PAMUKKALE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

AĞ İZLEME VE PERFORMANS DEĞERLENDİRME

YÜKSEK LİSANS TEZİ
Merve KEÇELİ

Anabilim Dalı : Bilgisayar Mühendisliği

Tez Danışmanı: Yrd. Doç. Dr. Gürhan GÜNDÜZ

Mart 2011

YÜKSEK LİSANS TEZ ONAY FORMU

Pamukkale Üniversitesi Fen Bilimleri Enstitüsü 081281009 nolu öğrecisi Merve KEÇELİ tarafından hazırlanan "**AĞ İZLEME VE PERFORMANS DEĞERLENDİRME**" başlıklı tez tarafımızdan okunmuş, kapsamı ve niteliği açısından bir Yüksek Lisans tezi olarak kabul edilmiştir.

Tez Danışmanı : (Jüri Başkanı)	**Yrd. Doç. Dr. Gürhan GÜNDÜZ(PAÜ)**
Jüri Üyesi :	**Yrd. Doç. Dr. Emre ÇOMAK (PAÜ)**
Jüri Üyesi :	**Doç. Dr. Abdullah Tahsin TOLA (PAÜ)**

Pamukkale Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 01.06.2011 tarih ve ....16/29... sayılı kararıyla onaylanmıştır.

Fen Bilimleri Enstitüsü Müdürü
Prof. Dr. Nuri KOLSUZ

Bu tezin tasarımı, hazırlanması, yürütülmesi, araştırmalarının yapılması ve bulgularının analizlerinde bilimsel etiğe ve akademik kurallara özenle riayet edildiğini; bu çalışmanın doğrudan birincil ürünü olmayan bulguların, verilerin ve materyallerin bilimsel etiğe uygun olarak kaynak gösterildiğini ve alıntı yapılan çalışmalara atfedildiğine beyan ederim.

İmza : 

Öğrenci Adı Soyadı : Merve KEÇELİ

TEŞEKKÜRLER

Bu çalışmanın gerçeklenmesinde katkıda bulunan Tez Danışmanım ve Pamukkale Üniversitesi Bilgisayar Mühendisliği Öğretim Üyelerinden Yrd. Doç. Dr. Gürhan GÜNDÜZ'e, ve tez konusunun belirlenmesinde yardımcı olan hocam Yrd. Doç. Dr. A. Kadir YALDIR'a, tezin belirli aşamalarında desteğini aldığım meslektaşım PAÜ öğrencisi Mansur DİNÇER'e ve Pamukkale Üniversitesi Bilgi İşlem biriminde çalışan arkadaşlarıma teşekkür ederim.

İÇİNDEKİLER

Sayfa

KISALTMALAR

API : Application Programming Interface
ATM : Asyncronous Transfer Mode
AUI : Attachment Unit Interface
BNC : Bayonet Neill-Concelman
CLI : Command Line Interface
CPU : Central Processing Unit
CSMA/CD : Carrier Sense Multiple Access with Collision Detect
DHCP : Dynamic Host Configuration Protocol
FTP : File Transfer Protocol
ICMP : Internet Control Message Protocol
IPX/SPX : Internetwork Packet Exchange/Sequenced Packet Exchange
ISO : International Organization for Standardization
LAN : Local Area Network
MAC : Media Access Control
MAN : Metropolitan Area Network
MHz : MegaHertz
MSAU : MultiStation Access Unit
OSI : Open Systems Interconnection
NetBEUI : NetBIOS Extended User Interface
OLAP : Online Analytical Processing
OLTP : Online Transaction Processing
IOS : Internetwork Operating System
OSI : Open Systems Interconnection
PIB : Performance Monitoring Base
RAM : Random Access Memory
RPC : Remote Procedure Call
SMTP : Simple Mail Transfer Protocol
SNMP : Simple Network Management Protocol
STP : Shielded Twisted Pair
SQL : Structured Query Language
TCP/IP : Transmission Control Protocol / Internet Protocol
TFTP : Trivial File Transfer Protocol
UDP : User Datagram Protocol
UTP : Unshielded Twisted Pair
VLAN : Virtual Local Area Network
XML : Extensible Markup Language
WAN : Wide Area Network
WMI : Windows Management Instrumentation

TABLO LİSTESİ

Tablolar

ŞEKİL LİSTESİ

Şekiller

ÖZET

AĞ İZLEME VE PERFORMANS DEĞERLENDİRME

Günümüzde her kurum neredeyse bütün işlemlerini bilgisayar üzerinde yapmakta ve hayati önem taşıyan verilerini de yine bilgisayar ortamında saklamaktadır. Dolayısıyla bu verilerin işlendiği ve saklandığı makinalar da hayati önem taşır hale gelmiştir. Bu durum, bu makinaların öncelikle donanımsal kaynaklarının ne durumda olduğunun takibini yapmanın gerekliliğini ön plana çıkarmıştır. Öyle ki bahsedilen kaynaklar bilgisayarların verimli çalışması için temel oluşturmaktadır. Donanımsal kaynakları belirli nedenlerle normalden fazla kullanılan ya da tükenen bir bilgisayara belirli zamanlardaki durumu incelenerek müdahale edilebilir ve sorun çözülebilir. Bu yine bilgisayar üzerinde geliştirilen bir yazılımla toplanan verilerle mümkün olabilmektedir. Bu yazılım ağın tamamını bir döngü halinde tarayacak, belirlenen parametreleri veri tabanına kayıt edecek, kaydettiği verilerden de anlamlı raporlar sunacaktır. Oluşturulacak raporların ne kadar güvenilir olduğu ağın belirli zaman aralığında tarama sayısının ne kadar fazla olduğu ile doğru orantılıdır. Bunun sağlanabilmesi geliştirilen yazılım için kullanılan teknolojinin sunduğu imkanları verimli kullanmakla mümkün olacaktır.

Anahtar Kelimeler: Ağ değerlendirme, ağ izleme, performans sayacı, raporlama

SUMMARY

NETWORK MONITORING AND PERFORMANCE EVALUATION

Nowadays, almost every institution uses computer technology to handle their work load and to store vital data. In light of this fact, the computers that execute these operations themselves have also become crucial devices. Therefore, monitoring the hardware resources of these machines is a primary concern. So much so that these aformentioned resource checks form the basis of an efficently working computer system. The computers that overuse or deplete these resources quickly for one reason or another can be inspected and repaired in due time. This process is made possible with a software that is programmed by using such computers, as it records the necessary information. This software will keep scanning the whole network, will save the determined parameters in the database, and will present reports based on the saved data. The reliability of the reports will be directly proportional to the frequency of scans performed on the network. This process will be made possible by efficiently taking advantage of the technology designed specifically for the software.

Key words: Network evaluation, network monitoring, performance counter, reporting

1. GİRİŞ

Günümüzde artık kolayca elde edilen bilgi teknolojileri yardımıyla her veri sayısal ortamlara kaydedilebilir duruma gelmiştir. Örneğin bir kurum ağ yapısı içerisinde bulunan bilgisayarlarındaki gözlemlemek istediği verileri kayıt etmek ve sonrasında bu verileri raporlamak isteyebilir. Bu sayede sorun teşkil ettiğini düşündüğü bilgisayara kolayca müdahele edebilir.

Ağ izleme ve değerlendirme günümüzde çok sayıda ağ cihazının birbirine bağlı olarak çalıştığı karmaşık sistemlerde problemlerin tespitinde ve çözüme kavuşmasında önemli rol oynamaktadır. Ağ izleme ve değerlendirme kavramı kurumun ihtiyaçlarına göre çeşitlilik gösterebilen geniş kapsamlı bir kavramdır. Burada takip edilmek istenen parametrelere göre farklı projeler geliştirilebilmektedir. Bu proje kapsamında CPU (Central Processing Unit-Merkezi İşlemci Birimi), RAM (Random Access Memory-Rasgele Erişimli Bellek), ağ kullanım bilgileri parametre olarak belirlenmiş, toplanan veriler veri tabanına kaydedilmiş ve kaydedilen verilerden raporlar oluşturulmuştur. Projenin kullanım yeri Pamukkale Üniversitesi Hastaneleridir**.**

1.1 Tezin Amacı

Bu tezde amaçlanan yöneticinin belirlediği parametreye göre bir LAN'a (Local Area Network-Yerel Alan Ağı) bağlı, istenen bilgisayar grubuna erişimi sağlamak ve erişilen bilgisayarlar üzerinde bilgisayar adı, CPU, RAM, ağ kullanım bilgilerini almak, bu bilgileri veri tabanına kaydederek yöneticinin belirlediği aralığa göre raporlar halinde sunmak, eğer istenirse de istenen bilgisayara anlık olarak bağlanıp o bilgisayarın ağ kullanım bilgilerini grafiksel veya verisel olarak göstermektir. Bu sayede yönetici ağının proje kapsamında kaydedilen donanım kaynaklarının takibini yapabilecek, buradan ağının genel durumu hakkında bilgi sahibi olabilecek ve gerekirse de problem tespit ettiği bilgisayara ya da VLAN'a(Virtual Local Area Network-Sanal Yerel Alan Ağı) müdahale edebilecektir. Böylece olası sorunları gözlemleyebilecek ve ağının belirli kaynaklar açısından kontrolünü sağlayabilecektir.

Bu işlem ağın bir program tarafından sürekli taranmasını gerektirmektedir. Burada bir devir tarama işlemi ne kadar hızlı olursa ağ o kadar fazla sayıda taranacak ve böylece elde edilen veriler daha sağlıklı ve daha anlamlı olacaktır. Tez kapsamında tarama işleminin kısa sürede yapılabilmesi amaçlanmış, bu süreçte karşılaşılan olumsuzlukların üzerine gidilmiş ve program düzenli çalışma düzeyine ulaşmıştır.

1.2 Literatür Özeti

Yapılan araştırma ve incelemeler sonucunda geliştirilmeye başlanan proje bugünkü haline gelene kadar birçok aşama geçirmiştir. Bu kapsamda farklı farklı ağ programları ve projeleri incelenerek, avantaj ve dezavantajları irdelenmiş, uygun görülen uygulamalar geliştirilmek üzere proje bünyesine alınmıştır. Bu projeler zamanla ya programa entegre edilmiş ya da bu projelerden yola çıkılarak farklı bakış açıları kazanılmıştır.

A Performance Monitoring Tool for Predicting Degradation in Distributed System başlığı altında yapılan çalışmada PerfMon, izleme yapan sistem, incelenmiştir[1].

PerfMon çekirdek seviyesinde performans izleme yapan Linux tabanlı bir sistemdir. Kullanıcı ara yüzüne /Proc dizininden ilgili sistem bilgilerini almakta ve göstermektedir. /Proc dizini Linux işletim sisteminin standart bir özelliğidir. /Proc sanal dosya dizininin sağladığı şeyler yerel makinadaki birtakım verileri göstermektir. Sistem yükleri bellek kullanımı ve uygulama başına ne kadar yük kullanıldığı bunlara dahildir. Dahil olan düğümlerin herbirinden izleme verisi toplamak için PerfMon birçok izleme modülünden oluşmaktadır. PerfMon aşağıdaki fonksiyon ve gereksinimleri sağlamaktadır:

- PerfMon yapıldığı sisteme çok az yük getirir. İzleme aracı sadece kendisi için çok az ve kısıtlı bir hesaplama kapasitesi harcar. Bunlar da izleme verilerini toplamak için yeterlidir.
- Esnek analiz ve filtreleme sağlamaktadır. Dinamik olarak değişen kesin parametrelerin izlenmesinde basit bir yol sunmaktadır (istenen istatistikler ve izleme oranları gibi).
- Standart API(Application Programming Interface) kullanmaktadır. Uygulamalar açık bir şekilde izleme modülü kullanma ihtiyacı duymazlar. Uygulamalar izleme verilerini almak için PIB'e (Performance Monitoring Base) erişirler.

- Ölçülebilirdir. İsteğe bağlı olarak çok sayıda izleme düğümünün üstesinden gelebilir.

Perfmon uygulaması 4 büyük bileşen içerir.

- İzleme Yöneticisi
- İzleme Modülü
- PIB
- İsimlendirme ve Dizin Servisleri

İzleme Yöneticisi; PerfMon aktivitelerini yönetmek için oluşturulmuş çekirdek modüldür. 6 tane bileşen içermektedir. Bu bileşenler aşağıda açıklanmıştır.
Deamon: En önemli bileşendir. İzleme Yöneticisindeki bütün aktiviteleri kontrol eder. Olaylar başladığında Even Handler olayları parçalamak ve alt bileşenlere iletmek için çağrılır. Daha başka önemli rolleri de yönetmektedir. Konfigürasyon dosyasında belirtilen izleme oranlarına göre sistem kaynaklarının kullanımını toplamak için izleme modüllerini belirli aralıklarla çağırır. Daha sonra bütün kayıtlı filtrelere izleme verilerini yollar.
Son olarak şifrelenmiş verileri veri yollarına iletir.
Event Handler: Direk olarak kontrol yolu vasıtasıyla PIB'e bağlanır. Event Handler olaylar sonuca vardığında her zaman çağrılır. 5 olay tipine sahiptir.
REG_FILTER, CHG_FILTER, REG_MODULE, DEREG_MODULE ve MOD_PARAM REG_FILTER ya da CHG_FILTER olayları bir sonuca vardığında filtre formunda kabul etme kriterlerini belirleyebilir ya da değiştirebilir.
REG_MODULE ya da DEREG_MODULE olayları bir sonuca vardığında izleme modüllerine ilişkin kayıtları ya siler ya da ekler.
MOD_PARAM olayları bir sonuca vardığında parametre formunda kabul etme kriterlerini belirleyebilir ya da değiştirebilir. İki tür parametre tipi fark edilir. Birincisi bir uygulamanın sistem kaynaklarının kullanılabilirliğini ya da geçerli kullanım hakkında ne kadar sıklıkla bilgilendirilmek istediğini gösteren izleme oranlarıdır. İkincisi ise kaynak kullanımında alt ya da üst sınırların eşik değerleridir. İletilen olayları parçalamak ve bu olaylarla ilgili alt bileşenleri çağırmak event handlerın sorumluluğundadır.

Filter: İzleme verilerini filtrelemektedir. Kaydedilen filtreler konfigürasyon dosyası olarak yazılır. Filtrelerin çalışması izleme verileri veri paketleri olarak sunulmadan önce izleme yöneticisi tarafından sonlandırılır.

Data Wrapper: İzleme verilerinin şifrelenmesinde XML(Extensible Markup Language) tabanlı formatın kullanılmasını sağlamaktadır.

Register: Event Handler'dan alınan DEREG_MODULE ya da REG_MODULE olayları tarafından izleme düğümlerindeki izleme modüllerinin kayıt edilmesi ya da silinmesi işlemleri belirlenmektedir.

Data Path and Control Path: PIB'e izleme yöneticisinden izleme verisi göndermede kullanılır. Eksiksiz ve devam eden işletilen sistemdeki gerçek zamanlı performans verileri için sıklıkla ve hızlı izleme trafiği gereklidir. Düşük iletişim yükünü elde etmek için izleme yöneticisinden PIB'e izleme verilerinin iletilmesi için UDP(User Datagram Protocol) protokolu kullanılır. UDP hızlıdır fakat güvenilmeyen bir iletişim protokoludür. PerfMon TCP/IP(Transmission Control Protocol/Internet Protocol) kullanmaktadır.

Runtime System Metrics: PerfMon bütün sistemin kaynak kullanımının ve aktivitelerinin takibinin gerçekleştiği izleme modül çeşitliliğini gerçekleştirir.

CPU_MON: Bu modül filtreler tarafından belirlenebilen belirli bir zaman diliminde ortalama çalışan işlemlerin uzunluğunun sırasının takibini gerçekleştirmektedir.

MEM_MON: Kullanılan ya da boş kalan hafıza bilgileri gibi hafızaya ilişkin bilgileri sağlar.

NET_MON: Bu modül kurulan ağ bağlantılarını, bütün açık ağ trafiğini, iletilen ve kaybolan paketleri göstermektedir.

FS_MOD: Bu modül ortalama disk okuma ve yazmalarını ölçmektedir.

PROG_MON: Bu modül belirli bir işlemdeki bilgiyi verir. Belirli bir işlem tarafından kullanılan bellek bilgisi gibi.

PerfMon uygulamasının temel amaçları:

- İzleme Yöneticisi Kurulu olduğu izleme düğümünde kullanabildiği kadar az sistem kaynaklarını kullanmakta ve izleme sürecini hafifletmektedir.
- Filtreleme mekanizması izleme mekanizması tarafından oluşan karmaşayı ve ek yükü azaltabilir.
- Belirli bir izleme oranı durumunda veri toplamadaki kullanılan toplam zaman izleme zaman aralığından daha azdır.

PerfMon bazı düğümlerde çalışan işlemlerin durumunu izleyebilir. Bu çalışmada PerfMon üç farklı hesaplanmış düğümde bellek kullanımı, boş alan değiş tokuşu ve ağ trafiğini izler, izleme yöneticisi üzerindeki varsayılan bir etkiyi etkilemektedir[1]. PerfMon projesi dahilinde belirlenen izleme modüllerine baktığımızda geliştirilen proje ile benzerlikler göstermektedir. Bu bilgileri toplamak için PerfMon linux işletim sisteminin standart bir özelliği olan /Proc dizinini kullanmakta ve proje dahilinde de windows işletim sisteminin perfctrs.dll dosyası içerisindeki API kullanılmaktadır.

Bir diğer projede[2] incelenen, işlevi izleme yapmak olan, OpManager ağın performansının izlenmesinde kapsamlı ve uygun bir çözümdür. Büyüyen ağ karmaşıklığı ile sistem yöneticileri sağlıklı bir ağ sağlamak konusunda daha çok zorlanmaktadırlar. OpManager ağın, ağ kaynak ve raporlarının kullanılabilirliğini sağlar. OpManager domain kontroller, güvenlik duvarı, switchler, routerlar, serverlar ve bunlar gibi bütün ağa bağlı cihazlar üzerindeki kritik kaynakların kullanımına destek sağlamaktadır.

Diğer taraftan CPU, disk, bellek ve trafik kullanımı, kritik servislerin kullanılabilirliği, önemli Windows olaylarının logları gibi host kaynaklarının da dahil olduğu diğer kaynakları da izlemektedir.

Performans verileri bütün kaynaklar için kaynakların kullanılabilirliğini belirlemek amacıyla periyodik olarak toplanır ki, bu bütün ağ sağlığını belirlemektedir.

Anlamlı eşik değer konfigürasyonu tarafından pro_active izleme tatsız sürprizleri önlemeye yardım etmektedir.

İzleme Performansı SNMP(Simple Network Management Protocol), CLI(Command Line Interface), ve WMI(Windows Management Instrumentation) gibi standart protokoller kullanılarak yapılır. OpManager CLI kullanarak Linux ve solaris işletim sistemlerinde ve SNMP'nin yokluğunda WMI kullanarak Windows işletim sistemlerinde izleme yapabilir.

Program sayesinde ağ durumunun periyodik olarak raporlanması avantaj olarak eklenebilir. OpManager periyodik olarak seçilen gün ve zamanda raporları planlamaya izin verir. Ve raporlar otomatik olarak e-mail atılabilir. OpManager projesinin parametrelerine baktığımızda proje dahilinde belirlenen parametreler ile benzerlikler göstermektedir ve yine bu bilgilerin periyodik olarak toplanması ve raporlanması geliştirilen proje için temel oluşturmuştur.

Son projede ise[3] Cisco'nun Netflow protokolü incelenmiştir. Cisco IOS'un(Internetwork Operating System) bir parçası olan Netflow protokolünü 1996'dan bu yana geliştirmiştir. Netflow Cisco routerlarda önbelleklemeyi kontrol etmek için bir mekanizma olarak hayata başlamıştır. Ve bu ağ paketlerinin hızını arttırmıştır. Netflow açık kaynak kodlu olduğu için diğer router üreticileri tarafından kullanılabilir. Ağ protokolleri UDP ya da TCP olsun IP başlığı bilginin madenidir. Bu iş için router alınan her paketin başlığını denetlemeye ihtiyaç duymaktadır. Router diğer düğüme paket gönderdiğinde hedef ve port verisini kaynağa kayıt etmektedir. NetFlow hangi portun ve protokolün gerektiğini ve ne kadar veri alış verişi olduğunu göstermektedir.

Netflow, Cisco cihazlarıyla kurulu ağdaki trafiği detaylı analiz etme ve raporlama imkanı sağlamaktadır. Hangi kullanıcıların kaynakları tükettiği, hangi dönemlerde, ne kadar süre ile yavaşlama yaşandığı sorularına NetFlow analiz projesi ile cevap bulunabilmektedir.

2. AĞ

2.1 Ağ Nedir?

Birden fazla bilgisayarın çeşitli sebeplerden dolayı birbirlerine bağlandığı yapıya ağ denilmektedir. Bilgi paylaşımı, yazılım ve donanım paylaşımı, merkezi yönetim ve destek kolaylığı gibi konular düşünüldüğünde birden fazla bilgisayarın bulunduğu ortamlarda bir ağ kurulması zorunludur.

2.2 Ağ Tipleri

2.2.1 Peer-to-Peer Ağlar

Peer-to-Peer ağlarda tüm birimler servis istediğinde bulunma hakkına sahiptir. Ağdaki herhangi bir bilgisayar farklı zaman dilimlerinde hem istemci hem de sunucu olabilmektedir. Bütün birimler yönetim mekanizması bakımından birbirine benzemektedir.
Kullanıcıların bilgisayar kullanım bilgisinin iyi olması gerekmektedir. Her bilgisayar sadece kullanıcısı tarafından yönetilebilmektedir. Bu tip ağlarda genelde IPX/SPX(Internetwork Packet Exchange/Sequenced Packet Exchange) veya NetBEUI(NetBIOS Extended User Interface) protokolleri kullanılmaktadır. Kullanıcı sayısı günümüz ağ sistemlerini karşılayabilecek boyutta değildir[4].

2.2.2 İstemci Sunucu Ağlar

Ağlar genişledikçe peer to peer ağlar ihtiyaçları karşılayamaz duruma gelmişlerdir. Bu ihtiyaçları karşılayabilmek için sunucular ortaya çıkmıştır. Ağdaki bir sunucu sadece sunucu olarak davranabilir. Ağlar büyüdükçe İstemci Sunucu tipi ağlar standartlaşmaya başlamıştır.

