TÜRK HAVA KURUMU ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

KARMAŞIK OLAY ANALİZİ İLE DAĞITIK SİSTEMLERİN İZLENMESİ

YÜKSEK LİSANS TEZİ

Ali ARAS

Bilişim Teknolojileri Anabilim Dalı

Bilişim Teknolojileri Programı

EKİM, 2016

TÜRK HAVA KURUMU ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

KARMAŞIK OLAY ANALİZİ İLE DAĞITIK SİSTEMLERİN İZLENMESİ

YÜKSEK LİSANS TEZİ

Ali ARAS

1506050003

Bilişim Teknolojileri Anabilim Dalı

Bilişim Teknolojileri Programı

Tez Danışmanı: Yrd. Doç. Dr. Erhan MENGÜŞOĞLU

Türk Hava Kurumu Üniversitesi Fen Bilimleri Enstitüsü'nün 1506050003 numaralı Yüksek Lisans öğrencisi, Ali ARAS ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı Karmaşık Olay Analizi İle Dağıtık Sistemlerin İzlenmesi başlıklı tezini, aşağıda imzaları olan jüri önünde başarı ile savunmuştur.

Tez Danışmanı: Yrd. Doç. Dr. Erhan MENGÜŞOĞLU
Türk Hava Kurumu Üniversitesi .....

Jüri Üyeleri : Yrd. Doç. Dr. Oğuz ASLANTÜRK
Türk Hava Kurumu Üniversitesi .....

: Yrd. Doç. Dr. Fuat AKAL
Hacettepe Üniversitesi .....

: Yrd. Doç. Dr. Erhan MENGÜŞOĞLU
Türk Hava Kurumu Üniversitesi .....

Tez Savunma Tarihi: 10 Ekim 2016

TÜRK HAVA KURUMU ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRLÜĞÜ'NE

Yüksek Lisans Tezi olarak sunduğum, Karmaşık Olay Analizi İle Dağıtık sistemlerin İzlenmesi adlı çalışmamın, tarafımdan akademik etik ve kurallara aykırı düşecek bir yardıma başvurmaksızın yazıldığını ve yararlandığım kaynakların kaynakçada gösterilenlerden oluştuğunu, bunlara atıf yapılarak yararlanılmış olduğunu belirtir ve bunu onurumla doğrularım.

10.10.2016
Ali ARAS

ÖNSÖZ

Yüksek Lisans tez çalışma sürecinde beni yönlendiren, karşılaştığım zorlukları bilgi ve tecrübesi ile aşmamda yardımcı olan, desteğini ve yardımını hiçbir zaman esirgemeyen tez danışmanım değerli Yrd. Doç. Dr. Erhan MENGÜŞOĞLU'na teşekkürlerimi sunarım.

Her zaman yanımda olan, maddi ve manevi desteklerini hiçbir zaman esirgemeyen eşime ve biricik kızıma teşekkürlerimi ve sevgilerimi sunarım.

Ekim, 2016 Ali ARAS

İÇİNDEKİLER

TABLO LİSTESİ

ŞEKİL LİSTESİ

KISALTMALAR

CEP : Complex Event Processing
KOİ : Karmaşık Olay işleme
Oİ : Olay İşleme
GNU : GNU is Not Unix (GNU, UNIX Değildir)
LA : Lider Ahenk Merkezi Yönetim Sistemi
GPL : General Public License (Genel Kamu Lisansı)
CPU : Central processing unit
RAM : Random-access memory
FSF : Free Software Foundation
IFP : Information Flow Processing (Bilgi Akışı İzleme)
DBMS : Database Management system (Veritabanı Yönetim sistemi)
CCL : Continous Computation Language (Sürekli Hesaplama Dili)
FIFO : First In First Out

ÖZET

KARMAŞIK OLAY ANALİZİ İLE DAĞITIK SİSTEMLERİN İZLENMESİ

ARAS, Ali
Yüksek Lisans, Bilişim Teknolojileri Anabilim Dalı
Tez Danışmanı: Yrd. Doç. Dr. Erhan MENGÜŞOĞLU
Ekim-2016, 47 sayfa

Geleneksel sunucu izleme sistemlerinde (server monitoring systems) sistemden alınan veriler uzmanlarca yorumlandıktan sonra, nasıl bir işlem yapılması gerektiğine karar verilmektedir. Bu çalışma ile, sunucu ve bilgisayar kümelerinin yönetilmesinde izleme sonuçlarına göre alınması gereken kararların otomatikleştirilmesi için yeni bir yaklaşım sergilenmektedir. Belirlenen izleme sistemine eklenecek karmaşık olay işleme (Complex event processing) özelliği sayesinde gerçek zamanlı ve doğru kararlar alınması hedeflenmektedir. Bu alınan kararlar doğrultusunda üçüncü bir desteğe ihtiyaç duymaksızın sistemin kendi kendini sürdürülebilir bir yapıda tutması sağlanacaktır. Bu sürdürülebilirlik, alarm üretme veya duruma karşılık gelen iyileştirmenin otomatik olarak gerçekleştirilmesi ile sağlanabilecektir. İzleme sistemlerinde oluşan gereksiz maliyet ve insan hatasından kaynaklanan yanlış tanımlama ile yersiz alarm üretilmesi durumu ortadan kaldırılacaktır. Bu şekilde ortaya çıkacak durumsal farkındalık ile anormal olarak değerlendirilebilecek bazı sistem durumlarının aslında anormal olmadığı anlaşılabilecektir. Karmaşık olay analizi ve geleneksel izleme sistemlerinden elde edilen sonuçlar üzerinde yapılan karşılaştırma neticesinde, karmaşık olay analizi kullanılarak fark edilen aykırı durumlardaki yanlış alarm oranının çok daha düşük olduğu gözlemlenmiştir. Testler, Linux tabanlı milli işletim sistemi dağıtımı Pardus ve Pardus kapsamında istemci

yönetimi için geliştirilen ajan tabanlı Lider Ahenk Merkezi Yönetim Sistemi üzerinde gerçekleştirilmiştir. Bakım ile ilgili sistem tarafından alınan doğru kararlar sonucu, kendini yönetme yeteneği olan bir sistem geliştirilmiştir.

Anahtar Kelimeler: Karmaşık olay izleme, Pardus, Lider Ahenk Merkezi Yönetim Sistemi, sistem izleme

ABSTRACT

ONLİNE MONİTORİNG OF DİSTRİBUTED SYSTEMS USİNG COMPLEX EVENT PROCESSİNG

ARAS, Ali

Master, Information Technology

Thesis Supervisor: Yrd. Doç. Dr. Erhan MENGÜŞOĞLU

October-2016, 47 page

In traditional server monitoring systems data collected from system is analysed by the experts in order to decide what action will be taken for better performance. We propose a new approach for automated decision-making process according to server monitoring systems results. We intend that real-time and correct decisions will be taken with addition of the complex event process to monitoring system. These decisions will keep the system self-sustainable. Self-sustainability will be achieved with alarming mechanisms or improvements. Self-sustainability will also prevent human based errors and unnecessary costs. In this way, we can discover that some abnormal situations detected without self-sustainability are not real abnormal situations. Test results taken from traditional server monitoring systems and complex event processing systems clearly shows that complex event processing systems create less false alarm.The tests were carried out on the Turkish distribution of Linux operating system, Pardus through its recently developed agent based deployment and monitoring component called Lider-Ahenk. The system with self-management capability has an improved health resulted from correct decisions taken by the system itself about maintenance.

Key words: Complex event processing, Pardus, Lider Ahenk Central Management System, monitoring systems

GİRİŞ

Sunucu ve istemcilerin sağlık durumlarını izlemek (system health and check) ve izlenen kaynaklarda meydana gelen anomalileri tespit ederek değerlendirmek için kullanılan sistemler bütünü izleme sistemleri (monitoring systems) olarak bilinmektedir. İzleme sistemleri, basit yapıdaki istemcilerden dağıtık mimarideki sunuculara kadar işlemci, bellek gibi donanım kaynaklarının kullanım durumlarının toplanması, işlenmesi ve anlamlı bütün haline getirilerek görüntülenmesi; sistem üzerinde çalışan uygulamaların ve ağ parametrelerinin değerlendirilmesi amacıyla yaygın olarak kullanılmaktadır.

Verinin değerli olabilmesi için anlamlı bütünlüğe, kısaca bilgiye dönüştürülmüş olması gerekmektedir. Bilgisayar yönetiminde ise elde edilen veriler üzerinde karar destek sistemlerini çalıştırabilmek için elde edilen verinin başka bir işleme yön vermek üzere girdi olarak kullanılabilir olması gerekmektedir. Birçok izleme sistemi veriyi bilgiye dönüştürebilmekte, ancak gerçek veya yakın gerçek zamanlı karar alma eğilimi sergileyememektedir. Geleneksel izleme sistemleri, belirlenen eşik değerlerine göre izleme yapmaktadır. Koşulların belirlenmesi ve değerlendirilmesi insan uzmanlığı ile yorumlanmaktadır. İzleme sistemlerinde insan uzmanlığıyla yapılan analizlerde insan okur yazarlığından kaynaklanan hatalar kestirilememekte ve sistemlerden toplanan veriler gerçek zamanlı olarak tam değerlendirilememektedir. Bu belirli olmayan durumlar nedeniyle, izlenen sistemler eksik kapasiteyle çalışıp kaynak israfına sebebiyet verebilecekleri gibi kapasitelerinin üstünde çalışarak kararsızlıklara da sebebiyet verebilmektedir.

İzleme sistemleri ile düzenli takip edilemeyen bilgisayar kümeleri, üzerinde çalışan uygulama yazılımları veya sistemlerinin çalışmasını etkileyecek birçok soruna neden olabilmektedir. Bu çalışmada Karmaşık Olay İşleme (KOİ) ile farklı veriler birlikte değerlendirilerek dağıtık mimarideki sunucu veya istemcilerin belirlenen kalıplar çerçevesinde ortaya çıkaracağı karakteristik özelliklerine göre önceden tanımlanmış kurallara göre yönetilebilirliği incelenecektir.

Çok farklı donanım envanterine sahip dağıtık bilgisayar sistemlerinden alınan veriler insan uzmanlığı ile incelenip analiz edilmeye çalışıldığı için önemli bazı noktalar kaçırılabilmektedir. Elde edilen her türlü veri KOİ ile analiz edildikten sonra belirlenen eşik değerleri için kaliteli bir çıktı üretmesi sağlanacağından önemli noktalar tespit edilerek, meydana gelen anomaliler anında belirlenebilmektedir. İnsan uzmanlığı ile yönetilen sistemlerin performansı insan okuryazarlığı ile sınırlı olduğundan, kendi kendini yönetebilmesine olanak sağlayan sistemlere ihtiyaç her geçen gün artmaktadır. Bu yöntem ile bilgisayar sistemlerinin belirlenen kalıplara göre yönetilmesi sağlanmakta ve KOİ ile sürdürülebilir yönetim imkanı sunulmaktadır.

İş işleme yönetimi faydalı olduğundan bu çalışmada ondan ilham alınarak bazı olay ve koşulların gözlemlenip yorumlanması ve önceden oluşturulmuş iş mantıklarının otomatik olarak gerçekleştirilmesi sağlanmaktadır. İzleme sistemine sadece parametre girişi kullanıcı tarafından yapılmakta ve bu parametreler ise belirlenmek istenen kalıpların sınırlarını oluşturmaktadır. Parametre değişikliklerinden sistem etkilenmemektedir. Bu sistemin iş kuralları motorundan farkı anlık parametreler dışında sürekli gözlem yapması ve yaptığı gözlemlere göre değişiklikleri bir sonraki karar alma sürecinde de hesaplıyor olmasıdır. Gerçek zamanlı bilgisayar hareketlerine odaklanıp, bir durum gerçekleştiğinde şart olan bir unsurun tanımlanarak gerçekleşmesi sağlanabilmektedir. Önerilen sistem sayesinde dağıtık sistemlerin izlenmesi ve yönetilmesinde kolaylık sağlanacaktır.

Sunucu ve istemci bilgisayar izleme sistemlerinde; izleme sonuçlarından anlamlı bütün haline getirilmesi için geleneksel (ilişkisel) veritabanı mantığı aşılarak yeni bir sunucu yönetim yaklaşımı oluşturulmuştur. Bu yaklaşım, sunucu ve bilgisayar kümelerinin izlenmesi sırasında elde edilen işlemci ve bellek kaynak kullanım çıktılardan sebep sonuç ilişkisi kurarak, daha önce belirlenen kurallar çerçevesinde donanım kaynaklarının yükünü dengeleyen ve sistemin çalışmasında olumsuz etki oluşturacak eylemlerin tespit edilmesini amaçlamaktadır.

Örneğin, web tabanlı bir java uygulaması bir sistem üzerinde çalıştırıldığında işlemci kaynak kullanımı %80'lerde seyrediyor ise önerilen sistem tarafında bu belirli süre izlendikten sonra durum normal kabul edilerek, işlemci sınırı bahse konu uygulama için %80'e ayarlanmaktadır. Bu şekilde işlemci kaynak kullanımının değişmesi durumunda normal bir durumun anomali olarak tanımlanmasının da önüne

geçilmektedir. İnsan uzmanlığı ile bu tür bir durumun tespit edilmesi çoğu zaman mümkün olmamakta ve tam olarak kestirilememektedir. Önerilen sistem temel olarak işlemci ve bellek kaynak kullanımları üzerine test edilmiştir.

Önerilen sistem aşağıda belirtilen durumların gerçekleştirilmesinde kolaylık sağlayacaktır.

1. İşlemci kullanım değerleri yüksekte seyir eden uygulamanın tespiti
2. Bellekte gereksiz tutulan bilgilerin (son kullanım oranları, süresi, önemi) SWAP'a kaydırılıp bellek yükünün azaltılması ve performans artırımı
3. Büyüyen log verisinin, belirli bir zamana göre döndürülmesi (log rotate) ile sıkıştırılması ve eski verinin artan yedeğinin alınması
4. Uygulamanın türüne ve özelliğine bağlı olarak ağ bant genişliği kullanımının şekillendirilmesi
5. Uzaktan uygulama iyileştirme (fix) ve yeni versiyon yüklenmesinde (deploy) optimizasyon imkanı
6. Kullanılan bilgisayar ağında port, subnet vb. tarama yaparak saldırı olup olmadığının tespiti
7. Kullanılan bilgisayar ağında meydana gelen darboğaz (bottleneck) anlarında kısıtlama uygulanması gibi özellikler bu sistem sayesinde gerçekleştirilebilecektir.

Sistem izleme araçları, büyük ölçeklerdeki bilgisayar kümelerinin yüksek erişebilirliğinin sağlanmasında katkı sağlamakla birlikte kişisel kullanıma da olanak sağlayan yapıyı sunmaktadır. Bu tür araçların yaygın etkiye sahip olabilmesi belirlenen eşiklerde istenilen durumları gerçekleştirmesi ve konforlu sistem yönetimi sağlamakla mümkün olacağından, önerilen sistem sayesinde başkaca uygulamaların da bu sistemden faydalanması amaçlanmaktadır.

