDI Mesajı (2)

(1) Mesaj tipi ve uzunluğu

(2) EDI mesajı

Mesaj Bütünselliği ile EDI

Başlık (1)	Başlık (2)	EDI Mesajı (3)	Başlık (3)	Mesaj Bölümü (5)

(1) Mesaj tipi ve uzunluğu

(2) EDI mesaj etiketi ve uzunluğu

(3) EDI mesajı

(4) Mesaj bölümü etiketi ve uzunluğu

(5) Mesaj bölümü

- Kıyma algoritması (SHA)
- Mesaj bölümü

İnkar Etmeme ile EDI

(1) Mesaj tipi ve uzunluğu

(2) EDI mesaj etiketi ve uzunluğu

(3) EDI mesajı

(4) Dijital imza etiketi ve uzunluğu

(5) Dijital imza

- Kıyma algoritması (SHA)
- Göndericinin sertifika bilgileri
- Kıyma şifreleme algoritması
- Göndericinin özel anahtarı ile şifrelenmiş mesaj bölümü

IA Durumu

Başlık (1)	IA Durumu (2)

(1) Mesaj tipi ve uzunluğu

(2) Şifrelenmiş IA durumu[1]

7.2.2 Müşteri Belirtimleri

IA, EDI çeviriciden veri aldıktan sonra şu fonksiyonları gerçekleştirir:

1. Hedef IP adresi belirle

Uygun sunucuya veri iletmek için gerekli IP adresi ve kapı numarası belirlenir.

2. Sunucuya bağlan

a) SSL içeriğini yarat (bellek ayırma)

Bağlantı, sertifikalar ve diğer bilgilerle ilgili şifreleme verisi içerik bölümünde yer alır. Her bağlantı için bağımsız bir içerik belirlenmelidir.

b) SSL içeriğini ilet

SSL içeriği için değerleri belirler. Diğer SSL fonksiyonlarından önce çağırılmalıdır.

c) Yerel kütüphaneden sertifikaları Oku & Ekle

Bir SSL eş öğesini doğrularken kullanılan sertifikalar zincirine ekleme yapar.

d) Yerel kaynaktan özel anahtarı Oku & Ekle

İmzalama için kullanılan özel anahtarı belirler. Bu anahtar belirlenmiş kamusal anahtar ile uyumlu olmalıdır.

e) SSL protokol versiyonunu belirle

Bağlantıda kullanılacak SSL protokolü versiyonunu belirler. Normalde sadece SSL versiyon 3 seçilir.

f) SSL protokolünü sunucuya tanımla

Bağlantının müşteri kısmı belirlenir. Müşteriler sadece sunuculara bağlanabilir.

g) Prizi aç

Bir müşteri uygulaması bir priz yaratır ve bir servise bağlanır.

h) SSL I/O referansını belirle

Kütüphane tarafından I/O geri çağırma fonksiyonlarına iletilen referans parametreleri belirler. SSL içeriğinin yapısını belirler.

i) SSL I/O eş kimliğini belirle

SSL bağlantısı ile erişilen eş SSL ’nin kimliğini belirler. Bağlantıya bir aradan sonra yeniden devam edildiği durumlarda gereklidir.

j) SSL tokalaşma

- Müşteri Merhaba ile görevi, oturum kimliğini ve şifre bilgilerini gönderir.

- Sunucudan Merhaba mesajı aldıktan sonra, müşteri anahtarını sunucunun kamusal anahtarı ile şifreleyerek gönderir. Bu adım ilk seferde uygulanır, bağlantıya tekrar başlandığında uygulanmaz.

- Müşteri oturum kimliğini sunucunun anahtarı ile şifreleyerek Tamam işlemini yapar veya

- Sunucu Doğrulama fonksiyonu ile cevaplar ve müşterinin sertifikasını ister.

- Müşteri sertifika bilgilerini gönderdikten sonra Tamam işlemini yapar.

3. Sunucuya veri gönder

Sunucu ile bağlantı kurulduktan sonra, müşteri mesajları dizi şeklinde gönderir.

4. İletimleri kaydet

Sunucuya veri aktarımına cevap olarak kayıt işlemi başlatılır. Tarih/saat, uzak IP adresi ve kapı numarası, arabağdaşımdaki başarılı/hatalı uyarısı kaydedilmesi gereken minimum bilgilerdir.

5. Sunucunun bağlantıyı kesmesi için bekle

Hedef sunucu tarafından veri alımının başarılı olduğunun belirlenmesini sağlar. Müşteri Priz ve Zamanaşımı seçme ile bağlantı kesmeyi belirler. Şu anda kullanılan zamanaşımı süresi iki dakikadır.

6. Müşteri bağlantıyı kesme

Eğer sunucu zamanaşımından önce bağlantı kesme isteği gönderirse, müşteri şu adımları gerçekleştirir:

- SSL kapatmayı çalıştır

Bu fonksiyon eş öğe ile SSL oturumunu kapatır.

- Priz kapatmayı çalıştır

  Müşteri priz belirtecini bir parametre olarak ileterek priz bağlantısını kapatır.

- SSL içerik silmeyi çalıştır

  SSL bağlantısı tarafından kullanılan kaynakların bırakılmasını sağlar. Bitirilen oturumlar için uygulanır, daha sonra yeniden başlatılmak üzere ara verilenlerde uygulanmaz. Uygulamaya bağlı olarak içerik tarafından kullanılan bellek, içerik silmeden sonra boşaltılabilir.

Eğer sunucu bağlantı kesme isteği göndermemişse ve zamanaşımı gerçekleşmişse, müşteri hatalı durum kaydı alır ve şunları uygular:

- SSL kapatmayı çalıştır
- Priz kapatmayı çalıştır
- SSL içerik silmeyi çalıştır

7.2.3 Sunucu Belirtimleri

IA sunucu bağlantı kurulumu sırasında şu işlemleri gerçekleştirir:

Sunucuyu başlat

Bağlantı kabulü için bir priz belirlenir ve servis kapısına bağlanır. Gelen bağlantılar için bir kuyruk oluşturulur. Daha sonra bağlantılar kabul edilir. İşlem ve kapı bağlandığında sunucu kapıyı dinleyerek bağlantı isteklerini öğrenir.

Müşteriden Gelen Bağlantıyı Kabul Et

Müşteriden gelen IP adresi ve kapı numarası gibi bilgiler yardımıyla mesajlar uygun EDI çeviriciye gönderilir.

Mesaj Okuma Kurulumu

Aşağıdaki adımlar bu işlemi özetler:

SSL içeriğini yarat (bellek ayırma)

Bağlantı, sertifikalar ve diğer bilgilerle ilgili şifreleme verisi içerik bölümünde yer alır. Her bağlantı için bağımsız bir içerik belirlenmelidir.

SSL içeriğini ilet

SSL içeriği için değerleri belirler. Diğer SSL fonksiyonlarından önce çağırılmalıdır.

Yerel kütüphaneden sertifikaları Oku & Ekle

Bir SSL eş öğesini doğrularken kullanılan sertifikalar zincirine ekleme yapar.

Yerel kaynaktan özel anahtarı Oku & Ekle

İmzalama için kullanılan özel anahtarı belirler. Bu anahtar belirlenmiş kamusal anahtar ile uyumlu olmalıdır.

SSL protokol versiyonunu belirle

Bağlantıda kullanılacak SSL protokolü versiyonunu belirler. Normalde sadece SSL versiyon 3 seçilir.

SSL protokolünü sunucuya tanımla

Bağlantının sunucu kısmı belirlenir. Sunucular sadece müşterilerden gelen bağlantıları kabul edebilir.