2.3 Büyüklüğüne Göre Ağlar

Ağlar büyüklüklerine göre LAN, WAN (Wide Area Network) ve MAN (Metropolitan Area Network) olmak üzere üçe ayrılmaktadır.

LAN: Birbirine yakın yerlerde konumlandırılmış ve kablolar ile fiziksel olarak birbirlerine bağlanmış yapıdaki ağlar LAN olarak adlandırılmaktadır. Örneğin bir binada bulunan bütün bilgisayarların birbirlerine bağlanmasıyla oluşan yapı bir LAN dır.

WAN: İki ya da daha fazla LAN'ın telefon hatları, kiralık hatlar ya da benzer yollardan birbirlerine bağlanmasıyla oluşan yapı ise WAN olarak adlandırılmaktadır.

MAN: LAN'ın kapsadığı alandan daha geniş, fakat WAN'ın kapsadığından daha dar mesafeler arası iletişimi sağlayan ağlardır. Genellikle şehir içi bilgisayar sistemlerinin birbirleriyle bağlanmasıyla oluşturulur[4].

2.4 Ağ Teknolojileri

Bütün ağ teknolojilerine taşıyıcı protokoller denilebilmektedir. Aynı zamanda, bilgisayarların birbirleriyle konuşmalarının dil kurallarını koyan iletişim protokolleri vardır. TCP/IP, NetBEUI gibi protokoller iletişim protokolü olarak adlandırılabilmektedir. Bu bağlamda TCP/IP, NetBEUI protokollerinin kuralları ile oluşturulan veri paketleri, taşıyıcı protokollerinin veri paketlerine dahil edilerek ağ üzerinde taşınırlar. Bu olaya kapsülleme denilmektedir.

Ağ teknolojilerinin anlattığı şey, verinin bir yerden bir başka yere nasıl ve hangi kurallarla gittiğini belirlemektir. Bu kurallara uygun cihazlar ve bu cihazları kontrol edebilen işletim sistemleri, ağı oluşturan diğer öğelerdir.

Bir ağ teknolojisini diğer bir ağ teknolojisinden ayıran temel özellik; bu teknolojinin veriyi kablolar üzerinde nasıl ve hangi yolla aktardığıdır[4].

2.4.1 Ethernet

Ethernet, verilerin kabloyla iletilmesi sağlayan bir teknolojidir. Bu iletimde CSMA/CD(Carrier Sense Multiple Access With Collision Detection) tekniği kullanılmaktadır. Bu erişim yönteminde ağ üzerindeki bütün bilgisayarlar ağ kablosunu sürekli kontrol etmektedir. Kablonun boş olduğunu algılayan veriyi gönderir. Bu arada eğer kabloda veri varsa o zaman veri hedefine ulaşıncaya kadar

beklenmektedir. İki bilgisayarın paketleri kabloda karşılaşırlarsa çarpışma oluşur. Bu durumda her ikisinin de trafiği kaybolur. Ve hattın boş olduğu anı yakalamak için yeniden beklemeye ve hat dinlenmeye başlanır. CSMA/CD ağlarında, beklemelerin çoğalmaması için bus olarak tanımlanan kablonun iki ucunun sonlandırılması gerekmektedir. Sonlandırma bakır kablo üzerinde elektrik sinyalleri olarak taşınan paketlerin kablonun bittiği yerde gücünün alınmasına denilmektedir. Bu işlem elektrik sinyallerinin geri dönmesini önlemektedir. Yansımanın önüne geçilmesiyle kablonun sonuna çarpıp dönen sinyallerin yeniden bir trafik oluşturması engellenmiştir[4].

2.4.2 Token Ring

Token ring teknolojisi, bir ring topolojisi içinde uygulanmaktadır. Fiziksel olarak kullanılan topoloji bir star topolojisidir. Fakat, merkezdeki cihazın yapısı nedeniyle, veriler kapalı bir zincir üzerinde dolaşıyormuş gibi hareket etmektedir. Bir veri her bilgisayara teker teker gönderilmektedir. Ve veri merkezi birimden gereksiz yere birden fazla geçmiş olmaktadır. Token ring teknolojisinin kullanılmasının sebebi veri kaybının minumum düzeyde olmasıdır. Oldukça önemli verilerin taşınması için düşünülmüş bir teknolojidir. Yavaşlığı nedeniyle çok az uygulama alanı bulmuştur. Bu teknolojide veriler bir paket halinde her bilgisayara sıra ile merkezi birim tarafından gönderilmektedir. Bu sırada her makine paketin kendisine ait olup olmadığını kontrol eder. Bu sebeple merkezde yer alan hub ya da switch benzeri cihaza, MSAU (Multi Station Access Unit) adı verilmektedir[4].

2.4.3 ATM

ATM (Asyncronous Transfer Mode), 1988 yılında her tür veriyi, telefon hatları, ses, TV sinyali, ağ üzerinde gidip gelen veriler gibi, çeşitli veri sinyallerini taşıması için oluşturulmuş bir standarttır. ATM, paket anahtarlama yapan bir teknolojidir. Paket anahtarlama birden fazla parçası olan bir ağda verinin parçalara bölünüp, teker teker ayrı yollardan gönderilmesi ve ulaştığı yerde tekrar birleştirilmesi esasına dayanmaktadır[4].

2.4.4 FDDI

1986 yılında ANSI X3T9.5 komitesi tarafından tanıtılmış bir teknolojidir. FDDI, 100 Mbps'nin üzerindeki hızlarda veri aktarmak için, fiber optik kabloların kullanıldığı bir yapıyı oluşturmaktadır. FDDI, 1986 yılında ilk bulunduğunda yüksek kapasiteli bilgisayarlar için, o günlerde var olan 10 Mpbs'lik ethernet ve 4 Mbps'lik token ring teknolojilerine bir alternatif olarak sunulmuştur. FDDI, prensip olarak iki kapalı zincir üzerinde ters yönde hareket eden veri trafiğine göre yapılandırılmıştır. Bu kapalı hat ya da zincir olarak tabir edilen yapılardan biri boş olarak hazırda tutulur. Veri taşıyan zincirde bir problem olduğunda ikinci zincir devreye girer ve veriyi ters yönde taşımaya başlar. FDDI'da da token ring teknolojisinde olduğu gibi token isimli veri paketleri kullanılır. Paket yapıları birbirinden farklı olsa da veri bir zincir etrafında dolaştırılarak taşınır ve token ringdeki gibi her bilgisayardan bir kez geçer. FDDI, son derece yüksek bir güvenilirliğe ve veri aktarım hızına sahiptir[4].

2.4.5 Frame Relay

Frame relay, verileri paket anahtarlama prensibi ile taşıyan ve değişken veri paketlerinin kullanıldığı bir veri aktarım teknolojisidir. Veriler bir ağda bir noktada bir başka noktaya hedeflenerek gönderilmektedir. Bu verilerin ağ üzerinde gittiği yol, ulaştığı nokta tarafından bilinmez. Frame relay, bu sayede, ağdaki trafiğin kolayca gözlemlenebilmesini mümkün kılmaktadır. Frame relay ile veri paketleri istenildiğinde hep aynı yoldan gönderilebilmektedir. Bu sayede, ağdaki trafik istenildiği gibi düzenlenebilmektedir[4].

2.5 Ağ Cihazları

Bir ağı genişletmek, güvenliğini saklamak ve aynı zamanda hiyerarşi kazandırmak için bazı cihazlar kullanılmalıdır.

2.5.1 Microtransceiver

Ethernet ağlarında kullanılan kablo tiplerinden birini diğerine çevirmeye yarar. Üzerinde 2 port yer alır ve çeşitli tipleri vardır. BNC'yi(Bayonet Neill-Concelman) AUI'ya(Attachment Unit Interface) ya da UTP'yi(Unshielded Twisted Pair) AUI ya çevirmek örnek olarak verilebilir**.**

2.5.2 Transceiver

Transceiverlar kalın ethernet kullanılan ağlarda koaksiyel kabloya bağlantı yapmak için kullanılır. Transceiver'ın kabloya takılan kısmında iğne gibi bir parça kablonun iletken kısmıyla temas sağlamaktadır.

2.5.3 Hub

En basit ağ cihazıdır. Kendisine bağlı olan bilgisayarlara paylaşılan bir yol sunmaktadır. Yani huba bağlı tüm cihazlar aynı yolu kullanmaktadır ve bu da aynı anda haberleşmek isteyen ağ cihazlarının, bir tek yol olduğu için hattın boşalmasını beklemelerine sebep olmaktadır. Bir ağ kartı kendisine ait olmayan bir paketi aldığında, kendi adresinin paketin ulaştırılması gereken yerdeki MAC(Media Access Control) adresiyle aynı olmadığını fark ederse, paketi yok etmektedir.

2.5.4 Switch

Switch kendisine bağlı cihazlara anahtarlamalı bir yol sunmaktadır. Switchler, bir topolojinin merkezinde yer almakta ve onlara gönderilen verileri, bağlı olan bilgisayarlardan birine göndermektedirler. Bir switch ilk kez çalıştırıldığında, hangi portunda hangi MAC adresini taşıyan bilgisayarın bulunduğunu bilemez. Switch, bir süre hub gibi çalışarak bir tür öğrenme sürecine girmektedir. Bir bilgisayardan gelen paketi diğer tüm bilgisayarlara (sadece ilk anda) gönderir. Sonra bilgisayarların ağ kartlarının MAC adreslerini içerisindeki çiplerde tutmaya başlarlar. Switchin ilk anda yaptığı bu işleme flood adı verilmektedir.

2.5.5 Repeater

Repater bir ethernet segmentinden aldığı tüm paketleri yinelemekte ve diğer segmente yollamaktadır. Repeater gelen elektrik sinyallerini almakta ve binary koda yani 1 ve 0'lara çevirmektedir. Sonra da diğer segmente yollamaktadır. Repater basit bir yükseltici değildir. Yükselticiler gelen sinyalin ne olduğuna bakmadan sadece gücünü yükseltmektedir. Yolda bozulmuş bir sinyal yükselticiden geçince bozulma daha da artmaktadır. Repeater ise gelen sinyali önce 1 ve 0'a çevirdiği için yol boyunca zayıflamış sinyal tekrar temiz 1 ve 0 haline dönüşmüş olarak diğer segmente aktarmaktadır. OSI(Open Systems Interconnection) katmanlarından 1. katmanda çalışır[4].

2.5.6 Bridge

İki TCP/IP ağını birbirine bağlayan bir donanımdır. Fazla karmaşık aygıtlar olmayan bridge'ler gelen frameleri almakta ve yönlendirmektedirler. Bridgeler fiziksel bağlantının yanısıra ağ trafiğini kontrol eden aygıtlardır. Bridge bir çeşit yönlendirme yapmaktadır fakat OSI katmanlarından 2. katman yani veri bağlantı katmanında çalışmasıyla routerdan ayrılmaktadır[4].

2.5.7 Firewall

Türkçe güvenlik duvarı anlamına gelen firewall, özel ağlar ile internet arasında, her iki yönde de istenmeyen trafiği önleyecek yazılımsal ya da donanımsal sistemdir. firewallların verimli bir şekilde kullanılabilmesi için internet ve özel ağ arasındaki tüm trafiğin firewall üzerinden geçmesi ve gerekli izinlerin/yetkilerin, erişim listelerinin uygun bir stratejiyle hazırlanmış olması gerekmektedir[4].

2.5.8 Router

Router bir yönlendirme cihazıdır ve LAN-LAN ya da LAN-WAN gibi bağlantılarda kullanılır. Routerlar basit bir yönlendirici değildir. Routerlar bir işletim sistemine sahiptirler dolayısıyla programlanabilmektedirler ve gerekli konfigürasyonlar yapıldığında bir uzak ağa erişmek için mevcut birden fazla yol arasında kullanabilecekleri en iyi yolun seçimini yapabilmektedir. Üzerinde LAN ve WAN bağlantıları için ayrı portlar bulunur ve şaseli olarak da üretilebilmektedirler.

Gereksinime göre bu yuvalara LAN ya da WAN portları eklenebilir. OSI katmanlarının 3. katmanında çalışmaktadırlar[4].

2.5.9 Gateway

Gatewayler, routerların yaptığı ve farklı teknolojiler arasında gidip gelen veri paketlerinin dönüştürülmesi işlemini gerçekleştirmektedirler. Gateway, genellikle atanmış bir aygıt veya atanmış bir bilgisayar üzerinde çalışan bir grup servistir. Gatewayler örneğin FDDI ağından gelen paketleri almakta, gidecekleri bilgisayarın adres bilgisini koruyarak, ethernet ağında yol alabilecek şekilde yeniden oluşturmakta ve bunu bir ethernet ağına gönderebilmektedir. Bu işlemi yapabilmek için ağda kullanılan paket yapılarının, adreslerin ve adres yollarının çok iyi bilinmesi gerekmektedir. Gatewayler bu işlem için özelleştirilmiş cihazlardır. Aktarım teknolojisi açısından mimari bir farklılık varsa ve farklı fiziksel protokoller kullanılıyorsa gatewayler kullanılmalıdır[4].

2.6 Ağ Topolojileri

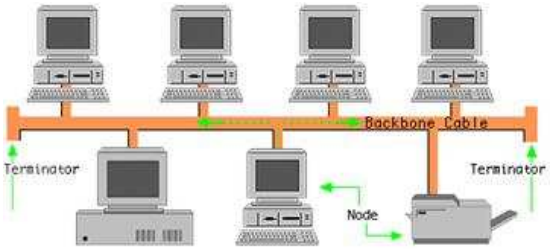
Topoloji, bir ağın fiziksel ve mantıksal yapısını ifade etmektedir. Ağı oluşturan bileşenlerin birbirlerine bağlanış şekilleri, kullanılacak aygıtlar, kablolama standartları, iletişim protokolünün seçimi ve bu protokollerin ağ yapısına uygulanabilirliği de yine topolojinin kapsamı içerisindedir. Bir topoloji hem fiziksel hem de kavramsal bir yapıdır. Fiziksel yapı bir ağda cihazların nasıl birbiri ile bağlanacağını ve ne tür araçlar kullanılacağını belirlemektedir. Kavramsal olarak bir topoloji ağındaki veri trafiğinin planlanmasını sağlamaktadır[4].

2.6.1 Fiziksel Topolojiler

Ağı oluşturan çevre birimlerinin birbirine bağlanırken kullanacakları fiziksel bağlantı metodlarını belirlemektedir. Ağın yapısında kullanılacak kablolama türü ve kullanılacak cihazlar da bu topolojide belirlenmektedir.

Bus Topoloji: Bütün terminaller tek bir doğrusal kablo ile birbirlerine bağlanmışlardır. Bu ağı taşıyan ana kabloya omurga adı verilmektedir. Burada hata gönderilen sinyal bütün terminallere gitmektedir. Sinyal hedefe ulaşana ya da bir

sonlandırıcıya gelene kadar hatta dolaşmaktadır. Bir bus topolojisindeki ağda aynı anda sadece bir bilgisayar veri gönderebilmektedir.

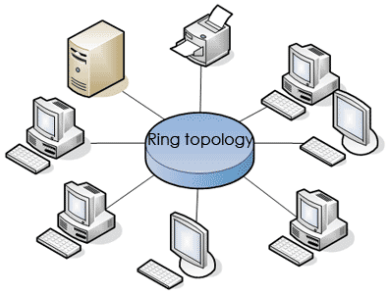


Şekil 2.1 Bus topolojisi

Bus topolojisinin avantajları; bilgisayarların ve diğer çevre birimlerinin ağa kolayca bağlanabilmesi, daha az kablo kullanılması, tasarımı ve genişletilebilirliği kolay olması, geçici amaçlı ve kalıcı olmayan ağların hızlı bir şekilde kurulabilmesi için ideal olması, switch veya hub gibi çevresel bağlantı aygıtlarının kullanılmaması ve böylece ek maliyetlerin ortadan kalkması, bir istasyonun çalışmaması durumunda diğerlerini etkilememesi, büyütülebilirlik açısından en ucuz topoloji olmasıdır.

Dezavantajları ise; sorun giderilmesi ve yönetimi zor olması, kısıtlı sayıda istasyon ve kısa mesafe kablo üzerinde olması, ana kabloda oluşan bir kopmanın tüm ağın çalışmasını engellemesi, eklenen her ilave istasyonun toplam ağ performansını kötü anlamda etkilemesi, omurga kablonun her iki ucunda sonlandırıcıların bulunma zorunluluğudur.

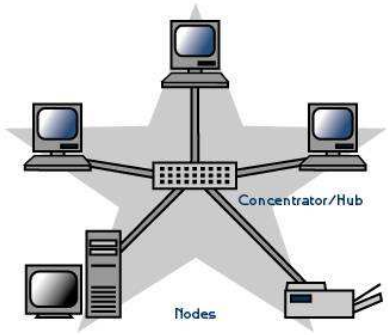
Ring Topoloji: Bu topolojide dairesel bir yapı söz konusudur. Hata gönderilen sinyaller hedefe ulaşana kadar tüm terminallere uğramaktadır. Tüm terminaller eşit haklara sahiptir. Genelde UTP korumasız çift dolanmış ya da STP(Shielded Twisted Pair) korumalı çift dolanmış kablo kullanılarak oluşturulmaktadır. Bilgisayarlarla bağlantı cihazının maksimum mesafesi 100 metredir.



Şekil 2.2 Ring topolojisi

Ring topolojisinin avantajları; ağın büyütülmesi, toplam sistem performansına çok az bir oranda olumlu etki yapmaktadır, tüm istasyonlar eşit erişim hakkına sahiptir. Dezavantajları ise; bilinen en pahalı topolojidir, oldukça karışıktır, bir istasyonun arızası durumunda tüm istasyonlar etkilenmektedir.

Star Topoloji: Star topolojide her bilgisayar switch, hub ya da başka bir server dediğimiz ağ cihazlarına direkt bağlıdır. Hata gönderilen sinyal önce switch ya da huba gelir ve buradan hedefe gönderilir.

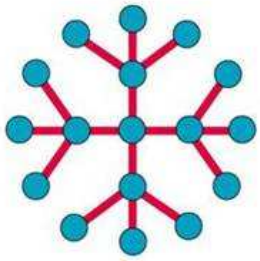


Şekil 2.3 Star topolojisi

Star topolojisinin avantajları; yeni istasyonların eklenmesi kolaydır, yönetimi ve hata tespiti basittir, birbirinden farklı kablolama metodları ile bağdaşabilir, herhangi bir istasyondaki arıza veya yeni bir birimin eklenmesi halinde bundan tüm ağ etkilenmez.

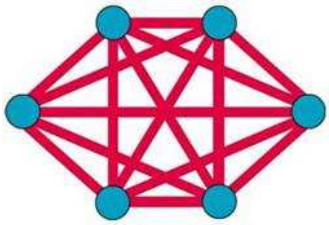
Dezavantajları; diğer topolojilere oranla, çok daha fazla kablo gereksinimi olur, hub veya switch cihazlarında ortaya çıkan sorunlarda tüm ağ etkilenmektedir, bus topolojisine göre maliyeti daha yüksektir.

Extended Star Topoloji: Star topolojinin geliştirilmesiyle ortaya çıkmıştır. Birden fazla star topolojinin bir araya gelmesiyle oluşmuş bir yapıdır. Bu yapıda kullanılan kablolama mesafesinin kısa oluşu ise bir avantaj olarak görülmektedir. Günümüzde telefon şebekelerinin yapıları bu topolojiye örnek gösterilebilir.



Şekil 2.4 Extended star topoloji

Mesh Topoloji: Ağda bulunan bütün bilgisayarlar diğer bütün bilgisayarlara direkt bağlıdırlar. Uçtan uca bütün bilgisayarlar birbirine direkt bağlı olduğu için hedefe kısa zamanda ulaşılmaktadır, iki bilgisayar arasındaki bağlantının kopması durumda alternatif yollar olacaktır.



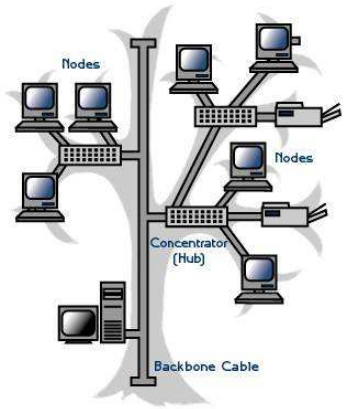
Şekil 2.5 Mesh topoloji

Bu yapının avantajları; her istasyonun kendi başına diğerleri ile uçtan uca bağlantı kurmasından dolayı, çoklu bağlantı oluşmakta ve böylece herhangi bir bağlantının kopması durumunda, sinyalin hedefine ulaşabilmesi için diğer bağlantıları kullanması en önemli avantajdır bir diğer avantajı ise bir istasyondan yayınlanan sinyal farklı hedeflere yöneldiğinde çoklu oluşan bağlantı sayesinde kısa süre içerisinde ağdaki hedeflerine varacaktır, böylece taşıma zamanı kısalacaktır.

Dezavantajları ise; ağ üzerinde az sayıda düğümün bulunduğu durumlarda ve ortam boyutunun küçük olması halinde ortaya çıkan bağlantı miktarının çok fazla gözükmesi ve bu durumda ağ hızının yavaşlaması dezavantaj olarak gösterilmektedir.

Hiyerarşik Topoloji: Üzerinde bus topoloji ve star topolojiden özellikler taşır.

Ağaç Ağ Bağlantısı: Temel olarak bus topolojisi ile star topolojisinin karakteristik özelliklerinin kombinasyonu şeklinde ortaya çıkan bir topoloji türüdür. Yıldız şeklinde bağlı istasyonların omurga üzerinde konumlanması sonucu oluşan bus modeli ağaç topolojisini oluşturmaktadır. Diğer bir yönden, ağaç topolojisi mantıksal açıdan gelişmiş star topolojisine benzemektedir. Tek farkları ise ağaç topolojisinin herhangi bir merkezi düğüme ihtiyaç duymamasıdır.

