
GAZİ ÜNİVERSİTESİ
Gazi
1926

NESNELERİN İNTERNETİ İÇİN SİS HESAPLAMA MİMARİSİ: BİR AKILLI EV UYGULAMASI ÖRNEĞİ

Aykut KANYILMAZ

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

TEMMUZ 2020

Aykut KANYILMAZ tarafından hazırlanan "NESNELERİN İNTERNETİ İÇİN SİS HESAPLAMA MİMARİSİ: BİR AKILLI EV UYGULAMASI ÖRNEĞİ" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ ile Gazi Üniversitesi Bilgisayar Mühendisliği Ana Bilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Danışman: Doç. Dr. Aydın ÇETİN

Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ……..……..

Başkan: Prof. Dr. O. Ayhan ERDEM ...……

Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum.

Üye: Doç. Dr. Osman ÖZKARACA

Bilişim Sistemleri Mühendisliği Ana Bilim Dalı, Muğla Sıtkı Koçman Üniversitesi ...……

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum.

Tez Savunma Tarihi: 16/07/2020

Jüri tarafından kabul edilen bu çalışmanın Yüksek Lisans Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum

……..…….

Prof. Dr. Sena YAŞYERLİ

Fen Bilimleri Enstitüsü Müdürü

ETİK BEYAN

Gazi Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

……..

Aykut KANYILMAZ

16/07/2020

NESNELERİN İNTERNETİ İÇİN SİS HESAPLAMA MİMARİSİ: BİR AKILLI EV UYGULAMASI ÖRNEĞİ

(Yüksek Lisans Tezi)

Aykut KANYILMAZ

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Temmuz 2020

ÖZET

Nesnelerin internetinin yaygınlaşmasıyla birlikte cihazlardan, algılayıcılardan, üretilen veri miktarı hızla artmış bulunmaktadır. Gelecekte üretilen verinin miktarının 3-4 kat daha artacağı öngörülmektedir. Artan veri miktarı bulut bilişim sistemleri ile işlenebilmektedir. Tüm bu verinin bulut bilişim altyapılarına taşınarak işlenmesi gelecekte mümkün olmayacaktır. Akıllı cihazların internet hızı, bant genişliği, internet sürekliliği ve hesaplama gücü kısıtı problemi nedeniyle sis bilişim(edge/fog computing) altyapıları tasarlamayı zorunlu kılmıştır. Sis bilişim sistemleri hesaplanacak verinin kaynağında işlenmesini öngörmektedir. Bu çalışmada sis bilişim mimarisi hazırlanmış ve akıllı ev örneği uygulanmıştır. Akıllı ev sisteminin farklı sis noktalarında dinamik olarak hizmet verebilmesi, yeni sis noktaları tanımlanması sağlanmıştır. Verilerin sis noktalarında işlenmesi sonucu hem veri gizliliği sağlanmış hem de internet bağlantısı kullanımı azaltılmıştır. Uygulamaların sistemden cevap alma hızında artış gözlemlenmiştir.

Bilim Kodu	:	92413
Anahtar Kelimeler	:	Nesnelerin interneti, sis bilişim, bulut bilişim
Sayfa Adedi	:	60
Danışman	:	Doç. Dr. Aydın ÇETİN

FOG INFRASTRUCTURE ARCHITECTURE DESIGN FOR INTERNET OF THINGS: A SMART HOME APPLICATION

(M. Sc. Thesis)

Aykut KANYILMAZ

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

July 2020

ABSTRACT

As the Internet of things became widespread, the amount of data produced by the devices, the sensors and the data has increased rapidly. It is predicted that the amount of data produced in the future will increase by 3-4 times. Increased data can be processed with cloud computing systems. However, due to the internet speed, bandwidth, internet continuity and computational power constraint problem of smart devices, it has made it compulsory to design fog computing infrastructures. The fog systems require processing of the data to be calculated at the source. In this study, a fog calculation architecture was prepared and a smart home application was applied. Smart home system can be served dynamically at different fog nodes and new fog nodes are defined. Processing of data at fog nodes has resulted in data confidentiality and reduced internet connection usage. An increase in the response rate of the applications was observed.