SSL I/O referansını belirle

Kütüphane tarafından I/O geri çağırma fonksiyonlarına iletilen referans parametreleri belirler. SSL içeriğinin yapısını belirler.

SSL I/O eş kimliğini belirle

SSL bağlantısı ile erişilen eş SSL ’nin kimliğini belirler. Bağlantıya bir aradan sonra yeniden devam edildiği durumlarda gereklidir.

SSL tokalaşma

- Müşteri Merhaba alındıktan sonra sunucu, bağlantı kimliği, oturum kimliğini ve şifre bilgilerini içeren Sunucu Merhaba mesajını gönderir.

- Sunucu Müşteri Tamam mesajını bekler. Buna cevap olarak Sunucu Doğrulama mesajında görevi şifreleyerek gönderir.

- Sunucu müşterinin sertifikasını ister.

- Müşteri sertifika bilgilerini gönderir.

- Sunucu şifrelenmiş oturum kimliğini içeren Tamam mesajını gönderir.

Bütünselliği korunmuş bir EDI mesajı alındıktan sonra, alıcı mesajda belirtilen algoritmayla mesaj bölümünü hesaplar ve mesajda gönderilenle karşılaştırır. Eğer değerler eşleşirse, mesaj EDI çeviriciye gönderilir ve başarılı bir alım gerçekleştiği kaydedilir. Eğer değerler eşleşmezse, EDI mesajı şüpheli bölümde saklanır ve hata mesajı kaydedilir.

İnkar etmeme özelliği korunmuş bir EDI mesajı alındıktan sonra, alıcı mesajda belirtilen algoritmayla mesaj bölümünü hesaplar. Daha sonra gelen mesajdaki şifre çözme algoritması ve anahtarı kullanarak alınan mesaj bölümünün şifresini çözer ve hesaplanan mesaj bölümüyle karşılaştırır. Eğer değerler eşleşirse, mesaj EDI çeviriciye gönderilir ve başarılı bir alım gerçekleştiği kaydedilir. Eğer değerler eşleşmezse, EDI mesajı şüpheli bölümde saklanır ve hata mesajı kaydedilir.

7.2.4 Arabağdaşımlar

IA müşteri ve sunucu arasındaki dış bağlantılardaki arabağdaşımlar:

Veri İletişimi Protokolü: EA veri iletimini SSL3 kütüphanelerini kullanarak yapar. SSL3 kütüphaneleri ile etkileşim SSL3 araç takımı fonksiyonlarıyla gerçekleştirilir.

EDI Çeviriciler: EDI çeviricileri için kullanılan arabağdaşımlar dosyalar, API ’ler ve mesaj kuyrukları olabilir.

7.2.5 Dizayn Tartışmaları

IA ’nın daha etkili kullanılabilmesi için gerekli dizayn şartları aşağıda incelenmiştir:

Çoklu işleme/Çoklu kullanım: Her IA sunucusu etkin 16 bağlantıyı destekleyebilmelidir. Bu özellik çoklu işleme veya çoklu kullanım gibi tekniklerle gerçekleştirilebilir.

Sürekli Olmayan Bağlantılar: Her SSL3 bağlantısı tek bir EDI mesajının taşınmasını desteklemelidir. Bu da oturumun sadece tek bir EDI mesajı iletimi süresince süreceği anlamına gelir. Bu özellik sayesinde bir kesilme neticesinde belleğin boşaltılması olasılığı ortadan kaldırılır. Güvenilirlik ve performans arasında bir seçimi gerektirir.

Yeniden Başlatılabilir SSL3 Oturumları: Performans sebebiyle yeniden başlatılabilir SSL3 oturumları önerilir. Yeniden başlama süresi iki tarafın anlaşmasıyla belirlenir. Eğer yeniden başlama süresi aşılırsa bir parametre ile belirtilir. Bu süre 1-30 dakika arasında önerilir.

Mesaj Önceliği: Belirlenmiş öncelikler TCP/IP kapılarının atanmasıyla uygulanır.

7.2.6 İşletimsel Konular

Güvenlik: Eğer güvenlik servisleri desteklenecekse gerekli şifreleme parametreleri belirlenmelidir. IA genellikle RSA, CBC kipinde DES ve SHA algoritmalarını seçer. Eğer gizlilik gerekmiyorsa DES kullanılmaz.

Akış Kontrolü: EDI/SSL3 etkileşimli aracı için kullanılan iletim mekanizması, her EDI mesajı için bir SSL3 oturumu kurmayı gerektirir. İletim müşteriden sunucuya olmak üzere tek yönlü kurulur. IA durumu mesajları hataları ve diğer akış kontrolü bilgilerini iletir.

Kayıt: IA iletimleri ve hata mesajları kayıt altına alınır. Standart olarak tarih/saat, uzak IP adresi ve kapı numarası ve arabağdaşımdaki başarılı/hatalı uyarısı kaydedilmesi gereken minimum bilgilerdir.

Ayrıca iletilen verinin sabit uzunluktaki özetleri de kaydedilebilir. Kayıt biçimi sabit uzunluktaki metinler olabilir.

Yönlendirme: Mesajda belirtilen IP adresi ve kapı numarası ile uygun sunucuya iletim yapılır. Daha sonra sunucu mesajı EDI çeviriciye gönderir.

Dijital Sertifikalar: SSL3 tokalaşma esnasında dijital sertifikalar değiştirilir. Eş öğeden sertifika bilgileri geldikten sonra SSL3 iletim katmanından IA 'ya iletilir ve burada saklanır.

7.2.7 Hata İşlemesi/Giderimi

Bir IA oturumu teknik nedenlerden sonlandırılabilir. Oturumun yeniden kurulması ve son verinin yeniden iletilmesi ile sorun giderilir. Fakat müşterideki bir nokta verinin sorunsuz iletildiğini düşünür. Bu noktada son SSL başarılı uyarısı alınmıştır. Bu noktadan sonra hata giderimi, zamanaşımı süresince alınamayan doğrulama bilgisiyle tetiklenen uçtan uca uygulama yöntemiyle gerçekleştirilir.

Uygulama katmanı hata giderimi, beklenen doğrulama gelmeden zamanaşımının gerçekleşmesiyle devreye girer. Farklı önceliklerdeki mesajlar için farklı zamanaşımı süreleri belirlenir.

7.2.8 Gerçekleştirme Konuları

Diğer tedarikçiler tarafından üretilen IA 'lar ile çalışma ve kapı atamalarını kontrol etmede karşılaşılan konular aşağıda belirtilmiştir:

Birlikte Çalışabilirlik: Farklı üreticiler tarafından üretilen IA modülleri birlikte çalışabilmelidir.

Kapı Atamaları: TCP/IP kapı atamaları karşılıklı olarak kabul edilmiştir. İletimin her iki tarafında aynı kapının kullanılmasına gerek yoktur. Farklı öncelikteki iletimler için farklı kapılar atanabilir.

Ortak Sorumlulukları: Ticari ortaklar kullanılacak mesaj tipine, güvenlik seviyesine, IP adresine, kapı numarasına, kıyma ve şifreleme algoritmalarına karar vermelidirler. Ayrıca IA yazılım değişiklikleri, IP adresler, kapı atamaları, test ve sertifikasyon kriterleri ve sertifika düzenleme yetkililerini değiştirme periyotlarını da belirlemelidirler.

8. CORBA TABANLI TMN GÜVENLİĞİ

TMN ’deki EDI gibi CORBA da TCP/IP üzerinden taşınır. Bu yüzden EDI güvenliği gibi, CORBA güvenliği de SSL3 iletim katmanını kullanır.