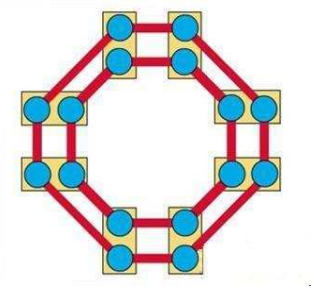


Şekil 2.6 Ağaç ağ bağlantısı

Bu yapının avantajları; her bir segment için noktadan noktaya bir kablolama yapısı kullanılmaktadır, böylece segmentlerde oluşan bir kesinti halinde diğerleri etkilenmemekte aynı zamanda birbirinden farklı donanım ve yazılım üreticilerinin sağladıkları ürünlerle uyum içerisinde çalışabilmektedir.

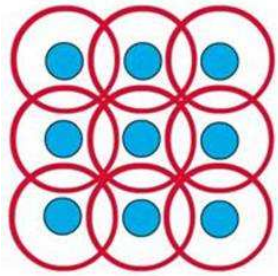
Dezavantajları ise; kullanılan kablolamanın tipine göre her bir segmentin ortalama uzunluğu belirli bir limiti geçemeyebilmekte, eğer ana omurga yapısında bir kopma olursa tüm ağ işlevini kaybetmektedir, kablolama açısından konfigürasyonu diğer tüm topolojilerden daha zordur.

Çift Halka Topolojisi: Çift halka topolojisi, birbirine eşmerkezli bir yapıda bulunan ve her bir halkanın kendi içinde birbirine bağlı istasyonlarının sadece kendisi ile komşu olan dış halkaya ait istasyon ile iletişim halinde bulunduğu bir yapıdır. Halkalar birbirine bağlı değildir ve aralarında herhangi bir sinyal alışverişi olmamaktadır. Geleneksel ring topolojisinin aynısıdır fakat birinci halkayı dıştan kuşatan ikinci bir halka bulunur ve bu dış halka sayesinde her bir istasyon kendilerine eş düzeyde bulunan diğer istasyonlar ile sinyal alışverişini sağlamaktadır. Böylece ağdaki esneklik ve güvenilirliği sağlamak üzere her aygıt kendi başlarına bağımsız olan iki halkanın ortak iletişim aygıtı haline gelmektedir.



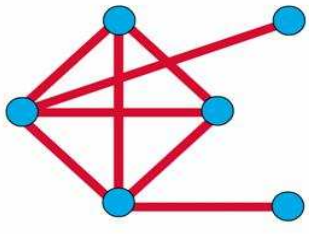
Şekil 2.7 Çift halka topolojisi

Hücresel Topoloji: Hücresel topoloji, her birinin kendi merkezi üzerinde birbirinden bağımsız düğümleri bulunan dairesel veya altıgen biçimindeki alanların oluşturduğu topoloji yapısıdır.



Şekil 2.8 Hücresel topoloji

En belirgin avantajı dünya atmosferi ve uzay boşluğu haricinde herhangi bir taşıyıcı ortamın bulunmamasıdır. Dezavantajı ise ortamda dolaşan sinyalin dinleme ve izlenmeye açık bir durumda bulunması ve bunun getirebileceği güvenlik tehditleridir.
Eğri Topoloji: Eğri topoloji, ağ bileşenleri arasında belirgin bir bağlantı şekli ve yolunun bulunmadığı, çarpık bir modelin ortaya çıktığı duruma denmektedir. Bu topolojide kablolama oldukça düzensizdir ve çok sayıdaki düğümün birçok kablo ile gelişigüzel bağlantısı ağın düşük performans sergilemesine ve güvensiz veri iletişimi yapmasına neden olmaktadır.



Şekil 2.9 Eğri topoloji

2.6.2 Mantıksal Topoloji

Ağların mantıksal topolojileri, ağ aygıtları ve istasyonların birbirleri ile nasıl iletişim kuracaklarını belirleyerek bunları ortak bir protokol çerçevesinde birleştirmektedir.
Yayın Topolojisi: Her istasyonun ağ ortamında sinyali diğer tüm istasyonlara aynı anda iletmesi kuralına dayanmaktadır. Yollayıcı, sinyali yayınladıktan sonra adresin eşleştiği istasyona bir aktarım söz konusu değildir.
Token Geçiş Topolojisi: Bu topoloji, elektronik bir token'ın her bir istasyona uğrayarak tüm ağı dolaşması esasına dayanmaktadır. Token, bir taşıyıcı görevindedir ve uğradığı her istasyon, o anda iletecek veya dağıtacak herhangi bir veriye sahip değilse tokenı bir sonraki istasyona aktarmaktadır. Böylece bir repeater görevi yapmış olmaktadır. Ağa sunulacak bir veri varsa, tokena o anda sahip olan istasyon veriyi ekleyerek dolaşıma sunmaktadır ve sinyal bu şekilde taşınmış olmaktadır.

2.7 OSI Referans Modeli

Kullanıcıların farklı talepleri ve dolayısıyla ağ üzerinde kullanılmak zorunda kalınan karmaşık uygulamalar, ağ kurulumlarında bir hiyerarşinin doğmasını gerekli kılmıştır. Bilgisayar ağları büyüdükçe bu ağları yönetmek ve sorun gidermek, standart bir yapı olmadığı göz önüne alınırsa çok daha zorlaşmaya başlamıştır. ISO(International Organization for Standardization-Uluslararası Standartlar Organizasyonu) bir çok ağ yapısını inceleyerek 1984 yılında OSI referans modelini geliştirmiştir. Donanım ve yazılım firmaları bu standarda uygun ürünler üretmeye başlamışlardır. OSI modelinde 7 katmanlı bir yapı kullanılmış ve bu model; karmaşıklığı azaltmış, insanların belli katmanlarda uzmanlaşması için referans olmuş, katmanların işlevlerinin öğrenilmesi ve öğretilmesi kolaylaşmış, farklı donanım ve yazılım ürünlerinin birbirleriyle uyumlu çalışmasını sağlamış ve bir katmanda yapılan değişiklikler diğer katmanları etkilemediği için işbirliği, görev paylaşımı, problem çözümü gibi konularda kolaylıklar getirmiştir.

OSI katmanlarını şu şekilde sıralanabilir.
Uygulama katmanı
Sunum katmanı
Oturum katmanı

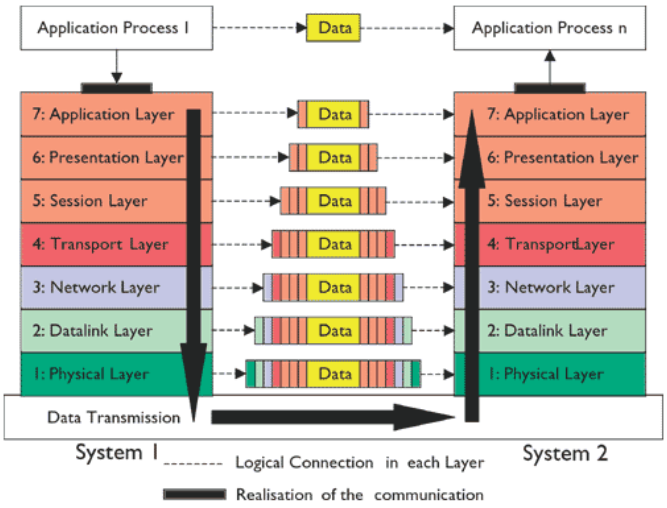
Nakil katmanı

Ağ katmanı

Veri bağlantı katmanı

Fiziksel katman

Burada uygulama, oturum ve sunum katmanları üst katmanlar olarak adlandırılırlar ve işlevlerini yazılımlar sağlamaktadır. Nakil, ağ, veri bağlantı ve fiziksel katmanlar ise alt katmanlar olarak adlandırılırlar ve işlevlerini bilgisayarların ve ağda kullanılan diğer cihazların donanımları ve bu donanımlar üzerindeki yazılımlar sağlamaktadır.



Şekil 2.10 OSI referans modeli

Uygulama Katmanı: Kullanıcıya en yakın olan katmandır ve diğer katmanlara herhangi bir servis sağlamamaktadır. Uygulama katmanı programların ağı kullanabilmesi için araçlar sunar. Microsoft API'leri uygulama katmanında çalışmaktadır. Bu API'leri kullanarak program yazan bir programcı, örneğin bir ağ sürücüsüne erişmek gerektiğinde API içindeki hazır aracı alıp kendi programında kullanmaktadır. Alt katmanlarda gerçekleşen farklı işlemlerin hiçbirisiyle uğraşmak zorunda kalmamaktadır. Kullanıcıların ağda veya yerel olarak kullandığı veya gereksinim duyduğu servislerin çalıştırılabildiği katmandır[5].

Sunum Katmanı: Gönderilecek verinin, veriyi alacak bilgisayar tarafından da anlaşılabilecek ortak bir biçime dönüştürüldüğü katmandır. Bu katmanda veri iletiminin güvenli olması için şifreleme de mümkündür. Bu katman eldeki verinin

içerik olarak belirlenmesine ve üzerine yazılacak bilgilerin bir üst veya alt katman tarafından bilinmesine yardımcı olmaktadır[5].

Oturum Katmanı: Oturum katmanı bir bilgisayar birden fazla bilgisayarla aynı anda iletişim içinde olduğunda, gerektiğinde doğru bilgisayarla konuşabilmesini sağlamaktadır. Bu katmanda çalışan NetBIOS ve Sockets gibi protokoller farklı bilgisayarlarla aynı anda olan bağlantıları yönetme imkanı sağlamaktadır[5].

Nakil Katmanı: Bu katman nakil edilecek verinin bozulmadan güvenli bir şekilde hedefe ulaştırılmasını sağlamaktadır. Üst katmanlardan gelen her türlü bilgi nakil katmanı tarafından diğer katmanlara ve hedefe ulaştırılmaktadır. Gönderilen verinin bozulmadan ve güvenli bir şekilde hedefe ulaşıp ulaşmadığını uygun protokollerle kontrol edebilmektedir. Bu katmanda çalışan protokollere TCP, UDP, NetBEUI örnek olarak verilebilir. Katmanın en önemli iki fonksiyonu güvenlilik ve akış kontroldür. Güvenlilik bilgisayarlar arasında gerçekleştirilen veri iletiminde verinin sağlıklı bir şekilde hedefe gönderilip gönderilmediğini yöneten, gönderilemediği durumlarda tekrar gönderilmesini sağlayan fonksiyondur.

İletişim halindeki bilgisayarlarda veriyi gönderen bilgisayar alıcının kapasitesinin üzerinde veri gönderebilir. Böyle bir durumda veriyi alan bilgisayar alamadığı paketleri yok edecektir ki, bunu önlemek için nakil katmanı ara bellekleme, tıkanıklıktan kaçınma ve pencereleme metotlarını kullanarak akış kontrolünü sağlamaktadır. Ara bellekleme de verinin akış hızına müdahale etmeden, kapasitenin üzerindeki verinin ara belleğe alınması, tıkanıklıktan kaçınma metodunda ICMP(Internet Control Message Protocol) kaynak kapatma mesajı ile gönderen bilgisayarın gönderimini yavaşlatması, pencereleme metoduyla paketlerin gruplar halinde gönderilmesi sağlanmaktadır[5].

Ağ Katmanı: Bu katman bir paketin yerel ağ içerisinde ya da diğer ağlar arasındaki hareketini sağlayan katmandır. Bu hareketin sağlanabilmesi için hiyerarşik bir adresleme yapısı gerekmektedir. Gelişen teknolojiyle birlikte mevcut ağlarında büyüme eğiliminde olması adresleme yapısının hiyerarşik olmasını gerektirmektedir. Ayrıca hiyerarşik sistem verilerin hedef bilgisayara en etkili ve en kısa yoldan ulaşmasını da sağlamaktadır. Bu katmanın bir özelliği olan adresleme sayesinde bu sağlanabilmiştir. Adresleme dinamik ya da statik olarak yapılabilmektedir. Sabit

adresleme el ile yapılan adreslemedir. Dinamik adreslemede ise otomatik olarak IP dağıtacak DHCP(Dynamic Host Configuration Protocol) gibi bir protokole ihtiyaç vardır. Ayrıca bu katmanda harekete geçen bir verinin hedefine ulaşabilmesi için en iyi yol seçimi de yapılmaktadır. Bu işleme yönlendirme bu işlemi yerine getiren cihaza ise Router denmektedir. Router gelen verilerin yönlendirme işlemini bu katmanda yapmaktadır. Ağ içi veri akışı için routerlarda belirlenen tablolar vardır. Bu tablolar sabit veya değişken olabilmektedir. Ağ içeriğindeki veri trafiği yine bu katman tarafından düzenlenmekte ve olası veri çakışmalarına karşı önlem alınmaktadır. IP protokolünün çalışma alanıdır[5].

Veri Bağlantı Katmanı: Fiziksel adreslemenin ve ağ ortamında verinin taşınma şeklinin tanımlandığı katmandır. Fiziksel adresleme MAC adresidir. Bu katmanın fonksiyonları hakemlik, adresleme, hata saptama, kapsüllenmiş veriyi tanımlamadır. Bu katman işlevinin büyük bir bölümü ağ kartı içinde gerçekleşmektedir. Veri bağlantısı katmanı ağ üzerindeki diğer bilgisayarları tanımlama, kablonun o anda kimin tarafından kullanıldığının tespit edilmesi ve fiziksel katmandan gelen verinin hata kontrolü görevini yerine getirmektedir.

Ethernet hakemlik için CSMA/CD algoritmasını kullanır.

Bu algoritma şu adımlardan oluşur;

- Hattın boş olup olmadığını dinler
- Boşsa veri gönderir
- Doluysa bekler ve dinlemeye devam eder
- Veri iletiminde çarpışma olursa durur ve tekrar dinlemeye başlar.

Fiziksel Katman: Fiziksel katman verinin kablo üzerinde alacağı fiziksel yapıyı tanımlar. Diğer katmanlar 1 ve 0 değerleriyle çalışırken, 1. katman 1 ve 0 değerlerinin nasıl elektrik, ışık veya radyo sinyallerine çevrileceğini ve aktarılacağını tanımlamaktadır. Gönderen tarafta 1. katman 1 ve 0'ları elektrik sinyallerine çevirip kabloya yerleştirirken, alıcı tarafta 1. katman kablodan okuduğu bu sinyalleri tekrar 1 ve 0 haline getirmektedir.

Fiziksel katman veri bitlerinin karşı tarafa, kullanılan ortam (kablo, fiber optik, radyo sinyalleri) üzerinden nasıl gönderileceğini tanımlamaktadır. İki tarafta aynı kurallar üzerinde anlaşmamışsa veri iletimi mümkün değildir. Üreticiler (örneğin ağ kartı

üreticileri) bu problemleri göz önüne alarak aynı değerleri kullanan ağ kartları üretmektedirler. Böylece farklı üreticilerin ağ kartları birbirleriyle sorunsuz çalışmaktadır. Kablolar, hub, repeater cihazlar bu katmanda yer alırlar. Bu katmanda herhangi bir protokol tanımlanmamıştır[5].

2.8 TCP / IP Katmanları

Günümüzde internetin temel protokolü olarak yerini almış TCP/IP'nin açılımı Transmission Control Protocol/Internet Protocol'dür. TCP/IP modeli OSI katmanlarından çok daha önce standartlaştığı için OSI içinde referans olmuş 4 katmanlı bir yapıdır.

Uygulama Katmanı: OSI modelindeki uygulama, oturum ve sunum katmanlarına karşılık gelmekte ve o katmanların işlevlerini yerine getirmektedir. Bu katmanda TFTP(Trivial File Transfer Protocol), FTP(File Transfer Protocol), SMTP(Simple Mail Transfer Protocol), SNMP gibi protokoller çalışmaktadır.

Nakil Katmanı: OSI modelindeki nakil katmanıyla bire bir eşleştirilebilmektedir. Bu katmanda iki farklı sınıfa ayrılabilecek iki protokol kullanılır: TCP ve UDP

- Bağlantı Odaklı: TCP
- Bağlantısız: UDP

İnternet Katmanı: OSI modelindeki ağ katmanına denktir ve adresleme, en iyi yol seçimi gibi işlevleri yerine getirmektedir.

Ağa Giriş Katmanı: OSI modelinde ki veri bağlantı ve fiziksel katmana denk gelmektedir.

2.9 Projenin OSI Modeline Göre Açıklanması

Uygulama Katmanı: Windows işletim sistemindeki perfctrs.dll dosyasının içerisinde bulunan API'lerin kullanıldığı katmandır. Bu API'ler, işletim sisteminin kullanıcıya sunduğu hazır araçlar, kullanılarak erişilen bilgisayarlardan belirlenen performans bilgileri toplanmaktadır.

Sunum Katmanı: Bu katmanda uygulama katmanından gelen veriler uygun başlıkları eklenip şifrelenerek oturum katmanına yollanır.

Oturum Katmanı: Bu katmanda projede erişilecek her bilgisayar için RPC (Remote Procedure Call) paketleri oluşturulur. Oturum katmanı oluşturulan RPC paketlerinin farklı amaçlarla kullanılan ağ trafiğine karıştırmadan paket trafiğinin düzenlenmesinden sorumludur.

Taşıma Katmanı: Oluşturulan RPC paketleri TCP veya UDP protokolleri aracılığı ile parçalara bölünür ve bu parçalar numaralandırılarak TCP veya UDP protokol başlığı eklenir ve ağ katmanına gönderilir.

Ağ Katmanı: Bu katmanda parçalara ayrılan paketlere gönderilecek ve gönderen bilgisayarın IP bilgileri başlık olarak eklenir ve bu haliyle veri bağlantı katmanına gönderilir.

Veri Bağlantı Katmanı: Ağ katmanından gelen paketlere kendinin ve bir sonraki düğümün MAC adresi bilgilerini ekleyerek verilerin bir sonraki düğüme hatasız aktarılmasını sağlar.

Fiziksel Katman: Bu katmanda, oluşturulan paketler fiziksel olarak taşınır.

2.10 Sanal Yerel Alan Ağı (VLAN)

Bir yerel alan ağı üzerindeki ağ kullanıcılarının ve kaynakların mantıksal olarak gruplandırılması ve switch üzerinde port'lara atanmasıyla yapılmaktadır. VLAN kullanılmasıyla her VLAN sadece kendi broadcast'ini alacağından, broadcast trafiği azaltılarak bant genişliği artırılmış olur. VLAN tanımlamaları, bulunulan yere, bölüme, kişilere ya da hatta kullanılan uygulamaya ya da protokole göre tanımlanabilmektedir. VLAN ağı birbirinden bağımsız segmentlere ayırma imkanı sağlayarak, riskleri önleme, karmaşıklığı ortadan kaldırma ve problem çözme konusunda büyük ölçüde fayda sağlamaktadır. Ağ yapısına hiyerarşi kazandırma yönünden en yararlı araçlardan birisidir[6]. VLAN oluşturulan bir switchte hostlar tarafından gönderilen broadcast paketleri sadece aynı VLAN'a dahil olan portlara gönderilir. Bu yüzdendir ki her VLAN kendi içerisinde bir broadcast domaindir. Bu

özellik sayesinde kaynağı bilinmeyen unicast paketlerde sınırlanmış olmaktadır. Bütün bunlar VLAN'ın kullanımını kaçınılmaz kılmıştır[4].

3. PROJEDE KULLANILAN TEKNOLOJİ VE KAVRAMLAR

3.1 Projede Kullanılan Teknolojiler

3.1.1 Microsoft Visual Studio 2008

Microsoft firması tarafından sunulan, Windows, web ve veri madenciliği uygulamaları geliştirilmesini sağlayan yazılım platformudur. NET çatısı, yazılım geliştiriciler için zengin uygulamalarının geliştirilebilmesi için temiz, nesneye dayalı ve genişletilebilir sınıf kümelerini sunar. Uygulamalar, çok katmanlı dağıtık çözümlerde yerel kullanıcı arayüzü gibi davranır[7].

3.1.2 Microsoft SQL Server 2008

SQL(Structured Query Language) serverı iki tür veritabanını yönetmek için kullanılmaktadır. Bunlar OLTP (Online Transaction Processing) veritabanları ve OLAP (Online Analytical Processing) veritabanlarıdır. Genel olarak farklı istemciler ağ üzerinden haberleşerek veritabanlarına erişmektedirler. SQL Server ile terabyte boyutundaki veritabanları yönetilebilir. Birden fazla server arasında Windows Clustering yaparak SQL Server kullanılabilmektedir. Microsoft SQL Server 2008 veritabanı yönetim sistemi, ilişkisel veri tabanı sistemidir. Express, Express Advanced, Web, Workgroup, Standard, Enterprise ve Development sürümleri mevcuttur[8].

Veri saklama modelleri: SQL Server OLTP ve OLAP veritabanları yönetebilmektedir.

OLTP Veritabanları: Bir OLTP veritabanı içinde veriler genellikle ilişkisel tablolar içinde organize edilmektedir. Bu gereksiz veri yığınları azaltmakta ve veri güncelleme hızını arttırmaktadır. SQL Server çok sayıda kullanıcının gerçek zamanlı olarak veri analiz edebilmesini ve güncellemesini sağlamaktadır. Örnek olarak OLTP veritabanları havayolu bilet satış bilgileri ve bankacılık işlemlerini içerir.

OLAP Veritabanları: OLAP teknolojisi büyük verilerin organize edilmesi ve incelenmesini sağlamaktadır. Örneğin bir analist büyük verileri hızlı ve gerçek

zamanlı olarak değerlendirebilmektedir. SQL Server Analiz Servisi toplu raporlama ve analizde, veri modelleme ve karar desteğe kadar geniş alanda çözümler sunmaktadır[7].