Seyir halindeki araçtan elde edilen (toplanan) bilgiler değerlendirilerek, seyir konforunun artırılması ve güvenlik gibi faktörlerin ön plana çıkarılması farklı disiplinlere örnek olarak verilebilmektedir. KOİ ile birlikte, hasta veya hasta adayından alınan bilgiler (kalp atış hızı, vücut sıcaklığı, nefes alıp verme süresi vb.) ışığında kalbe takılan pil (birlikte devre) kalp krizinin yaşanacağı an çok yaklaştığında anında şok vererek kalp krizinden kaynaklı kalp durmasının önüne geçebilmektedir.

KOİ ile birlikte finans alanında yapılacak yatırımlar ilgili parametrelerin alınarak değerlendirilmesi sonucu daha karlı bir yatırım imkanı sunabilmektedir. Bunun için yatırım verileri ile birbirine bağlı olayların izlenmesi ve analiz edilmesi gerekmektedir.

Bu kurallar çerçevesinde karar destek sistemleri kullanılabilir. Sistem başarılı olduğunda elde edilen verilerden etkin bir karar yönetimi sağlanabilecektir. Bu şekilde etkin karar alınabilmesi ve içinde bulunulan sistem riskleri en aza indirgenerek karar verileceğinden belirsizlik ortadan kaldırılacaktır.

Konu ile ilgili açık problemler:

1. Sunucu izleme sistemlerinin kendi kendine alarm üretebilmesi
2. İnsan faktörü ile incelenen sistemlerdeki hata payının azaltılması
3. Geri bildirim ile koşul uygulanan başkaca problemler
4. olarak tanımlanabilmektedir.

Tezin ikinci bölümünde izleme ve karmaşık olay işleme sistemlerinin temel kavramlarına değinilerek literatür özeti verilmektedir. Üçüncü bölümde materyal ve yöntem açıklanmıştır. Önerilen sistem mimarisi ve tasarımı dördüncü bölümde anlatılmıştır. Beşinci bölümde, önerilen sistem ile kurgulanan deneysel çalışmalar ve sonuçlar ele alınmıştır. Altıncı bölümde, önerilen sistemin uygulanması sonucu elde edilen sonuçlar ve önerilere yer verilmiştir.

BİRİNCİ BÖLÜM

LİTERATÜR ÖZETİ

Bilgisayar kümelerinin izlenmesi için oluşturulmuş birçok uygulama bulunmaktadır. Önerilen sistemin daha iyi açıklanabilmesi için bilgisayar izleme sistemleri ile karmaşık olay işlemenin belli başlı önemli çalışmalarını sunacağız.

1.1 Sunucu ve İstemci izleme sistemleri

1.1.1 IBM Tivoli Monitoring

Uygulama ve işletim sistemleri kaynaklarını izlemek için oluşturulmuş ticari bir ürün olan IBM Tivoli, sistemlerden aldığı durum bilgisini görselleştirmek ve kullanıcı yetkilerini yönetmek için oluşturulmuştur. Tivoli Enterprise Portal, Tivoli Enterprise Portal Server, Tivoli Enterprise Monitoring Server ve Agents olmak üzere çeşitli bileşenleri bulunmaktadır [36]. IBM Tivoli ile izlenmek istenen sistem kaynak sınırları operatör yardımı ile belirlenerek alarm üretilmesi sağlanmaktadır. Belirlenen sınır durumlarına ulaşan kaynak değerleri ile istenilen alarmların üretilmesi sağlanabilmektedir.

1.1.2 Nagios

Bilgisayarların ağ ve sistem kaynaklarını izlemek için geliştirilmiş GPL ve FSF lisanslı açık kaynak kodlu izleme sistemidir [21]. Bu sistemde alarm üretilmesi tamamen sistem yöneticisine bağlıdır. Sistem yöneticisi bir sınır belirler, izlenen sistem kaynakları bu sınırları aştığı durumlarda alarm üretilir. Sistemin kendi kendine eşik belirleyerek alarm üretmesi Nagios ile mümkün değildir. Nagios çoğunlukla ağ sistemlerinin yönetimi için kullanılmakla birlikte sistem kaynaklarının

izlenmesi ve problem olduğunda alarm üretilebilmesi için de kullanılabilmektedir. HTTP, NNTP, PING, SMTP vb protokollerin denetlenmesi gibi temel işlemlerin yanında mail, SMS, telefon gibi sistemler ile alarm üretilmesini mümkün kılmaktadır [20].

1.1.3 Cacti

Cacti; ağ aygıtları ve portların trafiği, sistemdeki kullanıcı yükü, bellek ve işlemci yükü, disk kapasitesi gibi bilgileri SNMP protokolü ile toplayarak görselleştiren bir uygulamadır. Cacti ile izlenmek istenen sistemler için alarm kalıpları oluşturulabileceği gibi ön tanımlı olarak gelen alarm kalıpları ile de kullanılabilmektedir. Veri alma, depolama ve sunum olmak üzere 3 farklı görev ile çalışmaktadır [51][52]. Cacti de Nagios gibi operatör yardımı olmaksızın kalıp belirleyememekte ve bu kalıba göre izleme imkanı sunmamaktadır.

1.1.4 Nigel's Monitor (NMON)

Linux ve Unix tabanlı işletim sistemlerinde kullanılan izleme aracıdır. İşlemci, bellek, disk, linux çekirdek durum ve kaynaklarının izlendiği sistemdir. NMON ile elde edilen bilgiler harici bir görselleştirme programı ile analiz edilmesi gerekmekle birlikte alarm üretme özelliği de bulunmamaktadır [15].

1.2 Karmaşık Olay İşleme

Karmaşık olay işlemeyi iyi anlayabilmek için öncelikle "Olay" nedir? Onu incelemek gerekmektedir. Gerçekleşen her bir durum olay'dır [5]. Olay işleme(Oİ), bilgi teknolojilerinde bir çok sistemin temelini oluşturmaktadır. Bu sistemler enerji, sağlık, çevre, taşımacılık, finans, hizmet ve üretimi kapsamaktadır. Oİ, belli bir zaman kısıtında olay kalıplarını tespit etme, dönüştürme, filtreleme gibi metot ve araçlar kullanmaktadır [6]. Karmaşık olay işleme (KOİ) belli bir zaman diliminde bir sistemin içinde üretilen ya da dışında üretilerek sistemin içine aktarılan iki ya da daha fazla basit olayın analiz edilmesi sonrası sonuç çıkaran bir yaklaşımdır [16]. KOİ sistemleri, otomatik algılama ve yanıtlama sistemlerinde işletme faaliyet monitörlerinde (Business Activity Montior-BAM) sürekli olarak gözlemleme/karar verme/aksiyon alma ihtiyacı bulunan tüm sistemlerde etkili bir şekilde

kullanılabilmektedir [18]. İzleme sistemlerinden elde edilen olay verileri karmaşık olay işleme motoruna aktarıldığında olaylar arasındaki ilişkinin kurulması ve kalıpların doğru tanımlanmış olması halinde belirlenen kalıplar çerçevesinde işlenmesi sağlanacaktır.

İzleme sistemleri istemci bilgisayar ve sunucu yönetim alanında aşağıdaki durumlarda kullanılabilmektedir:

1 İşlemci, disk, bellek, ağ vb kaynaklarının kullanım oranlarının tespiti

2 Sistemden gelen seyir kayıtlarının (log) izlenmesi ve istiflenmesi

KOİ dağıtık mesaj tabanlı sistemlerde bilgilerin ayıklanarak elde edilmesini sağlayarak farklı disiplinler ile birlikte kullanıldığından daha büyük etkiye sahip olmaktadır. Hizmet Seviyesi anlaşmaları (SLAs) ile servis sağlayıcı ve servis alıcılar arasında hizmet kalitesinin nasıl olması gerektiği hususunu belirlenmiştir [3]. Bu standart genellikle hizmet kalitesinin özniteliklerinin deterministik ve deterministik olmayan (non-deterministik) olarak sınıflandırılabileceği bildirilmiştir. Deterministik olmayan öznitelik için, sürekli izleme yaklaşımları ile mevcut hizmet kalite değerinin tespit edilebilmesinde hizmet kalitesi için iki ana faktör bulunmaktadır. Bunlar sunucu tarafı izleme, ikincisi istemci tarafı izlemesidir [4]. Alınan destek hizmetleri doğrultusunda sunucu ve istemci tarafı izlemeleri verilen hizmet kalitesinin belirlenmesine katkı sağlamaktadır.

Anton Michlmayr ve diğerleri tarafından yapılan araştırmada karmaşık olay analizine ilişkin üç ana çıkarım yapılmıştır. Bunlardan ilki web servis hizmet kalite özniteliklerini içeren iki yaklaşımın tanıtılmasıdır. İkincisi, belirlenen iki yaklaşımın Vienna Runtime Environment for Service-oriented Computing (VRESCo) [7] ile karşılaştırılması ve hizmet seviye anlaşmasına göre basitçe olay işlemede nasıl kullanılacağının tespit edilmesidir. Üçüncüsü ise iki farklı izleme yaklaşımının izleme çerçevesinde nasıl öncülük edeceği hususudur. Daha önce yapılan çalışmalarda hizmet kalite modeli önerilmiş, ancak hizmet kalite izleme yaklaşımının nasıl yapılacağına ilişkin tartışmaya gidilmemiştir [7][8][9]. Thio ve diğerleri tarafından hizmet kalite yaklaşımı web sistemleri üzerine sunulmuştur [10]. Yazarlar benzer teknikleri düşük seviye algılama veya ağ vekil tabanlı çözümler üzerine oluşturmuşlardır. Garg ve diğerleri tarafından WebMon sistem tasarlanmış, web işlemlerini sensör toplayıcı ile performans izlemesi yaptırmışlardır [11]. Raimonid ve diğerleri tarafından tasarlanan SLA izleme sisteminde gecikme ve erişim

zamanlanmış otomata ile incelenmiştir [12]. Chau ve diğerleri benzer bir yaklaşım sergileyerek eQoSystem projesi için olay bazlı izleme geliştirmişlerdir [2].

1.2.1 Geleneksel İçerik Tabanlı Abonelik- Yayınlama Sistemleri

Geleneksel içerik tabanlı abonelik-yayınlama sistemleri [30] karmaşık olay işleme sistemlerinin tarihsel temelini göstermekte ve bu modelinde, bilgi öğeleri sisteme çok sayıda *yayıncıdan* gelen mesajlar olarak akarken, basit kuralları ise *abonelerin* ilgisini tanımlamaktadır. Birçok abonelik-yayınlama sistemi [31] büyük ölçekli senaryolarda çalışmak üzere tasarlanmıştır.

1.2.2 Rapide

Rapide [32] karmaşık olay işleme sisteminin tanımlanmasına yönelik ilk adımlardan biri olarak nitelendirilir. Bir dil dizisinden ve kullanıcıların sistem mimarilerinin modellerini tanımlayıp çalıştırabilecekleri bir simülatörden oluşur. Rapide, bir dizi bileşeni kullanarak mimariyi, olayları kullanarak da bileşenler arasındaki iletişimi modeller. Hem bir olaylar örüntüsünün bir bileşen tarafından tespitinin diğer nesillere aktarımını açıklamak, hem de genel mimari ile alakalı nitelikleri belirlemek amacıyla kullanılan bir karmaşık olay algılama sistemini içerir. Rapide işleme modeli, belirlenmiş bir modelin tüm olası olay dizilerini yakalar bu da çoklu seçim ve sıfır tüketim politikaları uyguladığı anlamına gelir.

1.2.3 GEM

Dağıtık sistemler için genelleştirilmiş bir izleme dilidir [33]. Olay üreticilerle yan yana çalışan izleyicilere dağıtık uygulanmak amacıyla tasarlanmıştır. Dağıtık şekilde birden çok izleyiciyi ilgilendiren olayların tespiti mümkün kılınır. Algılama, ağaç-tabanlı bir yapı kullanılarak gerçekleştirilir: Bir kuralın eylem tarafı, bildirim teslimatı dışında, harici yordamların yürütülmesi olasılığını ve dinamik olarak kuralların etkinleştirilmesi ve etkisizleştirilmesini de içermektedir. GEM izleyicisi, iletişim ve konfigürasyon platformunu sunan REGIS/GARWIN [34] ortamı kullanılarak C++ ile gerçekleştirilmiştir.

1.2.4 DistCED

Başka bir içerik tabanlı abonelik-yayınlama sistemidir [1]. Geleneksel abonelik-yayınlama aracı sisteminin bir uzantısı olarak tasarlanmıştır, sonuç olarak mevcut pek çok sistemin üzerine uygulanabilir durumdadır. DistCED kuralları bir tespit dili kullanılarak tanımlanmaktadır.

1.2.5 CEDR

Yazarlar, CEDR'in temellerini, genel amaçlı olay aktarma sistemi olarak tanımlamışlardır [38]. Ayrıca bilgi akışları için yeni bir zamansal model önermişlerdir. Bu modelde üç farklı zamanlama vardır; sistem zamanı, geçerlilik zamanı ve oluş/değişiklik zamanı. Bu yaklaşım uygulamaların seçebileceği çeşitli tutarlılık modellerine imkan tanımaktadır.

1.2.6 Cayuga

Cayuga genel amaçlı bir olay izleme sistemidir [39]. CEL (Cayuga Event Language – Cayuga Olay Dili) adı verilen bir dil tabanlıdır. Dilin yapısı, geleneksel bildirimsel veritabanı dillerine çok benzemektedir. Tüm olayların bir süresi vardır ve operatörlerin birleştirilebilirliğine çok önem verilmiştir. Bunu yapabilmek için, tüm operatörlerin semantiği bağıntı cebri kullanılarak tanımlanmıştır [40].

1.2.7 NextCEP

NextCEP bir dağıtık karmaşık olay işleme sistemidir [41]. Cayuga'ya benzer şekilde, filtreleme ve isimlendirme gibi geleneksel SQL operatörleri ile diziler ve yinelemeler gibi model algılama operatörlerini bir arada içerir. Algılama, kuralları deterministik olmayan otomata çevirerek yapılır, bu da Cayuga yapı ve şemantiğinde tanımlananlara benzerlik gösterir. Ancak NextCEP'de algılama, sıkı bağlanmış düğüm dizileriyle dağıtık bir şekilde yapılabilir (kümeli ortam).

1.2.8 PB-CED

Bu modelde yazarlar, dağıtık kaynaklardan aldıkları verilerle karmaşık olay algılaması yapan bir sistem sunmaktadırlar [42]. Bu yaklaşıma Plan-Tabanlı Karmaşık Olay Algılaması (Plan-Based Complex Event Detection – PB-CED) adını vermişlerdir. Bu çalışmanın vurgusu, kuralların değerlendirilmesi için verimli bir plan tanımlamakta ve kaynaklardan yapılabilecek gereksiz veri aktarımını olabildiğince sınırlamaktadır.