CORBA ’nın zenginliği sayesinde yeni yaklaşımlar belirlenmiştir. Müşterinin ORB ’u CORBA servislerini kullanarak, uzak sistemden bazı işlemleri yapmasını isteyebilir. Eğer bu işlemler hassas bilgiler içeriyorsa, hangi sunucunun hangi işlemler için güvenilir olduğuna karar vermek gerekir. Çoğu TMN uygulamasında bu tip dağıtılmış işlemler gerekli değildir. Bu yüzden CORBA ile TMN güvenliği ele alınırken bu tür servisler etkisiz hale getirilir.

Uygulama katmanındaki CORBA güvenlik modülleri Güvenli İç ORB Protokolü (Secure Inter ORB Protocol – SECIOP) ile iletişimi sağlarlar. SECIOP güvenlik simgelerinin değişimini sağlar. Fakat SECIOP, GIOP mesajları parçalara ayrıldıktan sonra uygulanır ve bu yüzden inkar etmeme servisi için kullanılmaz. Bu servis daha yüksek bir seviyede ve tüm PDU için uygulanmalıdır. Taşıyıcılar servis emirleri için CORBA ’yı kullanmayı planladığından, inkar etmemeyi sağlayan ve birlikte çalışabilir ORB ’lar bulunmamaktadır. Bu yüzden uygulamaların inkar etmeme servisi için yöntem belirlenmelidir.

Güvenlik ihtiyaçları için bir bilgi modelini değiştirmemek gerekir. Bu yüzden, ilk olarak anlaşılan CORBA çözümü, CORBA mesajlarının değişimini etkilemez. Tüm inkar etmeme öğeleri ayrı mesajlarda taşınır. Sadece inkar etmeme özelliği için ayrı mesajlar taşımak sisteme yük getiren bir işlemdir. Bu özellikle ilgili bilgileri korunmuş mesajlardan ayrı olarak göndermenin bazı koşulları vardır:

Korelasyon: Bir mesajın inkar etmeme kanıtı gönderildiğinde onun hangi mesaja ait olduğu ile ilgili bir korelasyon kurulabilmelidir. Eğer bu kanıt bir cevaba aitse, hem isteğe hem de cevaba ait bilgiler yer almalıdır. Korunmuş mesaj ile birlikte inkar etmeme kanıtının gönderilmesi ile bu korelasyon sağlanmış olur.

Yeniden göndermenin önlenmesi: Bir servis sağlayıcı aldığı servis isteğindeki inkar etmeme kanıtını yeniden kullanarak, benzer servis isteklerini de karşıladığını iddia edebilir. Bu saldırıyı önlemek için her inkar etmeme kanıtı ayrı bir dizi içermelidir. Diğer güvenlik servisleriyle uyum için zaman işareti kullanılabilir.

Gönderme hatasının önlenmesi: Bir servis sağlayıcı, başka bir servis sağlayıcıya giden inkar etmeme kanıtını alarak servis isteğinin kendisine geldiğini iddia edebilir. Bu saldırıyı önlemek için ilgili alıcının belirtecini içermelidir.

Sonuçta oluşan Telekom İnkar Etmeme İç ORB Protokolüdür (Telecom Non-Repudiation Inter-ORB Protocol – TeNoRIOP).

8.1 GENEL İÇ ORB PROTOKOLÜ (GIOP)

TeNoRIOP 'u anlamak için öncelikle CORBA ve ORB arasında çalışan mesaj protokolü GIOP 'u incelemek gerekir.

Bir müşteri CORBA işlemini başlattığında istek mesajını sunucuya yollar. Bu mesaj bir GIOP **Request** mesajıdır ve nesnenin anahtarını, işlem parametrelerini ve işlemin adını taşır.

Sunucu isteğe cevap verdiğinde GIOP **Reply** mesajını müşteriye gönderir. Eğer sunucu normal işlem cevabı veriyorsa, mesajda NO_EXCEPTION durumu geçerlidir ve arabağdaşımın parametreleri bulunur. Eğer bir istisna varsa, mesajda USER_EXCEPTION durumu geçerlidir ve istisna ile ilgili bilgiler bulunur.

Müşteri istediği cevabı almadan bağlantıyı iptal edebilir. İstek ve cevap mesajlarının orijin ve alınma inkar etmeme özellikleri aşağıda anlatılmıştır. İptal isteği mesajları da aynı prensipleri izler.

ORB 'un istek ve cevap mesajlarını parçalara ayırmak için GIOP **Fragment** mesajını kullanıp kullanmaması, uygulama katmanındaki mesajın inkar etmeme özelliğini etkilemez.

8.2 TELEKOM İNKAR ETMEME İÇ ORB PROTOKOLÜ (TeNoRIOP)

TeNoRIOP tek bir ORB veya bir ORB alanında inkar etmeme özelliklerini sağlayan üreticiden bağımsız bir güvenlik mekanizmasıdır. Ayrıca eş öğe doğrulaması, gizlilik ve bütünsellik de sağlamasına rağmen, bu güvenlik fonksiyonları CORBA/SSL3'te desteklendiği için sadece inkar etmeme özelliğine odaklanır.

TeNoRIOP şu inkar etmeme servislerini sağlar:

- İsteğin orijinini inkar etmeme
- İsteğin alınmasını inkar etmeme
- Cevabın orijinini inkar etmeme
- Cevabın alınmasını inkar etmeme

İnkar etmeme (non-repudiation – NR) NREvidence nesnesi ile sağlanır. Bu nesne üç tip arabağdaşıma sahiptir:

1. **Yerel bir Uygulama Programlama Arabağdaşımı (API):** Yerel müşteri nesnesi (müşterini alanında) veya uygulama nesnesi (sunucunun alanında) ile NREvidence nesnesi arasındaki iç ORB arabağdaşımıdır. Müşterinin alanında, müşteri nesnesi NREvidence nesnesine gönderdiği isteklerin bir kopyasını iletir ve NREvidence nesnesinin aşağıdakilerden bir veya ikisini uygulamasını sağlar:

   - Orijin kanıtı için NR oluştur ve uzak NREvidence nesnesine gönder.
   - Uzak NREvidence nesnesinden alınma NR 'sini bekle ve kanıt geldiğinde doğrula.

   Sunucunun alanında, uygulama nesnesi NREvidence nesnesine müşteriden aldığı isteklerin bir kopyasını iletir ve NREvidence nesnesinin aşağıdakilerden bir veya ikisini uygulamasını sağlar:

   - Orijin kanıtı NR 'sini bekle ve kanıt geldiğinde doğrula.

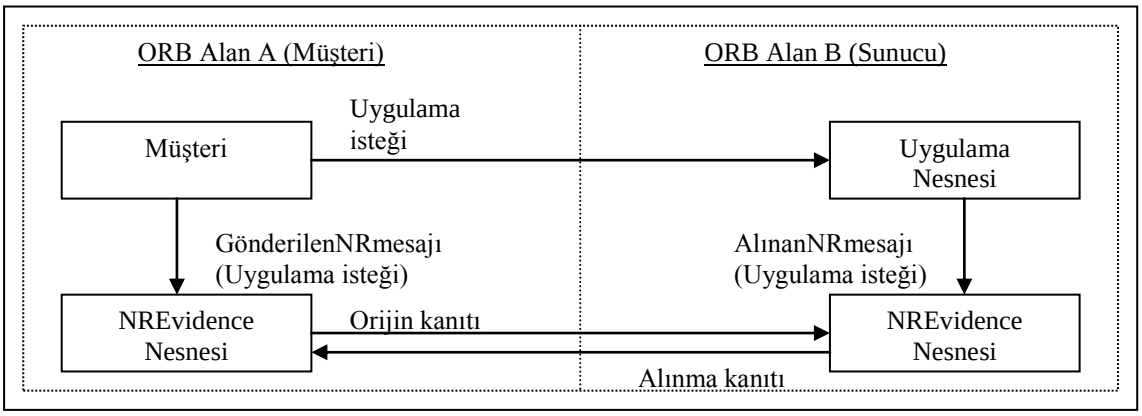
- Alınma için NR oluştur ve uzak NREvidence nesnesine gönder.