3.2 Projede Kullanılan Kavramlar

3.2.1 Thread

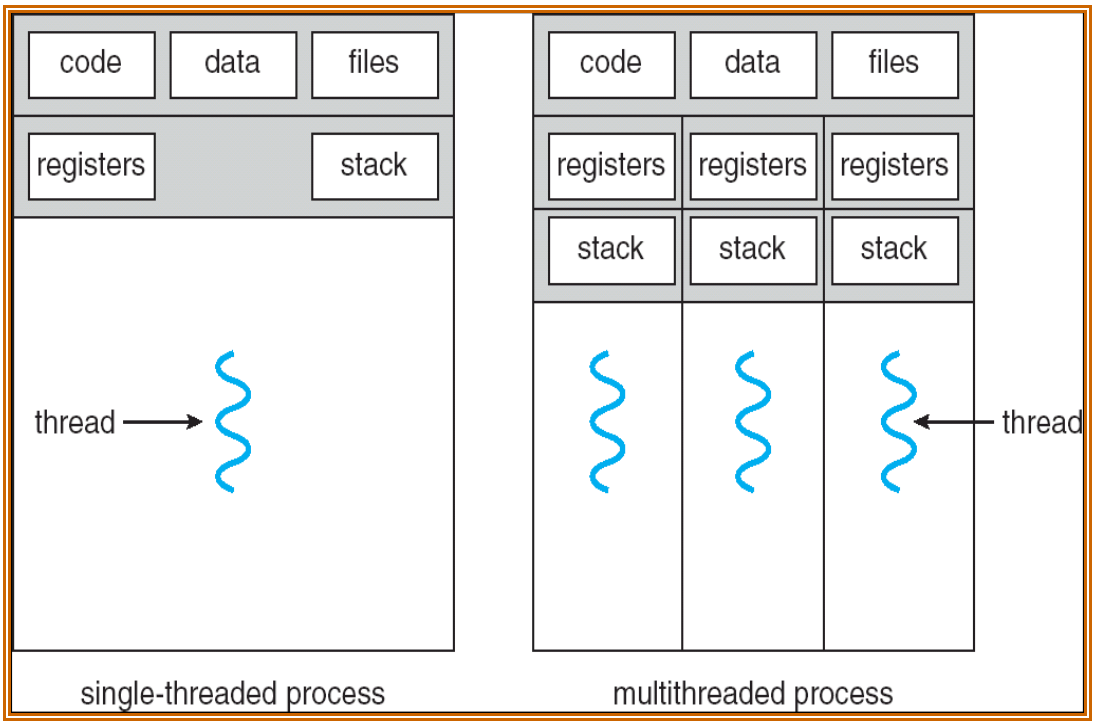
Aynı anda birden fazla program çalıştırılıp birden fazla işlem yapılırken işlemcinin kaynaklarını uygulamalar arasında paylaştırmayı işletim sistemi gerçekleştirmektedir. İşletim sistemi nazarında çalışan her uygulama bir işlemdir ve işlemler Windows'un görev yöneticisi penceresinde listelenmektedir. Uygulamalar dahilinde birbirine paralel yapılması istenen birden fazla işlem varsa bu işlemleri threadlerle gerçekleştirmek gerekmektedir. Başka bir deyişle gerçekte bir işlem olan uygulama dahilinde birden fazla işlemi birbirine paralel yaptırtma gereği duyuluyorsa 2. veya 3. bir kanal yani thread oluşturmak gerekmektedir.

Gerçekte çalışan uygulamaların tüm işlemleri aynı anda gerçekleşmemektedir. İşlemcinin yaptığı, çalıştırılan uygulamaya ait işlemleri iş parçacıkları (thread) halinde ele almaktır. Her bir iş parçacağı bir işlemin birden fazla parçaya bölünmesinden oluşmaktadır. İşlemciler her iş parçacığı için bir zaman dilimi belirlemektedir. T zaman diliminde bir işlem parçacığı yürütülmekte ve bu zaman dilimi bittiğinde işlem parçacığı geçici bir süre için durmaktadır. Ardından kuyrukta bekleyen diğer iş parçacağı başka bir zaman dilimi içinde çalıştırılmaktadır. Bu böyle devam ederken, işlemci her iş parçacığına geri dönmekte ve tüm iş parçacıkları sıra sıra çalıştırmaktadır[9].

Threadler sayesinde ayrı metodlar aynı anda çalıştırılabilmektedir. İşletim sistemi her threade çalışması için belli bir zaman aralığı vermektedir. Bu zaman aralığı dolduğunda çalışan threadden çıkılıp, program içerisindeki diğer bir metoda veya başka bir threade girilmektedir. Bir metod içerisinde yapılan iş ne kadar uzun sürüyorsa o metodun bağlı olduğu threade o kadar fazla zaman harcanmaktadır. Başlatılan threadlerin aynı anda eş zamanlı olarak çalışabilmesi için .NET'te senkronizasyon teknikleri kullanılaktadır[8].

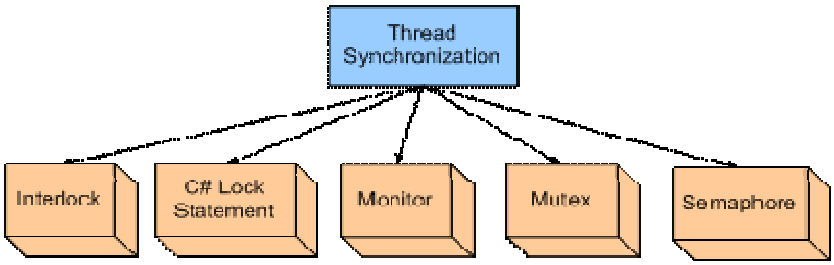
Bir C# uygulaması iki şekilde multithreaded olabilir; ya açık bir şekilde threadleri yaratıp çalıştırmakla olabilir ya da .NET framework'ünün BackgroundWorker, thread pooling, threading timer, remoting server, ASP.NET uygulaması, web servisleri gibi threadlerin üstü kapalı bir şekilde üretilip çalıştırıldığı durumlarda olabilir.

Multithreading dezavantajlarla birlikte gelir. En büyüğü çok karmaşık programlara neden olabilmesidir. Multiple threadlerin olması kendisi karmaşıklık yaratamaz; threadler arasındaki etkileşim karmaşıklığı yaratır. Etkileşimin kasıtlı olup olmaması ve uzun geliştirme döngülerine neden olabilir, ara ara olabilen ve geri döndürelemez hatalara karşı korunmasız olur. Multithreading eğer aşırı bir şekilde kullanılırsa aynı zamanda kaynak ve CPU maliyetiyle birlikte gelir. Bu nedenle threadler dikkatli kullanılmalıdır[10].



Şekil 3.1 Single and multithreaded processes[11]

Thread Senkronizasyon Teknikleri



Şekil 3.2 Thread senkronizasyon teknikleri

Interlocked sınıfı birden fazla kanal tarafından kullanılan değişkenler üzerinde çeşitli işlemlerin yapılmasına olanak sağlayan bir sınıftır. Buradaki değişken değer tipinde olabileceği gibi herhangi bir nesne de olabilmektedir. Interlocked sınıfının sadece Exchange metodu referans tipleri ile çalışabilmektedir. Exchange'in generic metot olarak tasarlanmış bir overload'u da mevcuttur. Increment, decrement metodları ise Int32, Int64 tipi değişkenleri parametre olarak kabul eder[12].

Lock Bir kanal içersindeki işlemlerin, diğer bir kanal tarafından müdahale edilmeden çalışabilmesi için lock anahtar kelimesi kullanılmaktadır. Lock ile bloklanmış olan işlemler bir kanal içersinde tamamlanıncaya kadar çalışırlar. Bu süre zarfında başka kanallar lock ile bloklanmış işlemlerin bitmesini beklerler. İşlem bitince diğer kanallar kendi metodlarını çalıştırmaya devam etmektedir. Bu şekilde senkronizasyon sağlanabilir. Lock parametre olarak herhangi bir nesne alabilir.

Monitor nesnelere kanalların senkronize bir şekilde ulaşmasını sağlayan bir sınıftır. Monitor sınıfı referans tipindeki değişkenleri senkronize etmek için kullanılmaktadır. Değer tipindeki değişkenler için monitor sınıfı kullanılmamaktadır. Monitor'ün kullanımı lock'un kullanımına benzemektedir. Monitor blokladığı kodların başka kanallar tarafından erişilmesini engellemektedir[12].

Mutex sınıfı monitor sınıfına benzer ancak System.Threading.WaitHandle sınıfından türetilmiştir ve türediği sınıfın daha kolay kullanılabilir bir genelleştirilmesidir. Mutex sınıfı kanallar tarafından ortak kullanılan nesnelere aynı anda ulaşılıp, işlem yapılmasını engellemek için kullanılmaktadır. Mutex sınıfı ortak kullanılan kaynaklara bir t zamanında sadece bir kanalın ulaşabilmesini garanti etmektedir. Mutex kendisini kullanan kanalın tekilliğini (identity) kontrol etmektedir. Bir

mutex'e sahip olan kanal WaitOne metodu ile onu kilitlemekte ve ReleaseMutex metodu ile mutex'i serbest bırakmaktadır. Bir mutex kullanan kanal sadece kendi mutex'ini ReleaseMutex metodu ile açabilmektedir[12].

Semaphore sınıfı .NET 2.0 framework ile gelen yeni bir sınıftır. Bu sınıf System.Threading.WaitHandle sınıfından türetilmiştir. Semaphore sınıfının Mutex sınıfından farkı, farklı kanalların birbirlerinin Semaphore'larının kilitlerini Release metodu ile açabilmeleridir. Bir kanal semaphore'un WaitOne metodunu birçok kez çağırabilmektedir. Bu kilitleri açmak için art arda Release metodunu çağırabileceği gibi, Release(int) overload'unu da kullanabimektedir. Semaphore kendisini kullanan kanalın identity'sine bakmamaktadır. Bu yüzden farklı kanallar birbirlerinin semaphore'larının WaitOne ve Release metodlarını çağırabilmektedir. Herbir WaitOne metodu çağırıldığında semaphore'un sayacı bir azaltılmaktadır. Herbir release metodu çağırıldığında ise sayaç bir arttırılmaktadır. Semaphore'un yapılandırıcısında sayacın minimum ve maksimum değerleri belirlenebilmektedir[12].

Çok kanallı programlamada karşılaşılan en büyük sorun kısırdöngü denilen programın kilitlenmesine yol açan kodlardır. Kısır döngülerden kurtulmak için ortak kullanılan değişkenler ve nesneler yukarıda anlatılan teknikler ile izole edilmelidir. Böylece bir proses içersinde birçok kanalda birçok metod paralel bir şekilde çalıştırılabilinir[12].

3.2.2 Ping Kavramı

Ping komutu network mühendislerine bir çok problemin teşhisinde yardımcı olmaktadır**.** Ping komutu ile ping isteğini gönderen cihaz ICMP Echo Request'te bulunmaktadır. ICMP mesajlarındaki Echo Request tipi 8 ve kodu 0'dir. Hedef IP adresi Echo Request mesajını aldığında gönderen cihaza Echo Reply ICMP mesajını gönderir. Bu mesajın tipi 0 ve kodu da 0'dir[4].

Ping paketi gönderilen makinenin o anda çalışmakta olduğunu teyid etmektedir. Ağın o anki paket kayıp oranı hakkında bir bilgi verebilmekte ya da kaynak makine ile karşı makine arasındaki iletişimin süresini gösterebilmektedir[13].

3.2.3 SYSTEM.DIAGNOSTICS

Network Performans adı altında incelenen ve geliştirilen programlar C#'ın Ağ ile ilgili gerekli namespace ve sınıfları incelemeyi, araştırmayı gerekli kılmıştır. Bu namespace System.Diagnostics ve sınıf performans counter sınıfıdır. System.Diagnostics namespace'i system process, event logs ve performans counter ile etkileşimli olunmasına izin veren sınıflar sağlamaktadır. Burada proje kapsamında irdelenmesi gereken System.Diagnostics namespeces'i altındaki sınıf performans counter sınıfıdır[14].

Performance counter

.NET uygulamaları içinde performance counter verilerine erişilebilmesine imkan tanımaktadır. Counterlar işletim sistemi veya bir uygulama, hizmet veya sürücünün ne kadar iyi performansla çalıştığı hakkında bilgi vermek için kullanılmaktadırlar. Counter verileri sistem darboğazları, ince ayar sistemi ve uygulama performansını belirlemeye yardımcı olabilmektedir. Performans Counter sınıfları mevcut yüksek performanslı sağlayıcıları tarafından hesaplanan sistem performans verilerine erişime izin vermektedir. Performance counter sistem üzerinde çalışan tüm işlemleri, cpu kullanımını, memory kullanımını, sql server durumu, IIS durumu, servisler ve yönetici araçları seçeneklerinin kontrol edilebileceği ve anlık verilerin alınabileceği bir bileşendir. Microsoft Windows Task Manager'da görülebilen tüm sistem istatistiklerine bu bileşen sayesinde erişilmektedir. Yukarıda açıklanan System.Diagnostics namespacesi altındaki performance counter sınıfı bu proje kapsamında erişilen bilgisayarın CPU kullanım, kalan RAM miktarı ve ağ kullanım bilgilerine yani performans verilerine erişmeye imkan sağlayan sınıftır[14]. Performance counter sınıfı farklı kuruculara sahiptir, bunlar Tablo 3.1'de belirtilmiştir. Projede System.Diagnostics name spacesi altındaki performance counter sınıfının NextValue methodu geçerli counter'ın o anki hesaplanmış değerini döndürür. Hesaplanmış değer proje kapsamında belirlenen performans parametrelerinin erişilen bilgisayardaki o anki değeridir. Bu değerler rapor oluşturmada temeldir. Bu nedenle NextValue metodu proje için temel oluşturan fonksiyondur.

Tablo 3.1 Performance counter kurucuları

PerformanceCounter()	Performance counter sınıfının yeni, salt okunur bir nesnesini yaratır. Bu nesneyi sistem ya da özel bir performance sayacı ile ilişkilendirmez.
PerformanceCounter(String, String)	Performance counter sınıfının yeni, salt okunur bir nesnesini yaratır ve nesneyi sistem ya da özel bir yerel bir bilgisayardaki performans sayacı ile ilişkilendirir. Bu kurucu tek bir örneği olan kategoriye ihtiyaç duyar.
PerformanceCounter(String, String, Boolean)	Performance counter sınıfının yeni, salt okunur ya da okunur yazılır bir nesnesini nesneyi sistem ya da özel bir yerel bir bilgisayardaki performans sayacı ile ilişkilendirir. Bu kurucu tek bir örneği olan kategoriye ihtiyaç duyar.
PerformanceCounter(String, String, String)	Performance counter sınıfının yeni, salt okunur bir nesnesini yaratır ve nesneyi özel bir sistem ya da özel bir yerel bilgisayardaki kategori örneği ya da performans sayacı ile ilişkilendirir.
PerformanceCounter(String, String, String, Boolean)	Performance counter sınıfının yeni, salt okunur ya da okunur yazılır bir nesnesini yaratır ve nesneyi özel bir sistem ya da özel bir yerel bilgisayardaki kategori örneği ya da performans sayacı ile ilişkilendirir.
PerformanceCounter(String, String, String, String)	Performance counter sınıfının yeni, salt okunur ya da okunur yazılır bir nesnesini yaratır ve nesneyi özel bir sistem ya da özel bir belirli bilgisayardaki kategori örneği ya da özel performans sayacı ile ilişkilendirir.

Parametreler:

Tablo 3.2'de proje dahilinde kullanılan kurucunun parametre açıklamaları verilmiştir.

Tablo 3.2 Performance counter kurucuları parametre açıklamaları

categoryName	Performans sayacının ilişkilendirildiği performans kategorisinin örneğidir.
counterName	Performance Counter'ın ismidir.
instanceName	Performans sayacının kategori örneğinin ismi ya da eğer kategori sadece bir örnek içeriyorsa boş string ("")'dir.
machineName	Performans sayacının olduğu ve ilişkilendirdiği kategorinin olduğu bilgisayardır.

Tablo 3.3 Performance counter sınıfından oluşacak nesnelerin özellikleri

CanRaiseEvents	Bileşen bir olay oluşturup oluşturmayacağına ait değeri alır (Component'ten kalıtılmıştır.)
CategoryName	O performans sayacı ile ilgili performans sayacı kategorisinin adını alır veya döndürür.
Container	Componenti kapsayan IContainer'ı alır. (Component'ten kalıtılmıştır.)
CounterHelp	Performans sayacının anlatımını alır.
CounterName	Performance Counter örneğiyle ilişkilendirilmiş olan performans sayacının adını alır veya döndürür.
CounterType	İlişkilendirilmiş performans sayacının sayaç tipini alır.
DesignMode	Componentin tasarım modunda olup olmamasını işaret eden değeri alır. (Component'ten kalıtılmıştır.)
Events	Bu Componente eklenmiş durum yakalayıcıların listesini alır. (Component'ten kalıtılmıştır.)
InstanceLifetime	Bir processin yaşam ömrünü alır veya döndürür.
InstanceName	Bu performans sayacının örnek adını alır veya döndürür.
MachineName	Bu performans sayacı için bilgisayar adını alır veya döndürür.
RawValue	Bu sayaç için ham işlenmemiş değeri alır veya döndürür.

ReadOnly	Performans sayacı örneğinin salt okunur modda olup olmadığını alır veya döndürür.
Site	Componenet'in ISite'ını alır veya döndürür. (Component'ten kalıtılmıştır.)

Tablo 3.4 Performance counter sınıfından oluşacak nesnelerin metodları

BeginInit	Form tarafından kullanılan ya da başka bir bileşen tarafından kullanılan performans sayacı örneğinin oluşturumunu başlatır. Bu oluşturma çalışma zamanında yapılır.
Close	Performans sayacını kapatır ve bu performans sayacı örneği tarafından kullanılan tüm kaynakları geri verir.
CloseSharedResources	Sayaçlar tarafından paylaşılmış performans counter kütüphanesini sisteme geri verir.
CreateObjRef	Uzaktaki bir nesneyle iletişimde bulunmak için gerekli olan Proxy'i üretmek için gerekli olan tüm bilgileri kapsayan bir nesne yaratır. (MarshalByRefObject'ten kalıtılmıştır.)
Decrement	Etkin bir atomik işlemle ilişkili performans sayacını azaltır.
Dispose	Component tarafından kullanılan tüm kaynakları sisteme geri verir. (Component'ten kalıtılmıştır.)
Dispose(Boolean)	Component tarafından kullanılan yönetilmeyen kaynakları bırakır ve seçeneksel olarak yönetilmiş kaynakları da bırakır. (Component'ten kalıtılmıştır.)
EndInit	Bir form ya da başka bir bileşen tarafından kullanılan PerformansCounter örneğinin oluşumunu önler. Bu çalışma zamanında olur.
Equals(Object)	Belirlenen nesnenin şimdiki nesneye eşit olup olmadığına karar verir. (Component'ten

	kalıtılmıştır.)
Finalize	Yönetilmemiş kaynakları bırakır ve bileşen garbage collection tarafından işlenmeden temizleme işlemine girer. (Component'ten kalıtılmıştır.)
GetHashCode	Belirli bir tip için hash fonksiyon görevi görür. (Object'ten kalıtılmıştır.)
GetLifetimeService	Servis nesnesinin şimdiki ömür süresini servisini alır. (MarshalByRefObject'ten kalıtılmıştır.)
GetService	Component ya da onun Container'ı tarafından sağlanan servisi temsil eden nesneyi döndürür. (Component'ten kalıtılmıştır.)
GetType	Şimdiki örneğin tipini alır (Object'ten türetilmiştir.)
Increment	Etkin atomik bir işlemle ilişkilendirilmiş performans sayacını bir arttırır.
IncrementBy	İlişkilendirilmiş performans sayacını belirtildiği şekilde etkin atomik bir işlemle arttırır ya da azaltır.
InitializeLifetimeService	Örneğin yaşam süresi politikası için yaşama süresi servisi elde edilir.
MemberwiseClone	Geçerli objenin yüzeysel bir kopyasını oluşturur. (Objeden kalıtılmıştır.)
MemberwiseClone(Boolean)	Geçerli MarshalByRefObject objesinin yüzeysel bir kopyasını oluşturur. (MarshalByRefObject objesinden kalıtılmıştır.)
NextSample	Counter örneğini alır ve bunun için ham ya da hesaplanamamış bir değer döndürür.
NextValue	Counter örneğini alır ve bunun için hesaplanmış bir değer döndürür.
RemoveInstance	PerformanceCounter nesnesinin InstanceName özelliği tarafından oluşturulan kategori nesnesini siler.

3.2.4 CPU

İşlemci, bilgisayarın beyni niteliğindeki en önemli bileşendir. Diğer aygıtlardan gelen verileri matematiksel işlemler yardımı ile işler, sonuca ulaşır ve sonucu gerekli yerlere gönderir. Çalışabildikleri maksimum saat hızları yani frekansları sınıflandırılmalarına yardımcı en önemli kriterdir. Hesaplaması veya karar verilmesi gereken her şeyde işlemci devreye girer. Hesaplama işlemlerinde aritmetik, karar verilmesi gereken işlemlerde de mantık ünitesi devreye girer. İşlemcilerin yapısı hiç bir mekanik parçası olmayan tamamen devreler ve transistörlerden oluşur. İçlerinde milyonlarca transistör bulunur ve bu transistorlerin sayısı ne kadar fazla olursa işlemci o kadar hızlı olur. İşlemcilerin hızları MHz (MegaHertz) cinsinden ölçülür. Bu sayı ne kadar yüksek olursa, hızı da o kadar yüksek olur.
CPU utilization: Harddiskten diğer birimlerin arayüzüne veri aktarılırken bazı sistem kaynakları kullanılmaktadır. Bunların en kritik olanı işlemler gerçekleştirilirken ne kadar CPU kullanıldığıdır. CPU kullanımı ne kadar yüksekse diğer görevler için ayrılan CPU gücü daha az olmaktadır[15].

3.2.5 RAM

İşletim sisteminin, çalışan uygulama programlarının veya kullanılan verinin işlemci tarafından hızlı bir biçimde erişebildiği yerdir.
İşletim sistemi işlem yapacağı zaman, istenilen veriler bellekte yazılı oldukları adreslerden geri alınırlar. Bellek adreslerine hızlı bir şekilde ulaşılması sistemin genel performansını olumlu yönde etkiler. RAM'ler birbirinden tamamen bağımsız hücrelerden oluşur. Bu hücrelerin her birinin kendine ait sayısal bir adresi vardır. Her hücrenin çift yönlü bir çıkışı vardır. Bu çıkış veri yolunda mikroişlemciye bağlıdır. Bu adresleme yöntemiyle RAM'deki herhangi bir bellek hücresine istenildiği anda diğerlerinden tamamen bağımsız olarak erişilebilir. Rastgele erişimli bellek adı buradan gelmektedir. RAM'de istenen kaynak ya da hücreye anında erişilebilir.