1.2.9 Raced

Raced bir dağıtık karmaşık olay işleme ara yazılımıdır [43]. Padres'e oldukça benzeyen basit bir algılama dili sunar. Kümeleri hesaplamak için herhangi bir işleme yeteneği sunulmamaktadır. Padres gibi, Raced de büyük çaplı senaryolar için tasarlanmıştır ve dağıtık algılamayı destekler.

1.2.10 Amit

Amit, reaktif ve proaktif uygulamaların hızlı ve güvenilir gerçekleştirim sağlamak için geliştirilmiş bir uygulama geliştirme ve işleyiş kontrol aracıdır [44]. Amit, *durum yöneticisi* olarak adlandırılan bir bileşeni yerleştirir, bu bileşen özellikle farklı kaynaklardan gelen bildirimleri işlemek, bunlardan *durumlar* olarak adlandırılan ilgili kalıpları tespit etmek ve talep eden *abonelere* yönlendirmek için tasarlanmıştır. *Durum yöneticisi* ifade etme gücü yüksek ve son derece esnek bir algılama diline dayanmaktadır. Bu dil birleşimler, olumsuzlamalar, parametreler, diziler ve tekrarları (sayım operatörleri şeklinde) içerir. Zamanlama operatörleri de kuralların periyodik değerlendirmesini sağlamak amacıyla sunulmuştur. Amit, Java kullanılarak gerçekleştirilmiştir ve IBM Global Services'ın E-Ticaret Yönetim Hizmeti'nin arkasındaki çekirdek teknoloji olarak kullanılmaktadır [45]. Merkezi bir tespit stratejisini kullanır, tüm olaylar tek bir düğümde saklanır ve geçerli olabilecekleri yaşam sürelerine göre bölümlendirilirler. Bir sonlandırıcı alındığında, Amit bu durumun tespiti için tüm koşulların karşılanıp karşılanmadığını değerlendirir ve eğer karşılanmışsa, abonelere bildirim yapar.

1.2.11 Sase

Sase, RFID okumalarının gerçek-zamanlı akışlarının üzerinde karmaşık sorgular yapmak için dizayn edilmiş bir izleme sistemidir [46]. Sase, algılama kuralları dilini modeller üzerinde tanımlar. Her kural üç parçadan oluşur: *olay, nerede* ve, *neyin dahilinde*. *Olay*, hangi bilgi öğelerinin algılanması gerektiğini ve bu öğelerin arasındaki ilişkiyi belirler. İlişkiler mantıksal operatörler ve diziler kullanılarak ifade edilir. *Nerede* olduğu, olay teriminin içinde bulunan bilgi öğelerinin iç yapısı üzerindeki sınırlamaları belirler. *Neyin dahilinde* olduğu, kuralın geçerli olacağı süreyi ifade eder; bu sayede zaman-tabanlı, kayan pencereler tanımlamak mümkündür. Sase tarafından kabul edilen dil sadece bilgi öğelerinin modellerinin algılanmasını mümkün kılar, ancak hiçbir şekilde kümeleşme kavramını içermez.

1.2.12 Sase+

Sase+, Sase'nin yazarlarının ortaya koyduğu bir ifadeci olay işleme dilidir [47]. Sase+, Sase'nin ifadeciliğini, yinelemeler ve kümeleri model algılamanın olası parçalarına dahil ederek geliştirir. Model algılamak ve dilin kesin anlanmasını sağlayabilmek için deterministik olmayan otomata kullanılmaktadır. Sase+ kullanıcılara seçim politikalarını özelleştirmek için *stratejileri* kullanma imkanı vermektedir. Seçim stratejileri otomata geçişi için hangi olayların geçerli olduğunu belirlemektedir. Yapılan önemli bir katkı da dilin ifade gücü ve algılama algoritmasının karmaşıklığının biçimsel bir analiz haline getirilmesidir [48]. Bu analiz bir yandan diğer dillerle doğrudan kıyas yapılabilmesini sağlarken, öte yandan sadece limitli bir model-tabanlı dil grubuna uygulanabilmektedir ve IFP alanında tanımlanan çok sayıda dili henüz ele alamamaktadır.

1.2.13 Peex

Peex (Probabilistic Event Extractor), RFID verisinden karmaşık olayları çıkarmak için tasarlanmış bir sistemdir [49]. Peex'in en önemli katkısı veri belirsizliğine verdiği destektir. Özellikle de veri hatalarına ve belirsizliğine yönelmekte ve tüm bilgi öğelerine bir olasılık atayarak veri modelini değiştirmektedir. Sistem yöneticileri, kuralları tanımlarken olaylara olasılık

atayabilirler. Örneğin verilmiş bir zaman diliminde üç okuma algılandı ise "John 50 numaralı odaya girer" olayı %70 olasılıkta, "John 51 numaralı odaya girer" olayı ise %20 olasılıkta olabilir (olasılıkların toplamı %100 olmak zorunda değildir). Peex, kullanıcıların olay tanımlarını diğer olayları tanımlarken tekrar kullanmalarına izin verir. Peex'in bir başka önemli yanı da *kısmi olaylar* üretilmesine imkan tanımasıdır, olayı oluşturan bazı parçalar kayıpsa bile bir olay oluşturulabilir. Gerçekleştirim açısından bakıldığında, Peex kaynaklardan gelen tüm bilgiyi ve güvenilirlik bilgisini sakladığı ilişkisel DBMSler kullanır. Kurallar SQL sorgularına dönüştürülür ve periyodik olarak çalıştırılır. Bu sebeple, olayların algılanma zamanı, gerçekten meydana geldikleri zamandan farklı olabilir.

1.3 Ticari İzleme Sistemleri

1.3.1 Coral8 CEP Engine

Coral8 CEP Engine [19] ismine rağmen, bir veri akış sistemi olarak sınıflandırılabilmektedir. Aslında, bilgi akışlarının bir ya da daha fazla işleme adımıyla, SQL-benzeri ve bildirimsel bir dil olan CCL (Continous Computation Language–Sürekli Hesaplama Dili) kullanılarak dönüştürüldüğü bir işleme paradigması kullanır. CCL tüm SQL ifadelerini içerir; buna ek olarak zaman-tabanlı veya sayım-tabanlı pencereler oluşturulabilmesi, tanımlanmış bir pencerede veri okunup yazılabilmesi ve çıktı akışının bir parçası olarak öğelerin teslimatı için terimler sunar. CCL ayrıca model eşleştirme için birleşimler, olumsuzlamalar ve serilerden oluşan basit yapılar sunmaktadır, ancak yineleme desteklenmez. Aleri'de olduğu gibi, Coral8 motoru mutlak zaman modelinin varlığına güvenmemekle birlikte, kullanıcılar akıştan akışa işlem sırasını seçebilirler. İşlemenin sonuçları iki şekilde elde edilebilir: bir çıktı akışına abone olarak (itme) veya umumi bir pencerenin içeriklerini okuyarak (çekme).

Coral8 CEP motoruyla birlikte geliştirme ve yerleştirme için Coral8 Studio olarak adlandırılan bir grafik ortam sunmaktadır. Bu araç veri kaynaklarını belirlemede ve farklı işleme kurallarını, bir bileşenin çıktısı diğerinin girdisi olacak şekilde bir plan çizerek, grafiksel olarak birleştirmede kullanılabilir.

Aleri gibi, Coral8 CEP motoru da merkezi ve kümelendirilmiş bir ortamda çalışabilir. Kümelendirilmiş yerleştirmeye destek verilerek sistemin mevcudiyeti,

hatalar oluşması durumunda bile, daha kararlı sağlanmış olur. Yük atma (Load shedding), yöneticiye her bir girdi akışı için maksimum bir oran belirleme imkanı sunularak gerçeklenmiştir.

1.3.2 StreamBase

StreamBase [53] farklı cinsten kaynaklardan bilgi toplamak için bir grup bağdaştırıcı, veri akışı işleme sistemi ve Eclipse üzerinde geliştirici aracı içeren bir yazılım platformudur. Coral8 CEP motoruyla birçok benzerliği vardır: Özellikle kural tanımlaması için StreamSQL adında SQL-benzeri, bildirimsel bir dil kullanmaktadır. Kullanıcı tarafından oluşturulmuş Java ya da C++ ile yazılmış fonksiyonlar kolaylıkla özel kümeler olarak eklenebilmektedir. Bir başka özellik ise, sistemden işlenmiş verinin bir kısmının tarihsel analiz için kalıcı olarak saklanabilmesidir. StreamBase hem merkezi hem de kümeleşmiş yerleştirmeleri destekler; ağ bağlantılı yerleştirmeler de kullanılabilir ve sorun oluşması durumunda mevcudiyetin korunmasını sağlar. Kullanıcılar her sunucu için maksimum yükü tanımlayabilir, fakat dokümantasyon bu sınırlamaların sağlanması için yükün gerçekte nasıl dağıtılacağını belirtmemektedir.

1.3.3 Oracle CEP

Oracle [50] olay-güdümlü mimari yazılım paketini 2006'da piyasaya sürmüş, 2008'de ise BEA'nın WebLogic Olay Sunucusu'nu ekleyerek bugün “Oracle CEP” olarak bilinen, gerçek-zamanlı bilgi akışı işleme sistemini ortaya çıkarmıştır. Oracle CEP, kural tanımlama dili olarak CQL kullanmaktadır. Ancak Coral8 ve StreamBase'e benzer şekilde; birleşimler, parçalanmalar ve sıralar içeren model algılama sunmak için bir grup ilişkiden ilişkiye operatör eklenmiştir. Bu model dilinin bir başka özelliği olarak, kullanıcılar kuralların seçim ve tüketim politikalarını programlayabilmektedir. Coral8 ve StreamBase’de olduğu gibi görsel, plan-tabanlı bir dil Eclipse’e dayalı bir geliştirme ortamı içerisinde kullanılabilir. Bu araç kullanıcılara karmaşık bir yürütme planı içine basit kuralları bağlamak için olanak sağlar. Oracle CEP, mevcut Oracle çözümleri ile entegre haldedir. Ayrıca tarihsel verilerin analizi için kullanılan araçların yanı sıra kümelenmiş bir ortamda dağıtık işlemeye imkan tanıyan bir teknoloji içerir.

1.3.4 Esper

Esper [35] önde gelen bir açık kaynak KOİ sağlayıcı olarak kabul edilir. Esper kural tanımlamaları için zengin bir bildirimsel dil tanımlar, buna Olay İşleme Dili (EPL-Event Processing Language) denir. EPL, SQL'in tüm operatörlerini içerir; buna pencere tanımlama, etkileşimi ve çıkış üretimi için doğaçlama (ad-hoc) bir yapı ekler. Esper dili ve işleme algoritması Java ve.Net (NEsper)'e kütüphaneler olarak entegre edilmiştir. Kullanıcılar kendi programlarından yeni kurallar ekleyebilir ve itme-tabanlı modda (dinleyiciler kullanarak) ya da çekme-tabanlı modda (yineleyiciler kullanarak) çıkış verilerini alabilirler. Esper hem merkezi hem de kümelenmiş yerleştirmeleri destekler; EsperHA (Esper High Availability) mekanizmaları kullanılarak farklı, iyi-bağlantılı düğümlerin işlem gücünden faydalanmak ve bu sayede sistemin mevcudiyetini arttırmak ve sistem yükünü özelleştirilebilir QoS (Servis Kalitesi) politikalarına göre paylaştırmak mümkündür.

1.3.5 IBM WebSphere Business Events

IBM, 2008'te KOİ'nin öncü sistemlerinden AptSoft'u bünyesine katması ile adını WebSphere Business Events olarak değiştirmiştir [29]. Bugün, WebSphere platformu içine tümüyle bütünleşmiş daha hızlı bir işlem için kümelenmiş bir ortama yerleştirilebilen bir sistemdir. IBM WebSphere Business Events grafiksel, model-tabanlı bir dilde kullanıcıların kuralları yazmalarına yardımcı olan bir ön-yüz sağlamaktadır. Olaylar arasında mantıksal, nedensel, ve zamansal ilişkilerin algılanmasına imkan tanıyan bir dildir, Rapide ve Tibco Business Events'e benzer bir yaklaşımı kullanır.

1.3.6 IBM System S.

System S'nin temel fikri, yeni bilgi ortaya çıkarmak için yüksek hızlı veri akışlarını işleyen bir hesaplama altyapısı sağlamaktır. İşleme kısmı İşleme Elemanları (Processing Elements – Pes) olarak adlandırılan basit operatörlere bölünmüştür. System S, SPADE diye adlandırılan SQL-benzeri bildirimsel bir dil kullanarak belirlenen kuralları kabul etmektedir. Sunulan sistemlerin çoğundan farklı olarak, System S kullanıcıların kendi işlem elemanlarını tam özellikli bir programlama dili kullanarak yazmalarına izin verir. Bu şekilde kullanıcılar görsel

olarak her türlü fonksiyonu yazabilirler, tam olarak ne zaman giriş akışından bir bilginin okunacağına, neyin kaydedileceğine, bilginin nasıl birleştirileceğine ve ne zaman yeni bilgi üretileceğine karar verebilirler. System S, işlem elemanlarını kümelenmiş ortamlara yerleştirerek büyük çaplı senaryoları destekleyebilecek şekilde tasarlanmıştır. Bu şekilde farklı makineler birlikte çalışarak istenen sonuçları üretebilmektedirler. İlginç şekilde, System S önceki işlem yükü hakkında bilgileri kullanarak daha iyi bir işleme planı hesaplayan ve istenen QoS (Servis Kalitesi) için operatörleri alandan alana taşıyabilen bir izleme aracına da sahiptir.

1.3.7 Event Zero

Bağlantılı olay işleme aracılarına dayalı benzeri bir mimari Event Zero içinde de kullanılmıştır (EventZero 2009) [28]. Event Zero, bilgi öğeleri geldikçe onları yakalama ve işleme amaçlı bir yazılım setidir. Event Zero'nun önemli bir özelliği de kullanıcıları için gerçek-zamanlı sunumlar ve analizleri desteklemesidir.

1.3.8 Progress Apama Event Processing Platform

Progress Apama Olay İşleme Platformu [26], sunduğu çözümler ve piyasadaki güçlü varlığından ötürü bir piyasa lideri olarak tanınmıştır (Gualtieri ve Rymer 2009)[22]. Kural tanımlama, test etme ve yerleştirme için bir geliştirme aracı ile algılama için yüksek performanslı bir motor sunmaktadır.

Yapılan bu çalışmalar incelenmiş, ancak sunucu izleme sistemlerinde karmaşık olay işleme ile kalıp çıkartan ve kalıp işleyen bir yapı ile karşılaşılmamıştır. Çalışmamızın başaralı olması ile önerilen tekniğin başkaca problemlerin çözümü uygulanmasını mümkün kılacaktır. Karmaşık olay analizi sistemlerine uygun ve açık kaynak kodlu olması bakımından Lider Ahenk Merkezi Yönetim Sistemi seçilmiştir.