2. **Farklı alanlardaki iki NREvidence nesnesi arasındaki iç ORB arabağdaşımı:** Sunucu tarafındaki NREvidence nesnesinin, bir isteğin orijin ve bir cevabın alındı NREvidence nesnesini almasını sağlar. Ayrıca müşteri tarafındaki NREvidence nesnesinin, bir cevabın orijin ve bir isteğin alındı NREvidence nesnesini almasını sağlar.

3. **NREvidence nesnesi ve bir yerel, iç ORB güvenlik yönetim nesnesi arasındaki yönetimsel arabağdaşım:** Bu arabağdaşım, güvenlik yönetimi nesnesinin hangi imzalama algoritmasını ve hangi anahtarı kullanacağını belirler. Ayrıca aldığı mesajları, NREvidence'leri (örneğin; alınan mesajların kayıtları, NREvidence nesnesi ile uyma istekler) ve NREvidence nesnesinden gelen bilgileri (örneğin; henüz NREvidence almamış isteklerin listesi) ne yapacağına karar vermesini sağlar. Bu arabağdaşım iki ORB alanı arasındaki birlikte çalışabilir ve NR servisi için gerekli değildir.

8.2.1 İstek İçin İnkar Etmeme

Şekil 8.1'de bir uygulama isteği (GIOP istek mesajı) için orijin ve alınma inkar etmeme özellikleri gösterilmiştir.



Şekil 8.1: İstek için inkar etmeme

İstek için NR şu şekilde oluşturulur:

1. Müşteri uygulama nesnesindeki işlemi başlatır ve sonuçta oluşan uygulama istek mesajı sunucuya gönderilir.

2. Müşteri istek için mesaj bölümünü hesaplar. Mesaj bölümünü kullanarak dijital imzayı hesaplar ve istek için orijin inkar etmeme kanıtını oluşturur.

3. Müşteri orijin inkar etmeme kanıtını sunucuya gönderir.

4. Sunucu uygulama isteğini aldıktan sonra isteğin mesaj bölümünü hesaplar.

5. Sunucu, müşterinin kamusal anahtarı ile gelen inkar etmeme kanıtının şifresini çözer ve hesapladığı mesaj bölümü ile bu değeri karşılaştırır. Eğer iki değer eşleşiyorsa, istek orijin inkar etmeme için korunmuştur. Eğer iki değer eşleşmiyorsa, güvenlik politikasına göre işlem yapılır.

6. Sunucu, isteğin alındı inkar etmeme kanıtı için bir dijital imza oluşturur.

7. Sunucu alındı inkar etmeme kanıtını müşteriye gönderir.

8. Müşteri alındı kanıtını alınca sunucunun kamusal anahtarı ile şifresini çözer ve bu değeri isteğin mesaj bölümü ile karşılaştırır. Eğer iki değer eşleşiyorsa, istek alındı inkar etmeme için korunmuştur. Eğer iki değer eşleşmiyorsa, güvenlik politikasına göre işlem yapılır.

Cevap için inkar etmeme kanıtı da benzer şekilde hesaplanır.[1]

8.3 İNKAR ETMEME KANITININ ZAMANLAMASI

8.3.1 Zamanlama Anlaşmaları

Orijini İnkar Etmeme: Mesaj gönderildikten sonra en fazla bir dakika içinde inkar etmeme kanıtı da gönderilmelidir.

Ağdaki gecikmelerin en fazla iki dakika olduğu hesaplanarak alıcı kanıtı en fazla üç dakika içinde bekler.

Alındıyı İnkar Etmeme: Mesajı alan taraf kanıtı en fazla bir dakika içinde hesaplayıp göndermelidir. Tek taraflı ağ gecikmesi dört dakika olarak düşünülürse ilk mesaj gönderiminden alındı doğrulamasının gelmesi arasındaki süre en fazla dokuz dakika olabilir.

Kanıtın beklenen zamanda alınmama nedenleri aşağıdakiler olabilir:

- Gönderilen mesajın alıcı için geçerli bir mesaj olmaması.
- İnkar etmeme kanıtının iletim sırasında kesintiye uğraması.
- Kanıtı oluşturmakla görevli sistemin kanıt oluşturmaması.
- İnkar etmeme kanıtının iletim sırasında kaybolması.
- Başlatıcı sistemin geçici olarak devre dışı kalması.
- İnkar etmeme kanıtının ağdaki yük nedeniyle gecikmeye uğraması.

Bekleme süresine ilişkin aşağıdaki davranışlar gösterilebilir:

Güvenen: Karşı tarafa ve iletim hattına yüksek derecede güvenildiği için mesajların orijin kanıtları gelmeden işler ve hatta alındı kanıtını gönderir. Ancak her mesajın ve inkar etmeme kanıtının ulaşma saati kaydedilir. Belirli zamanlarda bu kayıt kontrol edilir ve kayıp kanıtlar varsa karşı taraftan istenir.

Dikkatli: Karşı tarafa ve iletim hattına orta derecede güvenildiği için mesajların orijin kanıtları gelmeden işler ve hatta alındı kanıtını gönderir. Ancak her mesajın ve inkar etmeme kanıtının ulaşma saati kaydedilir. Beklenen zaman sonunda kanıt alınmadığında karşı taraf uyarılır ve iletişim bir süre askıya alınır.

Şüpheci: Karşı tarafa ve iletim hattına düşük derecede güvenildiği için mesajların orijin kanıtları gelmeden işlemez. Ayrıca ilk mesajın kanıtı gelene kadar, daha sonra kanıtı ulaşan mesajlar gelse bile, iletişim askıya alınır.

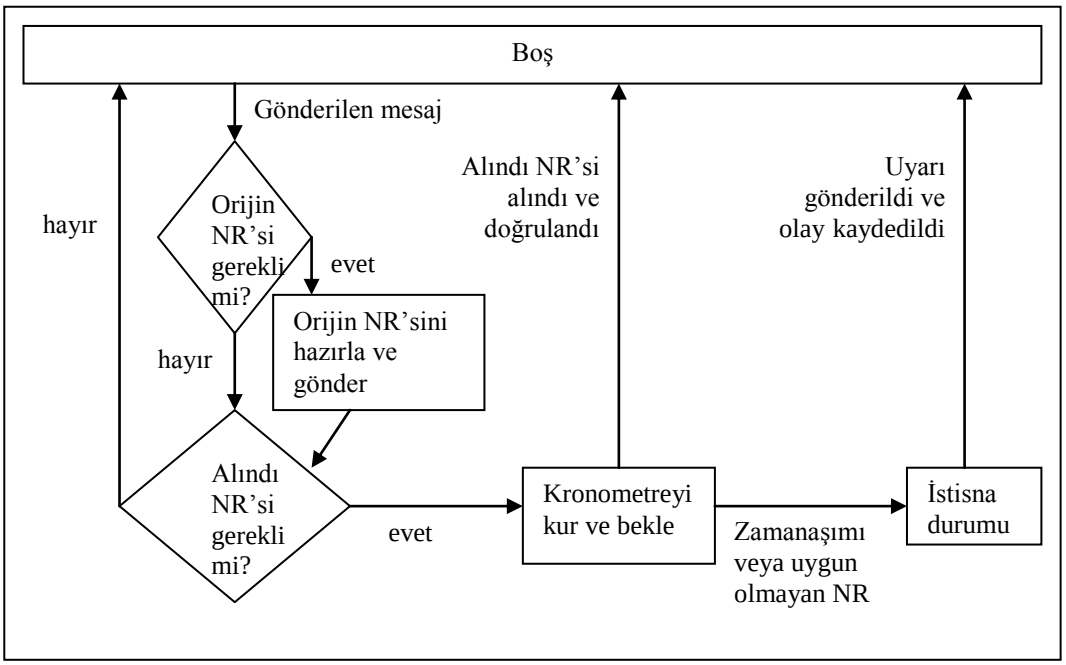
İki taraf iletişime başlamadan önce hangi davranışı göstereceklerini belirlemelidir.