Science Code	:	92413
Key Words	:	Internet of things, fog computing, cloud computing
Page Number	:	60
Supervisor	:	Assoc. Prof. Dr. Aydın ÇETİN

TEŞEKKÜR

Çalışmalarım boyunca yardım ve katkılarıyla beni yönlendiren değerli hocam Doç. Dr. Aydın ÇETİN'e, yine kıymetli tecrübelerinden faydalandığım Gazi Üniversitesi Bilgisayar Mühendisliği Bölümü öğretim üyelerine, tez sürecinde yardımlarını esirgemeyen değerli arkadaşım Mustafa ÖZCAN'a ve manevi destekleriyle beni hiç yalnız bırakmayan eşim Bahar Şebnem KANYILMAZ'a teşekkürü bir borç bilirim.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklamalar**
GB	Gigabayt
ms	Milisaniye
TB	Terabayt
ZB	Zetabayt

Kısaltmalar	**Açıklamalar**
ADN	Uygulama dağıtım ağı (Application Delivery Network)
AWS	Amazon web servisleri
CDN	İçerik dağıtım ağı (Content Delivery Network)
EC2	Esnek bulut hesaplama (Elastic Cloud Computing)
I/O	Giriş/Çıkış (Input/Output)
JDK	Java geliştirme kiti (Java Development Kit)
JSON	Javascript nesne notasyonu
JVM	Java sanal makinası (Java Virtual Machine)
PoP	Erişim noktası (Points of precense)
SOA	Servis yönelimli mimari
XML	Genişletilebilir İşaretleme Dili
XMPP	Genişletilebilir Mesajlaşma ve Varlık Protokolü

1. GİRİŞ

Nesnelerin interneti cihazlarından ve mobil cihazlardan üretilen veri büyüklüğünde ciddi bir artış bulunmaktadır. Mevcut nesnelerin interneti uygulamaları yoğun bir şekilde bulut bilişim altyapılarını kullanmaktadır. En büyük bulut bilişim altyapısı sağlayıcılarının bile sınırlı konumda veri merkezleri bulunmaktadır. Nesnelerin interneti ağına bağlanmak için cihazlar dünyanın belirli noktalarındaki veri merkezleriyle veri iletişimi halindedir. "Nesnelerin internetinde kullanılan cihazlar ise sınırlı enerji ve hesaplama gücüne sahiptirler" [1]. Bu limitler nedeniyle tüm hesaplama, bulut bilişim altyapılarına yüklenmiştir. "Bulut bilişim ölçeklenebilirlik ve erişilebilirlik dikkate alındığında ideal çözüm olsa da cihazların internet hızı, bant genişliği, internet sürekliliği ve hesaplama gücü kısıtları nedeniyle üretilen verinin kaynağında işlenmesi önerilen bir yaklaşımdır" [2-3]. Bu sorunların çözümü için sis bilişim veya uç hesaplama önerilmiştir. Gartner'a göre üretilen verinin kaynağına yakın yerde işlenmesi sis hesaplamadır. "Uç/sis hesaplama hücre ağları, bulut bilişim ve veri merkezlerinin dağıtık olarak genişletilmesidir" [4].

Verinin üretildiği/tüketildiği yere yakınlaştırılması yeni bir yaklaşım değildir. CDN ve APN yaklaşımıyla uygulamalar ve sabit içerikler kullanıcılara yakın yerleştirilir. PoP noktaları aracılığıyla farklı coğrafi noktalardaki kullanıcıların veriye ve uygulamaya erişimi hızlandırılır. Bu yaklaşımları da göz önüne aldığımızda Nesnelerin interneti verileri sis hesaplama altyapılarında saklanabilir/işlenebilirdir.

Sis bilişim kavramı ilk olarak Cisco tarafından önerilmiştir. Veri merkezleri ve uç cihazlar arasında bir katman olarak önerilen sis bilişim mimarisi hesaplama, veri saklama ve ağ servisleri sağlamaktadır. Sis bilişim sayesinde verinin hızlı işlenmesi gereken sağlık, üretim, acil durum gözlemleme (yangın), lojistik gibi birçok sektöre sis bilişim mimarileri uygulanabilir.

Bu çalışmada sis bilişim mimarisine uygun bir platform geliştirilmiştir. Geliştirilen sis bilişim mimarisi içerisinde örnek olarak bir akıllı ev sistemi geliştirilmiştir. Platform ile cihaz erişilebilirliği ve düğüm erişilebilirliği çözümlerinin matematiksel çözümleri paylaşılmıştır. Sis bilişim mimarisinin etkin olabilmesi için düğüm sayısının fazla olması gerekmektedir. Bu çalışma ile 3. kişilerinde platforma sis düğümü eklemesi sağlanmıştır. Sis