İKİNCİ BÖLÜM

MATERYAL VE YÖNTEM

Önerilen sistemin gerçekleştirilebilmesi için Pardus işletim sistemi ve açık kaynak alt projesi Lider Ahenk Merkezi Yönetim Sistemi (LA) kullanılacaktır. Sistemin tasarımı için KOİ motoru oluşturularak LA içine eklenti olarak eklenecektir. Önerilen sistemin örneklenmesi için işlemci ve bellek kaynak kullanımları izlenerek analiz edilecektir. Analiz LA içine konuşlandırılmış KOİ motoru ile gerçekleştirilecektir. KOİ moturu, toplanan verileri gerçek veya yakın gerçek zamanlı olarak analiz ederek, daha önce tanımlanmış şablonlar dahilinde alarm üretilmesini sağlayacaktır.

2.1 Pardus İşletim Sistemi

Linux çekirdeği 1991 yılında, Finlandiya'lı bilgisayar mühendisi Linus Torvalds tarafından geliştirilmeye başlanmış, tüm dünyadan bir çok programcının desteğini de alarak hızla gelişmiş özgür ve açık kaynak kodlu bir yazılımdır.

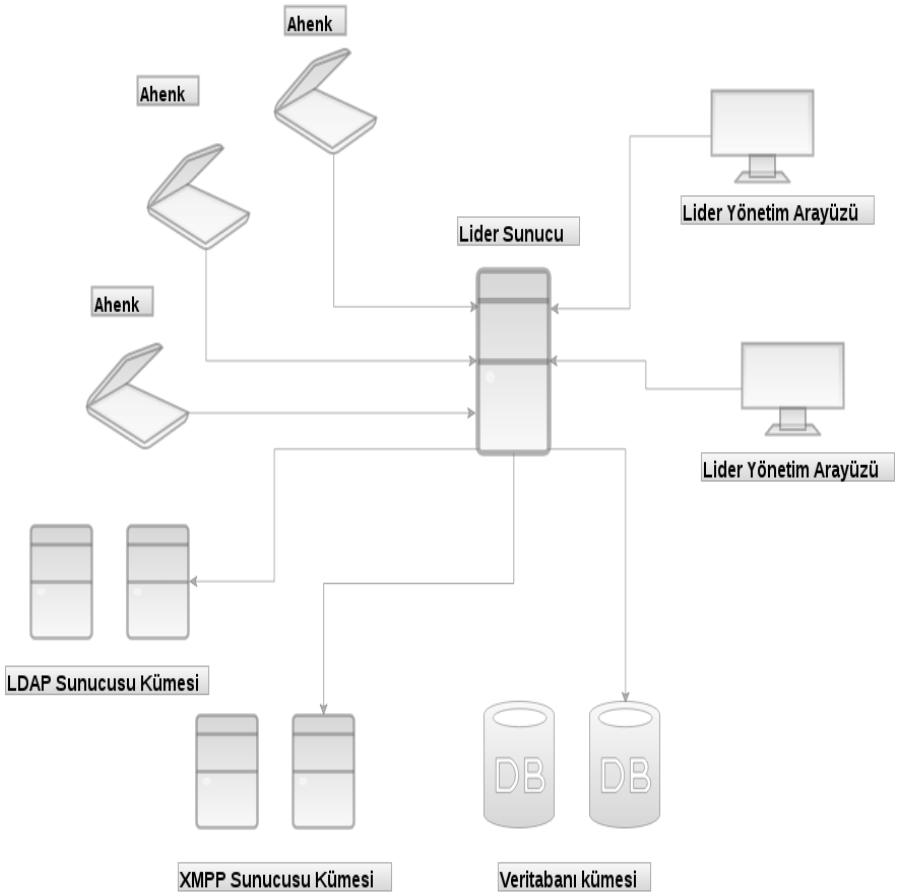
Pardus, 2003 Yılında TÜBİTAK tarafından Linux çekirdeği kullanılarak PİSİ paket sistemi ile oluşturulan milli işletim sistemi dağıtımıdır [27]. PİSİ paket altyapısı 2012 yılında revize edilerek DEB paket altyapısına dönüşüm sağlanarak sistem güncelliği ve paket oluşturma için harcanması gerekli zamandan tasarruf sağlanmıştır. Pardus işletim sisteminin GNOME, KDE, XFCE masaüstü ortamlarını barındıran çeşitli dağıtımları bulunmaktadır. Pardus işletim sisteminin kurumsal desteği TÜBİTAK tarafından sağlanmakta, ayrıca açık kaynak topluluğu tarafından da destek verilmektedir. Pardus diğer Linux işletim sistemleri gibi özgür bir yazılım olarak topluluk tarafından da destek verilmektedir. Pardus'un açık kaynak kodlu yapısı sayesinde art niyetli bir kod bütünü bulundurup bulundurmadığı programlama bilgisi olanlar tarafından denetlenebildiğinden kullanıcı bakımından sistem

mahremiyeti bulunmamaktadır. Bu şekilde kullanıcı sistemi istediği gibi denetleyebilmektedir. Pardus işletim sistemi kişisel ve kurumsal kullanım için uygun olmakla birlikte, kurumsal kullanım için çeşitli alt projeleri bünyesinde barındırmaktadır. En önemli alt projelerinden bir tanesi ajan yüklü bilgisayarları ve kullanıcıları uzaktan yönetmeye yarayan bir sistemi yazılımıdır. Pardus'un kamu kurum ve kuruluşlarında yaygınlaştırılması ve sistem uzmanları tarafından kolay yönetilebilmesi için Lider Ahenk Merkezi Yönetim sistemi TUBİTAK tarafından geliştirilerek sunucu ve istemci bilgisayarların izleme ve yönetim faaliyetlerinin yapılabileceği platform haline getirilmiştir [17].

2.2 Lider Ahenk Merkezi Yönetim Sistemi (LA)

Lider Ahenk Merkezi Yönetim sistemi [13] orta ve büyük ölçekli kurum ve kuruluşların Pardus Linux tabanlı (Pardus, Debian vb) istemci ve sunucularını dağıtık veya merkezi bir mimaride yönetmelerine imkân sağlayan yönetim sistemi olarak tanımlanmaktadır [14].

LA mimarisi Şekil 2.1'de belirtildiği gibi büyük ölçekli bilgisayar kümesine sahip kamu kurum ve kuruluşları ile KOBİ ihtiyaçlarının verimli bir şekilde karşılanabilmesi için Lider Sunucu, Lider Arayüz ve Ahenk olmak üzere üç temel bileşenden oluşmaktadır. OSGI teknolojisi kullanılarak java platformunda geliştirilmiş REST ve XMPP iletişim protokolü kullanılarak istemci yönetimi sağlamaktadır. Üzerindeki her bir yönetim özelliği bir bileşen ve/veya eklenti olarak adlandırılmakta olup, uzaktan yönetme/izleme, uygulama yönetimi, betik çalıştırma başta olmak üzere birçok özelliği mevcuttur.



Şekil 2.1: Lider ahenk merkezi yönetim sistemi mimarisi.

2.2.1 Lider Sunucu

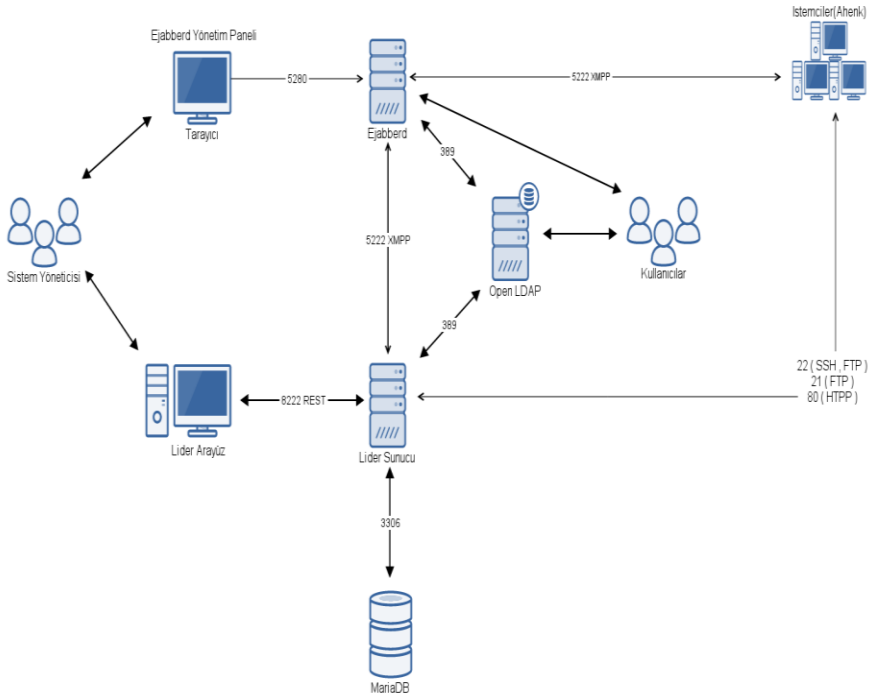
Eklentiler için ortak altyapı ve servis hizmeti sunan, saha eklenti yöntemi ile uzaktan kurulum, güncelleme, eklenti sürüm kontrolü ve sistem izlemesi yapan bileşendir. Tüm hesaplama ve depolama işlemleri lider sunucu üzerinde gerçekleştirilmektedir. Lider Arayüz üzerinden gönderilen talimatlar lider sunucuda toplanmakta ve işlem görmektedir. Lider sunucunun temel bileşeni Apache Karaf'tır. Bu özelliği sayesinde konteyner yazılım geliştirme yöntemi uygulanabilmektedir.

2.2.2 Lider Arayüz

Eclipse RCP üzerine Java ile geliştirilmiştir. Temel amacı, Lider Arayüz ile tanımlanan işleri Lider Sunucu'ya aktararak gerçekleştirilmesini sağlamaktır. Kullanıcı ve bilgisayar yönetimleri için LDAP editörü barındırmaktadır. Bu LDAP editörü aracılığı ile kullanıcılar ve bilgisayarlar tanımlanabilmekte, tanımlı bilgisayarlara da yönetim talimatları gönderilebilmektedir.

2.2.3 Ahenk Ajan

Python ile geliştirilmiş lider sunucu ile XMPP ve REST haberleşmesi kullanan eklentilerin uygulandığı bütünüdür. Gerektiğinde aktif olabildiği gibi üzerindeki eklentiler için geliştirmeye uygun ortak bir altyapı sunmaktadır. Aynı anda birden fazla bilgisayar yönetimi için çok kodları ve mimarisi minimal tutulmuştur. Lider sunucu, Ahenk ajan kurulu bilgisayarlarda işlemci ve bellek kaynak kullanım miktarlarını sorunsuz toplayabilmektedir. İşlemci ve bellek kaynak kullanımlarının analiz edilmesi için Lider Sunucu üzerinde toplanmaktadır. Analiz edildikten sonra ortaya çıkan durum Lider Arayüz üzeriden kullanıcıya gösterilmektedir. Ayrıca alarm şablon değişiklikleri ile alarm tanımlamaları Lider Arayüz üzerinden gerçekleştirilebilmektedir. Kamu kurumlarında çoğunlukla, lisans ve destek ücretleri karşılığında yardım alınabilen kapalı kaynak işletim sistemleri kullanılmaktadır. Bu işletim sistemlerinin yönetilebilmesi bilgisayar sayısına, sistemlerin ölçeğine (bilgisayar ve kullanıcı sayısına) göre yüksek maliyetler ortaya çıkmaktadır. LA, GPL lisansı ile açık kaynak kodlu oluşturulduğundan ücretsiz indirilip kullanabilmekte ve geliştirilebilmektedir. Lider Sunucu, Arayüz ve Ajan arasındaki haberleşme protokolleri Şekil 2.2'de belirtilmiştir.



Şekil 2.2: LA Mimari yapı ve kullanılan protokoller.

2.2.4 Lider Ahenk Merkezi Yönetim Sistemi Yetenekleri

Kullanıcı ve bilgisayar sistemlerinin takip ve izlenmesi için gerekli yönetim özelliği ve araçlarını barındıran LA Tablo 2.1'de belirtilen 28'dan fazla eklentiyi barındırmaktadır. Her bir eklenti bilgisayar ve kullanıcı yönetimi için amacına özel oluşturulmuş özelliklerini kapsamaktadır. İstemciler için görev, kullanıcılar için politika uygulanmaktadır. Sistem yöneticisi uygulamak istediği politika ve göreve karşılık gelen işi yerine getirecek betiği oluşturarak bu eklenti üzerinden uygulayabilmektedir.

Tablo 2.7: Lider ahenk merkezi yönetim sistemi temel özellikleri.

USB Yönetimi	Güncelleme Yönetimi
Optik Sürücü Yönetimi	Bildirim Yönetimi
Yazıcı Yönetimi	Ekran Koruyucu Yönetimi
Görüntü Ses Kayıt Politikaları Yönetimi	Ekran Görüntüsü Alma
Kablosuz Ağ Yönetimi	Masaüstü Arkaplan Yönetimi
Disk Kotası Yönetimi	Uygulama Yönetim,
Bağlı Dizin Yönetimi	Dosya Paylaşım
Ağ Yönetimi	Kaynak Kullanımı
Yedekleme Yönetimi	Oturum Açma
Servis Yönetimi	Betik Yönetim
Dosya Yönetimi	Yerel Kullanıcı Yönetimi
Yetkili Kullanıcı Yönetimi	Ateş Duvarı Yönetimi
Parola Değiştirme Yönetimi	Antivirüs Yönetimi
Parola Politikaları Yönetimi	Web Browser Yönetici

Önerilen sistem *Kaynak Kullanım* eklentisi temel alınarak geliştirileceğinden sistem mimarisi ve geliştirilen kodlar GPLv3 ile açık kaynak gönüllüleri ile paylaşılacaktır.

ÜÇÜNCÜ BÖLÜM

ÖNERİLEN SİSTEM

Bilgisayar ve sunucu sistemlerinin kaynak kullanım durumlarında anomali meydana geldiğinde tespit edilmesi ve sistemin normal durum bilgisinin dinamik olarak izlenebilmesi açısından günlük tutulması gerekmektedir. Sistemin normal durumuna, kısaca günlükteki son değere *bilgi nesnesi* (BN) olarak isimlendirilmektedir. BN anlık elde edilen veriler ile kıyaslanarak sistemdeki anlık değişikliklerin (anomali) tespitini mümkün kılmaktadır.

İzlenmek istenen sistemden toplanan işlemci ve bellek kullanım verilerinin KOİ motoru ile bilgi nesnesi haline dönüştürülmektedir. Bu bilgi nesneleri yeni gelen veriler ile devamlı güncellenmektedir. Bu güncellemeler ile sistem güncelliği sağlanarak önerilen sistemin başarısını artıracaktır. Sistemden alınan olay nesneleri Linux sistemlerin elverdiği ölçüde hazır kalıplar halinde toplanmakla birlikte bu hazır kalıp nesnelerin mesaj kuyruğu şeklinde tanımlanarak işlem kümelerine iletilmesi sağlanmaktadır.

