Bir Ağ Yönetim Sistemi: GuardiLAN

Erhan Altıntaş, Onur Özardıç, Salih Talay, Tolga Ayav
Bilgisayar Mühendisliği Bölümü
İzmir Yüksek Teknoloji Enstitüsü, İzmir
{altintas, ozardic, stalay, tayav}@arf.iyte.edu.tr

ÖZET

Bilgisayar ağlarındaki hızlı gelişime rağmen ağ yönetim sistemleri bu gelişimin gerisinde kalmaktadır. Ağların hızla gelişmesi, ağda oluşan trafik tiplerindeki çeşitlilik, ve kaynaklardaki kısıtlılık **Servis Kalitesi** ve **Network Yönetimi** konularını daha önemli hale getirmiştir. Bu bildiride, İzmir Yüksek Teknoloji Enstitüsü'nün yerel bilgisayar ağı için geliştirilen GuardiLAN projesi sunulacaktır. GuardiLAN ilk aşamada 3 kısımdan oluşan, modüler ve web tabanlı bir ağ yönetim sistemidir. Birinci kısım ağ topolojisinin belirlenmesinden ve takibinden sorumlu olup, ikinci kısım kullanıcıların IP adreslerinin ve MAC adreslerinin düzenli olarak kontrolüyle ağ güvenliğinin yükseltilmesini sağlamaktadır. Son kısım ise geniş alan ağı bağlantısının etkin kullanımı için geliştirilmiş bir akıllı bant genişliği yönetim aracıdır.

ÖZGEÇMİŞ

Erhan Altıntaş
1980 yılında Hasköy (Bulgaristan)'da doğan Erhan Altıntaş, 1998 yılında İzmir Şirinyer Lisesi'ni, 2003 yılında da İzmir Yüksek Teknoloji Enstitüsü Bilgisayar Mühendisliği Bölümü'nü bitirdi. Bugüne dek çeşitli web tabanlı yazılım geliştirme projelerinde görev almıştır.

Onur Özardıç
1981 yılında Bartın'da doğan Onur Özardıç, 1999 yılında Bartın Davut Fırıncıoğlu Anadolu Lisesi'ni, 2003 yılında da İzmir Yüksek Teknoloji Enstitüsü Bilgisayar Mühendisliği Bölümü'nü bitirdi.

Salih Talay
1982 yılında Sakarya'da doğan Salih TALAY, 1999 yılında İzmir Buca Anadolu Lisesi'ni, 2003 yılında da İzmir Yüksek Teknoloji Enstitüsü Bilgisayar Mühendisliği Bölümü'nü bitirdi.

Tolga Ayav
1974 yılında İzmir'de doğdu. 1991 yılında İzmir Atatürk Lisesini, 1995 yılında Dokuz Eylül Üniversitesi Elektrik-Elektronik Mühendisliği Bölümü'nü bitirdi. 1998 yılında İzmir Yüksek Teknoloji Enstitüsü Bilgisayar Müh. Bölümünde Yüksek Lisans öğrenimini tamamladı. Halen bu bölümde Öğretim Görevlisi olarak görev yapmakta olup Ege Üniversitesi Bilgisayar Müh. Bölümünde de doktora öğrenimini sürdürmektedir.

ABSTRACT

Present day's growth of networks and diversity in types of traffic may cause congested networks, which brings a great demand for **Quality of Service** (QoS) and **Network Management**. In this paper, the project namely GuardiLAN, which has been developed for Izmir Institute of Technology's local area network, is presented. GuardiLAN consists of three sub-projects; network topology discovery and management, IP-MAC address matching, and intelligent bandwidth management for the WAN connection.

1. GİRİŞ

Bilgisayar ağlarında oluşan trafik, yüksek oranlarda farklılıklar göstermekte ve her bir trafik tipi; bant genişliği, gecikme, gecikme sürelerindeki değişim ve bulunabilirlik faktörleri açısından kendisine özgü gereksinimlere sahip olmaktadır. Internet'in kullanımının patlamasıyla, bu trafiklerin büyük bir kısmı IP tabanlı olmuştur. Tek bir transfer protokolünün kullanılması, hem gerekli olan donanımın daha az karmaşık olması, hem de işletim bedellerinin düşük olması açısından yararlı olmaktadır. IP'nin paket anahtarlamalı bağlantısız bir protokol olması bilgisayar ağlarında Servis Kalitesi açısından belirsizlikler getirmektedir. Tüm bunlar ağ yönetimini daha zor hale getirmektedir. Bu bildiride sunulan ve üç kısımdan oluşan GuardiLAN projesi İzmir Yüksek Teknoloji Enstitüsü yerel ağının yönetimsel ihtiyaçlarını karşılamak için geliştirilmektedir. İzmir Yüksek Teknoloji Enstitüsü Kampüs alanı ve buna bağlı olarak kampüs yerel alan ağı hızla genişlemektedir. Bunun sonucunda da ağ içinde bazı kontrol ve yönetim gereksinimleri ortaya çıkmıştır.

En temel ihtiyaç ağın topolojisinin belirlenmesi ve bir haritasının oluşturulmasıdır. Bu harita sayesinde her türlü cihazın çalışma durumu, hat durumları ve hatların kullanım oranları izlenebilir. Bu sayede ağda oluşabilecek problemler cok daha hızlı bir şekilde tespit edilebilir. Takip eden bölümde topoloji belirleme çalışması detaylı olarak anlatılmaktadır.

Ağ içindeki sorunlardan biri de yerel alan ağının kendi içindeki güvenlik gereksinimleridir. Ağ içindeki bir kullanıcı, bir başka kullanıcının IP adresini ele geçirip kullanarak, o kullanıcıya tanınmış ve başkalarının erişimi ya da kullanımına izin verilmeyen hizmetlere ya da bilgilere ulaşabilir; o kullanıcı adına ağ içinde trafik oluşturabilir ve hatta bunu değişik kaynaklara saldırı amaçlı kullanabilir. Bu durumun önüne geçebilmek için, her kullanıcının sürekli aynı IP adresiyle ağa giriş yapıp yapmadığının kontrol edilmesi gerekmektedir. Bu konuyla ilgili başka bir sorun ise, ağa yeni dahil olan ve daha önceden ağın bir elemanı olduğuna dair güvenilir terminal olarak belirlenmemiş terminallerin belirlenmesidir. Bu durum, ethernet ortamında çok büyük bir sorun oluşturmayabilir; çünkü bu yeni terminalin ağa giriş yapabileceği fiziksel noktalar genellikle denetim altındadır; ancak saldırının içeriden de gelebileceği göz önünde bulundurulduğunda bu durum da bir risk faktörü oluşturmaktadır. Ancak, kampüs ağını oluşturan alt ağlardan herhangi birinin kablosuz ağ olması durumunda, fiziksel olarak kontrol edilemeyen bir noktadan gelebilecek bir dış saldırı önemli bir risk oluşturur. Saldırı, ağ içindeki herhangi bir kaynağa zarar verilmesi olabileceği gibi, servislerde kesintilere sebep olunması, korumalı bir bilgiye ulaşılması ve hatta yalnızca ağ içindeki akan trafiğin dinlenmesi olarak da karşımıza çıkabilir. Bu ve benzeri durumlarda da, ağ içinde güvenilen terminaller belirlenmeli ve ağa eklenen her yeni terminal bulunmalı, bu terminal hakkında güvenilir ya da güvenilmez kararının verilmesi ve o terminale tahsis edilmiş olan bant genişliğinin buna göre ayarlanması gerekmektedir. Bu projenin ikinci kısmı IP ve MAC adreslerini karşılaştırmak suretiyle uyumsuzlukarı tespit ederek ve ağa yeni eklenen kullanıcıları belirleyerek ağ güvenliğinin yükseltilmesinde rol oynamaktadır.

Yerel ağdaki diğer bir problem de WAN bağlantısındaki sıkışma veya verimsiz kullanımdır. Bu bağlantının sıkışmasını önlemek amacıyla mevcut bant genişliğinin artırılması, akla gelen ilk çözümdür. Ancak sorun, basit bir kapasite sorunundan daha ötesidir. Durum, trafiğin hacimsel olarak artışının yanında yapısal olarak da değişmesidir. IP tabanlı uygulamalar çok sayıda yeni trafik tipleri doğurmakta ve oldukça farklı yapıda operasyonel gereksinimler ortaya çıkarmaktadır. Son zamanlarda peer-to-peer programların gelişmesiyle mevcut bant genişliklerinin büyük bir kısmı bu programlar tarafından harcanmaya başlamıştır. Bu ise bir çok kurum için, o kurumun Internet kullanım politikasına bağlı olarak, önemli bir sorun teşkil etmektedir. Projenin üçüncü ve son kısmı mevcut bant genişliğini kullanıcılara kurumun politikalarına bağlı olarak otomatik olarak paylaştıran bir akıllı bant genişliği yönetim programıdır. Bu projeyle ilgili detaylı bilgi son bölümde verilmektedir.

Bunların dışında, ağ içindeki trafiğin ve kullanıcılara ait belli başlı kullanım verilerinin derlenmesi, bunların işlenmesi ve oluşan bilgilerin ağ yöneticisi tarafından yönetsel karar verme aşamasında kullanılması ağın verimliliğini arttıracak bir faktördür. Bunun için üç proje tarafından kullanılabilecek ortak bir veritabanı oluşturulmuştur. Veritabanı ile ilgili detaylar da takip eden bölümlerde verilecektir.