RAM miktarının az olması işletim sistemi ve üzerinde çalışan uygulamalara daha az miktarda fiziksel bellek ayrılmasına neden olacaktır. Ve bu durum mikroişlemcili bir sistemin işlem kapasitesini düşürmesine neden olmaktadır[16].

3.2.6 Ağ Kullanımı

Ağ kullanımı, o anki kullanılan ağ trafiğinin portun kullanabileceği maksimum trafiğe oranını gösterir. Bu da o an kullanılan bant genişliğinin açıklamasıdır. Ağ kullanımı yüksekse ağın meşgul olduğu, düşükse boşta olduğunu göstermektedir. Ağ kullanımı normal değerleri aşarsa bu iletim hızının düşmesine kesintilerin olmasına ve isteklerin gecikmesine yol açabilir[17].

4. AĞ İZLEME VE PERFORMANS DEĞERLENDİRME

Ağ izleme; belirlenen parametrelere göre ağın sistematik olarak takip edilmesi ve toplanan bilgilerin veri tabanına kayıt edilmesi; performance değerlendirme ise toplanan verilerden anlamlı raporlar oluşturulması, oluşturulan raporların kurumun ağ yapısını değerlendirmek amacıyla kullanılması esasına dayanmaktadır.

Projede ulaşılan her bilgisayardan ağ kullanımı, CPU kullanım ve kalan RAM miktarı bilgileri alınmakta; raporlar oluşturmak üzere veri tabanına kaydedilmektedir. Bu verilerle oluşturulan raporlar değerlendirilerek kurumun ağ yapısı içerisindeki kaynak darboğazları takip edilebilir, ağ yapısı içerisindeki tüm makinaların performansını olumsuz etkileyebilecek kaynak takipleri yapılabilir ve bu gözlemlerin sonucunda olası yaşanacak problemlere karşı önlemler alınabilir.

4.1 Veri Tabanı

Proje dahilinde oluşturulan veri tabanı üç temel tablodan oluşan sade bir yapıdadır. Aşağıda proje dahilinde veritabanında kayıtlı olan tablolar ve açıklamaları verilmiştir.

Şekil 4.1 İstisna tablosu

Projede erişilen bilgisayarlar donanımsal ya da yazılımsal bazı problemlerden dolayı istenen sonuçları döndüremeyebilmektedir. Bu durum programın o bilgisayarda çok uzun süre takılmasına ve böylece sistemin optimum sürede taranmasına engel olabilmektedir. Optimum sürede sistemin taranması doğru sonuçlar elde edilmesi ve dolayısıyla doğru tespitler yapılması için kritik önem teşkil eder. Bunu sağlamak için belirlenen bu bilgisayarlar istisna tablosuna eklenmektedir. Program çalıştığında öncelikle bu liste taranmakta ve bu listedeki bilgisayarların dışındaki bilgisayarlara erişilip bilgileri toplanmaktadır.

taramaId	int	☐
ipAddress	varchar(15)	☐
bilgisayarAdi	varchar(100)	☑
pingSuresi	numeric(8, 3)	☑
threadCalismaSuresi	numeric(8, 3)	☑
networkUtil	numeric(8, 3)	☑
cpuYuzdesi	numeric(8, 3)	☑
kalanRam	numeric(8, 3)	☑
kayitZamani	datetime	☑
		☐

Şekil 4.2 Ağ değerlendirme tablosu

Ağ değerlendirme tablosu sistem için temel tablo konumundadır. Raporlamada kullanılan veriler bu tablodan gelmektedir. Bütün ağ periyodik olarak taranmaktadır ve tarama sırasında ulaşılan bilgisayarlardan alınan belirlenen parametreler ağ değerlendirme tablosuna kayıt edilmektedir.

taramaId	int	☐
makinaSayisi	int	☑
threadSayisi	int	☑
calismaSuresi	varchar(50)	☑
		☐

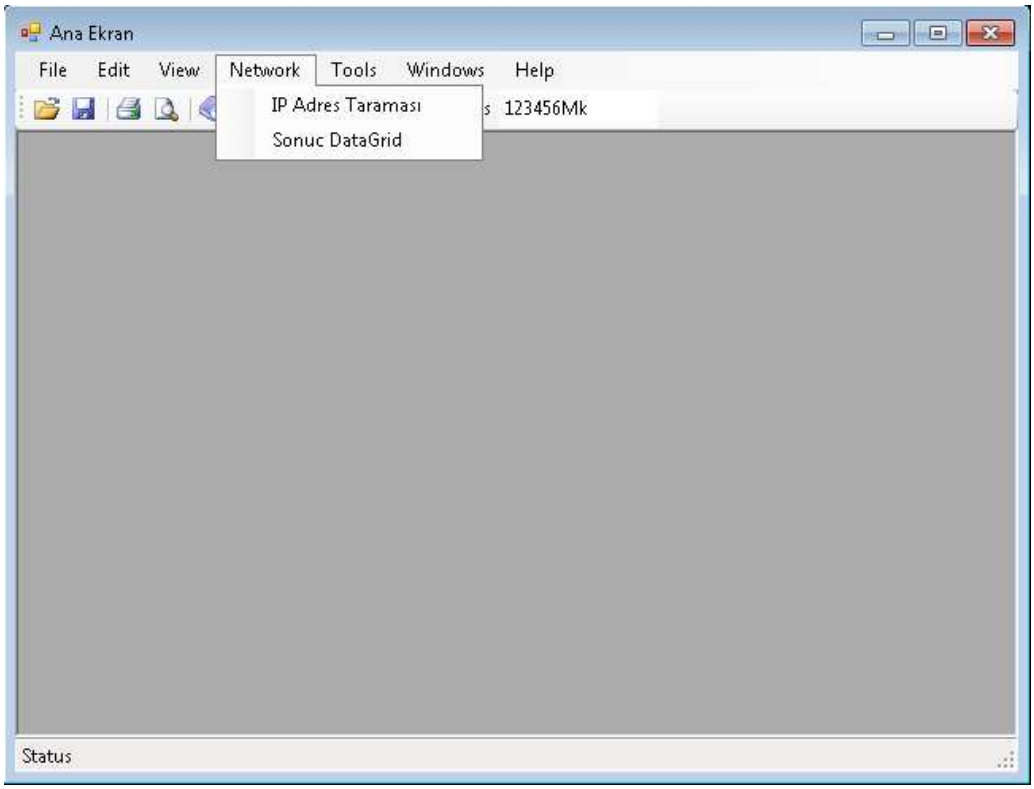
Şekil 4.3 Thread sayısı tablosu

Thread sayısı tablosu programın çalışması için uygun thread sayısının belirlenmesi için oluşturulmuş bir tablodur.

Bu tabloda tarama yapılırken, belirlenen thread sayısına göre kaç makinanın ne kadar sürede tarandığı bilgisi kaydedilmektedir. Bu verilere bakılarak proje ve projenin çalıştığı makine için çalışacak en uygun thread sayısı belirlenebilmektedir.

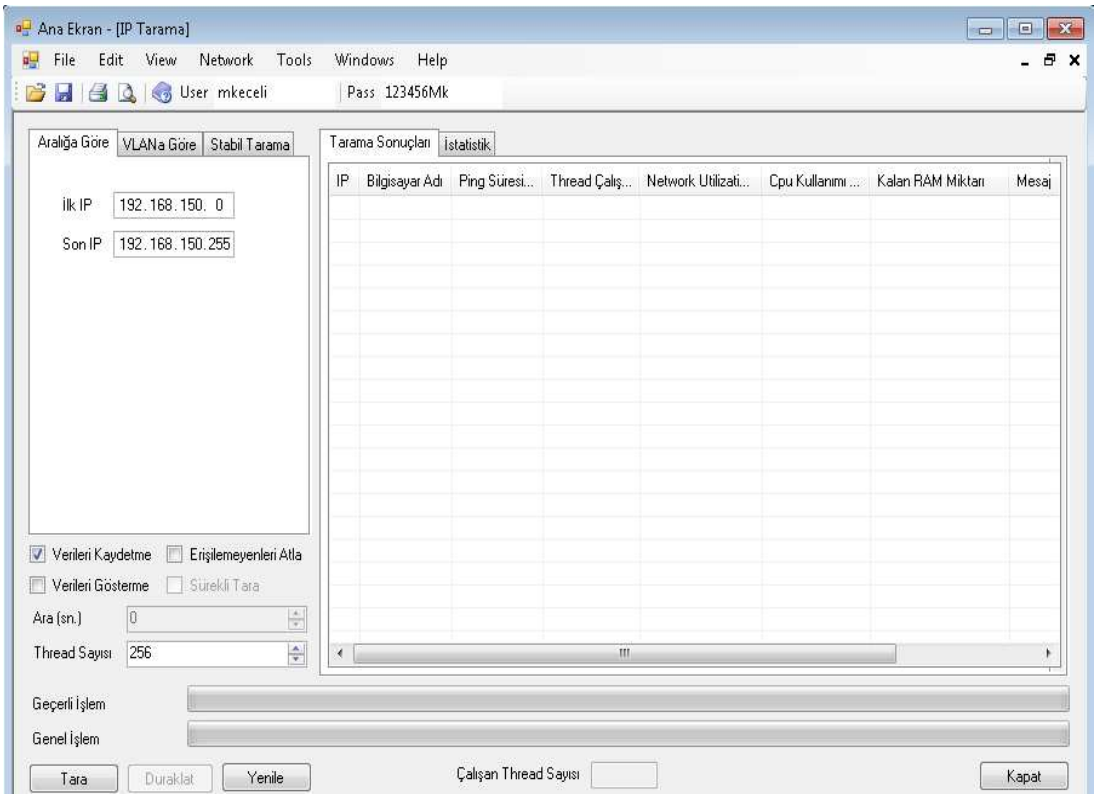
4.2 Ekranlar

Şekil 4.4 de programı ilk çalıştırdığımızda karşımıza çıkacak olan genel program ekranı verilmiştir. Bu ekran IP adres taraması seçeneği ile belirlenen kritere göre ağın taraması işlemini ve Sonuc DataGrid seçeneği ile de tarama sonucunda kaydedilen verilerin raporlaması işlemini yapmaya olanak sağlamaktadır.



Şekil 4.4 Genel ekran

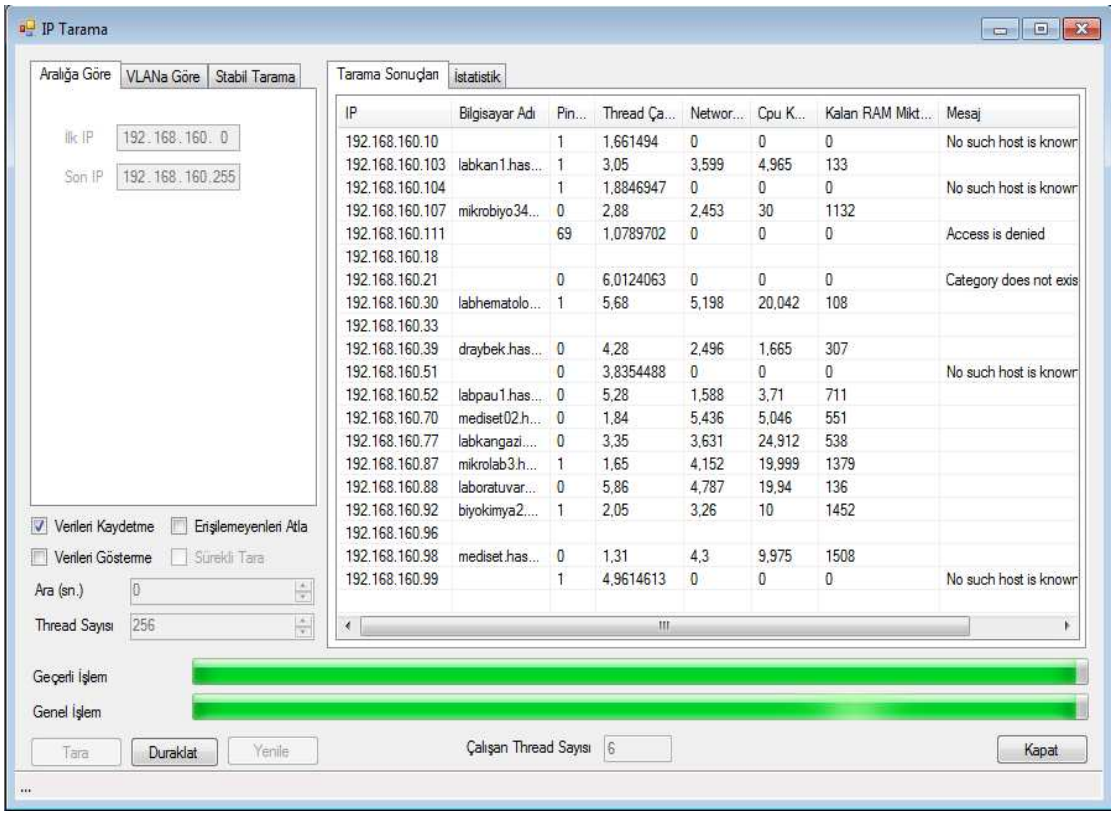
Şekil 4.5'de IP adres taraması ekranı açıldığında karşılaşılacak ekran verilmiştir. Bu ekran genel tarama işleminin belirlenen aralığa, VLAN'a ya da stabil tarama kriterlerine göre tarama yapılanabilecek ekrandır.



Şekil 4.5 IP adres taraması ekranı

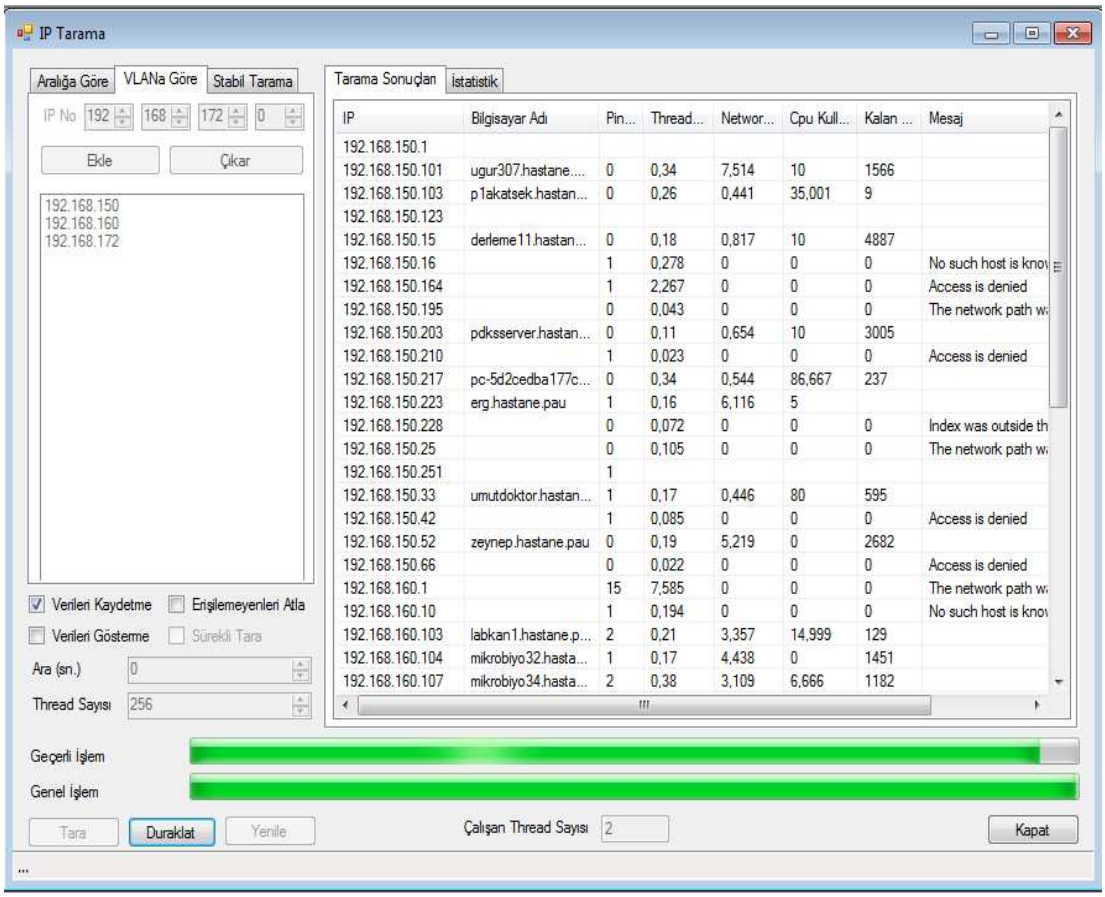
Belirlenen kriterlere göre yapılan taramalar ayrıntılı olarak anlatılmıştır.

Belirli Aralığa Göre Tarama: Belirli aralığa göre tarama kullanıcının istediği aralığa göre tarama yapmasını sağlamaktadır. Amaç kullanıcının tarama yapmayı istediği bir aralık varsa sadece bu aralıkta işlem yapmasını sağlamaktır. Bu anlık olarak kullanılacak tarama türüdür. Şekil 4.6'da belirli aralığa göre tarama yapılması durumunda oluşan ekran çıktısı gösterilmiştir.



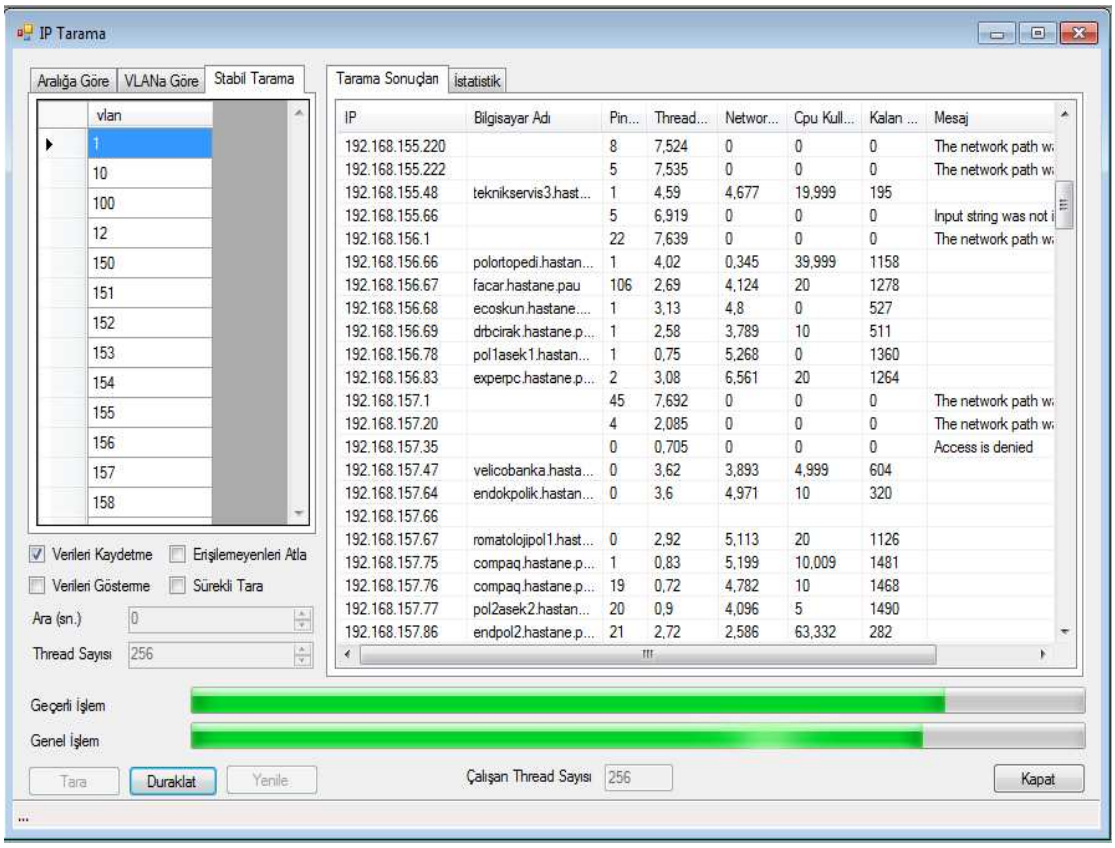
Şekil 4.6 Belirli aralığa göre tarama yapılması durumunda oluşmuş ekran çıktısı

VLAN a Göre Tarama: VLAN'a göre tarama yapmak kullanıcının taramak istediği ve veritabanında kayıtlı olan yani gerçekten sistemin çalıştığı ağdaki bütün VLAN'ları kapsayan VLAN'lardan istenenleri seçip sadece bu VLAN'ların taramasının yapılmasıdır. Burada amaçlanan kullanıcının incelemek istediği VLAN'ları seçip sadece bu VLAN'ları incelemesini sağlamaktır. Bu yine anlık olarak kullanılacak bir ekrandır. Şekil 4.7'de VLAN'a göre tarama yapılması durumunda oluşan ekran çıktısı gösterilmiştir.



Şekil 4.7 VLAN'a göre tarama yapılması durumunda oluşmuş ekran çıktısı

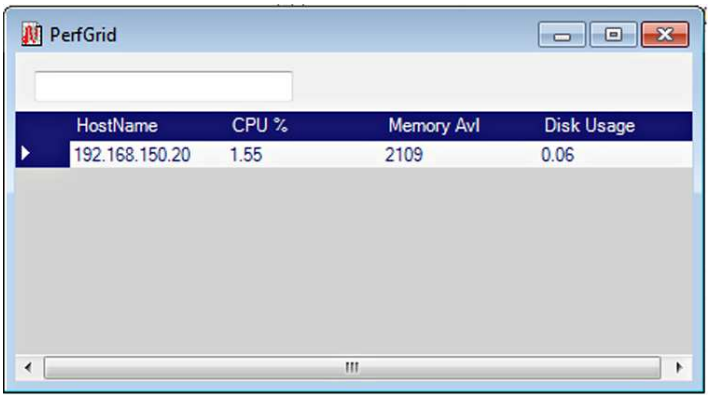
Stabil Tarama: Stabil tarama bu projede temelde ve sürekli kullanılacak tarama olup bu tarama ile veri tabanında kayıtlı, projenin çalıştığı ağdaki bütün VLAN'ların taraması yapılmaktadır. Bu tarama, sürekli tarama işlemine tabi tutulup projenin çalıştığı ağdaki makinaların döngüsel olarak taranması şeklinde gerçekleşmektedir. Buradan da ulaşılan makinaların bilgileri veri tabanına kaydedilmekte ve bu veriler raporlama yapmak için kullanılmaktadır. Raporlama için kullanılacak asıl veriler bu tarama türünden oluşmaktadır. Bu tarama türünde kurum içinde var olan ve veri tabanına kayıtlı olan bütün VLAN'lar sürekli olarak taranmakta ve verileri kaydedilmektedir. Şekil 4.8' de stabil tarama yapılması durumunda oluşan ekran çıktısı gösterilmiştir**.**



Şekil 4.8 Stabil tarama yapılması durumunda oluşmuş ekran çıktısı

Erişilen bilgisayarlar üzerinde de anlık olarak detaylı ve özet bilgi görüntülemesi yapılmaktadır. Bu iki ekran farklı iki projenin sisteme entegre edilmesinden oluşmuştur.