3.1 Önerilen Sistem Kuralları ve Çalışma Şekli

Önerilen sistem, işlemci ve bellek donanım kaynak kullanımını toplamaktadır. Toplanan kaynaklarının ortalaması alınarak bilgi nesnesi oluşturulmaktadır. Kuyruğa her eklenen yeni veri ile bilgi nesnesi güncellenmektedir. Ayrıca oluşan son durum ile alarm kalıpları karşılaştırılmaktadır. Sistemin bir bütün halinde bu işlemleri gerçekleştirebilmesi için üç temel bileşen ile önerilen sistemin çalışma şeklini açıklamak mümkündür. Bunlar; olay algılama, analiz/hesaplama ve aksiyon'dur [23],[24],[25].

3.1.1 Olay Algılama

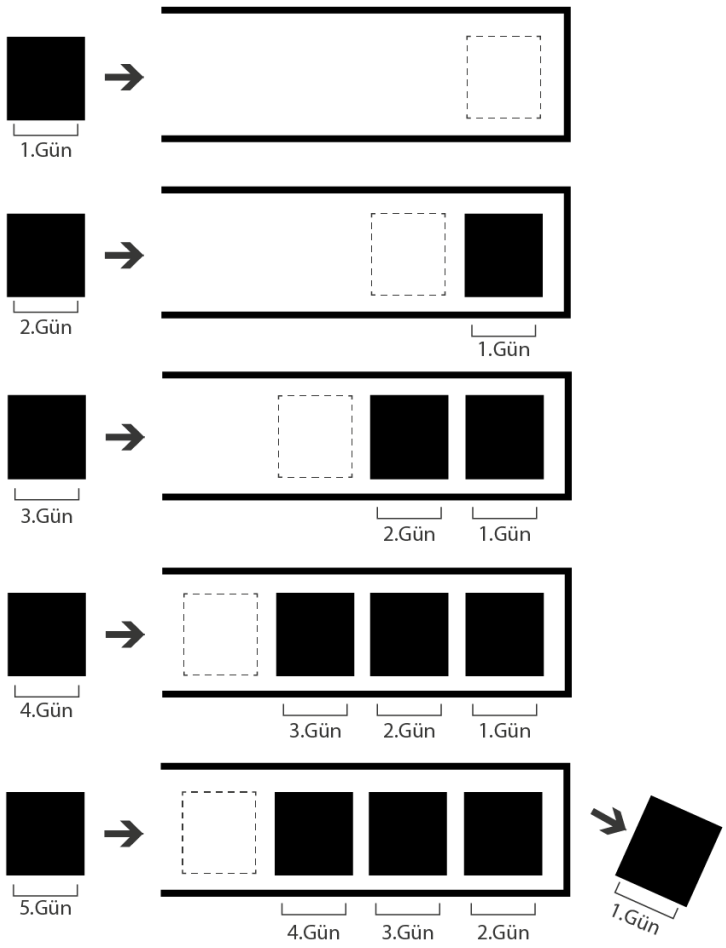
Öncelikle olaylar tespit edilmektedir. Gerçekleşen her bir durum olayı ifade etmektedir. Önerilen sistemde ise olay, işlemci ve bellek kaynak kullanım durumları olarak kabul edilmektedir. Tespit edilen olaylar kategorilerine göre filtrelenmekte, kümeleştirme işlemi yapılarak anlamlı kalıplara dönüştürülmektedir. Bu kalıplarda olay hiyerarşisi modellenmekte ve olaylar arası ilişkiler nedensellik ve sonuç odaklı olarak tespit edilmektedir. İstemcilerden veya sunuculardan toplanan kaynak kullanım verileri olaylar halinde tanımlanmaktadır. Toplanan veriler ahenk ajan bünyesinde oluşturulmuş çeşitli araçlar aracılığı ile toparlanıp işlenmeye hazır hale getirilmektedir.

3.1.2 Bilgi Nesnesi Oluşturma ve Öğrenme Yöntemi

BN anlık elde edilen veriler ile kıyaslanarak sistemdeki anlık değişikliklerin (anomali) tespitini mümkün kılmaktadır. BN'nin tespit edilebilmesi için denetimli (Semi-supervised) ve denetimsiz (unsupervised) olmak üzere iki farkı yaklaşım kullanılacaktır. Bu yaklaşımlar ile beklenen değer tespit edilerek her gün toplanan değerlerin ortalama kuyruğuna eklenmesi ile yeni bir değer ortaya çıkmaktadır. Bu yöntem Kayan ortalama (Floating Average) yöntemi olarak isimlendirilmiştir.

Örneğin, 1 Nisan 2016 tarihinde 4 günlük BN hesaplansın. Bu değer 5 Nisan 2016 tarihinde tekrar hesaplanacaktır. Kuyruğa her eklenen güne ait kaynak kullanım verileri ortalamaya dahil edilerek güncellik sağlanacaktır. Ancak beklenen değerinin güncelliğinin devamlı takip edilebilmesi için bu değer her geçen gün güncellenecek ve BN bu yeni değerler ile karşılaştırılacaktır. İlk başta 4 günlük bir çerçeve oluşturulması için ayar girildiğinden, her yeni gün eklendiğinde son 4 güne ait değerler hesaplanacaktır. Bu şekilde ilk giren kuyruktan ilk çıkmış olacaktır. Bu durumda 6 Nisan 2016 tarihinde beklenen değerin hesaplanması için 2,3,4 ve 5 Nisan 2016 günlerine ait verilerin ortalaması alınacaktır. 7 Nisan 2016 tarihinin beklenen değerin hesaplanması için 3,4,5 ve 6 Nisan 2016 günlerine ait veriler hesaplanacaktır. Görüleceği üzere hesaplamaya yeni bir gün dahil edildiğinde hesaplama kuyruğundaki ilk gün ortalamadan çıkarılmaktadır. Buna biz kayan pencere (floating window) yöntemi denilmektedir. Bu yöntem ile veri devamlı güncellenecek ve yeni gelen değerler ile karşılaştırmaları yapılabilecektir.

Yukarıdaki örnekte bahsedildiği gibi istemcilerde ve sunuculardaki tüm süreç ahenk ajan ile takip edilmektedir. Toplanan veriler Lider Sunucu veri tabanına kayıt edilerek verilerin güvenliği sağlanmaktadır. Kaç günün ortalaması alınması isteniyorsa o gün sonunda ortalama hesaplanarak tekrardan izlemeye devam ettirilebilmektedir. Bu şekilde işlemci ve bellek için Bilgi Nesneleri gibi kalıp oluşturulabilmektedir. Bilgi nesnesi oluşturabilmek için dinamik kuyruk yapısı kullanılacaktır. İzleme yapılması için oluşturulan gün sayısı her yeni günün verisinin alınması ile kayan pencere yaklaşımı ile güncellenerek FIFO yaklaşımı gözetilmiştir. Örneğin: 4 Günlük periyotlar ile oluşturulacak bilgi nesneleri için çalışacak kuyruk yapısı Şekil 3.1'de açıklanmıştır. İzlenecek gün sayısı operatör aracılığı ile belirlenebilmektedir. Kuyruk yapısına örnek olması bakımından 4 gün belirlenerek kuyruk içeriği bunun üzerine çizilmiştir. Her gün, izlenen kaynakların ortalama değerleri alınarak veritabanına kayıt edilmektedir. Bu kayıt edilen değerler ile anlık alınan değerler alarma girdi olmak üzere karşılaştırılmaktadır.



Şekil 3.1: Bilgi nesnesi kuyruk yapısı.

3.1.3 Analiz

Toplanan donanım kaynak kullanım verileri sayısına göre ortalaması alınmaktadır. Bu ortalamalar bilgi nesnesini oluşturduğundan toplanan her değer bilgi nesnesinin ortalamasına etki etmektedir. Sonucu elde edilen referans değerleri ile kayan olay verilerinin karşılaştırılması sonucu ortaya standart kabul ettiğimiz bir değer çıkmaktadır. Bunun için filtreleme ve kıyaslama yapılabilmektedir. [Olay güdümlü süreç soyutlaması yapılacak, tüm bunlar yapıldıktan sonra karar destek sistemi gibi çalışan olay yönetim yazılımı, belirtilen tanımlamalar doğrultusunda karar aldırılması sağlanmaktadır.] Bu tür bir yönetim yazılımı için açık kaynak kodlu Lider-Ahenk Merkezi Yönetim sisteminden faydalanılacak ve bu yönetim yazılımına sistem izleme yaparak karar almasının kolaylaşması için KOİ özelliği kazandırılacaktır.

3.1.4 Aksiyon

Daha önce tanımlanmış alarm kalıpları ile analize konu veriler karşılaştırılmakta, karşılaştırma sonucuna göre de alarm gereksinimleri yerine getirilmektedir.

3.1.5 Bilgi Nesnelerinin İşlenmesi (Profil İşleme)

İzlenen donanım kaynakları, aşağıda belirtilen olaylara göre kıyaslanarak analiz edilmektedir. İşlemci kaynak kullanım verilerinin değerlendirilmesine ilişkin olaylar aşağıdaki belirtilmiştir. İstenirse bellek gibi diğer kaynaklara da bu olaylar uygulanabilmektedir.

1. CPU kullanımı yüzdesi bilgi nesnesinin %Y' miktarından daha yüksek olduğu zaman bilgi maili gönder.
2. CPU kullanım yüzdesi bir gün içinde X defa bilgi nesnesinin %Y miktarını aşarsa uyarı maili gönder (alarm seviyesinin yükseltilmesi).

Bu olay durumları toplayabildiğimiz tüm veriler için geçerli olup, istenilen miktarda bu olay nesneleri artırılabilmektedir.

3.1.6 Denetimli Bilgi Nesnesi

Sistemin mevcut durumunun manuel olarak sistem yöneticisi tarafından belirlendiği durumdur. Bu durumda sistem operatörü dilediği yük durumunu sistemin karşılaştırma yapması için tanımlayabilmektedir. Denetimli sistem ile öğrenme yapılmaksızın doğrudan alarm mekanizmasına değer tanımlanmaktadır. Bu tanımlanan değerler toplanan kaynak kullanımları ile karşılaştırılarak alarm üretilmesi sağlanmaktadır. Bu şekilde sistemin herhangi bir veriyi analiz etmesinden önce beklenen değer ile karşılaştırma yapılabilmektedir. Beklenen değerin manuel belirlenmesi ile otomatik tespit edilmesi arasındaki fark, tartışma bölümünde ayrıntılı olarak ele alınacaktır.

3.1.7 Denetimsiz Bilgi Nesnesi

Denetimli sistemde beklenen değerin tespiti için sistem aktif hale getirildiğinde denetimsiz yöntem otomatik olarak aktif hale gelmektedir. Sistemin kaç gün izlenerek kalıp oluşturulacağı ve beklenen değerin hesaplanacağı kullanıcı tarafından belirleneceği gibi, sistem tarafından otomatik olarak gün tanımı yapılarak hesaplanabilmektedir. Bu şekilde her bir gün sonunda beklenen değer tekrardan hesaplanacak ve günlük olarak beklenen değer değişecektir. Bu yöntem, tez çalışmasının temelini oluşturmakla birlikte ayrıntılar tartışma bölümünde belirtilecektir. Sistem kuralları, donanımlardan elde edilen verilerin değerlendirilmesi ve elde edilen değerlere göre alarm seviyelerinin değiştirilmesinin gerçekleştirilmesine yönelik tanımlanacaktır. İşlemci ve bellek bileşenlerinin izlenmesi sonucu elde edilen değerler daha önce tanımlanan alarm seviyelerine göre sınıflandırılmaktadır.

İki çeşit alarm seviyesi tanımlanmıştır. Birinci seviye alarm, sistemin çalışmasına doğrudan olumsuz etkisi olmayan, ancak bilinmesi sistem hakkında bilinmesi gereken konular hakkında iletim yapması için tanımanmıştır. İkinci seviye alarm sistemin çalışmasına olumsuz etki yapacak anomalilerin bildirildiği durumları kapsamaktadır. Alarm seviyeleri arasındaki geçişler, belirlenen eşiklerin sistem tarafından sınırlarına ulaşıldığında gerçekleşmesi ile mümkün olacaktır. Kalıp oluşturmada (beklenen değer tespiti) algoritması aşağıda belirtilmiştir. Örneklemeler için işlemci ve bellek bilgileri belirtilmiştir.

Birinci Alarm seviyesi, bilgi mail gönderme.

İkinci Alarm seviyesi, kritik düzey mail gönderme şeklindedir.

3.2 Önerilen Sistem Veri Seti

İşlemci ve bellek kullanımı veri seti (Sistem Kullanım Örüntüsü)

1. Kayıt Tarihi
2. Kullanılan bellek
3. Toplam bellek
4. Memory Kullanımı %
5. CPU Kullanımı %

bellek işlemleri ve Alarm

Veri işlemleri:

1. Kontrol aralığı süresi (saniye):
2. Örüntü(pattern) işlem süresi (gün)

Alarm Seviyeleri bilgisi (İşlemci ve bellek kullanımı):

1. *CPU kullanımı yüzdesi bilgi nesnesinin %Y' miktarından daha yüksek olduğu zaman bilgi maili gönder.*
2. *CPU kullanım yüzdesi bir gün içinde X defa bilgi nesnesinin %Y miktarını aşarsa kritik uyarı maili gönder (alarm seviyesinin yükseltilmesi).*

Uyarıların gönderilebileceği iletişim yöntemleri

1. mail address (mail)
2. SMS number (sms)

3.2.1 Önerilen Sistem Algoritması

Her bir işlem thread olarak tanımlanmış olup, toplamda 3 thread yapısı kurulmuştur.

"İş parçacığı 1" temel algoritması:

1: Lider konsol üzerinden Lider sunucu'ya REST talebi gönderilir.

2: Lider sunucu, REST talebini XMPP mesajına çevirir.

3: Lider sunucu XMPP mesaj nesnesi olarak Ahenk'e gönderir.

4: Ahenk XMPP mesaj nesnesini alır.