2. FİZİKSEL AĞ TOPOLOJİSİ BELİRLEME

2.1. GİRİŞ

Bir bilgisayar ağının fiziksel topolojisi, o bilgisayar ağına bağlı kullanıcıların bilgisayarları, ağ yazıcıları, kesintisiz güç kaynağı takip sistemleri, bu sistemleri birbirlerine bağlayan ethernet anahtarları, yönlendiriciler ve bunların birbirleriyle olan fiziksel (kablolu veya kablosuz) bağlantılarını ifade eder. Fiziksel topoloji, Veri Bağlama katmanı (OSI – 2) seviyesindeki bağlantıları içerir. Mantıksal ağ topolojisi ise Ağ katmanı (OSI – 3) seviyesinde bağlantıları ifade eder ve fiziksel topolojiden farklı olarak yerel ağı oluşturan bağlantıları görmezden gelir. Mantıksal topoloji daha çok Geniş Alan Ağları seviyesinde anlam taşır.

Modern IP tabanlı ağların yönetimi gün geçtikçe daha önemli bir hale gelmektedir. Ağa katılan yeni kullanıcılar, eklenen yeni ethernet anahtarları, bu aygıtların takibi, yönetimi, ağ trafiği incelemesi ve değerlendirmesi gibi günlük sorunların çözümünde ağ yöneticilerine yönelik olarak hazırlanmış pek çok yazılım mevcuttur, ancak bunların hepsi belirli bir görevi yerine getirecek şekilde geliştirilmiştir. Özel talepler ve görev grupları için özel geliştirilmiş yazılımlar ancak özel olarak geliştirilmektedir.

Fiziksel ağ topolojisinin bilinmesi ağ yönetimi çalışmaları içinde önemli bir yer tutmaktadır. Sistemin yük dağılımının incelenmesi, genel eğilimlerin ortaya çıkarılması, ihtiyaçların belirlenmesi, istenen türdeki trafiğin takibi ve yönetimi, kullanıcıların ve haklarının yönetimi, sistemde ortaya çıkan sorunların takibi ve giderilmesi yine ağ yöneticisinin sorumluğundadır. Bu görevlerin etkin ve yeteri kadar yerine getirilebilmesi için ağın topolojisinin kontrol altında tutulması ağ yönetiminde oldukça önemli yer tutar.

Ağ topolojisi belirleme çalışmaları Internet'in dünyaya yayılması ile hız kazanmıştır. Ağ Katmanı düzeyinde yapılan çalışmalar daha çok ana omurgaların bulunduğu Geniş Alan Ağlarını ve ağ katmanı seviyesinde topoloji gösterimini hedeflemiş, [1], [2] ve [3] çoğu durumda yerel alan ağlarını ve onların bileşenlerini gözardı etmiştir. Ticari ürünlere de yansıyan bu durum daha sonra pek çok üretici ve geliştiricinin ortaya çıkardığı ürünlerle kapatılmaya çalışılmıştır [4], [5].

Fiziksel topoloji belirleme konusundaki ilk önemli akademik çalışma Breitbart tarafından Bell Laboratuarları'nda yapılmıştır [6]. Lucent grubu, standart SNMP MIB'lerini kullanarak yaptığı çalışmada ethernet anahtar aktarım tablolarında eksiklik olmadığı kabul edilerek bir sistem tasarlamıştır. Ancak bu şartı yerine getirebilmek için ağ üzerinde sürekli olarak ağ elemanlarına yönelik paket trafiği oluşturma zorunluluğu vardır. Daha sonraki önemli çalışma ise Lowekamp tarafından yapılmıştır [7]. Bu çalışmada fazladan paket trafiği oluşturmadan ve anahtarların aktarım tablolarının içerdiği bilgiler yeterli kabul edilmiştir. Topoloji belirleme işlemi kullanıcı tarafından belirtilen IP adresleriyle sınırlıdır ve ethernet anahtarları hariç diğer akıllı cihazların yönetimini içermememektedir. Bu uygulama İYTE yerel alan ağında denenmiş ancak tatmin edici bir sonuç elde edilememiştir.

Konuyla ilgili en güncel çalışma ise Bejerano'nun çalışmasıdır [8]. Farklı bir algoritma ortaya koyan grup, çalışmasında uygulamanın sonuçlarından bahsetmemiştir. "Spanning Tree" protokolünü kullanan ağlardaki fiziksel topolojiyi belirlemek için yapılmış çalışmalar da bulunmaktadır [9] .

2.2. YAPILAN ÇALIŞMA

GuardiLAN projesi çerçevesinde yapılan çalışma önceden belirlenmiş genel ve özel tüm Internet adres blokları üzerindeki sistemlerin belirlenmesine ve önceden belirtilmiş tüm ağ yönetim topluluklarının yönetimine imkan sağlamaktadır. Bunun yanında toplanan verilerin GuardiLAN veritabanına aktarımı ve böylece diğer ağ yönetim araçlarıyla yardımlaşması ve ağ yönetimi görevlerini kolaylaştırması hedeflenmiştir.

Aşağıda verilen algoritma ile önceden belirlenmiş Internet adres bloklarında çalışmakta bulunan sistemlerin belirlenmesi ve sınıflandırılması mümkün olmaktadır. Bu yöntemle hem ağa yeni bağlanan bilgisayarlar belirlenmekte hem de topoloji belirleme esnasında kullanılacak olan ethernet anahtarları adres aktarım tablolarındaki eksiklikleri gidermektedir.

2.2.1 SNMP, SMI, MIB HAKKINDA TEMEL BİLGİLER

Ağ yönetimi görevlerinin daha kolay ve uzaktan yerine getirilebilmesi için tasarlanan SNMP protokolu topoloji belirlemede temel bileşeni oluşturmaktadır [10]. SNMP'nin 3 versiyonu bulunmaktadır: v1, v2c, v3. Kimlik doğrulama işlemleri SNMP v2c'de topluluk isimleri vasıtasıyla yapılmaktadır: örn. "public". SNMP ile akıllı cihazlarla ilgili genel ve özel pekçok bilgiye erişim mümkün olmaktadır. Bütün bu yönetimsel bilgiler ve bunların saklanma ve erişim biçimi SMI (Structure of Management Information) ile tanımlanmıştır. SMI ile tanımlanan yönetimsel nesneler, yine SMI'da tanımlanan hiyeraşik yapıya uyarlar. SMI ile ağacın yalnızca gövdesi ve bileşenlerin yapıları tanımlanır. Ağacın yaprakları ise MIB (Management Information Base) ile belirlenmiştir [11] . MIB'lerle tanımlanan yönetimsel bilgiler Sistem Adı gibi tek bir satır ile ifade edilebildiği gibi bir tabloyu da ifade edebilmektedir. Örneğin Ethernet Anahtarlarının hangi MAC adresine hangi porttan ulaşabileceğini bulduğu adres aktarım tabloları (Address Forward Table - AFT) bir tablo olarak BRIDGE-MIB'de tanımlanmıştır [12].

Genel amaçlı MIB'ler RFC belgeleriyle tanımlanır ve güncellenir. Ancak cihazın kendisine özel yönetimsel bilgilerin tutulduğu özel MIB'ler cihazı üreten kuruluş tarafından düzenlenir ve güncellenir. MIB-II, RFC-1213 ile tanımlanmıştır ve daha sonra başka RFC'ler ile değişiklikler yapılmıştır. Herhangi bir akıllı cihazla ilgili genel bilgilere erişmek için akıllı cihazın SNMP Agent'ında (akıllı cihazın SNMP sorgularına cevap vermesini sağlayan yazılım) implemente edilmiş olması yeterlidir. Cihazların cinsine özel yönetimsel bilgiler, örneğin Ethernet Anahtarının Port-Arabirim tabloları veya ağ yazıcısının iş kuyruğu bilgileri veya bir sunucunun üzerinde çalışan süreçler gibi cihaz tipine özel MIB'lerde tanımlıdır (BRIDGE-MIB (RFC 1493), HOST-RESOURCES [13]) ve yüklüdür. Örneğin bir ağ yazıcısının SNMP Agent'ı ethernet anahtarında bulunması gereken Port-Arabirim tablosu ile ilgili sorguya cevap vermez.

2.2.2 AKILLI CİHAZLARIN TİPLERİNİN BELİRLENMESİ

Algoritmada kullanılan sistem tanımı bilgisi "`.iso.org.dod.internet.mgmt.mib-2.system.sysDescr.0`" nesnesi olarak bütün akıllı cihazlarda ortaktır, cihazda SNMP bulunduğunu işaret eder. Sistemden gelen yanıt ile sistemin yazılım ve/veya donanım üreticisini öğrenmek de mümkün olmaktadır.
BRIDGE-MIB ethernet anahtarlarında ve yönlendiricilerde bulunur. Bunları birbirinden ayıran özellik köprü yöntemidir: günümüzde ethernet anahtarları "Transparent" modda çalışır. Yönlendiriciler ise "Source Route Transparent" modda çalışır. Bu ikisini birbirinden ayırmak için BRIDGE-MIB' deki (RFC-1493'te `.iso.org.dod.internet.mgmt.mib-2.dot1dBridge` olarak tanımlanmıştır) "`.dot1dBase.dot1dBaseNumPorts.0`" (port sayısı) nesnesi sorgulanır. Cevap 0'dan büyükse ethernet anahtardır, değilse yönlendiricidir.
C = { belirli SNMP yönetimsel topluluk isimleri : örn. "public" } cl_i C'nin herhangi bir elemanını temsil etmektedir.
ip = { belirli bir IP adres bloğu : örn. "193.140.248.0/22" veya "192.168.0.0/24"} ip_{j-k} j'inci IP bloğunun k'ıncı elemanını temsil etmektedir.
IPB = { belirli IP adres blokları }
M = {IPB kümesinde tanımlı IP adreslerini kullanan ve C'nin elemanlarından herhangi biri ile SNMP sorgularına cevap veren akıllı cihazları ifade eden IP adresi – topluluk ismi ikilisi }
B = { M'nin sadece ethernet anahtarı olan elemanları}
H = { SNMP ile yönetim imkanı bulunmayan cihazlar}

ifade etmek üzere

Prosedür : Cihazların cinslerinin belirlenmesi :
Başlangıç :

/* Akıllı cihazların belirlenmesi */
C'nin elemanı her bir c_i için
Başla:
IPB'nin elemanı her bir ip_j için
Başla:
ip_j 'nin her bir elemanı ip_{j-k} için
Başla:
(cl_i) ile (ip_{j-k}) adresinde sistem bilgisini sorgula
Eğer cevap gelirse M'ye (ip_{j-k} , cl_i) ikilisini ekle
Diğer durumda ip_{j-k} adresini H'ye ekle
Bitir.
Bitir.
Bitir.