Özet bilgi görüntüleme ekranında, istenen bilgisayarın anlık olarak CPU kullanım, disk kullanım ve kullanılabilir RAM miktarları görüntülenebilmektedir. Şekilde 4.9'da projenin çalıştığı ağdaki bir bilgisayarın özet bilgi görüntüleme ekranı gösterilmektedir[18].



Şekil 4.9 Özet bilgi görüntüleme ekranı

Detaylı bilgi görüntüleme ekranında ise yine istenilen bilgisayarın anlık olarak ağ performans bilgileri görüntülenebilmektedir. Bu ekran anlık olarak o bilgisayarın verilerinin grafiksel takibine imkan sunmaktadır[19].

Şekil 4.10'da projenin çalıştığı ağdaki bir bilgisayarın detaylı bilgi görüntüleme ekranı gösterilmektedir.

Şekil 4.10 Detaylı bilgi görüntüleme ekranı

Verilen ekranlar program çalışırken anlık gözlem yapabilmeyi sağlamaktadır.

4.3 Raporlama

Kurumlar yaptıkları işin performans değerini ölçmek için raporlamaya ihtiyaç duyarlar. Proje kapsamında raporlama kurumların değişen kaynaklarını takip etmesi açısından önem teşkil eden bir araçtır diye tanımlanabilmektedir. Toplanan doğru verilerin tablolar halinde sunulması kurumun verileri doğru şekilde yorumlaması açısından önemlidir. Kurum bu raporlar sayesinde takip etmek istediği verileri kolayca takip edebilir ve müdahale etmesi gereken noktalara kısa sürede müdahale edebilmektedir.

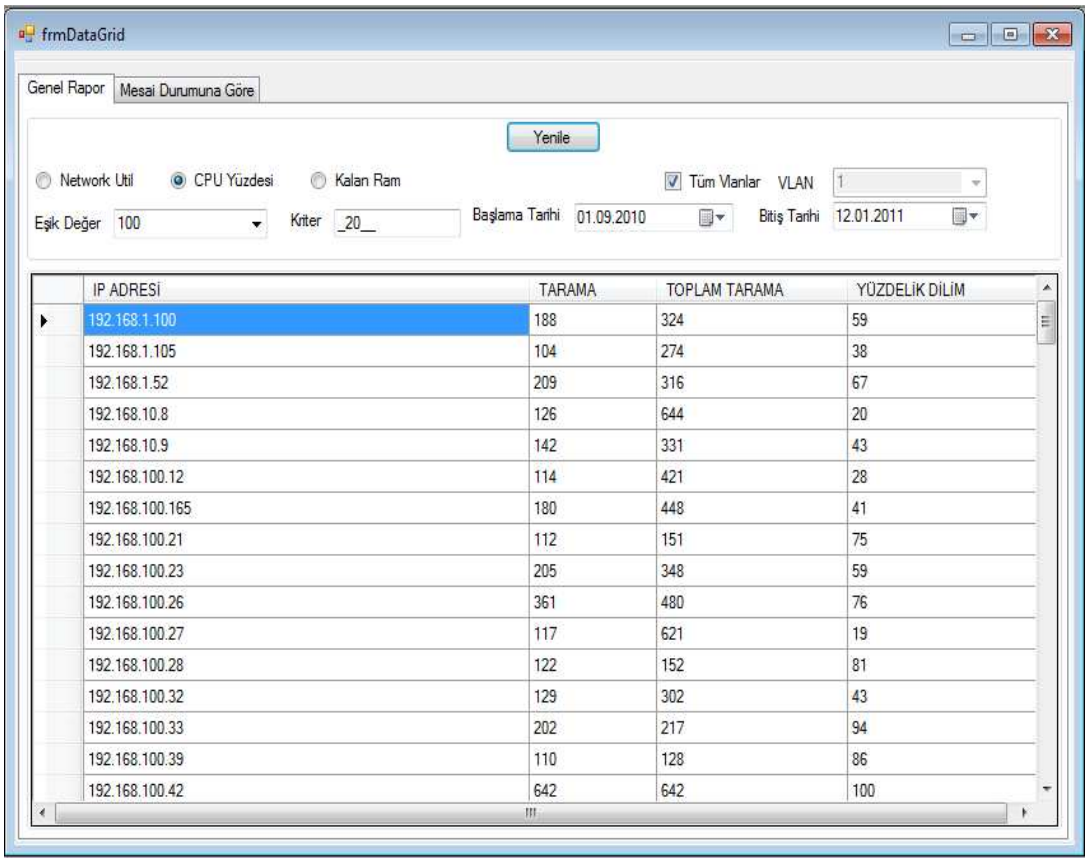
Projedeki asıl amaç ağdaki bilgisayarlara bir döngü dahilinde sürekli erişerek veri toplamak, toplanan verileri veri tabanına kaydetmek ve toplanan bu verilerden anlamlı raporlar hazırlamaktır. Buradaki anlamlı rapor terimi projenin kullanılacağı ağın durumuna göre anlamlandırılabilinir. Bu proje için raporlar hazırlanırken aşağıdaki kriterler göz önünde bulundurulmuştur.

- Projenin çalıştığı ağ faklı VLAN lar içermektedir
- Projenin çalıştığı sistem 7/24 hizmet veren bir sistemdir fakat 8:00-17:00 aktif 17:00-8:00 kısmen hizmet veren bir sistemdir.

Yukarıdaki kriterler göz önünde bulundurularak projenin çalıştığı sistem için 2 farklı rapor oluşturulabilmektedir.

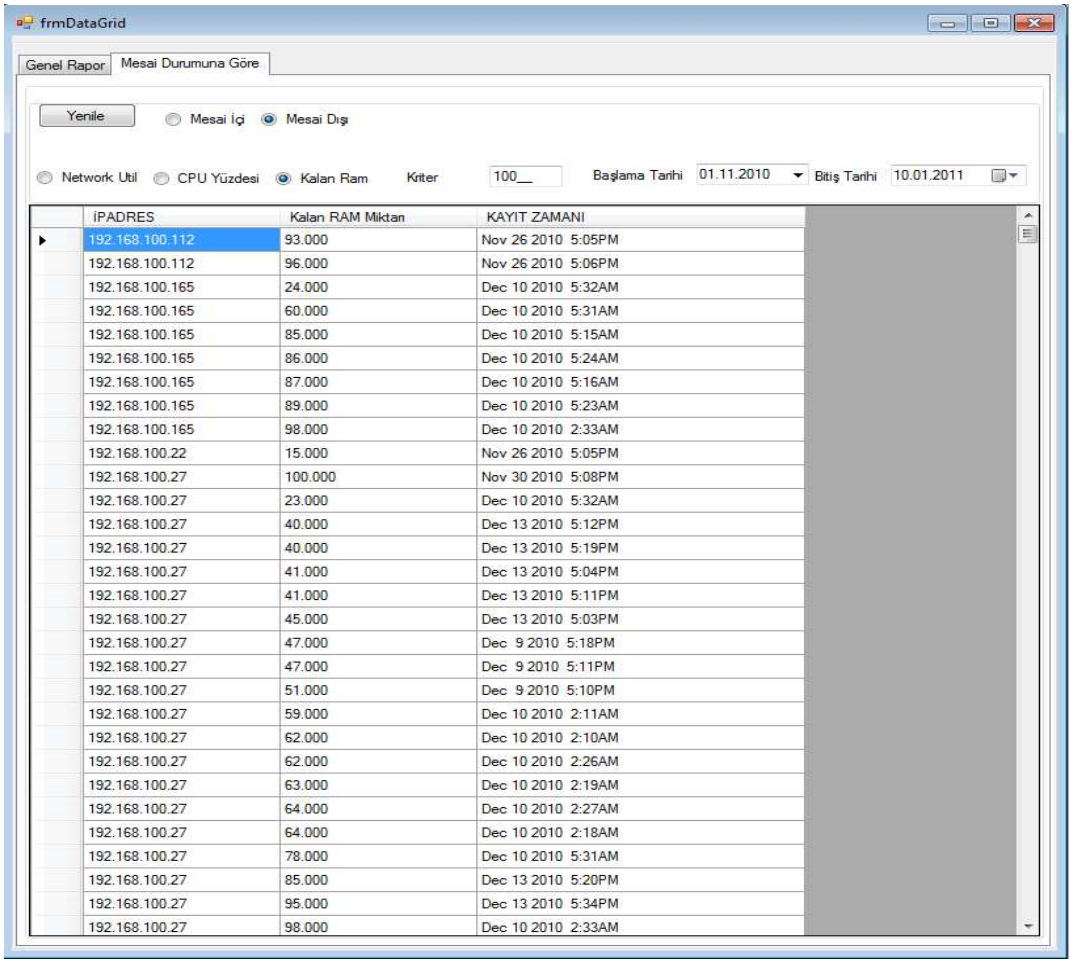
Genel Raporlama: Kullanıcı proje kapsamında veri tabanına kaydedilen performans bilgilerini sisteminde var olan seçmiş olduğu VLAN bazında incelemek isteyebilir. Bu sayede seçilen VLAN'a ait makinaların kayıtlı performans bilgilerinin karşılaştırması yapabilmektedir. Aynı zamanda genel raporlama ekranında tüm VLAN'lar bazında da raporlama yapılabilmektedir.

Genel rapor oluşturma ekranı Şekil 4.11'de verilen ekran olup bu ekranda istenen tarih aralığı, istenen kriter, istenen eşik değer ve istenen kayıtlı VLAN'a göre rapor oluşturulabilmektedir. Bahsedilen eşik değer belirtilen tarih aralığındaki taranan bilgisayarların belirlenen kritere göre toplam ulaşıldığı sayıyı verir. O tarih aralığında belirlenen kriteri aşan bilgisayarların toplam tarama sayısı eşik değeri aşarsa dikkate alınır. Yüzdelik değer ifadesi de yine o tarih aralığında yapılan toplam taramanın eşik değeri aşan tarama sayısına oranını vermektedir. Sonuç olarak bu tablo seçilen kriterlere göre genel bir değerlendirme yapılmasını sağlamaktadır.



Şekil 4.11 Genel raporlama ekranı

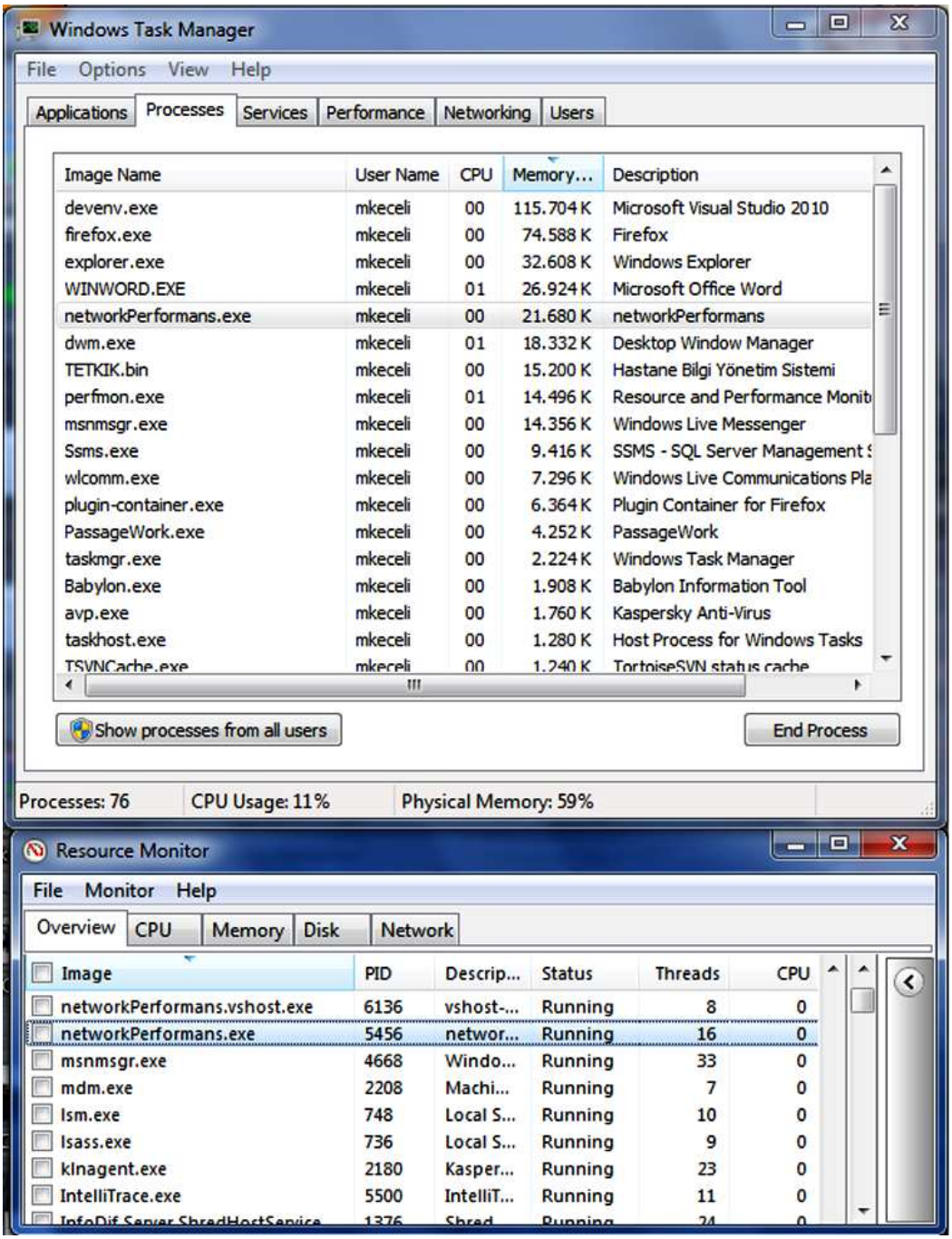
Mesai durumuna göre raporlama: Projenin çalıştığı sistem 8:00 -17:00 aktif 17:00-8:00 kısmen hizmet veren 7/24 çalışan bir sistemdir. Bu durum göz önünde bulundurulduğunda mesai durumuna göre rapor oluşturulması anlamlı hale gelmektedir. Kullanıcı sistemin ister mesai içi isterse mesai dışı çalışması durumlarını karşılaştırmak, gözlemlemek isteyebilir. Bu raporlama türünde mesai içi veya mesai dışı duruma göre ve belirlenen kriter göz önüne alınarak raporlanmak istenen veriler tablo halinde sunulabilmektedir. Şekil 4.12'de mesai durumuna göre raporlama ekranı verilmiştir.



Şekil 4.12 Mesai durumuna göre raporlama ekranı

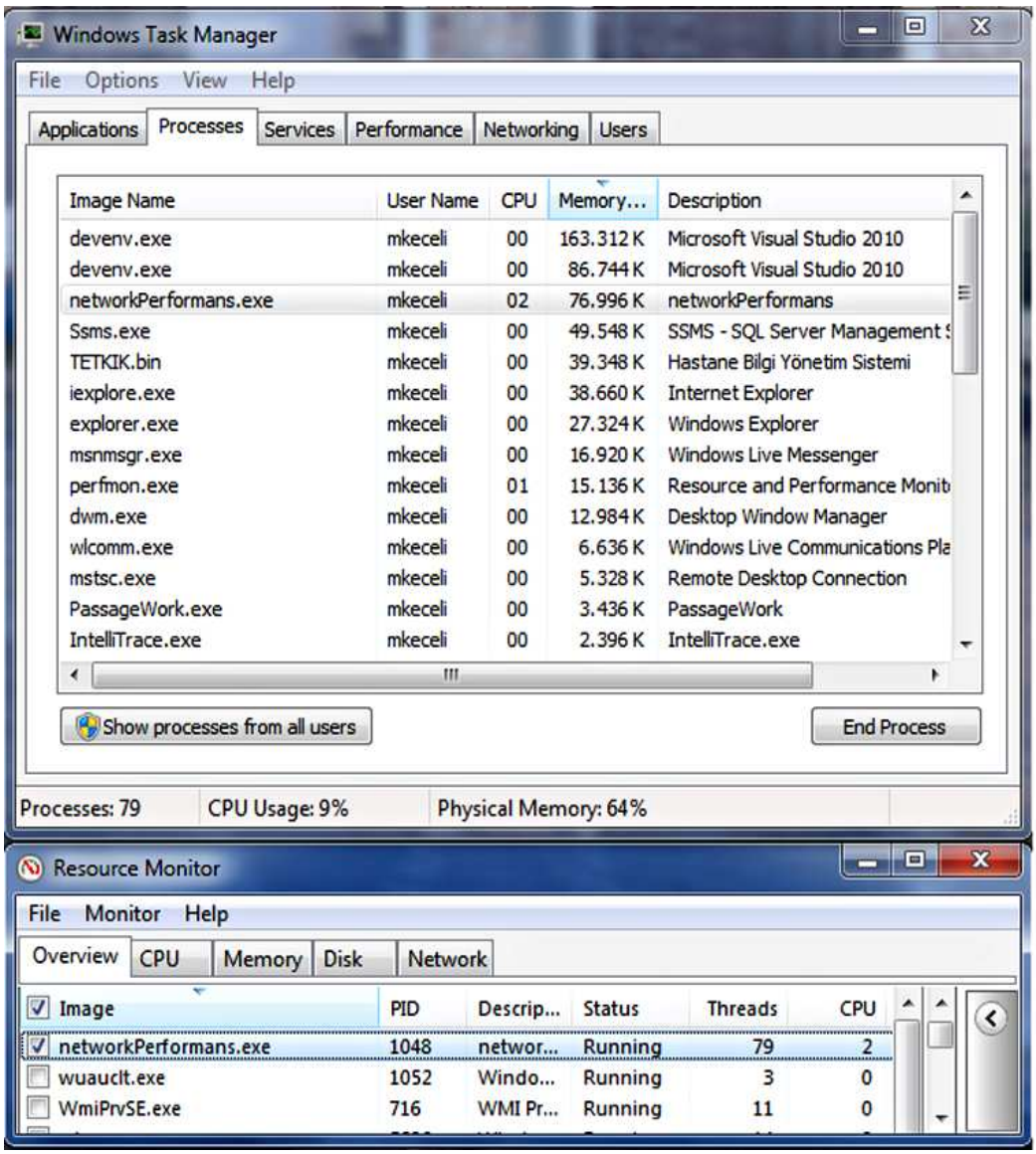
4.4 Ağ İzleme ve Performans Değerlendirme Programının Optimum Çalışma Şartlarının belirlenmesi

Proje dahilinde projede kullanılan teknoloji ve kavramlar bölümünde açıklanan thread kavramı verimli olarak kullanılmıştır. Projenin amacı düşünüldüğünde veriler toplanması ve elde edilen bu verilerden anlamlı ve gerçekçi raporlar oluşturulması temel amaçtır. Bu raporların anlamlı olmasında projenin çalıştığı ağın ne kadar sıklıkla tarandığı önemlidir. Ağın fazla taranması uygun thread sayısı ile alakalıdır. Diğer taraftan kontrolsüz thread kullanımının ağı yoracağı da düşünülerek en uygun thread sayısını belirlemek önemlidir. Proje kapsamında uygun thread sayısının belirlenmesi için kodlar farklı sayıda threadler ile çalıştırılmış ve uygun thread sayısı bu denemeler sonucunda belirlenmiştir. Aşağıda farklı thread sayılarına göre çalışan programın ekran çıktıları verilmiştir.