8.4 İNKAR ETMEME PROTOKOL MAKİNESİ

İnkar etmeme protokol makinesinin iki elemanı vardır: mesaj gönderici ve mesaj alıcı. İstek ve cevap için kullanılan protokol makinesi aynıdır.

8.4.1 Mesaj Gönderici

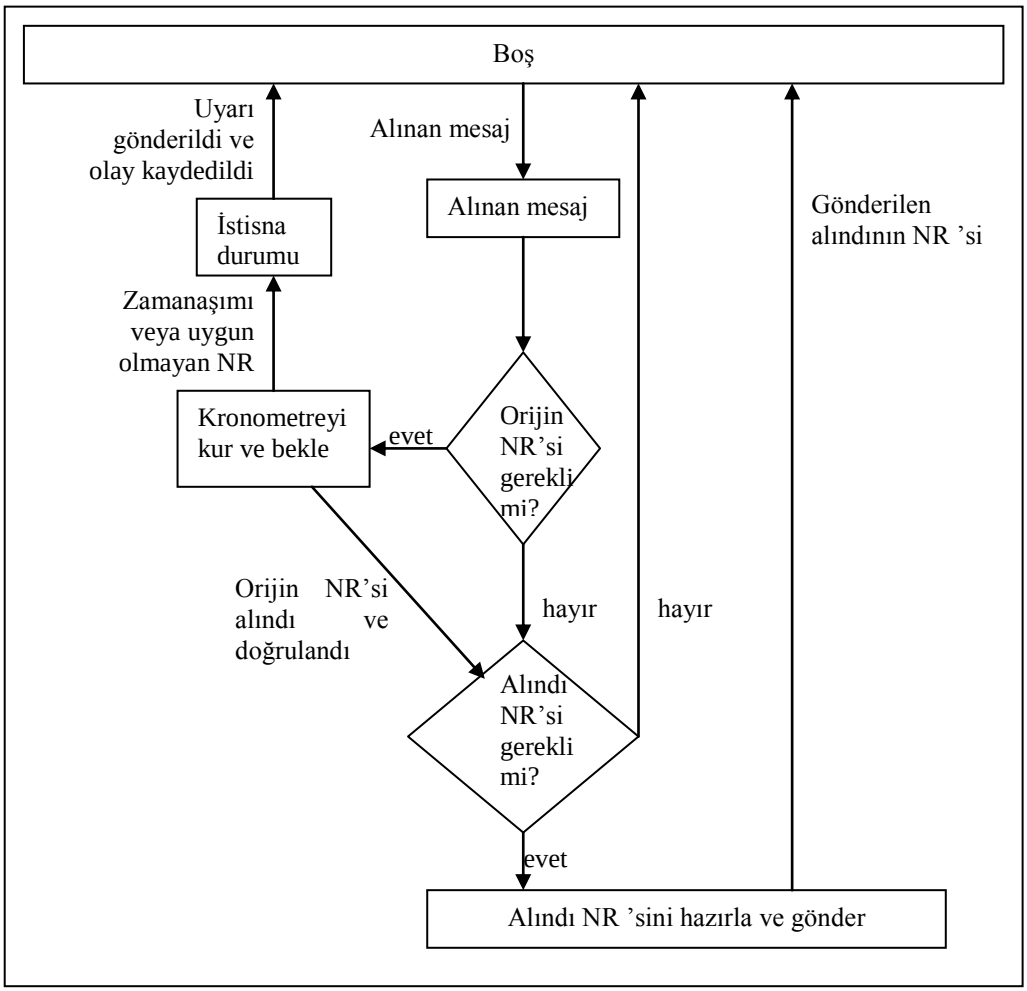
Mesaj gönderici için protokol makinesi Şekil 8.2'de gösterilmiştir.



Şekil 8.2: Mesaj gönderici protokol makinesi

8.4.2 Mesaj Alıcı

Mesaj alıcı için protokol makinesi Şekil 8.3'te gösterilmiştir.



Şekil 8.3: Mesaj alıcı protokol makinesi

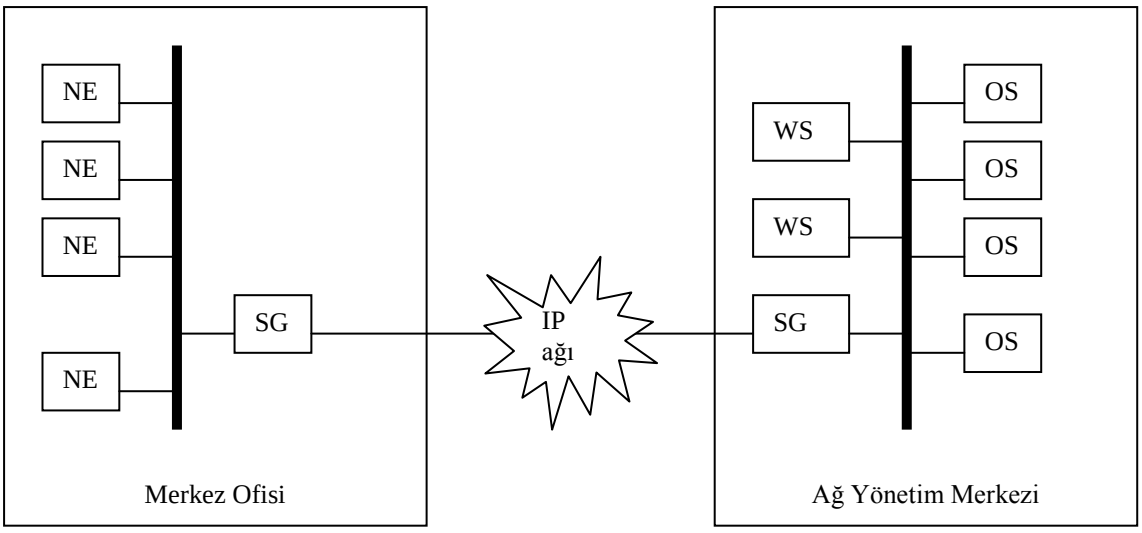
9. SNMP TABANLI TMN GÜVENLİĞİ

SNMP üç farklı versiyona sahiptir ve bu üç versiyonun güvenlik yapıları da farklıdır.

9.1 SNMPv1 GÜVENLİĞİ

SNMPv1'de güvenlik tanımlanmamıştır. Ancak SNMP iletimlerinin güvenliği IPsec kullanılarak sağlanabilir. Bu durumda SNMPv1 ile yönetilen her öğeye IPsec güvenlik parametrelerini tanıtmak gereklidir. IPsec 'in minimum protokolleri şifreleme için DES ve kıyma için MD5 ve SHA 'dır. Bu protokoller SNMP dışı TMN iletimleri için de en uygun protokol grubudur.