/* Ethernet Anahtarlarının belirlenmesi */

M'nin elemanı olan her bir (ip, topluluk) ikilisi için
Başla:
"topluluk" ismi ile "ip" adresinde port sayısını sorgula
Eğer cevap gelirse ve
Eğer 0'dan büyükse "ip" bir ethernet anahtarına aittir
Diğer durumda "ip" bir yönlendiriciye aittir.
Diğer durumda "ip" adresini kullanan sistem normal kullanıcı, ağ yazıcısı, UPS takip sistemi olabilir.
Bitir.
Bitir.

2.2.3 AÇIK BİLGİSAYAR SİSTEMLERİNİN BELİRLENMESİ

Ağ yönetimi ile ilgili çalışmaların etkin ve etkili olarak yapılabilmesi için akıllı cihazlar kadar diğer sistemler hakkında bilgi sahibi olmak da önem taşır. Ağa bağlı kullanıcıların çoğunda SNMP bulunmaz, bu yüzden zaten bağlantısız bir protokol olan UDP'yi kullanan SNMP sorgularına cevap vermezler. Dolayısıyla SNMP sorguları açık sistemleri belirlemede etkisiz kalmaktadır. Ancak normal kullanıcılar yerel ağ trafiğin büyük bölümünü oluşturmaktadır ve takibinin faydaları açıktır. Hangi sistemlerin o anda açık olduğunu ve/veya açık olması gerektiğini bulabilmek veya tahmin edebilmek için birkaç yöntem vardır :

1. ICMP Ping. Hedef IP adresine ICMP-Echo paketi yollar. UNIX türevi ve Windows sistemlerde bu işlem en basit olarak "ping" komutuyla yapılmaktadır, dolayısıyla yöntem de bu şekilde ismlendirilmiştir. ICMP Paketleri Raw Soket kullandığı için platforma özel uygulama gerektirir. Güvenlik duvarı bulunan sistemler bu paketleri değerlendirmeye almazlar, bu yüzden bu şekilde elde edilen veriler çok da güvenilir olmamaktadır.

2. TCP Ping. Hedef IP adresinin önceden belirlenmiş bir portuna (örn. 80/tcp) bağlantı isteği yollanır. Bağlantı kabul edilirse sistemin açık olduğu ortaya çıkar, bağlantının karşı taraftan reddedilmesi de açık olduğunu ortaya çıkarır. Güvenlik duvarı bulunan sistemlerde kısmen daha iyi sonuç verir, ancak ICMP Ping'e göre daha yavaş sonuç alınır.

3. Pasif Dinleme. Yukarıdaki yöntemler gerek sisteme, gerekse ağa yük getirmektedir. Aynı zamanda güvenlik duvarı bulunan sistemleri belirlemede başarı oranı düşüktür. Pasif dinleme yaklaşımı sistemlerin farklı hizmetler için yolladıkları paketlerin dinlenmesi ve değerlendirilmesi esasına dayanır. Örneğin güvenlik duvarı kullanan bir kullanıcı bilgisayarı gelen ICMP Echo ve TCP Bağlantı isteği paketlerini reddedecektir, ancak aynı kullanıcının bir ftp sunucusuna bağlanırken oluşturduğu trafik yakalanabilirse sistemin açık olduğunu ispat edecektir. Bu paket trafiğinin iyi seçilmiş bir örnekleme aralığı içinde (örn. 5 dk.) dinlenmesi ise zamansal lokalite prensibine dayanarak açık sistemler hakkında fazladan bilgi sağlayacaktır.

Açık sistemlerin belirlenmesi için ping ve pasif dinleme teknikleri kendi başlarına kullanılabilir. Ancak bu iki yöntemin kombinasyonunun kullanılmasının sağlayacağı fayda açıktır.

A = { Pasif dinleme sonucu açık olduğu bilinen veya tahmin edilen sistemlerin IP adresleri }
A^* = {Açık olduğu kabul edilen sistemlerin IP Adresleri}
ip = { belirli bir IP adres bloğu : örn. "193.140.248.0/22" veya "192.168.0.0/24" gibi } ip_{j-k} j'inci IP bloğunun k'ıncı elemanını, ve
IPB = { belirli IP adres blokları }
temsil etmek üzere

/* Açık ve ağa bağlı bilgisayarların belirlenmesi*/
Prosedür : Açık Sistemlerin Belirlenmesi :
Başla:
IPB'nin her elemanı ip_j için
Başla :
ip_j'nin her elemanı ipj-k için
Başla :
Eğer $ip_{j-k} \in A$ devam et
Diğer durumda ping(ip_{j-k})
Eğer cevap gelirse $A^* = A^* \cup ip_{j-k}$
Bitir.
Bitir.
Bitir.

2.2.4 TOPOLOJİ BELİRLEME ALGORİTMASI VE İŞLEYİŞİ

Bu aşamadan önce yapılan işlemler ile elde edilen bilgilere bakıldığında sisteme bağlı cihazların o anki durumları, akıllı olup olmadıkları ve tahmini olarak cinsleri belirlenmektedir. Yukarıdaki bilgiler ağ yönetim sistemine ve topoloji belirleme algoritmasına ağa bağlı bulunan ethernet anahtarlarının listesi, kullanıcıların ve diğer akıllı cihazların listesi gibi bilgileri sağlamak içindir. Sonraki aşamada ethernet anahtarları ile ilgili olarak özel bilgiler toplanmaktadır. Ethernet anahtarlarından aşağıdaki bilgiler alınmaktadır :

- Sistem Bilgileri (örn. Sistemin isim, yer, tanım, çalışabilirlik süresi bilgileri ...)
- Arabirim Bilgileri (örn. Arabirim sayısı, özellikleri ve MAC adresi ...)
- Temel anahtarlama bilgileri (örn. sistemin kendi MAC Adresi, Port sayısı ...)
- Anahtar Bilgileri (örn. Port-Arabirim Tablosu, MAC Adresi – Port Tablosu, ...)

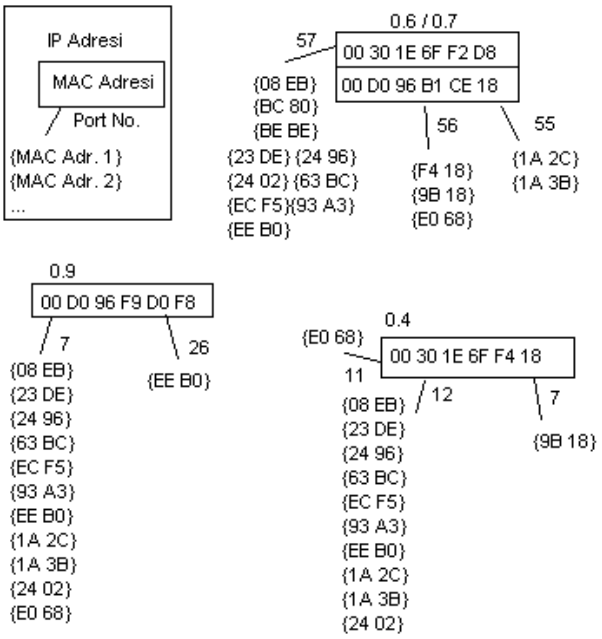
Aşağıda örnek bir MAC Adresi port tablosunun dökümü verilmiştir:

Index No:	MAC Adresi	Port	Durum
0.32.175.242.1.30	00 20 AF F2 01 1E	2	learned (3)
0.96.151.119.25.49	00 60 97 77 19 31	12	learned (3)
0.96.151.159.237	00 60 97 9F ED A7	26	learned (3)

Şekil 1. Örnek aktarım tablosu

Ethernet Anahtarları MAC çerçevelerinin aktarımını yaparken hangi adrese hangi porttan yapılacağı bilgisini bu tabloda saklamaktadır. Şekil 1 aktarım tablolarına bir örnektir. Normal çalışma saatlerinde ana omurgayı oluşturan anahtarların tuttukları tablolar yüzlerce satır içermektedir ve sürekli olarak yenilenmektedir. Şekil 2 İYTE'de bulunan 3 Ethernet Anahtarı ile ilgili bilgileri göstermektedir.