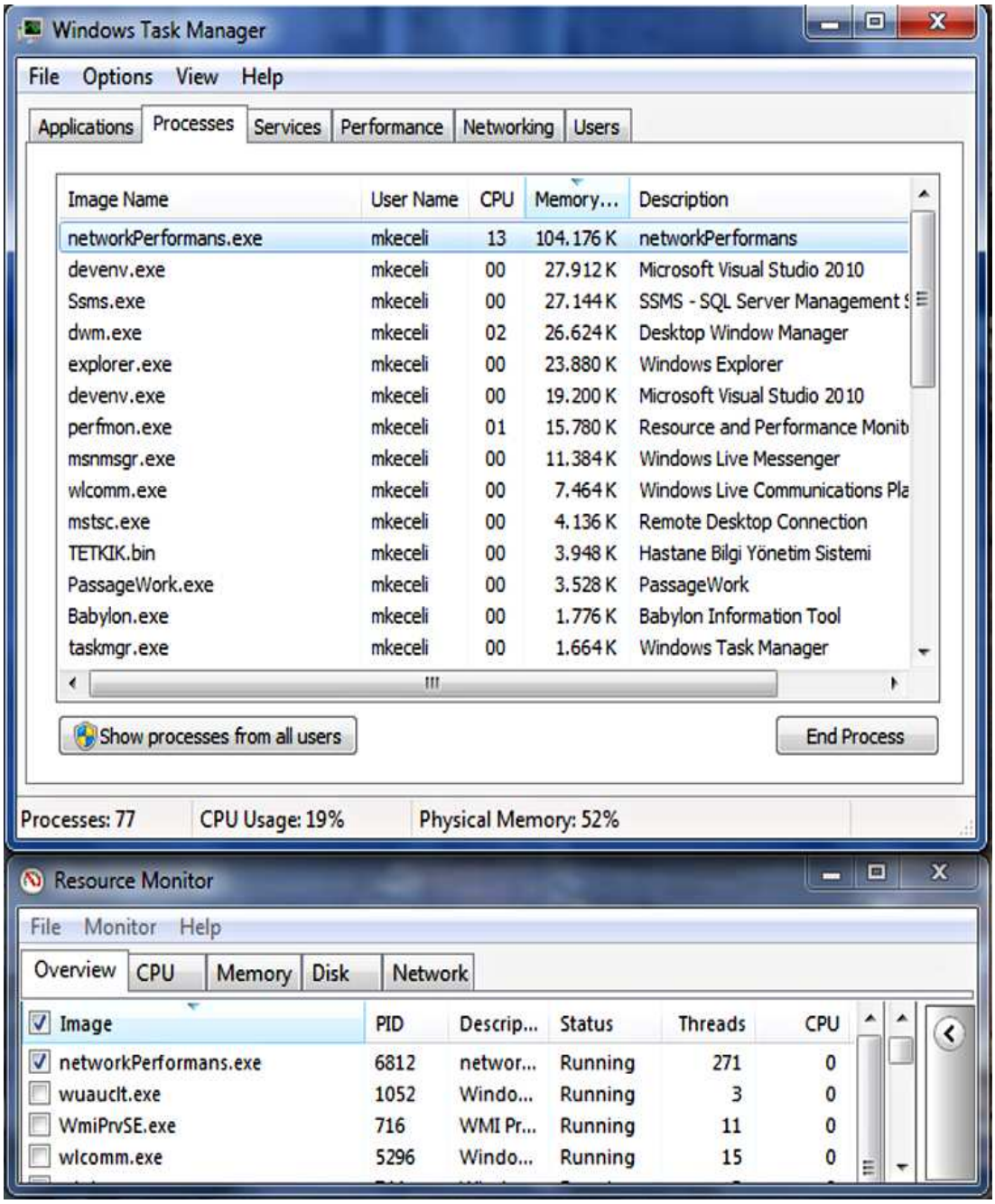
Şekil 4.13 1 thread ile çalışan programın çalıştığı bilgisayarın performansına etkisi

Şekil 4.13'e baktığımızda 1 thread ile çalışan sistemin çalıştığı bilgisayarın işlemcisini yormadığı görülmüştür.



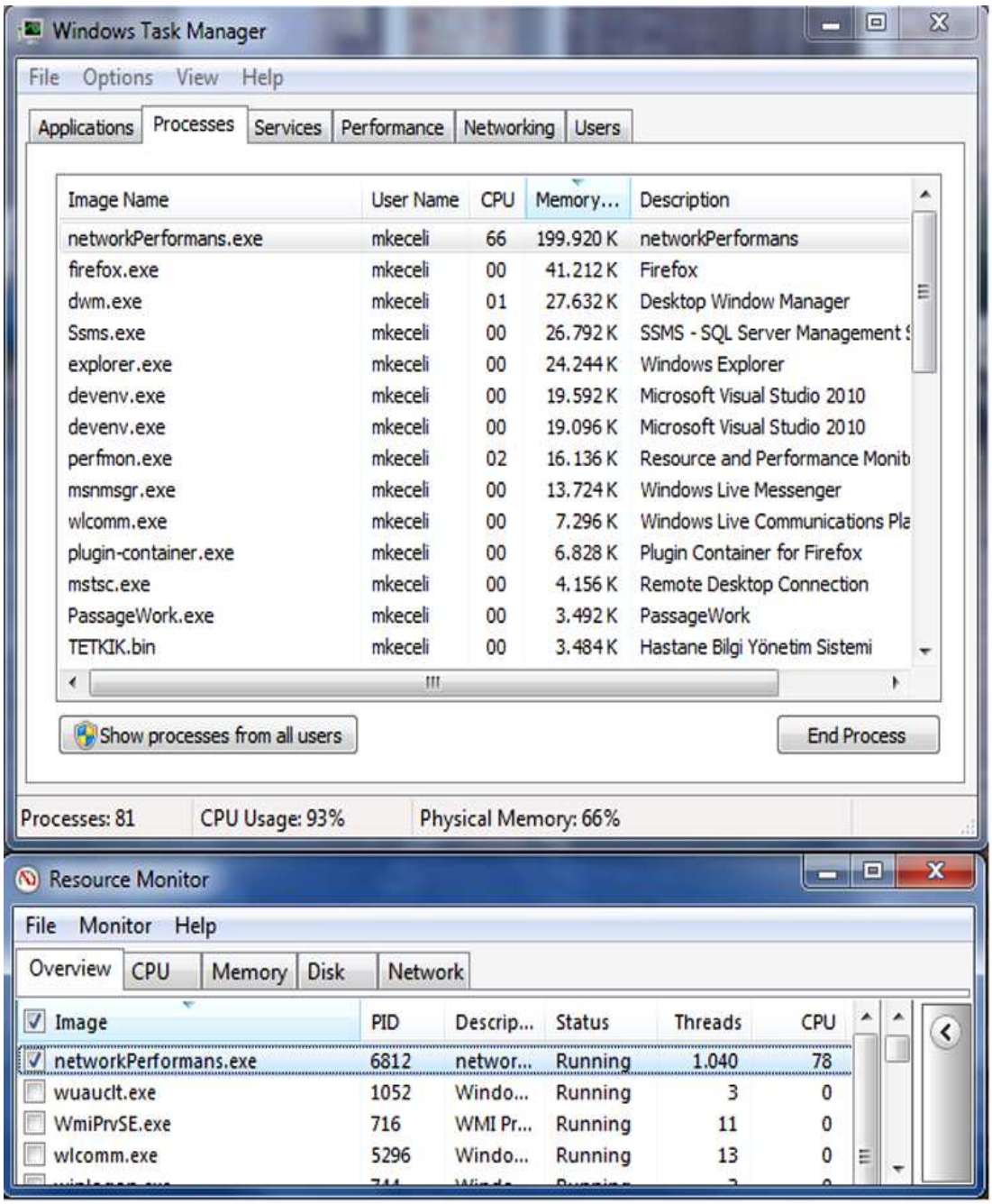
Şekil 4.14 64thread ile çalışan programın çalıştığı bilgisayarın performansına etkisi

Şekil 4.14'e baktığımızda 64 thread ile çalışan sistemin çalıştığı bilgisayarın işlemcisini çok yormadığı fakat 1 thread'e göre de daha fazla işlemci tükettiği görülmüştür.



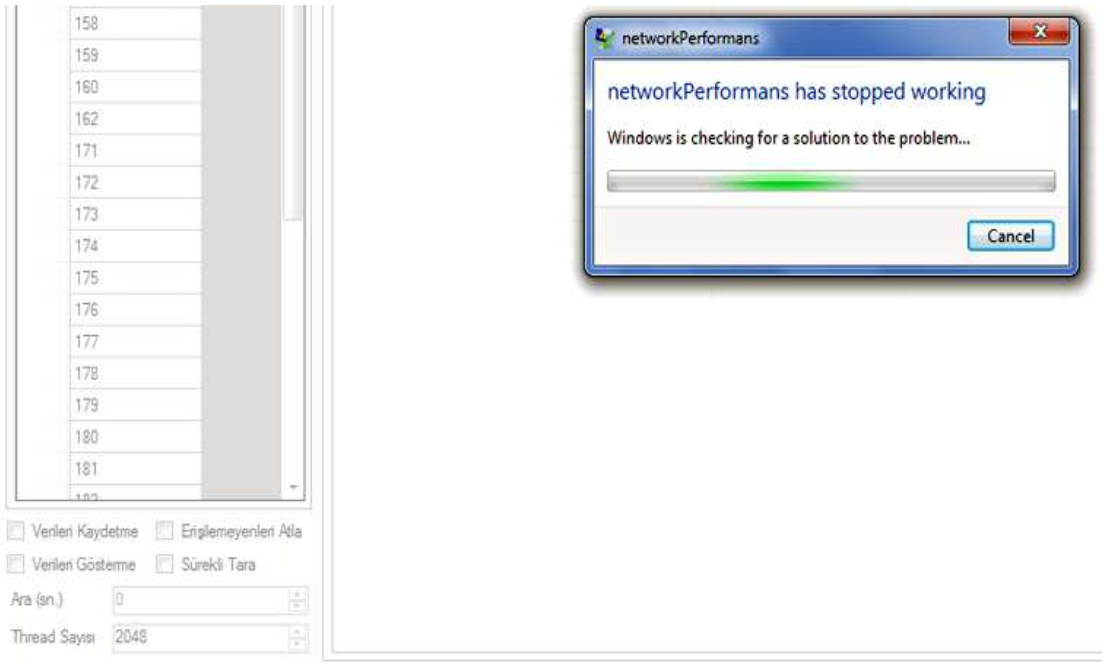
Şekil 4.15 256 thread ile çalışan programın çalıştığı bilgisayarın performansına etkisi

Şekil 4.15'e baktığımızda 256 thread ile çalışan sistemin çalıştığı bilgisayarın kaynaklarını olumsuz yönde etkilemediği görülmüştür.



Şekil 4.16 1024 thread ile çalışan programın çalıştığı bilgisayarın performansına etkisi

Şekil 4.16'ya baktığımızda 1024 thread ile çalışan sistemde sistemin çalıştığı bilgisayarın işlemcisini yorduğu görülmüştür.



Şekil 4.17 2048 thread ile çalışan programın çalıştığı bilgisayarın performansına etkisi

Şekil 4.17'ye baktığımızda 2048 thread ile programın çalışmadığı görülmüştür. Bu 2048 threadin programın çalıştığı bilgisayara fazla geldiği anlamına gelmektedir.

Programa genel olarak baktığımızda kullanılan thread sayısı ile programın, çalıştığı bilgisayarın işlemcisini kullanması doğru orantılıdır. Belirlenecek thread sayısı sistemin çalıştığı bilgisayarın özellikleri ile de ilgilidir. Bu sebeple program farklı özellikteki bilgisayarlarda farklı sonuçlar verebilmektedir. Programın proje aşamasında çalıştığı bilgisayar 2 GB RAM'e ve Intel(R) Core(TM)2 Duo E7300 @ 2,66 GHz 2,67 GHz işlemcisine sahip 32 bit işletim sistemli bir makinadır. Elde edilen sonuçlar bu özelliklere göre değerlendirilmelidir.

Tablo 4.1 Farklı threadlerle çalışan sitemin iş bitirme zamanları

taramaId	makinaSayisi	threadSayisi	CalismaSuresi
1149	382	1	13:53:20
1151	411	2	06:54:29
1152	333	2	04:31:38
1153	319	2	04:31:54
1155	543	32	00:26:29
1156	561	64	00:13:33
1159	560	128	00:07:29
1162	550	256	00:03:52
1163	531	256	00:03:51
1164	531	256	00:03:51
1165	533	512	00:02:20
1166	509	512	00:02:18
1167	517	1024	00:01:41
1168	517	1024	00:01:36

Tablo 4.1‘e bakıldığında uygun thread sayısı 1024 olarak görülse de program periyodik olarak çalıştığında verimli sonuçlar elde edilmemektedir.1024 thread ile belirli bir tarama sayısına eriştiğinde çalıştığı bilgisayarda kilitlenmelere neden olabilmektedir. Bunun sebebi işlemcinin o bilgisayar üzerinde çalıştırdığı programlara ayırdığı kaynakların sınırlı olmasıdır. Sistemin çalıştığı makina için uygun thread sayısının belirlenmesi kaynak kullanımını da düzenleyecek ve bu sayede uygun threadlerle program verimli çalışacaktır.

Programın da periyodik olarak çalışması gerektiği göz önünde bulundurulursa 1024 threadle çalışması uygun değildir. Bahsedilen verimsizlik kavramı projenin kilitlenmesi olarak belirlenebilir. Genel olarak bakıldığında ise projenin çalıştığı makinada en uygun thread sayısı 256 olarak belirlenmiştir.

Lock

Projede kullanılan teknoloji ve kavramlar bölümünde bahsedilen lock kavramı thread kullanılan programlarda kullanılan thread senkronizasyon tekniklerindendir ve threadleri etkin bir şekilde kullanmamızı sağlamaktadır. Bu kapsamda proje dahilinde de gerekli görülen yerlerde kullanılmıştır. Ve programın aksamadan çalışmasında oldukça etkili olmuştur.

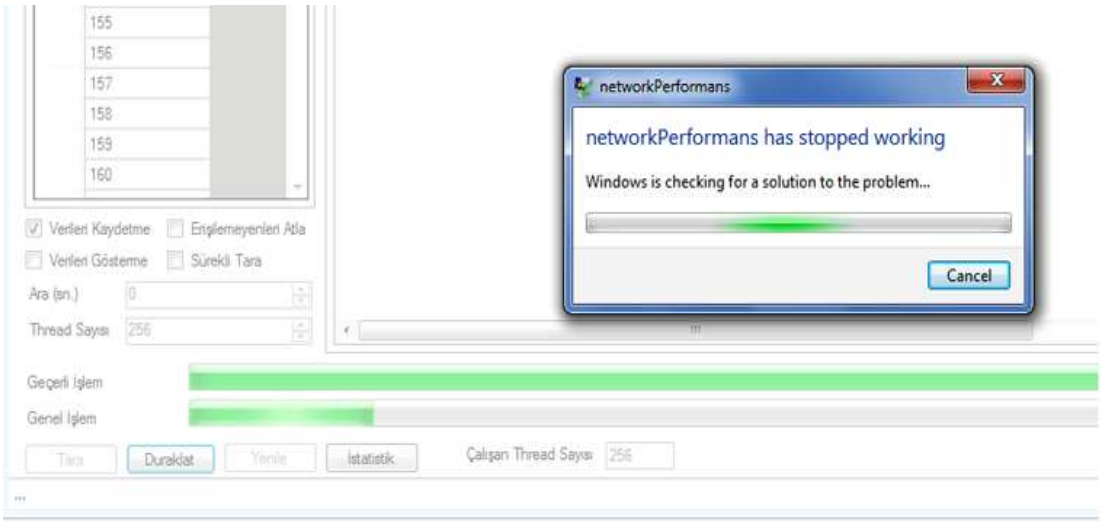
Lock işlemi projede Tablo 4.2'deki kodlarla gerçekleşmektedir ve bu 4 satır kod sistemin çalışması için hayati önem taşımaktadır.

Tablo 4.2 Projede kullanılan lock kavramı kod parçacığı

```
lock (frm.pbGecerliIslem)
        {
          frm.pbGecerliIslem.Value++;

          if (frm.pbGecerliIslem.Value == frm.pbGecerliIslem.Maximum)
          {
            frm.pbGecerliIslem.Value = 0;
          }
        }
```

Proje lock satırları kapatılıp çalıştırıldığında kilitlenmektedir. Şekil 4.18'de lock satırları kapatılarak çalıştırılan programın ekran çıktısı gösterilmiştir.



Şekil 4.18 Lock olmadan çalışan programın ekran çıktısı

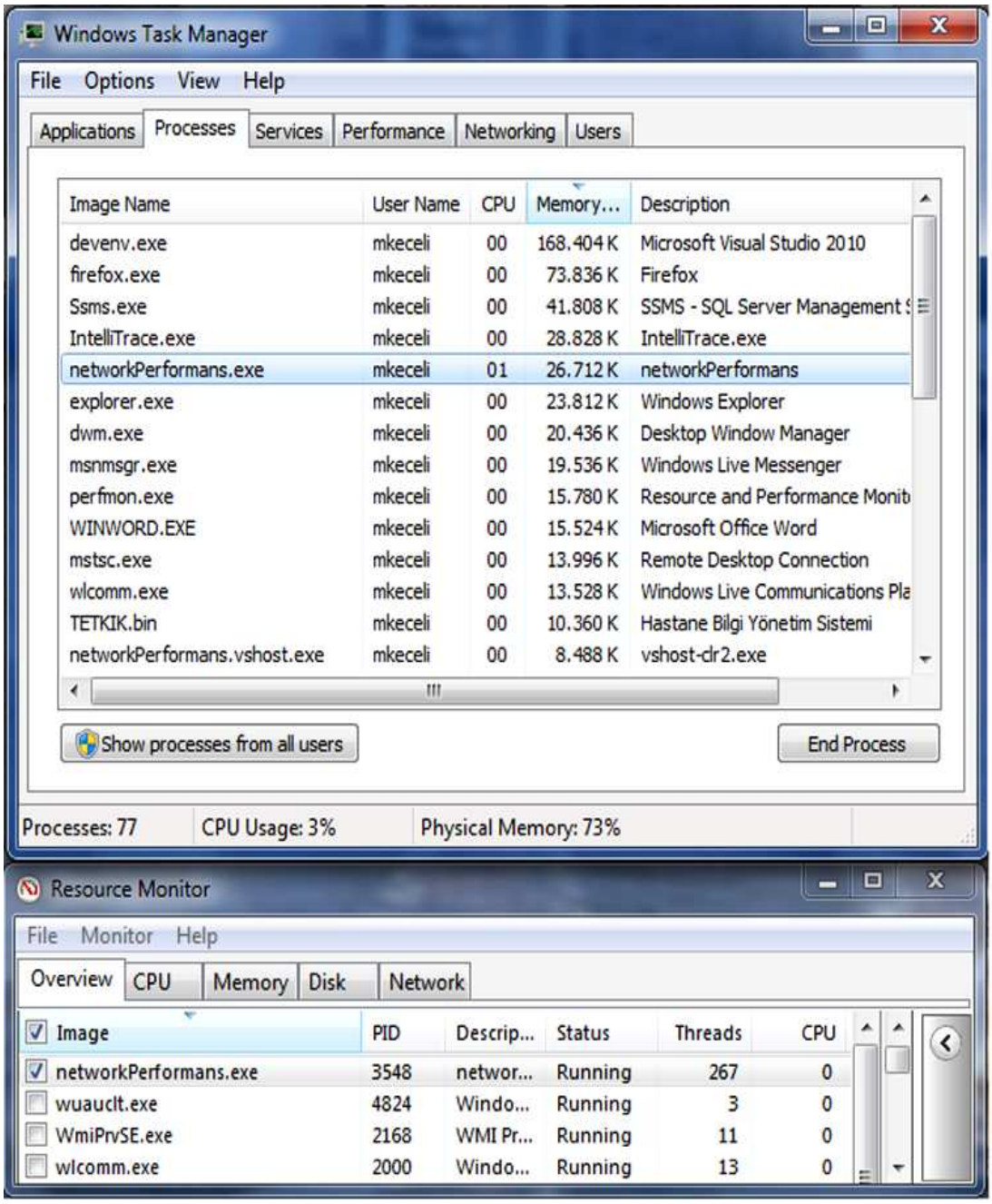
Ping

Ping kavramı basit olmakla birlikte ağ problemlerini belirlemede sıkça kullanılan bir kavramdır. Proje kapsamında da projeyi hızlandırmada bir adım olarak kullanılmış ve daha verimli sonuçlar elde edilmesini sağlamıştır. Tablo 4.3'deki küçük kod parçası programın verimini arttırmada önem arz etmektedir. Bu kodla öncelikle bilgileri alınmak için erişilmek istenen bilgisayarlara ping atılarak dönen sonuca göre işlem yapılmaktadır. Dönen cevap true ise o bilgisayara ulaşılmakta ve belirlenen parametre bilgileri alınmakta false ise o bilgisayar atlanmaktadır.

Tablo 4.3 Projede kullanılan ping fonksiyonu kod parçacığı

```
public string pingStatus(string ip)
      {
         using (Ping p = new Ping())
         {
            PingReply reply = p.Send(ip);
            pingSuresi = reply.RoundtripTime;
            return reply.Status.ToString();
         }
}
```

Şekil 4.19'da ping fonksiyonu kullanılmadan tarama yapılması durumunda programın çalıştığı bilgisyarın performansına etkisi gösterilmiştir.



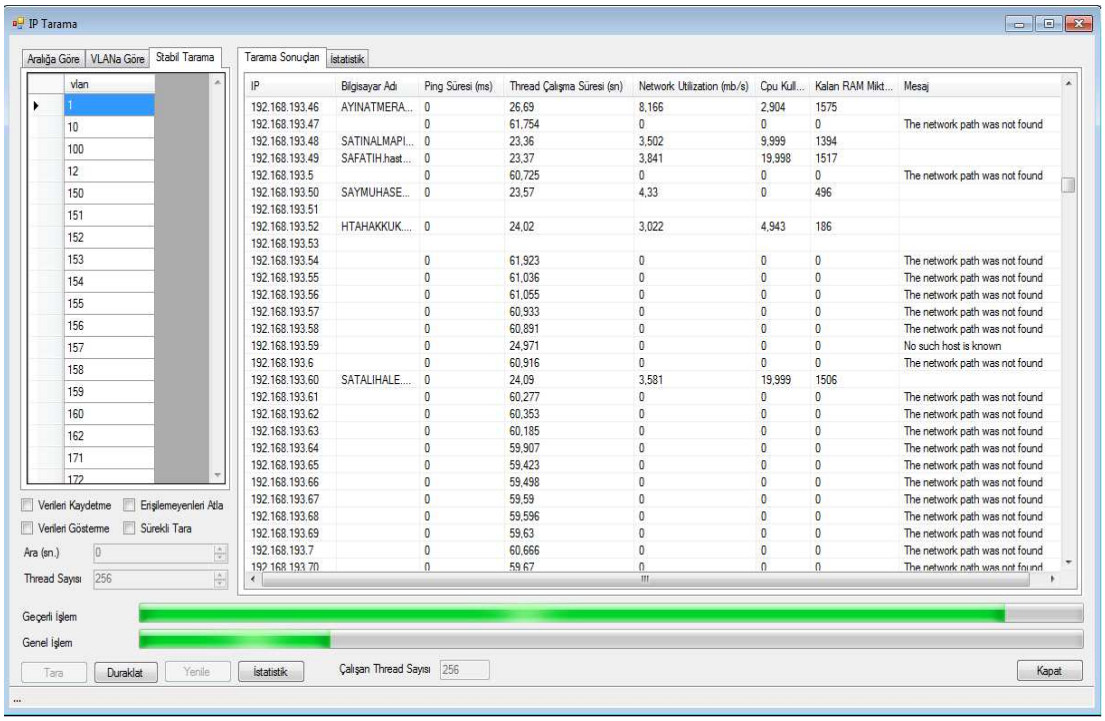
Şekil 4.19 Ping fonksiyonu olmadan çalışan programın çalıştığı bilgisayarın performansına etkisi

Şekil 4.19'a bakıldığında ping fonksiyonunun kullanılmaması programın çalıştığı bilgisayarın performansına herhangi bir etki yapmamıştır. Ping fonksiyonu programın iş bitirme zamanına olumlu yönde bir etki yapmıştır. Tablo 4.4'de kırmızı ile belirlenen alan ping fonksiyonu olmadan çalışan sistemin iş bitirme zamanını göstermektedir. Bu sonuçlardan da anlaşılacağı gibi ping kavramının kullanımı proje için göz ardı edilemeyecek öneme sahiptir.