SNMPv1 güvenliği için IPsec kullanımını kolaylaştıracak bir yöntem, TMN öğelerinin Güvenlik Ağ Geçidi (Security Gateway – SG) kullanmalarıdır. Bu durum Şekil 9.1'de gösterilmiştir.



Şekil 9.1: TMN öğelerini koruyan Güvenlik Ağ Geçitleri (SG)

Şekil 9.1'de merkez ofisteki bir SG birçok NE 'ye girişi kontrol eder. Ağ yönetim merkezindeki diğer bir SG de OS ve WS 'lere girişleri kontrol eder. Eğer bu iki yer de güvenli ise IPsec SG 'den SG 'ye bir güvenlik tüneli oluşturarak, iki yer arasında değiştirilen tüm SNMP mesajlarını koruyabilir. Çoğu durumda doğrulama yeterli

olmaktadır. Bu durumlarda IPsec 'in sadece Doğrulama Başlığı (Authentication Header – AH) protokolü gereklidir. SG korunmamış bir IP adresi yerine doğrulanmış göndericinin belirtecini kullanır ve bir ateş duvarı gibi davranır.

IPsec inkar etmeme servisi sağlamaz. Fakat SNMP NE 'ler ve öğe yönetim sistemleri arasında bir arabağdaşım görevi görür. Bu arabağdaşımların inkar etmeme servisine ihtiyaçları yoktur.

9.2 SNMPv2 GÜVENLİĞİ

SNMPv2 mesajları da SNMPv1 mesajları gibi IPsec yardımıyla korunabilir. Fakat daha iyi bir yöntem SNMPv2 içerisindeki güvenlik özelliklerini kullanmaktır.

9.2.1 Uygun ID Gerekliliği

Güvenliğin en temel öğesi haberleşen öğelerin birbirlerini doğrulamalarıdır. Versiyon 2 haberleşen her öğeye özel bir **SnmpParty** belirler. **SnmpParty** sadece bir ID değil; kullanılacak iletim servisini, iletim adresini, kabul edilebilir maksimum mesaj uzunluğunu, doğrulama protokolünü, yerel zamanı, özel ve kamusal doğrulama anahtarlarını, alınan mesajlar için kabul edilebilir maksimum gecikmeyi, şifreleme protokolünü, özel ve kamusal şifreleme anahtarlarını da içeren bir dizidir. Tüm bu bilgiler yerel Yönetim Bilgi Tabanında (Management Information Base – MIB) saklanır. Bu sebeple uzaktaki bir güvenlik yöneticisi tarafından da idare edilebilir.

Bir Versiyon 2 yöneticisi bir aracının MIB 'ine girmeye çalıştığında sadece uygun ID'yi değil giriş sebebini de açıklamalıdır. Aracı yöneticiden bir mesaj aldığında değişkenler kısmındaki bilgileri ve istenen içerik için kullanma hakkını kontrol eder.

9.2.2 Zaman

Yeniden gönderme saldırılarına, sıralama problemlerine ve mesaj zamanlamasına karşı zaman işaretleri kullanılabilir. Bu işlem için kullanılan saatlerin senkronize olmaları gerekir. OSI'de evrensel zaman kavramıyla bu senkronizasyon sağlanır, ancak IP'de her öğenin kendi saati vardır. Bu saat öğe üretildiğinden veya son olarak yeniden yüklendiğinden itibaren saniyeyi sayan bir tamsayı sayacıdır. Dolayısıyla

gelen bir mesajdaki zaman işaretini yerel zamanla karşılaştırmak faydasızdır. Gönderilen mesajın zamanlamasını öğrenebilmek için göndericinin saati bilinmelidir. Eğer iki öğe SNMPv2 üzerinden haberleşeceklerse birbirlerinin saatini kendi MIB 'lerinde tutmalıdırlar. İlk olarak bu saat 0'a ayarlanır. Öğelerden biri setRequest mesajı ile diğerinin MIB 'indeki saatini ayarlar. Bu mesaj korunmasız bir mesajdır ve sadece karşı tarafın saat bilgilerini kaybettiği veya geride kaldığından şüphelenilen durumlarda gönderilir. İki öğe de hem kendi saatlerini hem de MIB 'lerinde tuttukları saatleri saniyede bir arttırırlar.

Öğelerden biri karşı taraftan korunmuş bir mesaj aldığında iki zaman işareti içerir. Karşı tarafın saati ve kendi saatinin karşı tarafın MIB 'inde duran kopyası. Eğer kendi MIB 'inde tuttuğu kopya, aldığı mesajdaki karşı tarafın saati artı izin verilen mesaj yaşam süresinden büyükse mesaj reddedilir.

Eğer iki öğe bir süredir haberleşmiyorlarsa, saatleri kaymış olabilir. Yeniden senkronizasyon için alınan korunmuş mesajdaki zaman işaretleri kullanılır. Eğer mesajdaki zaman MIB 'de saklanandan büyükse, MIB 'deki saat mesajdakine ayarlanır. Eğer mesajdaki zaman kopyası gerçek zamandan büyükse, gerçek zaman karşı tarafın düşündüğü değere göre ayarlanır. Eğer alınan mesajdaki saatler alıcıdakilerden küçükse herhangi bir işlem yapılmaz. Ancak bu durum farklı öğeler ile haberleşirken de meydana geliyorsa saati kalibre etmek gerekir.

9.2.3 Güvenli PDU 'lar

Güvenli v2 mesajının getirdiği ek yük SnmpMgmtCom olarak adlandırılır ve alıcının kimliğini, gönderenin kimliğini, hedef MIB görüntülenme kimliğini ve korunmamış SNMPv2 mesajını içerir.

SNMPv2 doğrulaması kıyma fonksiyonu kullanarak gerçekleştirilir. Doğrulama mesaj bölümü; göndericinin özel doğrulama anahtarı, göndericide bulunan karşı tarafın saatinin kopyası ve göndericinin saatine kıyma fonksiyonu uygulanmasıyla oluşur.

Gerçek doğrulama verisi yukarıdaki gibi hesaplanan mesaj bölümü, göndericide bulunan karşı tarafın saatinin kopyası ve göndericinin saatinden oluşur.

SNMP bağlantısız bir protokol olduğundan güvenlik içeriği üzerinde anlaşma söz konusu değildir. Bunun yerine her SNMPv2 öğesi güvenlik içeriklerini kendi MIB'lerinde tutar ve bu bilgiler uzaktan yönetilebilir.

9.3 SNMPv3 GÜVENLİĞİ

v1 ve v2'de olduğu gibi SNMPv3 de IPsec kullanılarak korunabilir. Fakat SNMPv3'ün kendi güvenlik özellikleri daha kolay yönetilebilir. SNMPv3 güvenliğinin iki temel öğesi vardır: SNMPv3 iletimlerini korumak için Kullanıcı Tabanlı Güvenlik Modeli (User-based Security Model – USM) ve giriş kontrolü sağlamak için Görüntü Tabanlı Giriş Kontrol Modeli (View-based Access Control Model – VACM).

9.3.1 Kullanıcı Tabanlı Güvenlik Modeli

USM, gecikmeyi sezme ve gizlilik koruması ile birlikte veya tek başına bütünsellik sağlar.