Lowekamp, topoloji belirlemek için varolan adres aktarım listelerinin yeterli olduğunu ve fazladan trafik oluşturmaya gerek olmadığını savunmuştur. Ancak yine de anahtarları belirlemek ve gerekli bilgileri yükleyebilmek için herbir anahtarın teker teker sırayla sorgulanmasında büyük yarar vardır. Bunun nedeni ileride açıklanacaktır.



Şekil 2. Anahtarların aktarım tablolarına ilişkin örnek.

2.2.4.1 BASİT BAĞLANTI TEOREMİ VE MİNİMUM BİLGİ GEREKSİNİMİ ŞARTLARI

Lowekamp adres aktarma tabloları kullanılarak topolojiyi belirlemede Breitbart'ın kullandığı "Doğrudan Bağlantı" teoremi yerine "Basit Bağlantı" teoremini önermiştir. Basit Bağlantı Teoremi, iki cihazın birbirlerine doğrudan yada dolaylı olarak birbirlerine bağlı olmaları durumlarını ifade eder, varolan ağdaki yerleşim hakkındaki bilgilerde topoloji hakkında çelişki oluşturacak ve olası olmayan durumları dışlayarak Ethernet anahtarlarının birbirlerini 'gör'dükleri portların belirlenebilmesi için üç şartın gerekli ve yeterli olduğunu belirtir.

Anahtarlar ve MAC Adresleri A ve B olsun. A'nın i'nci portundan gördüğü MAC Adresleri Ai ile belirtilsin. Bi ise B'nin j'nci portundan gördüğü MAC Adresleri olsun.

1. A, B'nin MAC Adresini A'nın i.'nci portundan, B de A'yı j'nci portundan görmektedir.
2. A, B'nin MAC Adresini A'nın i'nci portunda görmektedir, B nin j'inci portunda gördüğü iki MAC adresi C ve D olsun. A, herbiri i'den farklı k ve l'ıncı portlardan aktarılmaktadır.
3. A'nın i'inci portunda gördüğü bir MAC Adresi E olsun. Bu B'nin j'inci olmayan bir portunda gözükmektedir. B'nin j'inci portunda gördüğü iki MAC Adresi, C ve D, A'da herbiri i'den farklı k ve l'nci portlardan aktarılmaktadır.

Akıllı Ethernet anahtarları Spanning Tree Protocol uygulanmadığı durumlarda sadece kendilerine paket geldiği zaman cevap verirler. Bu yüzden birbirlerinin yerleri hakkında sahip oldukları bilgiler ancak başka bir anahtara yönelik paketleri aktarmaları ile olur. Anahtarların birbirlerini de görebilmeleri için en başta bütün anahtarlara paket yollamak daha sonra da adres aktarımı bilgilerini yüklemek büyük kolaylık sağlamaktadır. Bu işlemin yapılmaması birinci şartın ortaya çıkmasını zorlaştırmaktadır. Şekil 2'de adres aktarım tabloları verilen üç anahtardan IP Adresi 0.4 olan anahtar 0.6 ve 0.7 IP adreslerini kullanan anahtara 12. portundan bağlıdır. 0.6/0.7 adreslerini kullanan anahtarın 56.portunda 0.4 adresli anahtarın MAC Adresi kayıtlıdır. Bu durumda 2. şarta dayanarak 0.4'ün 12. portu 0.6'nın 56. portuna bağlıdır denebilir. Bu bağlantıyı 0.4:12 → 0.6:56 olarak göstereceğiz. 0.6 ve 0.9 adreslerini kullanan anahtarlarda ise ilk iki şart bağlantıyı belirlemede yetersiz kalmaktadır. Bu durumda üçüncü şart ile bağlantı belirlenebilmektedir. 0.9 adresli anahtarın 7. portundan aktardığı MAC adreslerinden en az ikisi 0.6'nın en az iki ayrı portu üzerinden aktarılmaktadır. Bu durumda 0.9:7 kabul edilir. 0.9'un 7. portu dışındaki portlardan aktardığı MAC Adresi ise 0.6'nın 57. portu üzerinden aktarılmaktadır. 3. şarta dayanarak 0.9:7 → 0.6:57 bulunur.

2.2.4.2 ALGORİTMA

Aşağıdaki algoritmada bütün anahtarların kök kabul edilen anahtara bağlı oldukları port, kök port kabul edilmektedir. Buna göre topoloji ağacının kökü dışıdaki bütün anahtarların belirli bir kök portu olacaktır.
Algoritma, ağı oluşturan Hubları ve SNMP yüklü olmayan veya SNMP özelliği kapatılmış anahtarları da kapsamaktadır.

SW={Topoloji Belirlemede Kullanılacak Ethernet Anahtarlar }
Kök={ Topoloji ağacının kökü kabul edilecek anahtar }

Prosedür TopolojiBelirle(Kök)
Başla:
    BaşlangıçKökünüBelirle();
    Kök Anahtarın her bir portu i için
    AltTopolojiBelirle(Kök, i);
Bitir.

Prosedür AltTopolojiBelirle(Kök Anahtar, i)
Başla.
  Kök = **KökAnahtarıBelirle**(Adaylar, Liste);
  Kök Anahtarın her bir portu i için
    AltTopolojiBelirle(Kök, i);
Bitir.

Prosedür KökAnahtarıBelirle(Kök, KökPort, Adaylar, Liste)
Başla:
  Her bir aday anahtarın (A_i) kök portunu belirle;
  Kökbulundu = Yanlış;
  Eğer OrtakSegmentiBul(Kök, KökPort, Adaylar) ≠ {}
  Başla:
    H = **HubOluştur**();
    YeniKök = H;
    Geri dön;
  Bitir.
 Her bir aday anahtar A_i için ve Kökbulundu = Yanlış iken
  Başla :
    KökOlabilir = Doğru ;
    Listedeki A_j ($i \neq j$) için ve KökOlabilir = Doğru iken
    Başla:
      Eğer A_i, A_j ile kök portundan bağlıysa
        KökOlabilir = Yanlış;
      Diğer durumda
        Devam et;
    Bitir.
    Eğer KökOlabilir = Doğru ise
      Başla:
        KökBulundu = Doğru;
        YeniKök = A_i ;
        Geri dön;
      Bitir.
  Bitir.

/* Hiçbir anahtar kök olamaz veya Aday = {} ise – Hub veya SNMP bulunmayan Ethernet Anahtarı */
OrtakSegmentBul(Kök, KökPort, Adaylar);
H = **HubOluştur**();
YeniKök = H;
Geri dön;
Bitir.

2.2.4.3 PAYLAŞILAN SEGMENTLERİN TESPİTİ

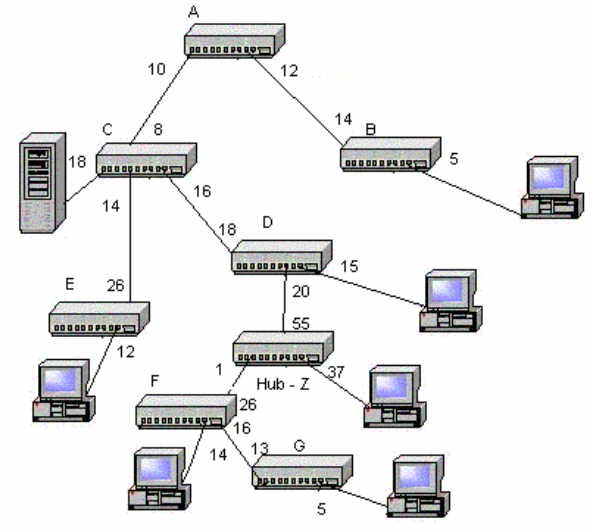
Eğer ağ içinde iki anahtar birbirine bir Hub veya SNMP özelliği bulunmayan veya kullanılamayan bir anahtar ile bağlanmış ise bu iki anahtar aralarında paylaştıkları bir segmentin varlığından haberdar olmayacaktır. İki anahtar da de bu segmente bağlı MAC adreslerini sıradan adresler olarak göreceklerdir. Ancak ortak segmentin varlığı aşağıdaki yöntem ile ortaya çıkarılabilmektedir. Böylece iki anahtarın arasında bir hub olduğuna (SNMP özelliği kullanılmadığı takdirde anahtarlar da hub olarak kabul edilebilir) karar verilir, topolojideki hata giderilebilir.

A_i : A'nın i'nci portunda kayıtlı MAC Adresleri Listesi

Prosedür OrtakSegmentiBul(Kök, kökünPortu–k , Adaylar)
OrtakSegment = $Kök_k$
Başla:
Adaylar içindeki her bir eleman A_i için
Ortak Segment = Ortak Segment $\cap$ A_i
Bitir.