Tablo 4.4 Ping fonksiyonuna göre sistemin iş bitirme zamanlarının karşılaştırılması

taramaId	makinaSayisi	threadSayisi	calismaSuresi
1149	382	1	13:53:20
1151	411	2	06:54:29
1152	333	2	04:31:38
1153	319	2	04:31:54
1155	543	32	00:26:29
1156	561	64	00:13:33
1159	560	128	00:07:29
1162	550	256	00:03:52
1163	531	256	00:03:51
1164	531	256	00:03:51
1165	533	512	00:02:20
1166	509	512	00:02:18
1167	517	1024	00:01:41
1168	517	1024	00:01:36
1170	9945	256	1.00:01:15

Ping fonksiyonu aynı zamanda gereksiz alanlarıda ortadan kaldırmaktadır. Ping fonksiyonu kullanılmadan Şekil 4.20'deki gibi bir ekranla kaşılaşmaktayız. Bu görüntüden de anlaşılacağı gibi burada gereksiz olan alanlar oldukça fazladır. Ping fonksiyonunun kullanılması hem programın döngüsünü daha kısa zamanda bitirmesini sağlayacak hemde gereksiz alanların tekrarını engelleyecektir.



Şekil 4.20 Ping fonksiyonu kullanılarak oluşan ekran çıktısı

4.5 Program Gelişim Aşamaları

Proje son haline gelene kadar bir çok aşama atlatmıştır. Atlatılan her aşama projenin farklı bir boyut kazanması sağlamıştır. Öncelikle bu alanda geliştirilen birçok farklı ağ programları incelenmiş, incelenen projeler yorumlanarak proje için bir yön belirlenmiştir ve böylece projenin temeli oluşturulmuştur. Sonrasında, oluşturulan bu temelin nasıl devam edeceği üzerine gidilmiştir. Programın yapısı ana hatlarıyla belirlendikten sonra geliştirilmeye başlanmıştır. Geliştirilmeye başlayan projede başlangıç aşamasındaki temel sorun ağın tamamının nasıl taranacağıydı. Bu sorun ağ yapısı içerisindeki bütün VLAN'ların veri tabanına kayıt edilmesi, veri tabanına kaydedilen her VLAN'da bulunan IP bloğu içerisindeki her bir bilgisayara sırasıyla erişilmesi şeklinde aşılmıştır. Bu haliyle çalışan program kısmen amaca uygun olsa da performans açısından bakıldığında bir tam tur taramasını tamamlayamamaktadır. Tarama işleminin kısa sürede ve bir döngü halinde sürekli olarak yapılması gerektiği göz önünde bulundurulursa bu sorunun aşılmasının gerekliliği ortaya çıkmaktadır. Araştırmalar sonucunda sorunun çözümü için, bir işin eş zamanlı olarak işlenen her bir bölümünü ifade thread kavramının kullanılması uygun görülmüştür. Proje kapsamında çoklu thread kullanımı ilk olarak .NET frameworkün üstü kapalı bir

şekilde thread üretip çalıştırdığı BackgroundWorker komponenti kullanılarak yapılmaya çalışılmıştır. Belirli ölçülerde olumlu sonuç veren BackgroundWorker kullanımından kontrolun tamamen kullanıcıda olmaması nedeniyle vazgeçilmiştir. Bunun yerine kontrolün tamamen kullanıcıda olduğu, açık bir şekilde threadleri oluşturup çalıştırmak yöntemi tercih edilmiştir. Threadlerin verimli şekilde kullanılmasıyla ağın bir tam tur taranması sorunsuz olarak gerçekleştirilebilinmiştir. Fakat multithreading dezavantajlarla birlikte gelmektedir. Bunların başında threadler arasındaki etkileşim karmaşıklığı gelir. Ve bu sorun thread senkronizasyon teknikleri kullanılarak aşılabilir. Farklı thread senkronizasyon teknikleri incelenmiş ve bu teknikler içinden lock tekniği kullanım için uygun görülmüştür. Thread senkronizasyon teknikleriyle birlikte kullanılan threadler projenin büyük ölçüde şekillenmesini sağlamıştır. Bu aşamaya gelen projeye bakıldığında bir sonraki hedef, programın bir tam tur taramasını daha kısa sürede yapmasıdır. Network mühendislerinin ağ ile ilgili problemlerinin çözümünde sıklıkla kullandıkları ping fonksiyonu problemin aşılması için kullanılacak yöntem olarak belirlenmiştir.Ping fonksiyonu, ping paketi gönderilen makinenin o anda çalışmakta olduğunu teyid etmektedir. Ping fonksiyonu da etkin olarak kullanıldıktan sonra ulaşılan nokta projenin son haliyle büyük ölçüde benzerlikler göstermektedir.

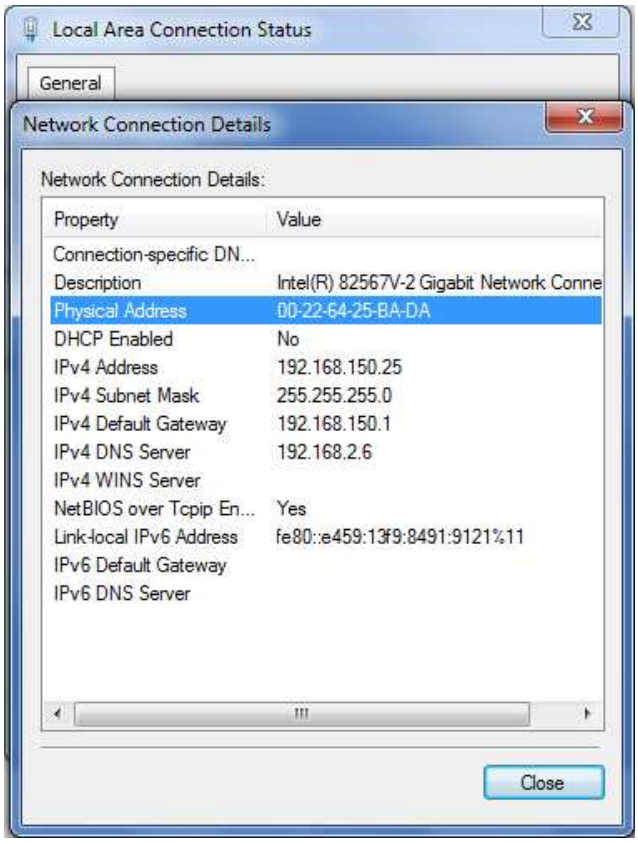
Karşılaşılan ara problemler yapılan araştırmalar sonucunda belirlenen uygun yöntemlerle aşılmıştır.

4.6 Program Ağı Ne kadar Yorar

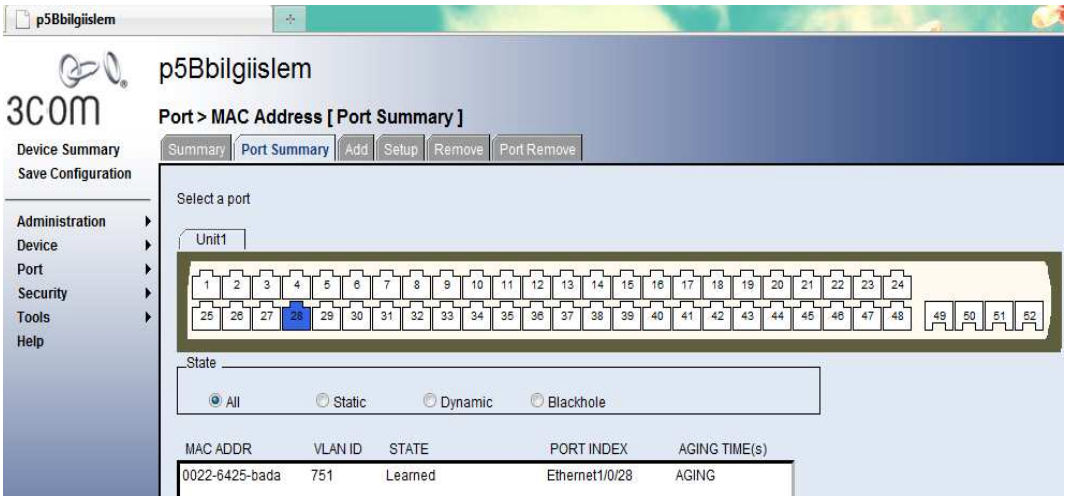
Programın bir bilgisayar üzerinde sürekli çalışacak olması, ağdaki tüm bilgisayarlarla kısa süre içinde haberleşip program dahilinde belirlenen parametre bilgilerini alması, ve aynı anda çok sayıda bilgisayara erişecek olması programın ağı ne kadar yorduğu sorusunu akla getirmiştir.

Programın ağı ne kadar yorduğunu anlamak için projenin çalıştığı bilgisayarın bağlı olduğu kenar switch üzerindeki portu belirlenmiş ve bu port bilgileri program çalışmadan önce ve program çalıştıktan sonra kaydedilmiştir. Bu verilerden program çalışırken o porta gelen/giden byte bilgileri kullanılarak programın ağı ne kadar yorduğu tespit edilmiştir.

Şekil 4.21 ve Şekil 4.22 programın çalıştığı makinanın MAC adres bilgisinin kenar switch üzerinde karşılığını karşılaştırmak ve iki bilginin aynı olduğunu göstermek için verilmiştir.

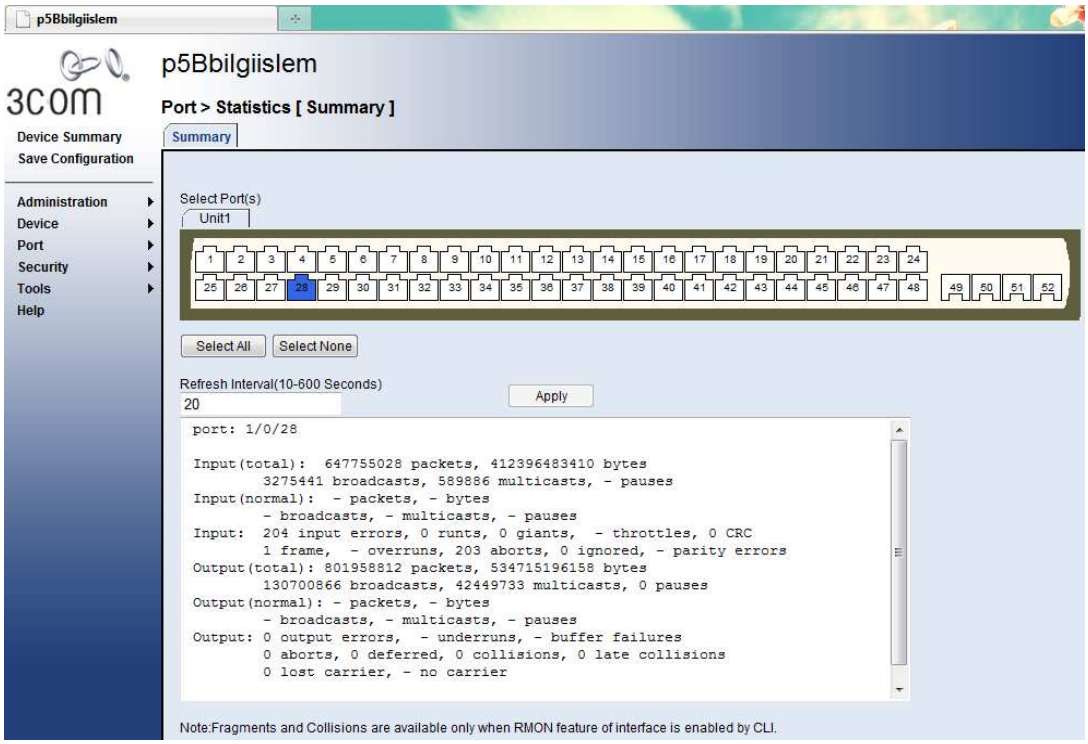


Şekil 4.21 Programın çalıştığı bilgisayarın MAC adresi

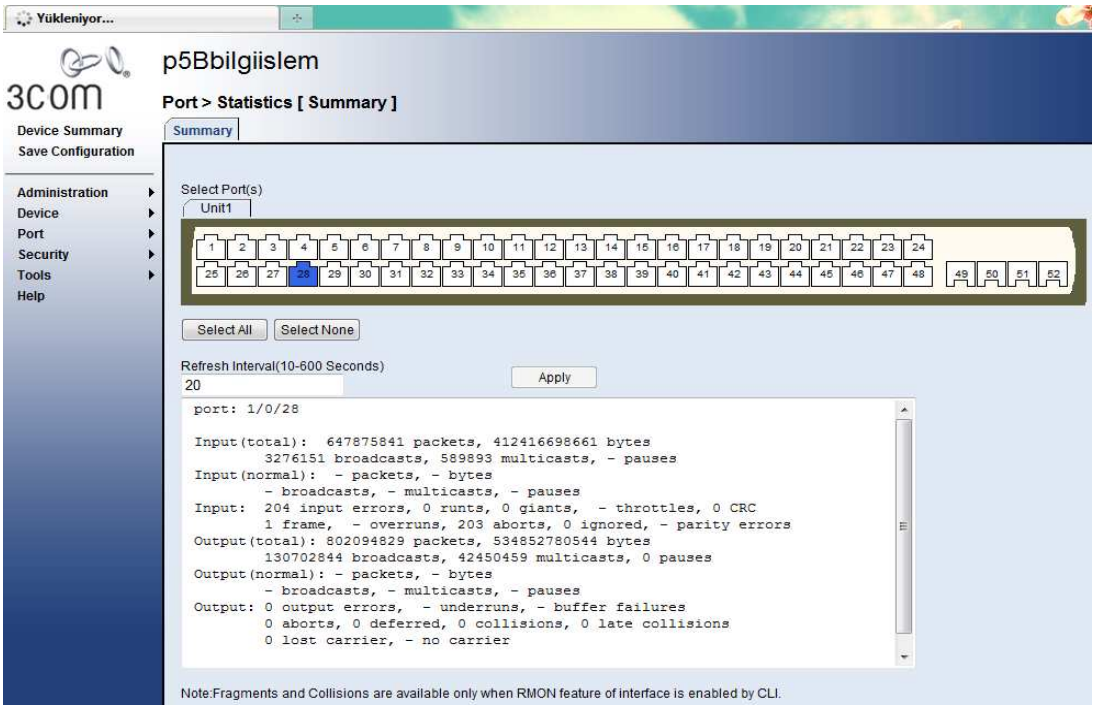


Şekil 4.22 Programın çalıştığı bilgisayarın bağlı olduğu switch port bilgisi

Şekil 4.21'e ve Şekil 4.22'ye baktığımızda programın çalıştığı makinanın kenar switch üzerindeki yirmi sekizinci portuna karşılık geldiği görülmektedir.



Şekil 4.23 Program çalışmadan önceki port istatistikleri



Şekil 4.24 Program çalıştıktan sonraki port istatistikleri

Program bu deneme için bir devir taramasını yaklaşık 4 dakikada gerçekleştirmiştir. Aşağıda program çalışmadan önceki ve sonraki gelen giden byte'lara bakılarak, programın çalıştığı ağı ne kadar yorduğu hesaplanmıştır.

Tablo 4.5 Gelen byte göre program ağı ne kadar yorar

(Çalıştıktan sonraki gelen byte)-(Çalışmadan önceki gelen byte) / zaman(sn)
412416698661 - 412396483410 = 20215251 byte
20215251 byte / 240 sn= 84230 byte/sn = **0.08 MB/sn**

Gelen byte'a göre programın ağı ne kadar yorduğunu anlamak için yapılan işlemin Tablo 4.5'deki sonucuna bakıldığında saniyede 0.08 MB gelen veri ile programın ağı yormadığı tespit edilebilir.

Tablo 4.6 Giden byte göre program ağı ne kadar yorar

(Çalıştıktan sonraki giden byte)-(Çalışmadan önceki giden byte) / zaman(sn)
534852780544-534715196158 =137584386 byte
137584386 byte /240 sn= 573268 byte/sn= **0.5 MB/sn**

Giden byte'a göre programın ağı ne kadar yorduğunu anlamak için yapılan işlemin Tablo 4.6'daki sonucuna bakıldığında saniyede 0.5 MB giden veri ile programın ağı yormadığı tespit edilebilir.

5. SONUÇ VE ÖNERİLER

Ağ değerlendirme programı ile çalışmanın plot yeri olan Pamukkale Üniversitesi Hastanelerinin ağı üzerindeki bütün bilgisayarlarının proje kapsamında alınmak istenen performans verileri C# yazılım teknolojisi kullanılarak alınmış ve veriler anlık olarak SQL veri tabanına kaydedilmiştir. Veri tabanına kaydedilen bu veriler üzerinde kullanıcının görmeyi isteyebileceği raporlar kullanıcıya sunulmuştur. Bu raporlara bakarak kullanıcı ağı üzerinde değerlendirme yapabilir ve değerlendirmesi sonucunda gerekirse istediği bilgisayarlara müdahale edebilir duruma gelmiştir. Raporda da ayrıntılı olarak anlatıldığı üzere projede sağlanmak istenen, programın sürekli bir döngü halinde ağı dolaşması ve ulaştığı bilgisayarlardan proje kapsamında belirlenen verileri alması ve aldığı verilerden anlamlı raporlar oluşturmasıdır. Projenin geçirdiği temel aşamalardan sonra, son haline bakıldığında bu başarının büyük ölçüde sağlandığı görülmektedir.

Oluşturulan raporlardan birkaç tanesine baktğımızda projenin çalıştığı sisteme katkısının olduğu kesin olarak belirlenebilir. Örnek genel rapora baktığımızda CPU kullanımının belirlenen tarih aralığında belirli bilgisayarlar için her erişildiğinde %90 dan fazla olduğu görülmüştür. Bir diğer örnekte ise yine genel raporlamadan alınan bir sonuca göre kalan RAM miktarının belirlenen tarih aralığında belirli bilgisayarlar için her erişildiğinde 60 MB'dan az olduğu görülmüştür. Bu örnekler çoğaltılabilir. Örnek raporlar, programın uzun vadede çalışması halinde çalıştığı sistemin programda belirlenen parametrelere göre kaynak durumunu takip etmek açısından yardımcı olacağına işarettir.

Gelecekte proje daha verimli çalışır ve daha anlamlı raporlar oluşturabilir hale getirilebilir. Raporlama kısmında kullanıcının isteklerine göre farklı raporlar oluşturulabilinir ve oluşturulan raporlarda sorgular farklı fonksiyonlardan geçirilip daha iyi sonuçlar elde edilebilirdi. Bu fonksiyonlara bir örnek vermek gerekirse herhangi bir ölçek veya test üzerinden alınmış puanların dağılım ortalamasından kaç birim saptığını göstererek sonuçlar arası tutarlı bir karşılaştırma yapma olanağı sağlayan istatiksel bir fonksiyon z-value fonksiyonudur. Z-value hipotezin reddedilip

edilmemesi konusunda karar vermeye yardımcı olan bir testtir. Projede kaydedilen veriler z-value fonksiyonundan geçirilirse daha anlamlı sonuçlar elde edilebilir[20].

KAYNAKLAR

[1] Xu,J.,Xu,M.,2009: Performance Monitoring Tool for Predicting Degradation in Distributed Systems, Nanjing University, China

[2] http://www.monitortools.com/performance

[3] http://www.computerperformance.co.uk/HealthCheck/netflow_monitoring.htm

[4] http://www.ciscotr.com/forum/cisco-associate/195-turkce-cisco-notlari-ccna-dokumanlari-cisco-certified-network-associate.html

[5] http://www.bilgiportal.com/v1/idx/35/911/Network/makale/OSI-Katmanlar.html

[6] http://www.hakanuzuner.com/index.php/category/cisco

[7] http://www.aspnedir.com/Article/DisplayArticle.aspx?ID=655

[8] İşli, D., 2009: Veri Ambarı ve OLAP Teknolojilerinden Yararlanılarak Raporlama Aracı Gerçekleştirimi, Bilgisayar Mühendisliği Anabilim Dalı, Fen Bilimleri Enstitüsü, Pamukkale Üniversitesi

[9] http://www.csharpnedir.com/articles/read/?id=176

[10] http://www.tidhar.co.il/Data/Uploads/threading.pdf

[11] http://ube.ege.edu.tr

[12] http://www.csharpnedir.com

[13] http://tr.wikipedia.org/wiki/Ping

[14] msdn.microsoft.com

[15] http://www.webhatti.com/lemciler/583949-cpu-central-processing-unit-merkezi-islem-birimi.html

[16] http://www.webhatti.com/donanim-bilesenleri/19857-ram-nedir.html

[17] http://nayyeri.net/how-to-calculate-network-utilization-in-net

[18] http://www.codeproject.com/KB/cs/perfgrid.aspx

[19] http://msdn.microsoft.com/en-us/library/aa645516%28v=vs.71%29.aspx

[20][http://resources.esri.com/help/9.3/arcgisdesktop/com/gp_toolref/spatial_statistics_toolbox/what_is_a_z_score_what_is_a_p_value.htm]

ÖZGEÇMİŞ

Ad Soyad: Merve KEÇELİ

Doğum Yeri ve Tarihi: 03. 10. 1984 Bafra/SAMSUN

Adres: İncilipınar mah. 1243 sok. No:17 kat 3 daire 4

Lisans Eğitimi: Pamukkale Üniversitesi Bilgisayar Mühendisliği Bölümü