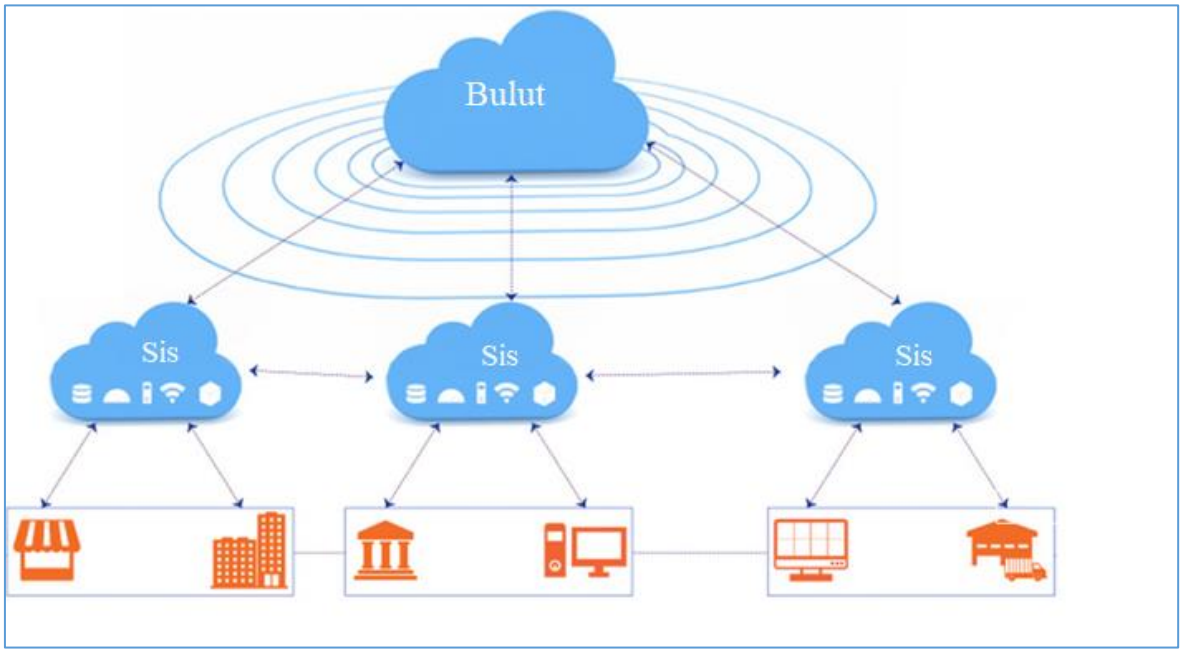
bilişim platformuna kullanıcıların kendi sis düğümlerini sisteme dâhil ederek hem veri gizliliği hem de performans artışı sağlanmıştır. Genel ve özel sis düğümleri ile sis bilişim altyapısının yatay ölçekleme yöntemiyle genişletilmesine imkân sağlanmıştır.

Bölüm 2'de sis bilişim mimarisi, sis bilişimin ana özellikleri, yurtiçinde ve yurtdışında yapılan çalışmalar, akıllı cihazların kısıtları ve mevcut altyapı sorunları ele alınmıştır. Bölüm 3'de Sis bilişim altyapısı oluşturabilmek için kullanılan teknolojiler ve araçlar hakkında bilgiler bulunmaktadır. Bölüm 4'de önerilen sis bilişim altyapısı ve sis bilişim altyapısına ait akışlar, süreçler, iş parçacıkları bulunmaktadır. Bölüm 5'de sis bilişim altyapı mimarisinin etkileri, performans ve erişilebilirlik açısından değerlendirilmesi yapılmıştır. Bölüm 6'da yapılan çalışmanın sonuçları ve değerlendirmeleri yer almaktadır.

2. SİS BİLİŞİM

Bu bölümde sis bilişim hakkında bilgiler paylaşılacaktır. Sis bilişimin uygulama alanları, sis bilişim altyapılarının sahip olabileceği yetenekler, sis bilişim altyapılarına neden ihtiyaç duyulduğu ve sis bilişim hakkında ülkemizde ve dünyada yapılan çalışmalar hakkında bilgi paylaşılacaktır.

Üretilen verilerin bulut sistemlere aktarılması ve işlenmesi performans ve gizlilik problemlerine yol açmaktadır. Gaz, petrol, üretim, taşıma, madencilik, hizmet sektörü gibi endüstriler hızlılık gerektiren alanlardır. Veri işlem hızın artması çıktıları, servis kalitesini ve güvenilirliği arttıracaktır. "Sis hesaplama, üretilen verinin üretildiği noktaya en yakın noktada işlenmesidir. Bu işlem noktalarına sis düğümleri denmektedir" [5]. Şekil 2.1'de sis bilişim mimarisinin bileşenleri sis düğümleri, bulut katmanı ve uç cihazların yerleşimi gözükmektedir.