2.2.4.4 ALGORİTMANIN İŞLEYİŞİNE ÖRNEK

Aşağıdaki şekilde küçük bir bilgisayar ağı görülmektedir. Ağa bağlı 5 akıllı anahtar, bir adet Hub, bir sunucu ve birkaç tane de kullanıcı gözükmektedir. Topoloji Algoritması aşağıdaki örnek ağda aşağıdaki gibi çalışmaktadır :



Şekil 3. Algoritmanın işleyişine dair örnek ağ

Basit Bağlantılarda A:10 → D:18 ifadesi A'nın 10.portunun D:18 portu arasında basit bağlantı bulunduğunu göstermektedir. Anahtar hiyeraşisinin belirlenmesi şu şekilde olmaktadır:

Kök : A
SW = {A, B, C, D, E, F, G}
Topoloji Belirle()
Kök = A;
A:10 için AltTopolojiBelirle(A);

AltTopolojiBelirle(Kök)
Adaylar = {A_{10} portu ucundaki anahtarlar }
Adaylar = {D, E, C, F, G }
OrtakSegment = {}
Kök Adayı : D
D:18 → E:26
/* D E ile bağlantısını kök port üzerinden yapmıştır. Bu durumda D,E,C arasında D kök olamaz.*/
E:26 → D:18
/* E, D ile bağlantısını kök portu üzerinden kurmuştur. Bu durumda E de kök olamaz. */
C:16 → D:18
C:14 → E:26
C:16 → F:1
C:16 → G:13
/* C iki bağlantısını da kök portu üzerinden yapmamıştır. Adaylar arasındaki kök C'dir.*/

Anahtar bulunmadığı durumlar:

D:20 için **AltTopolojiBelirle**(D)
Adaylar = {F, G}
OrtakSegmentBul(D, 20, {F, G})
Başla :
/* Anahtarların Adres Aktarım Tabloları :
D:20 = {F, User-1, User-2, User-3, G}
F:26 = { User-1 } F:14 = { User-2 }
G:13 = { User-1, User-2 } G:5 = { User-3 } */

OrtakSegment = D:20 = {F, User-1-2-3, G}
{F, G}'nin her bir elemanı A:kök
Başla :
OrtakSegment = OrtakSegment $\cap$ A_i
Bitir.
/* Eğer ortak segment yoksa **{ }** dönecektir. */
Bitir.
.....
OrtakSegment = { User-1}
H = **HubOluştur**(OrtakSegment , Hub-Z);
Kök = H;
Geri dön;
Bitir.
....
AltTopolojiBelirle(Hub-Z);
.....

2.3 SONUÇ

Fiziksel ağ topolojisinin bilinmesi ağ yönetimi açısından oldukça büyük önem taşımaktadır. GuardiLAN topoloji belirleme uygulamasında büyük oranda ilerleme kaydedilmiş, ancak gerek ağın iç sorunları gerekse uygulamadan kaynaklanan sorunların da eklenmesiyle henüz tamamlanamamıştır.

3. IP-MAC ADRESLERİNİN KONTROLÜ

3.1 GİRİŞ

Temel ağ protokolleri, yerel alan ağları için yeterli yönetim ve güvenlik kontrolü gereksinimlerini tam anlamıyla karşılayacak şekilde tasarlanmamıştır, ancak bu protokolleri ve bize sunduğu özelliklerini kullanarak, çeşitli tasarım araçları ile ağın kontrolü sağlanabilecek, dışarıdan girişler bakımından ağın güvenliği denetleyebilecek, IP ve MAC adresleri tablosu ile ağın mevcut yapısında karşılaşılabilecek uyumsuzluklar durumunda kullanıcılar hata mesajlarıyla uyarılacak ve programın internet arayüzü ile çeşitli kullanım bilgilerine ulaşılabilecek bir sistem oluşturmak mümkündür [14].

IP - MAC kontrolü projesi, İzmir Yüksek Teknoloji Enstitüsü kampüs alanı yerel ağı içerisinde yönetimsel bir araç olarak kullanılmak üzere güvenlik ve görüntüleme gereksinimleri karşısında ortaya çıkmıştır. Temeldeki amaç, bu proje ile birçok ayrı projenin birlikte güçlü bir ağ yönetim sistemi oluşturmasıdır.

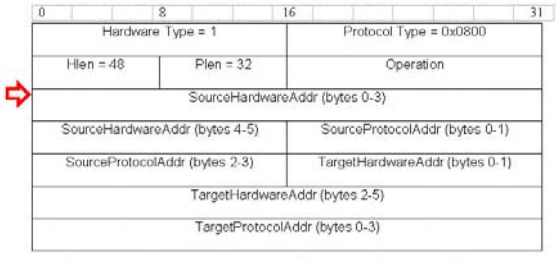
3.2 AĞ GÜVENLİĞİ - "Spoofing Attack"

Yerel alan ağları için önemli güvenlik tehditlerinden biri de “Spoofing Attack” saldırı türüdür. Burada saldırı, ağın içinde güvenilen bir kaynaktan geliyormuş gibi algılandığı için, bir saldırı olarak belirlemek zordur.

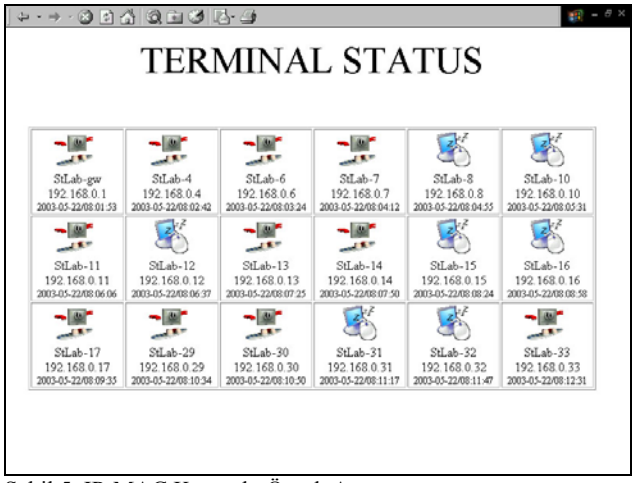
Güvenilen bir IP adresi ile paket alışverişi yapılırken, aynı zamanda o IP 'nin sahibinin gerçekten güvenilen terminal olup olmadığı kontrul edilmeli, "Authentication" işlemi yapılmalıdır. Kullanıcı, güvenilen bir terminalin IP adresini ele geçirerek bir saldırı düzenleyebilir ve sadece o terminalin ulaşmasına izin verilen servislere ya da kaynaklara ulaşabilir [15].

3.3 ADRES ÇÖZÜMLEME PROTOKOLÜ (ARP)

Adres Çözümleme Protokolü (ARP), her tür yayın ağında kullanılabilen OSI birinci katman adresleri ikinci katman adreslere çözümleyen, ikinci katmana ait genel bir protokoldür. Ethernet ortamında kullanılan ARP paket formatı Şekil 4' de gösterilmiştir. Paketin başlık bilgilerinden sonra gelen 64. bitten itibaren 6 bayt, yani 48 bit veri kaynak terminalin MAC adresi, sonraki 4 bayt IP adresi bilgilerini taşır. Bundan sonra gelen 6 baytta hedef terminalin MAC adresi ve ardından gelen 4 baytta de hedef terminalin IP adresi bilgileri yer alır [16].



Şekil 4 - ARP Paket Formatı



Şekil 5. IP-MAC Kontrolü Örnek Arayüzü

Terminaller, yerel alan ağına dahil olduklarında ağ içindeki diğer terminallerle veri alışverişinde bulunabilmek için ARP paketleri yayınlarlar. Bu paketler, ağa dahil olan tüm terminallere ulaşır ancak hedef IP adrese sahip olan terminal bu pakete cevap verir.

3.4 IP - MAC KONTROL PROGRAMI

IP ve MAC adres kontrol programı ilk olarak ethernet kartını dinleme moduna alır, bu şekilde ağ içinden programın çalıştığı terminale ulaşan tüm ARP paketlerinin birer kopyaları alınır ve kullanım amacına göre kontrol edilerek uygulamaya göre gerekli paketler süzülür. Bu süzme işleminden sonra program, dahil olduğu ağ içindeki terminalleri tespit etmek için iki temel yol kullanır, bunlar aktif ve pasif dinlemedir.

Pasif dinlemede temel mantık, ağa dahil olan tüm makinaların ağ geçidi olarak belirlenmiş makinaya bir ARP pakedi yollayacağı göz önünde bulundurularak, ağ içinde yayınlanan ve hedef terminal adresi ağ geçidi olan ARP paketlerinin dinlenmesi yoluyla herhangi bir makinanın açık olup olmadığı ve aktif olarak en son ne zaman kullanıldığı gibi bilgilerin elde edilmesidir.

Aktif dinlemede ise, veritabanında tutulan kullanıcı bilgilerine göre belirlenmiş olan sürenin üzerinde trafik oluşturmamış olan terminaller ile ICMP mesajları yoluyla bağlantı kurulmaya çalışılır. Eğer terminal açık ancak belirli bir süre boyunca ağ içinde trafik oluşturmamış ise de bu yolla ağa dahil olup olmadığı belirlenir.

Sonuç olarak, elde edilen ARP paketleri ikinci bir süzme işleminden daha geçirilir ve programın kontrol mekanizmalarına göre sınıflandırldıktan sonra, anlamlı bayt gruplarına parçalanır. Örneğin paketin 9. baytı ile15. baytı arasındaki bölüm kaynak terminalin MAC adres bilgisini, 19. baytı arasındaki bölüm IP adres bilgisini taşımaktadır. Bu şekilde kaynak ve hedef terminallerin IP ve MAC adresleri elde edilir.

ARP paketlerinin parçalanması ile kontrol işlemi için gereli bilgi toplanmış olur. Daha sonra program, güvenilir kullanıcı IP ve MAC adresleri eşlemesini içeren veritabanından, değişik sorgularla güvenilir IP ve MAC adresleri eşleştirmesini alır ve ARP paketlerinden elde edilen bilgilerle karşılaştırır. Karşılaştırma sonucunda herhangi bir uyumsuzluk tesbit edilirse, program veritabanındaki gerekli bölgeleri değiştirir ve

Java Sunucu sayfaları şeklinde tasarlanmış arayüzde uyarı mesajı oluşturulur. Bu işlem sonucunda Akıllı Bant Genişliği Yönetimi programı, tespit edilen IP adresine ait bant genişliğini kısıtlayacaktır. Herhangi bir uyumsuzluk saptanmadığı durumda ise program yakalanacak bir sonraki ARP paketine kadar bekleyecek ve bir sonraki ARP paketi için tekrar aynı işlemleri yapacaktır.