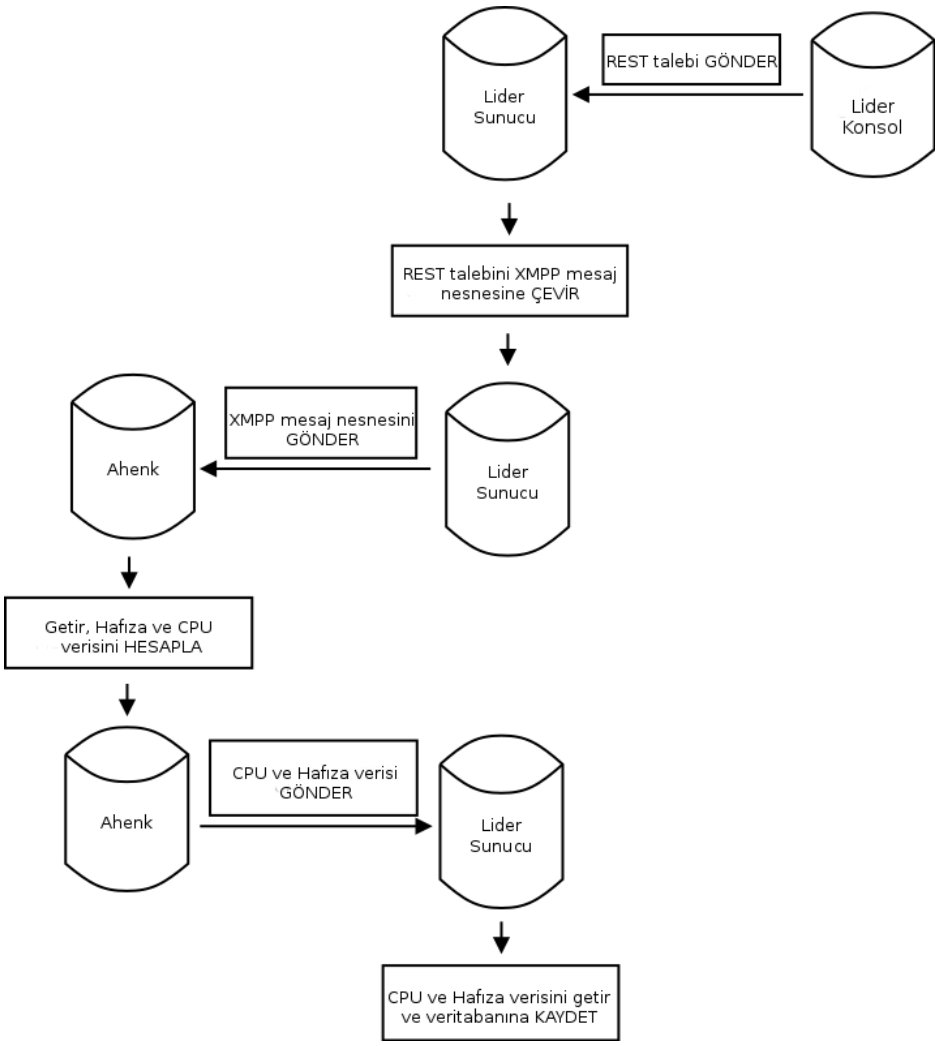
5: Ahenk mesaj nesnesindeki bellek ve CPU bilgilerini alıp, hesaplamalar

yapar.

6: Ahenk XMPP mesajını(bellek ve CPU verilerinin bulunduğu) Lider sunucuya gönderir.

7: Lider sunucu XMPP mesajını alır ve veritabanı üzerine kaydeder.

İş Parçacığı 1 akış diyagramı Şekil 3.2'de açıklanmıştır.



Şekil 3.2: İş parçacığı 1 akış diyagramı.

"İş parçacığı 2" temel algoritması:

1: Lider konsol üzerinden Lider sunucu'ya REST talebi gönderilir..

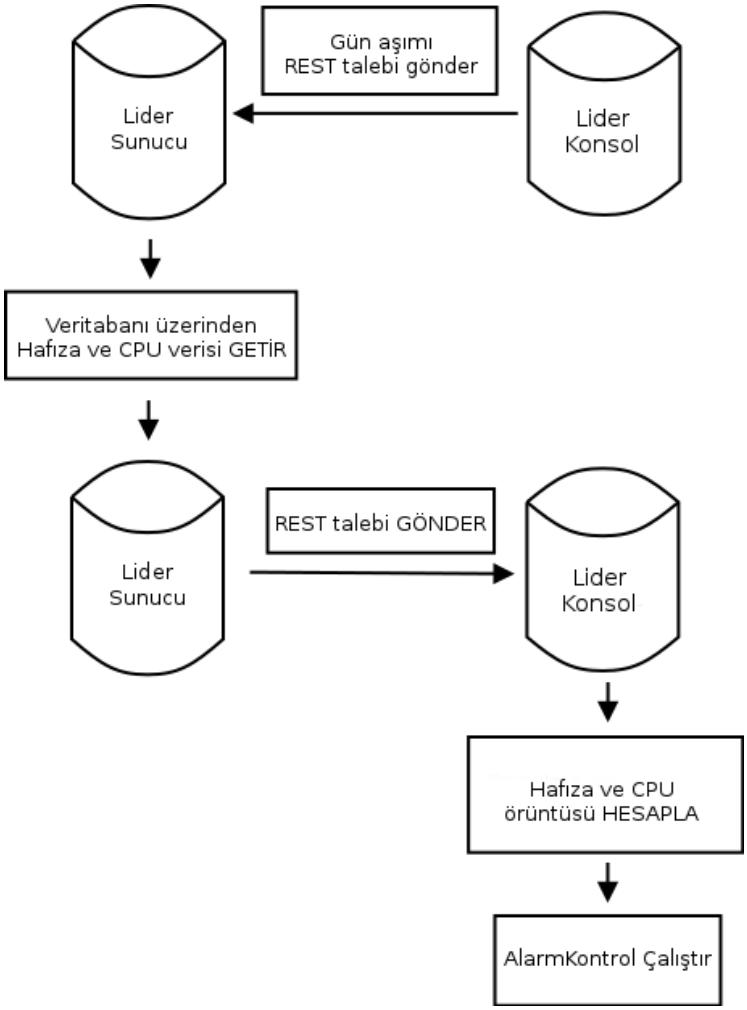
2: Lider sunucu, veritabanı üzerinden CPU ve bellek kullanım verilerine ulaşır.

3: Lider sunucu Lider konsolun talebine(günlük CPU ve bellek kullanımı) cevap verir.

4: Lider konsol cevabı alır, kullanılan CPU ve bellek örüntüsünü hesaplar.

5: Çalıştır "AlarmKontrol"

İş Parçacığı 2 akış diyagramı Şekil 3.3'de gösterilmiştir.



Şekil 3.3: İş parçacığı 2 akış diyagramı.

"AlarmKontrol" temel algoritması:

1: Eğer bellek aşımı veya CPU aşımı varsa:

2: Yap birinci alarm etkinliği (bellek veya CPU): "mail gönder"

3: Eğerson

4: Eğer bellek aşımı tekrarlanıyorsa veya cpu aşımı tekrarlanıyorsa:

5: Yap ikinci alarm etkinliği (bellek veya CPU): "kritik mail gönder"

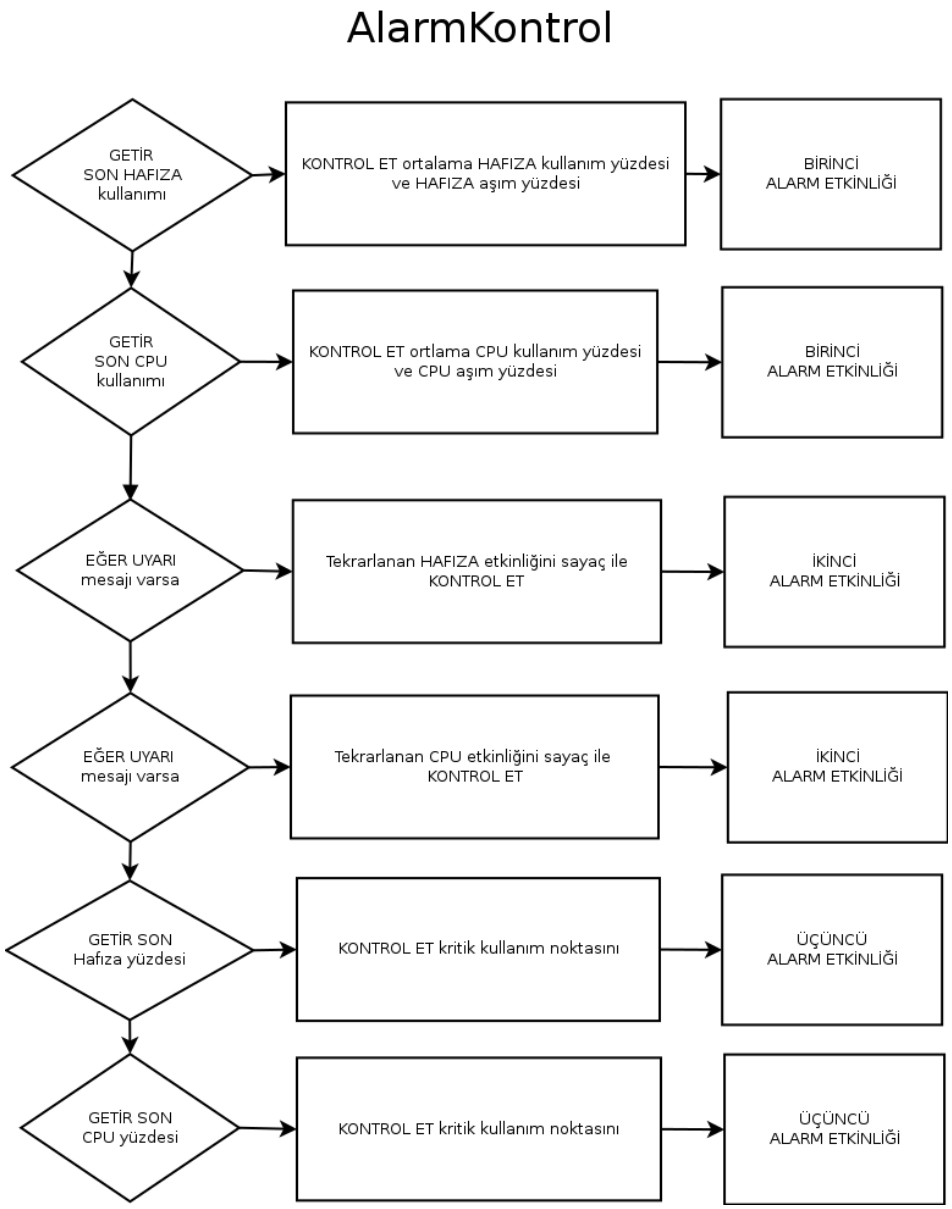
6: Eğerson

7: Eğer (kritik bellek kullanım limitini kontrol et) or (kritik CPU kullanım limitini kontrol et)

8: Yap üçüncü alarm etkinliği (bellek veya CPU): “kapat”

9: Eğerson

Alarm kontrol akış diyagramı Şekil 3.4’de gösterilmiştir.

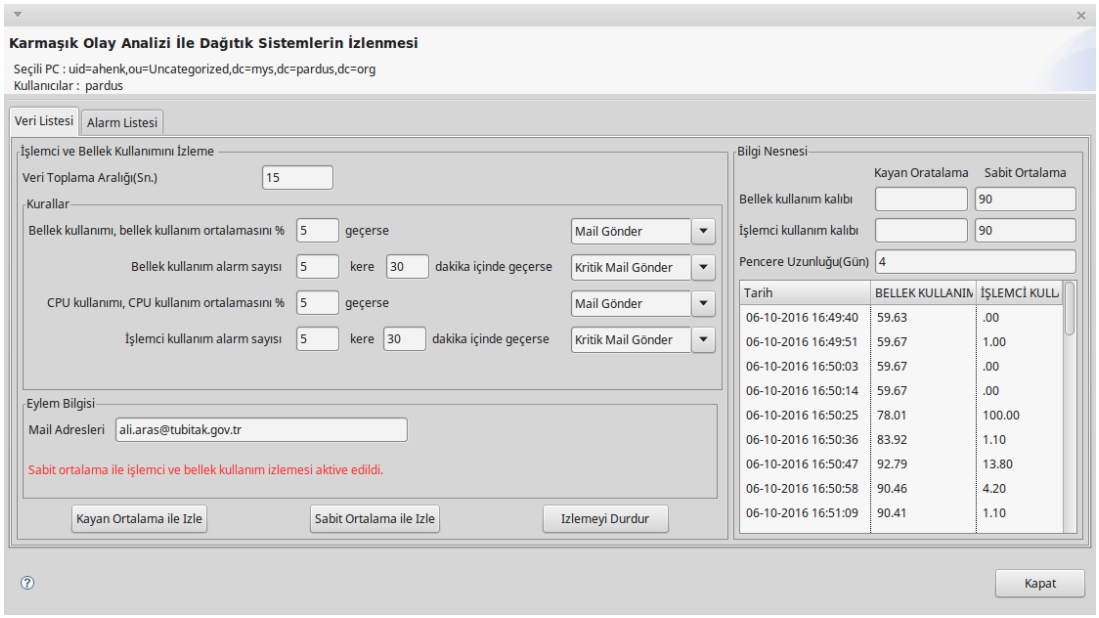


Şekil 3.4: Alarm kontrol algoritması akış diyagramı.

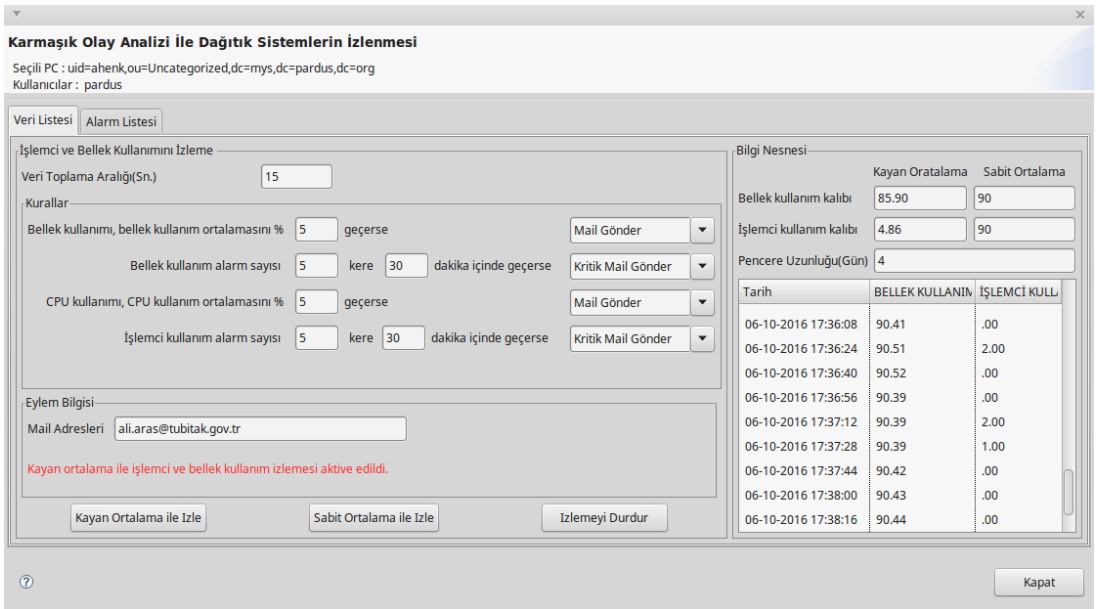
3.3 Arayüz Kullanımı Hakkında

Lider Ahenk Merkezi Yönetim sisteminin kaynak kullanımı eklentisi baz alınarak Python ve Java kullanılarak geliştirilen KOİ motoru, işlemci ve bellek kaynak kullanımlarını toplayabilmektedir. Kullanıcı arayüzü üzerinden bellek ve

işlemci kaynak kullanım izlemesi, kural tanımlamaları, bilgi nesnesi öğrenme ve girişi, hareket planı olmak üzere 4 bölümden oluşmaktadır. Şekil 4.5 ve 4.6'da kullanıcı arayüzü gösterilmektedir.