İki SNMPv2 öğesi haberleşecekleri zaman kendi MIM 'lerinde tuttukları diğerinin saatinin kopyasını kullanırlar. SNMPv3 bu işlemi kolaylaştırır. Yönetici aracının saatini tutar ancak aracı yöneticinin saatiyle ilgilenmez. SNMPv3 her mesaj için aşağıdaki gibi bir **yetkili SNMP öğesi** oluşturur:

- Cevap gerektiren bir mesaj için alıcı yetkili öğedir.
- Cevap gerektirmeyen bir mesaj için gönderici yetkili öğedir.

Her güvenli SNMPv3 mesajı kullanıcı adını, o mesaj için yetkili SNMP öğesini ve zaman gerekli ise yetkili öğenin saatini içermelidir. Saat iki tamsayı değerinden oluşur: SNMP öğesinin yeniden yüklenme sayısı (**snmpEngineBoots**) ve son yeniden yüklenmeden sonra geçen zaman (**snmpEngineTime**). Yetkili öğe için bu parametreler **msgAuthoritativeEngineBoots** ve **msgAuthoritativeEngineTime** olarak adlandırılırlar. Bu parametreler için izin verilen en yüksek değer 2.147.483.647 (2^{32}-1) olup oldukça uzun bir SNMP öğesi yaşam süresidir.

Aracının saatini öğrenmek için yönetici boş ve korunmamış bir mesaj gönderir. Bu mesajda msgAuthoritativeEngineBoots ve msgAuthoritativeEngineTime değerleri sıfıra ayarlanmıştır. Cevapta ise msgAuthoritativeEngineBoots ve msgAuthoritativeEngineTime ’ın gerçek değerleri yer alır.

Yetkili olmayan bir SNMP öğesi yetkili bir SNMP öğesinden bir mesaj aldığında, alınan mesajdaki msgAuthoritativeEngineBoots ve msgAuthoritativeEngineTime değerlerini kendi sakladığı değerlerle karşılaştırır. Eğer;

- alınan msgAuthoritativeEngineBoots değeri saklanan değerden büyükse, veya

- alınan msgAuthoritativeEngineBoots saklanan değerle aynıysa ve alınan msgAuthoritativeEngineTime değeri saklanan değerden büyükse

yetkili olmayan öğe sakladığı değerleri günceller.

SNMPv3 hem pek çok sisteme tek girişi hem de anahtar yönetimini destekler.

Tek giriş, kullanıcı (genellikle bir yönetici) ile hedef sistem (genellikle bir aracı) arasındaki paylaşılmış gizin kullanıcının gizli şifresi ve aracının belirteci yardımıyla hesaplanması ile sağlanır. Öncelikle kullanıcının şifresi gerektiği kadar tekrar edilir ve 1.048.576 oktet elde etmek için yuvarlanır. Oluşan dizi kıyma fonksiyonuna tabii tutulur ve mesaj bölümü oluşturulur. Kıyma fonksiyonu sonucu oluşan paylaşılmış giz, aracı SNMP ’ye yüklenir ve kullanıcı tek bir şifre ile farklı kaynaklara giriş yapabilir.

Kullanıcı şifresini her değiştirişinde diğer sistemlerle paylaştığı gizli anahtarlar da güncellenmelidir. Bu işlem bir SNMP iletimi ile sağlanır. Yeni anahtarı şifrelemek için kıyma fonksiyonu kullanılır.

USM PDU ’ları şu parametrelerin BER ile kodlanmasıyla oluşur:

- **msgAuthoritativeEngineID,** o mesaj için yetkili SNMP öğesini bildirir

- **msgAuthoritativeEngineBoots**

- **msgAuthoritativeEngineTime**

- **msgUserName,** mesajın kime gönderildiğini belirtir

- **msgAuthenticationParameters,** mesajı korumak için kullanılan doğrulama mekanizması için gerekli bilgileri taşır.

- **msgPrivacyParameters,** mesajı korumak için kullanılan gizlilik mekanizması için gerekli bilgileri taşır.

USM API 'leri TMN 'de tanımlı tüm öğeler için modülerlik ve yazılımın yeniden kullanımını sağlar.

Güvenli SNMPv3 mesajları gönderenin doğrulanmasını gerektirir. Gelen bilgilere göre istek kabul veya reddedilir. SNMPv3'nin bu özelliği bir sonraki bölümde anlatılmıştır.

9.3.2 Görüntü Tabanlı Giriş Kontrol Modeli

Bu özellik gelen mesajların kabul veya reddedilmesine karar verirken kullanılır. Bu karar gönderenin kimliğine ve yerel olarak saklanan giriş kontrol bilgilerine bağlıdır. Bu bilgiler her kullanıcının her bilgi öğesi için önceliklerini içerebilir. Fakat böyle bir yaklaşımda giriş kontrol bilgileri gerçek bilgiden daha fazla yer tutabilir. Bu durumda akıllı dizaynlar kullanılarak giriş kontrol bilgilerinin gösterimi değiştirilebilir. VACM hem potansiyel kullanıcıların hem de korunmuş bilgilerin gösterimlerini adresler.

Ağ yönetiminde pek çok kullanıcı aynı önceliklere sahip olabilir. Her kullanıcı için öncelikleri tekrar tekrar yazmak yerine aynı haklara sahip kullanıcıları listelemek daha pratiktir.

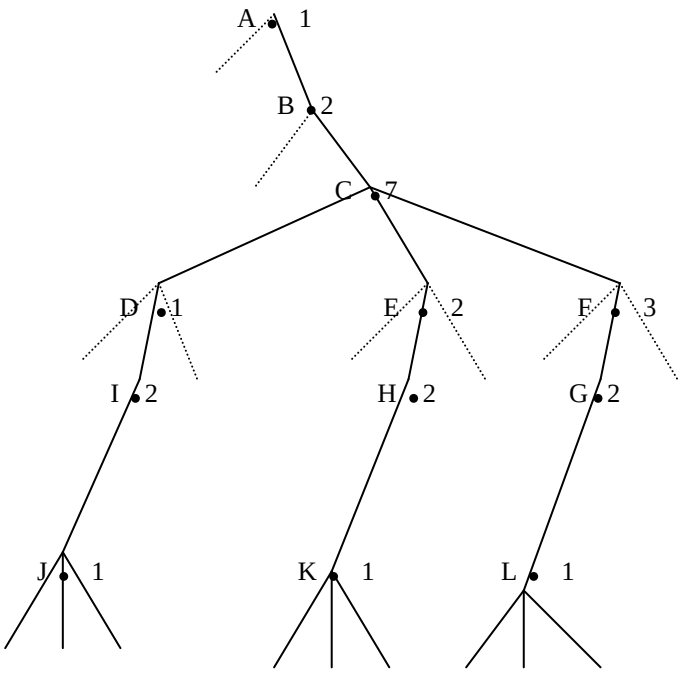
Tek kullanıcılar kendi güvenlik modelleri ve kendi özel güvenlik adları ile belirlenirler. Bir grup kullanıcı ise bir veya daha çok tek öğeyi içerir ve grup adı ile belirlenir.

Büyük bir MIB yüz binlerce yönetilen öğe içerebilir. Tüm öğelerin numaralandırılarak grupların kullanımına sunulması mantıklı değildir. VACM izin verilen giriş kipleri ile daha ayrıntıya inebilir.

SNMPv3 mesajlarının ek yükü içeriği belirten bir ifadeyle başlayan PDU 'lardır. Giriş kontrolünün ilk adımı mesajı gönderen kişinin bağlı olduğu grubun, içerikte belirtilen araç ile yasal bir işi olduğunun onaylanmasıdır. Aynı anda giriş kontrol fonksiyonu, içerikte belirtilen öğelerin PDU 'da bulunduğunu onaylar. Onaylama olmadığında veya içerik bilinmediğinde istek reddedilir.