Programın IP ve MAC adreslerinin kontrolü yanında, yakaladığı paketlerden topladığı çeşitli verileri veritabanına işlemesi ile ağ içinde sorgu yapıldığı anda tespit edilen açık terminallerin sayısı, bu terminallerin en son ne zaman trafik oluşturdukları ve terminallere ait bazı kullanıcı bilgileri yine programın ağ erişimli arayüzü ile edinilebilen bilgilerdir.

3.5 IP-MAC KONTROLÜ İŞLEVSEL VE MİMARİ YAPI

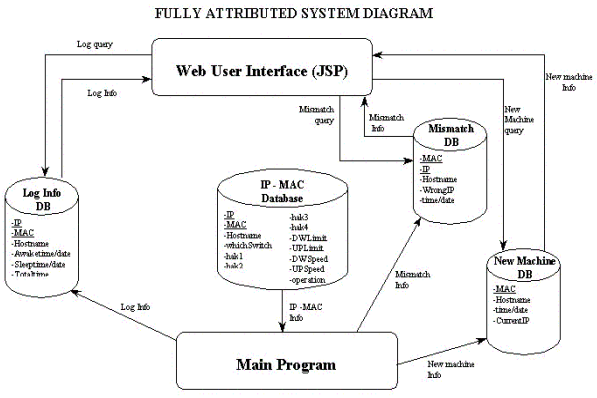
Proje işlevsel olarak değerlendirildiğinde iki temel görevi yerine getirmektedir. Bunlar ağ güvenlik fonksiyonu ve ağ görüntüleme fonksiyonları olarak isimlendirilebilir.

Güvenlik fonksiyonu kapsamında, "spoofing attack" olarak isimlendirilen, ağ içinde güvenilebilir olarak belirlenmiş kullanıcılar yerine geçme olarak açıklayabileceğimiz saldırılara karşın ağ içinde trafik oluşturan IP adreslerinin sürekli aynı MAC adresleri tarafından kullanıldığını kontrol eden bir program yer almaktadır. Şekil 6' da programın yapısını ve çalışma prensibini anlatan sistem diagramı görülmektedir.

Ağ görüntüleme fonksiyonunda ise güvenlik açısından yapılan kontroller sonucunda elde edilen verilerin işlenip terminallere ait kullanım istatistik bilgilerinin elde edilmesi ve bunların internet erişimli bir arayüzde birleştirilmesi sağlanmıştır.

İşlevsel olarak iki alt bölümde incelenebilen bu proje, mimari olarak incelendiğinde 4 ayrı bölümden oluşmaktadır. Bunlar sırasıyla şöyledir;

- Paket Yakalama
- Paket Süzme ve Verilerin Kontrolü
- Verilerin birleştirilip Veritabanına Girişi
- Verilerin Yorumlanması ve İnternet Erişimli Arayüz ile Raporlanması



Şekil 6. IP - MAC Kontrolü Sistem Diagramı

3.6. SONUÇ

IP – MAC kontrol programı, kampüs alan ağının değişik yoğunluklar gösterdiği gün ve saatlarde çalıştırılarak, ortalama ARP paket yayın sayıları ve ağ kullanım oranları belirlenmiştir. Bunun sonucunda, ARP histogramları oluşturulmuş ve programın bir terminal için pasif dinlemeden aktif dinlemeye geçiş süresinin 20 dakika olmasına karar verilmiştir. 20 dakika boyunca trafik oluşturmayan terminaller, aktif dinlemeye de cevap vermemeleri durumunda kapalı olarak belirlenir ve veritabanında o terminale ait alanlar düzenlenmesi sağlanarak ağ erişimli arayüzde, o terminal için açık/kapalı bilgisi, en son tarfik oluşturdukları tarih ve saat ile ağ içinde toplam açıklık/kapalılık bilgileri gösterilir.

Programın Akıllı Bant Genişliği Yönetimi ile birlikte çalışmasıyla ağ içindeki belirli saldırı ya da yeni eklenen güvenilmeyen terminallerin bant genişliğinin otomatik olarak sıfırlanması ve ağ yöneticisinin uyarılması sağlanmış, program ağ içindeki güvenlik düzeyini yükseltmiştir. Ayrıca programın ağ ile ilgili düzenli kayıt tutmasıyla, belli başlı ağ kullanım verileri zaman ekseni üzerinde sınıflandırılmış, ağ yöneticisi için kullanışlı olabilecek ağ kullanım veri blokları oluşturulmuştur.

4. AKILLI BANT GENİŞLİĞİ YÖNETİMİ

4.1. GİRİŞ

Internet'in kullanımı genişlerken, uzaktan öğrenim, video konferans, e-ticaret gibi yeni uygulamaların da geliştirilmesiyle daha yüksek bant genişliklerine ihtiyaç duyulmaktadır. Her yazılımın, bant genişliği, gecikme, gecikme sürelerindeki değişim ve bulunabilirlik açısından kendine özgü gereksinimleri olmaktadır.

Internet Protokol (IP) tabanlı ağlar en uygun eforda servis sağlamaktadır. IP, paketlerin hedefe zamanında veya belli bir gecikme süresi içerisinde ulaşma garantisi vermez. Bazı paketler sıkışma nedeniyle kaybolabilir. Bütün paketler farklı ihtiyaçlar duyabilmekle beraber, eşit olarak ele alınmaktadır. Çoğu uyguluma için bu kabul edilebilir değildir. Servis Kalitesi kullanıcıların temel olarak servis kullanımlarının garantilenmesi ve daha iyi hizmet verilmesidir. Bu da ancak, veri akışının sınıflandırılarak belli kurallara göre öncelik verilmesiyle olabilmektedir. Geçmişte, bu sorunun en kolay çözümü mevcut bant genişliği kapasitesinin artırılmasıydı. Fakat bant genişliğini sürekli artırmak mümkün değildir. Bugüne kadar bant genişliklerinin kullanıcı ve trafik bazında ayarlanmasını sağlayan bir çok mekanizma geliştirilmiştir. Ancak kullanıcıların kullandıkları bant genişliğini el ile ayarlamak çok zor ve çok zaman gerektiren bir işlemdir. Bu nedenlerden dolayı ağ üzerinde Servis Kalitesini sağlamak için **akıllı** bir yönetime gereksinim duyulmaktadır.

Bu bölümde İzmir Yüksek Teknoloji Enstitüsü yerel bilgisayar ağı için geliştirilen Akıllı Bant Genişliği Yönetimi uygulaması ayrıntılı olarak açıklanacaktır. Bu projenin geliştirilme amacı yerel bilgisayar ağı üzerindeki trafiğin en iyi şekilde kullanımını sağlamaktır.

4.2. SERVİS KALİTESİ NEDİR?

Servis Kalitesi (QoS) birçok farklı teknolojiyi kullanan ve ağ üzerinde kabul edilebilir bir trafik kullanımı sağlayan bir tekniktir. Servis Kalitesi, bant genişliği, gecikme, paket kaybı gibi parametrelere sahiptir. Normalde, ilk gelen ilk servis edilir mantığı geçerlidir. Fakat Servis Kalitesi bir trafik polisi gibidir. Trafik polisi önceliği olmayan araçları bekletir, ambulansa ise yola devam etme izni verir. Dikkat edilmesi gereken nokta; Servis Kalitesi sıkışmaları önlemez veya düşürmez, trafikteki sıkışma anlarında sadece trafiğin önceliklere göre akışını sağlar.

İki temel Servis Kalitesi tipi mevcuttur:

Kaynak Rezervasyonu : Ağ kaynakları, bant genişliği kontrol politikasına bağlı olarak, uygulamaların Servis Kalitesi isteklerine göre ayrılmaktadır.

Önceliklendirme : Ağ trafiği, bant genişliği politikasına bağlı olarak sınıflandırılmaktadır. Daha çok gereksinimi olan uygulamalara, bu sınıflandırma metoduyla öncelikli haklar tanınmaktadır.

4.3. 'tc' Komutu ve SNORT

Son zamanlardaki Linux çekirdeklerinde (2.4.20 ve üzeri), önceliklendirme mekanizmasının uygulanmasını sağlayan komutun adı 'tc' (**t**raffic **c**ontrol) dir. 'tc' komutu başlıca kuyruk mekanizmalarıyla (qdiscs), sınıflarla (classes) ve filtreleme mekanizmalarıyla (filters) uğraşmaktadır. Bütün bunlar sisteme sırayla **tc qdisc…** , **tc class…** ve **tc filter…** şeklinde bildirilmektedir.

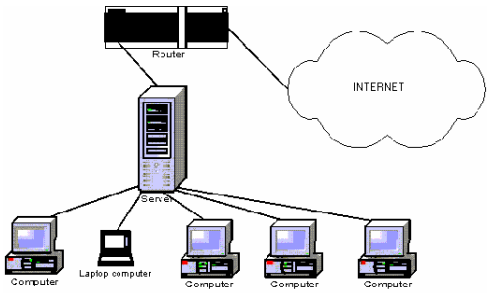
Snort ise bir "Ağ Temelli Saldırı Tespit Sistemi" dır . GPL lisanslı olarak dağıtılmaktadır. Farklı platformlarda (UNIX, MS-Windows) çalışabilmekte olup modüler bir mimariye sahiptir [17]. Snort belirlenmiş kurallara göre ağ üzerindeki her paketi inceler ve belirlenmiş kural meydana geldiğinde gerekli uyarıyı 'Snort uyarı' (/var/log/snort/alert) dosyasına yazar [18].