Şekil 3.5: Lider arayüz kullanım ekranı-1.



Şekil 3.6: Lider arayüz kullanım ekranı-2.

3.3.1 İşlemci ve Bellek Kaynak Kullanım İzlenmesi

Kurallar ve bilgi nesnelerinin oluşturulacağı sürenin tanımlanacağı bu bölüm üzerinden gerçekleştirilebilmektedir. Veri Toplama Aralığı (sn) alanına girilecek süre, hangi aralıklar ile işlemci ve bellek kaynak kullanım durumlarının alınacağı belirlemektedir. Kurallar bölümüne tanımlanan limitler bilgi nesneleri ile kıyaslanarak tanımlanan durumlara göre çıktı üretmektedir.

3.3.2 Kurallar

Alarmların sınırlarının çizildiği ve karmaşık olay işleminin çalıştırıldığı bölümdür. Burada 2 seviye alarm tanımı yapılabilmektedir. Birinci seviye alarm sistemin çalışmasına olumsuz etkisi olmayan, ancak farkındalığı artırmak için iletilen alarmları; ikinci seviye alarm ise sistemin çalışmasına olumsuz etkisi olan bu nedenle bilinmesinin kritik öneme sahip olduğu kritik seviye alarmı kapsamaktadır.

3.3.3 Bilgi Nesnesi

İşlemci ve bellek kaynaklarının o ana kadar izlenerek elde edilmiş değerlerin ortalamalarının tutulduğu alandır. Kayan pencere ile sistemin ne kadar süre izlenerek ortalamalarının tespitini bildirmektedir. Bilgi nesnesi bölümünde iki adet farklı kutucuk bulunmaktadır. Bunlar Kayan Ortalama ve Sabit Ortalama'dır. Kayan Ortalama ile sistemler durmadan incelenmekte ve kuyruk yapısı kullanılmaktadır. Ancak sabit ortalama ile sistem yöneticisi tarafından girilen değer değişmeksizin izleme imkanı yapılmaktadır.

3.3.4 Eylem Bilgisi

Bu bölümde alarm üretildiğinde bilgi verilmesi istenen yöntemlere göre farklı iletişim kanallarına göre bilgi tanımlamaları istenmektedir. Buraya tanımlanan bilgiler ışığında alarm bildirimleri gerçekleştirilecektir.

DÖRDÜNCÜ BÖLÜM

DENEYSEL ÇALIŞMALAR

4.1 Deneysel Çalışmalar

Lider Ahenk merkezi yönetim sistemi ile izlenen bilgisayarların sadece işlemci ve bellek kaynak kullanım değerleri biriktirilmektedir. Elde edilen değerlerin aritmetik ortalaması hesaplanarak izlenen sistemin durumuna özel bilgi nesnesi oluşturulmaktadır. Oluşturulan bilgi nesnesi zamana göre dinamik değişmektedir. Sistemin devamlı izlenmesi ile elde edilen değerler ile daha önceden oluşturulmuş bilgi nesnesi karşılaştırılarak anomali olup olmadığı belirlenmektedir. Yeni elde edilen değerler içinde, kalıp değerlerin sınırlarını aşan bir durum var ise alarm üretilmek üzere alarm kuyruğuna aktarılmaktadır.

Her 10 saniyede elde edilen işlemci ve bellek kaynak kullanım miktarları belirlenen bilgi nesnesi ile karşılaştırılmaktadır. Elde edilen bir değerin anomali olup olmadığı sistem tarafından belirlenen bilgi nesnesi ile karşılaştırılması sonucu anlaşılmaktadır. Karşılaştırma sonucu belirlenen alarm tiplerine göre uyarı gönderilmesi sağlanmaktadır. Bu bölümde, bir üst bölümde açıklamaları bulunan Lider Ahenk Merkezi Yönetim sistemine eklenen KOİ motoru eklentisinin uygulanması ile ilgili çalışmalar bu bölümde sunulacaktır. Çalışmalar sonucu elde edilen değerler, sistem tarafından belirlenen kriterlere göre ortaya çıkan durumun hangi alarm tipine göre sınıflandırıldığı kararlaştırılmıştır. Lider Ahenk Merkezi yöntemi sistemine eklenen KOİ eklentisi iki ayrı sistemi simüle edecek şekilde oluşturulmuştur. Birinci durum geleneksel izleme sistemlerini temsil eden denetimli bilgi nesnesi oluşturma tekniğini, ikinci durum da önerdiğimiz kayan pencere mantığı ile eşik değerlerin dinamik olarak güncellendiği denetimsiz bilgi nesnesi oluşturma durumunu ifade etmektedir.

Deneyler 8 GB belleği bulunan, intel i7 işlemcili 2.50 GHz frekansına sahip 4 bilgisayar üzerinde gerçekleştirilmiştir. Bu bilgisayarlar istemci veya sunucu olarak kabul edilerek tek merkezden yönetim yapılacak şekilde network sistemi oluşturulmuştur. Burada alınan sonuçlar sistemin doğru çalışmasına ilişkin fikir verecektir. Ayrıca bu tür sistemlerin dağıtık mimarideki sistemlere uygulanmasında fikir verecektir.

4.2 Deneysel Sonuçlar

Yapılan deneylerde, karmaşık olay analizinde en önemli faktörün sistemlerden alınan veriler üzerinde bir ortalama değer tespit edilebilmesidir. Beklenen değer tespit edildikten sonra iki farklı şekilde izleme yapılabilmektedir. İki farklı yöntem ile elde edilen değerlerin karşılaştırılması için yapılan deney sonuçları ayrıntılı olarak incelenecektir. Beklenen değer tespit edildikten sonra kuyruğa eklenen her bir kaynak kullanım değerinin karar destek sistemleri ile daha önceden tanımlanmış alarm şablonlarına göre alarm üretmesi ve sistemlerin izlenmesini kolay ve düzenli hale getirmesi sağlanmıştır. Birinci durumda, operatör yardımı ile bir değer girilerek denetimli oluşturulmuş bilgi nesnesi ile işlemci ve bellek kaynak kullanım verileri karşılaştırılmıştır. İkinci durumda ise denetimsiz sistem ile öğrenilen ve kuyruğa her eklenen kaynak kullanım değeri ile güncellenmiş yöntem ile karşılaştırma yapılmıştır.

4.2.1 Denetimli Bilgi Nesnesi ile İzleme

Bilgi nesnesi operatör yardımı ile belirlenmektedir. Belirlenen limitin gerçek bir sistem uzmanın sistem yönetiminde kullandığı ve hayatın olağan akışına uygun bir limit oması bakımından kaynak kullanımının %80'i olarak belirlenmiştir. Bu yöntem ile elde edilen işlemci ve bellek kaynak kullanım değerleri veritabanında saklanarak alarm grafiği ve sayıları çıkarılmıştır. Ayrıca bu yöntemde kuyruğa son eklenen kaynak kullanım değerleri ile ilk belirlenen bilgi nesnesi güncellenmemekte, aynı bilgi nesnesi ile alarm durumları karşılaştırılmaktadır. Tablo 4.1'den anlaşılacağı üzere, kullanıcı tarafından tanımlanan değer doğrultusunda sistem 2. seviye alarm seviyesine erişmiş olup, sistem durumu net belirlenemediği için alarm ve yük maliyeti artırmıştır. Ayrıca eşik değeri kullanıcı tarafından rastgele

belirlendiğinden normal çalışan bir sistemde izleme sisteminin faaliyette tutulduğu dört gün boyunca aralıksız olarak alarm üretildiği anlaşılmıştır. Bu izleme yönteminde sistemin boşta kaldığı durumlar ile gereksiz yere alarm üretildiği durumlar tam çıkartılamamıştır. Ancak dördüncü gün sonunda sistem ortalama bilgi nesnesinin tanımlanan değerinden daha da düşük seviyelerde seyir ettiği anlaşılmış, ancak ikinci seviye alarm üretilemediği için sistem durumu net bilinememiştir.

Günlere göre izleme sonuçları:

Birinci gün sonunda yapılan izleme değerlendirmesinde Tablo 4.1'den anlaşılacağı üzere kaynak kullanım miktarı manuel olarak belirlenen %80 durumunu aştığı için birinci ve ikinci seviye alarm üretilmiştir. Ancak anomaliler hakkında kesin bir bilgiye ulaşılamamıştır.

Tablo 4.1: Denetimli bilgi nesnesi ile izleme alarm durumu (1.Gün).

İZLENEN KAYNAKLAR	BELİRLENEN EŞİK	MAX KULLANIM	1. SEVİYE ALARM	2. SEVİYE ALARM
İşlemci	%80	%100	Evet	Evet
Bellek	%80	%88	Evet	Evet

İkinci gün sonunda yapılan izleme değerlendirme sonucunda Tablo 4.2'den anlaşılacağı üzere kaynak kullanım miktarı ilk belirlenen %80 durumunu aştığı için birinci ve ikinci seviye alarm üretilmiştir. Ancak anomaliler hakkında yine kesin bir bilgiye ulaşılamamıştır.

Tablo 4.2: Denetimli bilgi nesnesi ile izleme alarm durumu (2.Gün).

İZLENEN KAYNAKLAR	BELİRLENEN EŞİK	MAX KULLANIM	1.SEVİYE ALARM	2.SEVİYE ALARM
İşlemci	%80	%100	Evet	Evet
Bellek	%80	%90	Evet	Evet

Üçüncü gün sonunda yapılan izleme değerlendirmesinde Tablo 4.3'den anlaşılacağı üzere işlemci kaynak kullanım miktarında fazla kaynak kullanımın etkisiyle yukarı yönde değişim olması nedeniyle alarm üretilmesi durdurulamamıştır.

Bilgi nesnesi %80 belirlendiği ve belirlenen üst sınırına da durumunu aştığı için birinci ve ikinci seviye alarm üretilmiştir. Bellek kullanımında ise kaynak kullanım miktarı aşağı yönde değişiklik olması nedeniyle birinci ve ikinci seviye alarmlar da oluşturulmamıştır. Ayrıca anomaliler hakkında kesin bir bilgiye ulaşılamamıştır.

Tablo 4.3: Denetimli bilgi nesnesi ile izleme alarm durumu (3. gün).

İZLENEN KAYNAKLAR	BELİRLENEN EŞİK	MAX KULLANIM	1.SEVİYE ALARM	2.SEVİYE ALARM
İşlemci	%80	%90	Evet	Hayır
Bellek	%80	%70	Hayır	Hayır

Dördüncü gün sonunda yapılan izleme değerlendirmesinde Tablo 5.4'ten anlaşılacağı üzere işlemci bellek kaynak kullanım miktarında sistemin boşta kalması nedeniyle aşağı yönde kaynak kullanım miktarı olması nedeniyle alarm üretilmemiştir. Bilgi nesnesi %80 belirlendiği ve belirlenen üst sınırına da durumunu aşan bir anomali tespit edilmemesi nedeniyle alarm sistem durumu izlemeye alınmıştır. Ayrıca sistemin ne kadar boşta kaldığı veya ani yükselme ve düşmenin olup olmadığı hususlarında da ayrıntılı bilgi sahibi olunamamıştır.

Tablo 4.4: Denetimli bilgi nesnesi ile izleme alarm durumu (4. gün).

İZLENEN KAYNAKLAR	BELİRLENEN EŞİK	MAX KULLANIM	1.SEVİYE ALARM	2.SEVİYE ALARM
İşlemci	%80	%77	Hayır	Hayır
Bellek	%80	%75	Hayır	Hayır

4.2.2 Denetimsiz Bilgi Nesnesi ile İzleme

Tablo 5.5'ten anlaşılacağı üzere kayan ortalama ile yapılan izlemede,

1. gün İşlemci için %75,4; Bellek için %65,3
2. gün İşlemci için %82,9; Bellek için %55,8
3. gün İşlemci için %55,2; Bellek için %61,9
4. gün İşlemci için %79,6; Bellek için %72

5. gün İşlemci için %85,7;, Bellek için %48,1 ortalama değer (bilgi nesnesi) tespit edilmiştir. Alarm üretilebilmesi için belirlenen bilgi nesnesinin %110'unu aşması durumu uygulanmıştır. İşlemci ve bellek durumları belirlenen eşik değerleri arasında gidip geldiği için ortaya çıkan anomalilerde alarm üretilmesi sağlanmıştır. Bu şekilde sistemin boşta kalması ve bir anda sistemdeki yükün artması durumun haberdar edilmesini sağlamaktadır.

Tablo 4.5: Günlere göre denetimsiz bilgi nesnesi ile izleme.

GÜN	İŞLEMCİ%	İŞLEMCİ%110	RAM%	RAM%110
1.	75,4	82,94	65,3	71,83
2.	82,9	91,19	55,8	61,38
3.	55,2	60,72	61,9	68,09
4.	79,6	87,56	72	79,2
5.	85,7	94,27	48,1	52,91

Sistem durdurulunca kadar denetimsiz bilgi nesnesi oluşturma işlemi devam etmektedir.

Kurallar bölümüne işlemci ve bellek kaynak kullanım miktarlarının %110'unu geçen durumlar için birinci seviye alarm, bir saat içinde birinci seviye alarmın 4 defa tekrar ettiği durumlarda da 2. seviye alarm üretilmesine yönelik karar verilmiş, günlük maksimum kullanım değerleri incelemeye dahil edilerek Tablo 4.6'da belirtilen alarm durumları gerçekleşmiştir.

Tablo 4.6: Denetimsiz izleme alarm durumu.

GUN	İŞLEMCİ(max)	ALARM	RAM(max)	ALARM
1.	%78	H	%60	H
2.	%85	H	%55	H
3.	%89	E	%59	H
4.	%95	E	%62	H
5.	%60	H	%48	H

5 gün boyunca her 10 saniyede geri bildirim alınması ile izlenen sistemde işlemci ve bellek seviyeleri tespit edilerek alarm üretilmesi sağlanmıştır. 10 saniye boyunca belirlenen şablon değerini geçmeyen durumlarda alarm üretmeyen sistem, anomali olduğunu anlayabilmesi 10 saniyenin üzerinde belirlenen şablon değerin üstünde bir değerde çalışması ile alarm üretilebilecek şekilde ayarlanmıştır. Ciddi bir durum olmadığı varsayılarak alarm üretilmemiş ve boştan yere alarm üretmek için kullanılan sistem kaynakları sistem için ayrılmıştır.

Tablo 4.5'ten de anlaşılacağı üzere standart izlemelerde devamlı olarak alarm üretilirken Denetimsiz Bilgi Nesnesi ile izleme ile karmaşık olay analizi ile sadece 3. ve 4. günler için ilgili günlerdeki işlemcinin anomalilerini bildirmek üzere alarm üretilmesi sağlanmıştır.