Farklı gruplardaki üyeler bir SNMP öğesindeki herhangi bir cihaz için farklı haklara sahiplerdir. VACM cihaza yasal gereksinimi olan farklı gruplara MIB görüntüleri sağlar. MIB görüntüsü görüntüye dahil edilmiş MIB ağaçlarından ve görüntüden çıkarılmış MIB ağaçlarından oluşur.

MIB görüntüsündeki tüm ağaçları numaralandırmak zor olacağından, VACM ilişkili ağaçları bir araya getiren bir yöntem uygular. Şekil 9.2'de bir görüntü ağacı gösterilmiştir.



Şekil 9.2: Görüntü ağacı

Şekil 9.2'de MIB 'in bir kısmı gösterilmiştir. Her nokta onu diğerlerinden ayıran bir tamsayı ve harfe sahiptir. Numaralar kökten itibaren verilir ve OID adını alır. Örneğin (1,2,7,1,2,1) dizisi J düğümüne karşılık gelir.

VACM tek bir servis sağlar; gelen isteklerin veya giden bildirimlerin kabul edilip edilmeyeceğini belirler. Bu servis **isAccessAllowed** adını taşır ve şu parametrelerden oluşur:

- **securityModel**, mesaj için hangi güvenlik modeli uygulanacak (örneğin; USM)
- **securityName**, giriş yapmak isteyen kişi
- **securityLevel**, mesaj için hangi güvenlik seviyesi uygulanacak (örneğin; USM için güvenlik yok, sadece doğrulama ve gizlilik)
- **viewType**, okuma, yazma veya değiştirme
- **contextName**, PDU için
- **variableName**, yönetilen nesnenin OID adresi

isAccessAllowed şu cevaplarla dönebilir:

- **accessAllowed**, giriş isteğinin kabul edildiğini belirtir (diğer tüm cevaplar isteğin reddedildiğini belirtir)
- **notInView**, verilen OID görüntü ağacında değil
- **noSuchContext**, VACM verilen içeriği tanımıyor
- **noGroupName**, verilen ada karşılık grup bulunamadı
- **noAccessEntry**, verilen özelliklerde bir giriş bulunamadı
- **otherError**, belirlenmemiş hata

VACM giriş değerleri ile yerel olarak sakladığı giriş kontrol bilgilerini karşılaştırarak karar verir. Bu bilgiler MIB ’de tutulur ve uzaktan yönetilebilir.

10. SONUÇLAR VE TARTIŞMA

Bu çalışmada TMN 'in yapısı, güvenlik servisleri, güvenlik mekanizmaları ve destek mekanizmaları gibi temel güvenlik kavramları ve TMN 'in güvenliği için kullanılan dört ayrı yöntem incelenmiştir. Bu yöntemler: OSI tabanlı TMN güvenliği, EDI tabanlı TMN güvenliği, CORBA tabanlı TMN güvenliği ve SNMP tabanlı TMN güvenliği yöntemleridir.

TMN yapısı, fonksiyonel yapı, fiziksel yapı ve haberleşme / bilgi yapısı olarak ele alınmıştır. Bu yapıyı tamamlayan lojik katmanlanmış yapı da incelenmiştir.

Güvenlik servisleri içerisinde bağlantı giriş kontrolü, eş öğeleri doğrulama, veri orijini doğrulama, bütünsellik, gizlilik, inkar etmeme, giriş kontrolü, güvenlik alarmı ve güvenlik tetkiki ele alınmıştır.

Güvenlik mekanizmaları içerisinde kıyma fonksiyonu, şifreleme, dijital imzalar, sertifikalar, giriş kontrol mekanizmaları ve doğrulama protokolleri ele alınmıştır.

Destek mekanizmaları içerisinde ise güvenlik alarmları, güvenlik tetkik kaydı, anahtar dağıtımı, rehber, güvenlik protokolleri, GSS – API, GULS ve SSL3 ele alınmıştır.

TMN güvenliği yöntemleri içinde OSI tabanlı TMN güvenliği, temel OSI kavramlarını kullandığı için oldukça yaygın bir kullanım alanına sahiptir. Katmanlı yönetim yapısı ile uygulamalarla uyum sağlar. Fakat oldukça karmaşıktır.

EDI tabanlı TMN güvenliği yöntemi, basit doküman iletimlerinde kullanılmak üzere tasarlanmıştır. OSI gibi karmaşık modellerin kullanılmayacağı yerlerde kullanılır.

CORBA tabanlı TMN güvenliği yöntemi, dağıtılmış işleme ve yazılım yeteneğine sahiptir. Kullanım kolaylığı, düşük maliyeti ve etkili araçları sayesinde tercih edilir. CORBA 'da çalışan tüm uygulamalar, CORBA 'yı destekleyen platformlarda da çalışır. Ancak OSI tabanlı güvenlik kadar güçlü bir yapıya sahip değildir.

SNMP tabanlı TMN güvenliği, oldukça basittir ve uzaktan yönetimi destekler. İnternet uygulamaları için idealdir. Yapısındaki ve uygulamalarındaki kolaylık nedeniyle yaygın kullanıma sahiptir.

TMN güvenliği konusunda gelecekte beklenen gelişmelerden biri tek bir TMN içerisindeki farklı alanlar arasında güvenli iletişimin sağlanabilmesidir. Örneğin bir TMN içerisinde hem X.25 hem de TCP/IP alanları bulunabilir ve bu alanların sınırları çakışabilir. Bu çakışmaların ortadan kaldırılması ve birlikte çalışabilirliğin sağlanabilmesi için yöntemlerin geliştirilmesi gerekmektedir.

Güvenli olmayan bir aracı cihaz üzerinden haberleşebilmek için uygulama tabanlı güvenlik kullanılabilir. Bu aracı üzerinden farklı uygulama katmanı protokolleri kullanan (örneğin CORBA ve CMIP) haberleşme sistemleri, uygulamalarındaki bazı önemli bölümleri şifreleyebilirler.

Bir beklenti de tüm TMN sistemlerinin CORBA üzerinden haberleşmesidir. Şifrelenmiş CMIP PDU 'ları CORBA CIOP mesajları içerisinde taşınabilir. CORBA bu mesajlar için PDU seviyesinde güvenlik sağlar.

Güvenlik alarmları ve tetkiklerinin TMN içerisindeki tanımları geliştirilmeli ve kayıt yerleri, uyarının gideceği adresler ve kayıt saklama süreleri gibi konular belirlenmelidir.

KAYNAKLAR

[1] **Rozenblit, M.,** 2000. Security for Telecommunications Network Management, IEEE Press, Piscataway.

[2] **M.3010,** 2000. Principles for a Telecommunications Management Network, *ITU-T Recommendation*, Melbourne.

ÖZGEÇMİŞ

İlknur Koçoğlu 04.01.1981 yılında İstanbul'da doğdu. İlk ve orta okulu Bakırköy'de tamamladı. Bakırköy Yahya Kemal Beyatlı Lisesi'nden 1997 yılında birincilikle mezun oldu ve aynı yıl İstanbul Üniversitesi Elektronik Mühendisliği (İngilizce) bölümünü kazandı. Lisans eğitimini 2002 yılında üçüncülükle tamamladı ve aynı yıl İstanbul Teknik Üniversitesi Telekomünikasyon Mühendisliği bölümünde yüksek lisans eğitimine başladı. 2002 yılından beri Testo Elektronik firmasında görev yapmakta.