4.4 AKILLI BANT GENİŞLİĞİ YÖNETİM ARACININ GELİŞTİRİLMESİ

İzmir Yüksek Teknoloji Enstitüsü yerel ağında 1000'e yakın kişisel bilgisayar kullanıcısı bulunmaktadır. Toplam bant genişliğinin 8Mbps olduğu düşünülürse, her kullanıcıya ortalama 8kbps bant genişliği düşmektedir. Ayrıca, mevcut bant genişliğinin büyük bir miktarı mp3, video, divx, vb. dosya download işlemleri ile tüketilmektedir. Bu durumda mevcut bant genişliğinin daha etkili kullanımını sağlamak için akıllı bant genişliği yönetim aracı geliştirilmiştir.

Akıllı bant genişliği yönetim aracı şu görevleri yerine getirmektedir:

- Kullanıcı bilgilerini ve kullanım politakılarını tutmak için basit bir veritabanı
- Kullanıcı bant genişliği kullanımlarının hesaplanması
- Kullanım kuralları
- Bant genişliğini paylaştırmak için karar verme mekanizması
- Bant genişliği kullanımlarını izlemek ve değiştirmek için yönetici ara yüzü



Şekil-7. Bilgisayar Ağının Genel Görünümü

4.5 MODÜLLER

4.5.1 ORTAK VERİTABANI

Ortak veritabanı *Servis Kullanım Bulma Modülü*, *Java RMI Server ve Yönetici Arayüz Modülü ve Bant Genişliği Hesaplama ve Atama Modülü* tarafından ortak olarak kullanılmaktadır.

Veritabanı İsmi : Bandctl
Tablo İsmi : Users

Alanlar	**Özellikler**
uid:int	Kullanıcı No
ip: varchar	Kullanıcı Ip Adresi
up_min: int	Garantilenmiş upload bant genişliği
up_max: int	Maksimum upload bant genişliği
down_min: int	Garantilenmiş download bant genişliği
down_max: int	Maksimum download bant genişliği
prio: int	Kullanıcı önceliği (1-5)
serv1: boolean	Servis 1 Kullanımı
serv2: boolean	Servis 2 Kullanımı
serv3: boolean	Servis 3 Kullanımı
serv4: boolean	Servis 4 Kullanımı
serv5: boolean	Servis 5 Kullanımı
speed_down:int	atanan download bant genişliği
speed_up:int	atanan upload bant genişliği
Manual: boolean	atanan hızın türü, otomatik/el ile

Tablo 1. Ortak veritabanı yapısı

serv{1,2,3,4,5} alanlarında kullanıcının bu trafiği kullanıp kullanmadığını anlık olarak saklanır.

4.5.2 SERVİS TİPİ TESPİT MODÜLÜ

İlk olarak en çok kullanılan servisler belirlendikten sonra, bunlara kendi aralarında Tablo 2' de görülen öncelikler verilmiştir. Bu öncelik değerleri bant genişliği paylaştırma algoritması tarafından kullanılacak olup ileride açıklanacaktır.

Servisler ve öncelik katsayıları				
Servis1	Servis2	Servis3	Servis4	Servis5
Yasak Site Ziyaretleri	HTTP & FTP Kullanımı	POP3 & SMTP	.zip .rar vb. dosya transferleri	.divx .mpeg vb büyük dosya transferleri
0.05	0.3	0.3	0.2	0.15

Tablo 2: Servisler ve öncelik katsayıları

Ağ üzerindeki servis kullanımlarını izlemek içinse Snort kullanılmıştır. Tablo 2' de görülen 5 farklı servis kullanımı için Snort programı ayarları şu şekilde yapılmıştır:

Servis 1 : Yasak Site Ziyaretleri
alert tcp any any->any any
(content-list:"rule1.txt";msg:"rule1";)
Servis 2 : HTTP ve FTP Kullanımı
alert tcp any any -> any 80 (msg:"rule2";)
Servis 3 : POP3 & SMTP
alert tcp any any -> any 110 (msg:"rule3";)
Servis 4 : .zip .rar vb. dosya transferleri
alert tcp any any -> any any
(content-list:"rule4.txt";msg:"rule4";)
Servis 5 : .divx .mpeg vb büyük dosya transferleri
alert tcp any any -> any any
(content-list:"rule5.txt";msg:"rule5";)

Kural1, kural4 ve kural5 yazı dosyasına kaydedilmiş kelime izlerini içeren paketler içindir. Kural2 ve kural3 ise çok kullanılan servis portlarını izlemek içindir. Belirlenen servislere göre Snort ağı izlemekte ve uyarı mesajları vermektedir. Tablo 3' de yerel ağa bağlı olan 193.140.250.65 IP adresli istemci makinenin servis3 kullanımı için verilen uyarı mesajı görülmektedir.

```
[**] [1:0:0] rule3 [**] [Priority: 0]
05/27-14:25:39.705667      193.140.250.65:1150      ->
199.237.72.126:80
TCP TTL:128 TOS:0x0 ID:3903 IpLen:20 DgmLen:40
DF ***A**** Seq: 0x89F2C9A9  Ack: 0xA3E619B1
Win: 0x42C8  TcpLen: 20
```

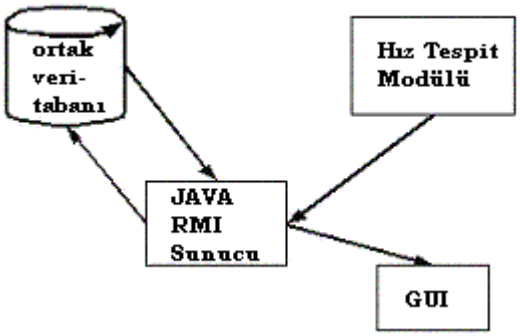
Tablo3. Örnek Snort uyarı mesajı

Tablo 3' deki Snort uyarı mesajlarından gerekli kısımlar bir perl programı yardımıyla alınır ve veritabanına yazılır.

4.5.3 JAVA RMI SERVER ve YÖNETİCİ ARAYÜZ MODÜLÜ

Java Server veritabanındaki verileri göstermek ve bant genişliği ayarlarını sistem yöneticisinden almak için kullanılır. Java Applet ve RMI teknolojisi kullanılarak web desteği sağlanmıştır.

Java Server, kullanıcıların bant genişliği kullanımlarını Hız Tespit Modülü'nden düzenli olarak alır ve bu verileri yönetici arayüzünde anlık olarak gösterir. Böylece kullanıcıların bant genişliği kullanımlarını ve ayrıca servis kullanım bilgilerini izlemek mümkündür. İstenilirse kullanıcı bant genişliği kullanım kısıtlamaları el ile de ayarlanabilir.



Şekil-8. Java RMI Server ve Yönetici Arayüzü Veri Akış Şeması

Şekil 9. Yönetici Arayüzleri

4.5.4. HIZ TESPİT MODÜLÜ

Hız tespit modülü ağ üzerinde geçen her paketi inceler, her paketin boyutunu alarak download / upload türüne göre her kullanıcı için kaydeder. Hızdaki ani değişimler için alçak geçiren filtreleme kullanılmakadır [19]. Buna göre hız değeri en son on değerin ortalaması alınarak elde edilir.

Her bir makine için bit/s cinsinden hesaplanan hız değerleri, Hız Tespit Modülünden Java RMI Server'a gönderilir.

Şekil 10. Java RMI Server ve Hız tespit Modülü için Veri Akış şeması

4.5.5 BANT GENİŞLİĞİ HESAPLAMA VE ATAMA MODÜLÜ

Bant genişliği hesaplama modülü veritabanından gerekli bilgileri alarak, aşağıda açıklanan algoritma yardımıyla her kullanıcı için bir bant genişliği değeri hesaplar.

4.5.5.1 ALGORİTMA

Terimler ve kullanılan değişkenler:
Toplam upload bant genişliği : *total_up_bw*
Toplam download bant genişliği : *total_dw_bw*
Üniversite kullanıcıları: $\{C_1, C_2,..,C_n\}$
Her kullanıcı için servis kullanım değişkenleri:

$\{s_1, s_2, s_3, s_4, s_5\}$
Her kullanıcı için öncelik değerleri :
$\{C_1p, C_2p,..., C_np\}$
Servis toplam upload bant genişliği kullanımları :
$\{S_1up, S_2up, S_3up, S_4up, S_5up\}$
Servis toplam download bant genişliği kullanımları :
$\{S_1dw, S_2dw, S_3dw, S_4dw, S_5dw\}$
Servis öncelik değerleri: $\{S_1p, S_2p, S_3p, S_4p, S_5p\}$
Öncelik değerleri toplamı: *Sp_total*
Kullanıcı öncelik değerleri toplamı: C*p_total*
Her servis için toplam kullanım değerleri:
$\{S_1k, S_2k, S_3k, S_4k, S_5k\}$
Her kullanıcı için istenildiğinde el ile atanan maksimum upload hızı: $\{C_1m\_up, C_2m\_up,...,C_nm\_up\}$
El ile atanan upload hızların toplamı: M_up
El ile atanan download hızların toplamı: M_dw
El ile atama ayarlarını tutan alan: m, eğer m 0 değerini alırsa hız atama otomatik, 1 olursa el ile olacaktır.
Otomatik atanan upload hızlarının toplamı: O_up
Otomatik atanan download hızlarının toplamı: O_dw
Her kullanıcı için istenildiğinde atanan maksimum download hızı: $\{C_1m\_dw, C_2m\_dw,\ldots,C_nm\_dw\}$
Her kullanıcı için otomatik olarak atanan maksimum upload hızı: $\{C_1up, C_2up,...,C_nup\}$
Her kullanıcı için otomatik olarak atanan maksimum download hızı: $\{C_1dw, C_2dw,\ldots,C_ndw\}$