BEŞENCİ BÖLÜM

SONUÇLAR VE ÖNERİLER

Lider Ahenk Merkezi Yönetim sistemine entegre edilen karmaşık olay analizi eklentisi ile Debian tabanlı Linux işletim sistemlerinin kaynak kullanım durumlarına ilişkin, denetimli (Rastgele değer atama) ve denetimsiz(kayan ortalama) karmaşık olay işleme sistemleri ile sistem izlemesi yapılmıştır. Denetimli yöntem, geleneksel izleme sistemlerindeki bilgi nesnesi oluşturma yönteminin birebir simülasyonu olarak değerlendirilerek elde edilen izleme sonuçları denetimsiz bilgi nesnesi oluşturma metodu ile karşılaştırılmıştır.

Geleneksel izleme sistemlerini temsil eden yöntem ile yapılan izlemelerde; öğrenilen veya belirlenen eşik değerleri dinamik olarak güncellenmediğinden kaynak kullanım durumunda meydana gelen anomaliler tespit edilememiş ve gereksiz alarmlar üretildiği tespit edilmiştir. Ayrıca eşik değerine yakın seyreden anomaliler de önceden tanımlanmış alarm sınır değerlerini geçmediği için tespit edilemediği görülmüştür.

Denetimsiz izleme yöntemi ile yapılan izlemelerde, elde edilen bilgi nesnesi belirlenen şablon aralıklarında otomatik güncellendiği için, sistemlerde meydana gelen anomaliler tespit edilebilmiştir. Anomalilerin tespit edilmesi sonrası alarm için belirlenen kalıplar doğrultusunda alarm üretilmesi sağlanmıştır. Bu yöntem ile işlemci ve bellek kaynak kullanımının olağan performansına binaen ortaya çıkan kaynak kullanımındaki değer artışı normal olarak kabul edilerek, gereksiz alarm üretilmesinin önüne geçilmiştir. Donanım kaynakları üzerine gereksiz maliyet oluşturan alarm yükü de ortadan kaldırılmıştır.

Sistemin gerçek yükü tespit edilerek, kullanılan uygulamaların standart kullanımda ne kadar kaynak tükettiklerine ilişkin analize esas bilgi olmaması nedeniyle, gelecek çalışmada uygulama bazında analiz yapılarak hangi uygulamanın

ne kadar kaynak tükettiğine ilişkin mikro analiz yapılması sistemin sürdürülebilirliği ve devamlılığı açısından faydalı olacaktır.

Ayrıca olay kalıplarında tanımlanan kurallar ile, sistemlerde oluşabilecek anomaliyi tespit etme kabiliyeti gelebilecek olası kötü durum senaryolarında erken müdahale yapılmasına olanak sağlayacak senaryolarının belirlenmesi sonrasında sistemin kapalı kalma süresine etkisi incelenebilecektir.

Önerilen yöntem ile, sistemlerin kendi kendini ne kadar süre izleyerek bilgi nesnesi oluşturacağı ve hangi olaylar için örüntü oluşturulacağı belirlenebileceği için sistemlerin kendi kendini yönetmesine yardımcı olduğu anlaşılmıştır. İşlemci kaynakları değerlendirilirken süreç yönetimi ve şablon oluşturma için mikro değerlendirme yapılmadığından gelecek çalışmada alt başlıkların da incelenmesine olanak sağlayacak yöntemin geliştirilmesi ile daha iyi tahmin yapar duruma getirilmesi sistem güvenirliğini artıracaktır.

KAYNAKLAR

[1] Pietzuch, P. R., Shand, B., and Bacon, J. 2003. A framework for event composition in distributed systems. In In Proceedings of the 2003 International Middleware Conference. Springer,62–82.

[2] T. Chau, V. Muthusamy, H.-A. Jacobsen, E. Litani, A. Chan, and P. Coulthard. Automating SLA Modeling. In Proc. of the 2008 Conference of the Center for Advanced Studies on Collaborative Research (CASCON'08), 2008

[3] A. Keller and H. Ludwig. The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services. Journal of Network and Systems Management, 11(1):57–81, 2003

[4] Anton Michlmayr, Florian Rosenberg, Philipp Leitner, Schahram Dustdar, Comprehensive QoS monitoring of Web services and event-based SLA violation detection, Proceedings of the 4th International Workshop on Middleware for Service Oriented Computing, p.1-6, November 30-30,2009, Urbana Champaign, Illinois [doi>10.1145/1657755.1657756]

[5] K. Mani Chandy, W. Roy Schulte: Event Processing, Designing IT Systems for Agile Companies. McGraw Hill, 200

[6] M. K. Chandy, O. Etzion, and R. von Ammon, "The Event Processing Manifesto," in Event Processing, ser. Dagstuhl Seminar Proceedings, K. M. Chandy, O. Etzion, and R. von Ammon,Eds., no. 10201. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, Germany, 2011. http://drops.dagstuhl.de/opus/volltexte/2011/2985 Erişim Zamanı 6 Mart 2016

[7] A. Michlmayr, F. Rosenberg, P. Leitner, and S. Dustdar. End-to-End Support for QoS-Aware Service Selection, Invocation and Mediation in VRESCo. Technical report, Vienna University of Technology, 2009. http://www.infosys.tuwien.ac.at/Staff/michlmayr/papers/TUV-1841-2009-03.pdf. Erişim Zamanı 6 Mart 2016

[8] S. Ran. A Model for Web Services Discovery with QoS. SIGecom Exchanges, 4(1):1–10, 2003.

[9] L. Zeng, B. Benatallah, A. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang. QoS-aware Middleware for Web Services Composition. IEEE Transactions on Software Engineering, 30(5):311– 327, May 2004

[10] N. Thio and S. Karunasekera. Automatic measurement of a QoS metric for Web service recommendation. In Proc. of the Australian Software Engineering Conference (ASWEC'05), 2005

[11] P. K. Garg, K. Eshghi, T. Gschwind, B. R. Haverkort, and K. Wolter. Enabling Network Caching of Dynamic Web Objects. In Proc. of the 12th Int. Conference on Computer Performance Evaluation, Modelling Techniques and Tools (TOOLS'02), 2002.

[12] F. Raimondi, J. Skene, and W. Emmerich. Efficient online monitoring of web-service SLAs. In Proc. of the 16th ACM SIGSOFT Int. Symposium on Foundations of Software Engineering (SIGSOFT'08/FSE-16), 2008.

[13] https://www.pardus.org.tr/web/projeler/lider-ahenk Erişim Zamanı 6 Mart 2016

[14] http://www.liderahenk.org/tr Erişim Zamanı 6 Mart 2016

[15] http://nmon.sourceforge.net/pmwiki.php Erişim Zamanı 17.09.2016 saat:12.21

[16] THOMPSON J., SHOLLER D., SCHULTE,W., 2010, Defining 'Events' and Clarifying Implementation Techniques, Gartner Research ID: G00205336.

[17] http://liderahenk.org/ Erişim Zamanı: 1 Mayıs 2016

[18] SCHULTE,W., GASSMAN,B., PEZZINI, M., 2008, Where and Why to Use Event Processing Platforms, Gartner Research ID: G00205939.

[19] Coral8. http://www.aleri.com/WebHelp/coral8documentation.htm. Erişim tarihi 3 Temmuz 2016

[20] https://seminer.linux.org.tr/wp-content/uploads/Nagios-OYLG.pdf, Erişim Zamanı 2 Mayıs 2016

[21] http://www.enderunix.org/docs/nagios.pdf, Erişim Zamanı 2 Mayıs 2016

[22] Gualtieri, M. and Rymer, J. 2009. The Forrester Wave TM: Complex Event Processing (CEP) Platforms, Q3 2009.

[23] SCHULTE W., 2008, Getting Business Value From Event Processing, Gartner ID:G00155981

[24] BRADLEY A., SCHULTE W., 2009, The Gartner Reference Architecture for Complex-Event Processing, Gartner ID: G00169043

[25] ADI A., BOTZER D., NECHUSHTAI G., SHARON G.,P and EDA in Business Intelligence, Gartner ID: G00166324

[26] http://www.complexevents.com/slides/Event_Processing_Symposum-Mark_Palmer_Progress.pdf Erişim Zamanı: 3 Temmuz 2016

[27] http://distrowatch.com/table.php?distribution=pisi, Erişim tarihi:19.03.2016 saat 22:33

[28] EventZero. 2009. http://www.eventzero.com/. Erişim Zamanı: 3 Temmuz 2016

[29] ftp://ftp.software.ibm.com/software/integration/wbe/5565_Empowering-the-Business-US-white-paper.pdf Erişim Zamanı: 3 Temmuz 2016

[30] Eugster, P., Felber, P., Guerraoui, R., and Kermarrec, A.-M. 2003. The many faces of publish/subscribe. ACM Comput. Surveys 2, 35 (June).

[31] Carzaniga, A., Rosenblum, D. S., and Wolf, A. L. 2000. Achieving scalability and expressiveness in an internet-scale event notification service. In Proceedings of the Nineteenth Annual ACM Symposium on Principles of Distributed Computing. ACM, Portland, Oregon, 219–227.

[32] Luckham, D. 1996. Rapide: A language and toolset for simulation of distributed systems by partial orderings of events.

[33] Mansouri-Samani, M. and Sloman, M. 1993. Monitoring distributed systems. Network, IEEE 7, 6, 20–30.

[34] Mansouri-Samani, M. and Sloman, M. 1996. A configurable event service for distributed systems. In ICCDS ’96: Proceedings of the 3rd International Conference on Configurable Distributed Systems. IEEE Computer Society, Washington, DC, USA, 210.

[35] Esper. 2009. http://www.espertech.com/. Erişim Zamanı: 3 Temmuz 2016

[36] http://www.ibm.com/support/knowledgecenter/SSTFXA_6.2.1/com.ibm.itm.doc_6.2.1/itm_highavail08.htm%23wq11 Erişim Zamanı: 17.09.2015 saat:12.06

[37] Magee, J., Dulay, N., and Kramer, J. 1994. Regis: A constructive development environment for distributed programs.

[38] Barga, R. S., Goldstein, J., Ali, M. H., and Hong, M. 2007. Consistent streaming through time: A vision for event stream processing. In In Proceedings of the 3rd Biennial Conference on Innovative Data Systems Research (CIDR 2007). 363–374.

[39] Brenna, L., Demers, A., Gehrke, J., Hong, M., Ossher, J., Panda, B., Riedewald, M.,Thatte, M., and White, W. 2007. Cayuga: a high-performance event processing engine. In SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Manage- ment of data. ACM, New York, NY, USA, 1100–1102.

[40] Demers, A., Gehrke, J., Hong, M., Riedewald, M., and White, W. 2006. Towards expressive publish/subscribe systems. In In Proc. EDBT. 627–644.

[41] Schultz-Moeller, N. P., Migliavacca, M., and Pietzuch, P. 2009. Distributed complex event processing with query optimisation. In International Conference on Distributed Event-Based Systems (DEBS'09). ACM, ACM, Nashville, TN, USA.

[42] Akdere, M., C¸etintemel, U., and Tatbul, N. 2008. Plan-based complex event detection across distributed sources. Proc. VLDB Endow. 1, 1, 66–77.

[43] Cugola, G. and Margara, A. 2009. Raced: an adaptive middleware for complex event detection. In ARM '09: Proceedings of the 8th International Workshop on Adaptive and Reflective MIddleware. ACM, New York, NY, USA, 1–6.

[44] Adi, A. and Etzion, O. 2004. Amit - the situation manager. The VLDB Journal 13, 2, 177–203.

[45] IBM. 2010. http://www-935.ibm.com/services/us/index.wss. Visited Feb. 2010.

[46] Wu, E., Diao, Y., and Rizvi, S. 2006. High-performance complex event processing over streams.In SIGMOD '06: Proceedings of the 2006 ACM SIGMO

[47] Gyllstrom, D., Agrawal, J., Diao, Y., and Immerman, N. 2008. On supporting kleene closure over event streams. In ICDE ’08: Proceedings of the 2008 IEEE 24th International Conference on Data Engineering. IEEE Computer Society, Washington, DC, USA, 1391–1393.

[48] Agrawal, J., Diao, Y., Gyllstrom, D., and Immerman, N. 2008. Efficient pattern matchingover event streams. In SIGMOD ’08: Proceedings of the 2008 ACM SIGMOD internationalconference on Management of data. ACM, New York, NY, USA, 147–160.

[49] Khoussainova, N., Balazinska, M., and Suciu, D. 2008. Probabilistic event extraction from rfid data. In ICDE ’08: Proceedings of the 2008 IEEE 24th International Conference on Data Engineering. IEEE Computer Society, Washington, DC, USA, 1480–1482.

[50] Oracle. 2009. http://www.oracle.com/technologies/soa/complex-event-processing.html. VisitedSep. 2009.

[51] http://www.cozumpark.com/blogs/3party/archive/2011/02/26/Cacti-sistem-ve-ag-cihazlari-izleme-yazilimi.aspx, Erişim Zamanı: 3 Mayıs 2016

[52] The Cacti Manual, by Ian Berry, Tony Roman, Larry Adams, J.P. Pasnak, Jimmy Conner, Reinhard Scheck, and Andreas Braun, 2016, Erişim Zamanı: 3 Mayıs 2016

[53] Streambase. 2016. http://www.streambase.com/. Erişim Zamanı: 3 Mayıs 2016

EKLER

1. Ek-A: Bilgisayar Programı CD’si

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı	:	Ali ARAS
Uyruğu	:	T.C.
Doğum Yeri ve Tarihi	:	Niğde – 05.12.1985
Medeni Hali	:	Evli
Adres	:	TUBİTAK ULAKBİM YÖK Binası B 5 Blok Bilkent/Ankara
E-Posta Adresi	:	ali.aras@tubitak.gov.tr
İletişim (Telefon)	:	0 505 958 16 51

EĞİTİM

Lise	:	Anadolu Dış Ticaret Meslek Lisesi - 2003
Lisans	:	Ahmet Yesevi Üniversitesi/Bilgisayar Mühendisliği (2011)
Yüksek Lisans	:	Türk Hava Kurumu Üniversitesi/Bilişim Teknolojileri (Ankara) – Devam Ediyor

PROJELER

Açık Kaynak ve Açık Standartları Yaygınlaştırma Projesi (TUBİTAK)
Kamu Kurumları Açık Kaynak Destek ve Entegrasyon Projeleri (TUBİTAK)
e-CODEX, e-SENS (Adalet Bakanlığı, Avrupa Komisyonu)

İŞ DENEYİMİ

TUBİTAK ULAKBİM Proje Yürütücüsü
08.08.2016-halen
Açık Kaynak ve Açık Standartları Yaygınlaştırma Projeleri
16.07.2015-halen

Proje Yöneticisi
Adalet Bakanlığı, Bilgi İşlem Dairesi Başkanlığı
14.02.2010-15.07.2015
Sistem Yönetimi, Teknik Destek
Adalet Bakanlığı, Mersin Cumhuriyet Başsavcılığı
10.07.2005 -14.02.2010