Hız Tespit Modülü Çalışma Algoritması:

1. Manuel atamalar için toplam upload ve download bant genişliği değerleri hesaplanır.

$$M\_up = \sum_{i=0}^{n} Cim\_up * mi$$

$$M\_dw = \sum_{i=0}^{n} Cim\_dw * mi$$

2. Elde edilen toplamlar genel toplamdan çıkarılır, geriye kalan değer diğer kullanıcılar arasında otomatik olarak paylaştırılır.

$$O\_up = total\_up\_bw - M\_up$$
$$O\_dw = total\_dw\_bw - M\_dw$$

3. Toplam servis kullanımları hesaplanır.

$$S_1k = \sum_{i=0}^{n} Cis1,$$

$$S_2k = \sum_{i=0}^{n} Cis2,$$

…

$$S_5k = \sum_{i=0}^{n} Cis5$$

Kullanılan servislerin öncelik değerleri toplanır.

$$Sp\_total = \sum_{i=0}^{n} \{Sip \mid Sik > 0\}$$

4. Servis kullanımlarına göre mevcut bant genişliği, kullanımdaki servisler arasında paylaştırılır.

$$S_1up = S_1p / Sp\_total * O\_up$$
$$S_1dw = S_1p / Sp\_total * O\_dw$$

Her bir servis için bu tekrarlanır.

5. Kullanıcı öncelik değerlerinin toplamı bulunur. Her bir kullanıcıya kullandığı servis için mevcut o servis kapatasitesinden kullanıcı önceliği ve toplam servis kullanımına göre bant genişliği atanır, atanan değerler toplanarak kullanıcı için toplam bant genişliği değeri bulunur.

$$Cp\_total = \sum_{i=0}^{n} Cip$$

$$Ciup = [\,(Cis1 / S_1k) + \ldots + (Cis1 / S_1k)\,] * (S_1up * Cip / Cp\_total)$$

$$Cidw = [\,(Cis1 / S_1k) + \ldots + (Cis1 / S_1k)\,] * (S_1dw * Cip / Cp\_total)$$

6. Her kullanıcı için hesaplanan $C_i$up, $C_i$dw değerleri veritabanına kaydedilir.
7. Bant genişliği atama modülü ile otomatik ve el ile atanan değerler veritabanından alınarak `tc` komutu yardımıyla sistem yeni değerlerle konfigüre edilir.

4.5.5.2 ALGORİTMANIN ÇALIŞMASINA ÖRNEK

Sistemi kullanan aktif 3 kullanıcımız olsun (C_1,C_2,C_3), birinci kullanıcı (C_1), servis1 (C_1s_1), servis2 (C_1s_2), servis5 (C_1s_5) kullansın, ikinci kullanıcı (C_2), servis2 (C_2s_2), servis3 (C_2s_3), servis4 (C_2s_4), üçüncü kullanıcı (C_3), servis2 (C_3s_2), servis3 (C_1s_2), servis5 (C_1s_5) kullansın. Bu kullanıcıların öncelik değerleri sırasıyla 4, 5, 4 ve toplam bant genişliği de 500Kb (Upload), 1000Kb (Download) olsun.

Tablo 1'e göre, her servisin öncelik değerleri sırasıyla 0.05, 0.3, 0.3 ve 0.15 tir. Toplam servis kullanım değeri 0.8, dolayısıyla servis1'in bant genişliği kullanım değeri 0.05/0.8*500 = 31.25Kb upload ve 0.05/0.8*1000 = 62.5Kb download'tur.

Servis toplam kullanımları ise: servis1:1, servis2:3, servis3:2, servis4:0, servis5:2 dir. Bu durumda,

Servis 1 = 31.25 kb (upload), 62.5 kb (download),
Servis 2 = 187.5 kb (upload), 375 kb (download),
Servis 3 = 187.5 kb (upload), 375 kb (download),
Servis 4 = 0 kb (upload), 0 kb (download),
Servis 5 = 93.75 kb (upload), 187.5 kb (download),

Kullanıcı öncelik değerlerinin toplamı: 4+5+4 = 13
Birinci kullanıcı için toplam upload bant genişliği = 1/1*31.25+1/3*187.5+0+0+1/2*93.75 = 140,63
Birinci kullanıcı için toplam download bant genişliği = 1/1*62.5+1/3*375+0+0+1/2*187.5 = 281,26

Görüldüğü gibi, servis1'i kullanan sadece 1 kullanıcı olduğundan servis1'e ayrılan bant genişliğinin tümünü kullanmaktadır.

4.6. SONUÇ

Bant genişliği yönetiminin otomatik ve manuel olarak yapılabilmesi sağlanmıştır. Kullanıcı sayısı ve kullanılan servisler sürekli büyürken, akıllı bant genişliği yönetiminin

optimum çözümü getireceği düşünülmektedir. Geliştirilen akıllı bant genişliği mekanizması sayesinde toplam bant genişliği kullanımının belirli bir değerde tutulması ve trafik tiplerini önceliklendirmek suretiyle bant genişliğinin paylaştırılması sayesinde geniş alan ağı çıkışının kurum politikasına uygun olarak kullanılması sağlanmıştır.

5. GELECEK ÇALIŞMALAR

GuardiLAN projesinde, topoloji belirleme çalışması veri bağlama katmanı seviyesinde yapılmakta olup, henüz tamamlanmamıştır. Bu projenin tamamlanmasının ardından, topolojinin ağ katmanı seviyesinde belirlenmesi yönünde çalışmaya devam edilmesi hedeflenmektedir.

Genel sistem güvenliğinin artırılması ve ağın takibinin daha kolaylaştırılması amacıyla geliştirilebilecek projeler sadece geliştiricilerin hayaliyle sınırlıdır. Geliştirilmiş ve gelecekte geliştirilebilecek olan ağ yönetimi ve güvenliği projelerinin GuardiLAN'a entegrasyonu ortak veri tabanı ve web ara yüzü sayesinde kolaylıkla sağlanabilecektir.

KAYNAKÇA

[1] Hewlett Packard Open View. http://www.openview.hp.com/

[2] CAIDA Tools : Skitter. http://www.caida.org/tools/measurement/skitter/

[3] Scotty - Tcl Extensions for Network Management Applications. http://wwwhome.cs.utwente.nl/~schoenw/scotty/

[4] Peregrine – Network Tools. http://www.peregrine.com

[5] Micromuse – Netcool Solutions. http://www.micromuse.com/

[6] Y. Breitbart, M. Garofalakis, C. Martin, R. Rastogi, S. Seshadri, and A. Silberschatz, "Topology discovery in heterogeneous IP networks," in *Proc. of the 2000 IEEE Computer and Communications Societies Conf. on Computer Communications (INFOCOM-00)*, (Los Alamitos, CA), pp. 265–274, IEEE, Mar. 26-30, 2000.

[7] B. Lowekamp, D. R. O'Hallaron, and T. R. Gross, "Topology discovery for large Ethernet networks," in *ACM SIGCOMM 2001*, (San Diego, CA), pp. 237–248, ACM, Aug. 27-31, 2001.

[8] Y. Bejerano, Y. Breitbart, M. Garofalakis, and R. Rastogi, "Physical Topology Discovery for Large Multi-Subnet Networks", July 2002, Bell Labs Tech. Memorandum.

[9] David T. Stott, "Layer-2 Path Discovery Using Spanning Tree MIBs", Avaya Labs. Research, 2002

[10] J. Case, M. Fedor, M. Schoffstall, and J. Davin, "A Simple Network Management Protocol (SNMP)", Internet RFC-1157 (http://www.ietf.org/rfc/rfc1157.txt), May 1990.

[11] K. McCloghrie and M. Rose, ""Management Information Base for Network Management of TCP/IP-based internets: MIB-II"," Internet RFC-1213 (http://www.ietf.org/rfc/rfc1213.txt), Mar. 1991.

[12] E. Decker, P. Langille, A. Rijsinghani, and K. McCloghrie, "Definitions of Managed Objects for Bridges"," Internet RFC-1493 (http://www.ietf.org/rfc/rfc1493.txt), July 1993.

[13] P.Grillo, S.Waldbusser, "Host Resources MIB" Internet RFC-1514 (http://www.ietf.org/rfc/rfc1514.txt) September 1993

[14] Otel, Daniel - Some security aspects of link layer Protocols Department of Computer Engineering, Chalmers University of Technology

[15] Tuğlular Tuğkan, "Çok Erişimli Ortamların Korunmasına Yönelik Bilgisayar Denetimli Güvenlik Spesifikasyonları", Ege Üniversitesi, Fen Bilimleri Enstitüsü, 1995.

[16] Tanenbaum A. S. , 1996; Computer Networks 3rd Edition Prentice Hall International Inc. , ISBN : 0-13-394248-1

[17] "The Need for QoS" White Paper. http://www.qosforum.com/

[18] Gaur, N., "Snort: Planning IDS for Your Enterprise", July 2001. http://www.linux.org/

[19] Çağrıcı Gökhan, "QoS in IP Networks", May 2002.