Şekil 2.1. Sis Bilişim Mimarisi

Nesnelerin interneti uygulamalarında ağ gecikme problemi, kısıtlı kaynak problemi, bağlantı sürekliliği problemi ve coğrafi uzaklık problemi bulunmaktadır. Gaz, petrol, üretim, taşıma, madencilik, hizmet, sağlık, lojistik gibi sektörlerde hızlı veri işleme çok önemlidir. Bir

üretim tesisinde gaz sızıntısının algılanması ve elektriğin kesilmesi arasında geçen süre milisaniye seviyesinde önemlidir.

Research Nester çalışmalarına göre “2015 yılında nesnelerin interneti ağının piyasa büyüklüğü 598.2 Milyar Amerikan Dolarına ulaştı ve 2023 itibariyle market büyüklüğünün 742 milyar Amerikan Doları olması öngörülmektedir” [6, 7]. Aynı zamanda bulut sistemlere aktarılacak verinin büyüklüğü de 2015’de 3.9 Zetabyte’a yükselmişken yılık trafiğin 2020’de 14.1 ZB’a yükselmesi öngörülmektedir.

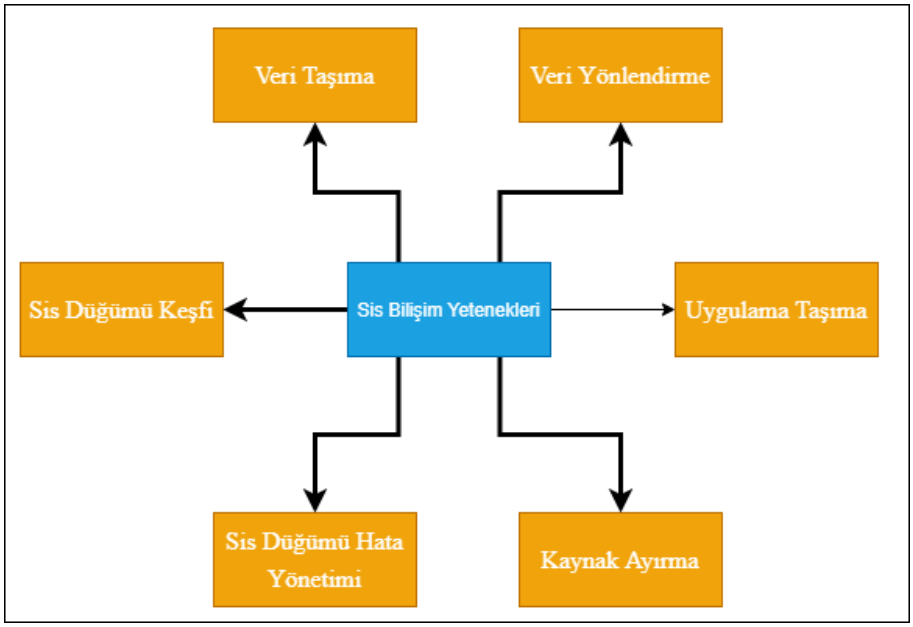
“Bir petrol istasyonu haftalık 500 GB veri üretir. Ticari bir jet uçak uçuş sırasında her 30 dakikada 10 TB veri üretmektedir. Bu kadar büyük verileri binlerce cihazdan, milyonlarca algılayıcıdan bulut sistemlere aktarmak uygun bir yöntem değildir" [5]. Verilerin bulut ortamlara taşınması ve işlenmesi sırasında da güvenlik ihtiyaçları ortaya çıkmaktadır. Yüzbinlerce cihazın, milyonlarca algılayıcının farklı coğrafi konumlardan verileri tek bir merkeze toplaması da uygun bir yöntem değildir.

2.1. Sis Bilişim Uygulama Alanları

Sis bilişim, dağınık bir coğrafi alanda bulunan, internet erişimi ve altyapısıyla ilgili kısıtlar bulunan, uç cihazların sınırlı kaynaklarının bulunduğu ve kısa vadeli veri toplama, hızlı karar alma gibi gereksinimlerin bulunduğu nesnelerin interneti uygulama alanlarının tamamında uygulanabilir. Bu uygulama alanlarını şöyle örneklendirebiliriz: akıllı şehirler, üretim, inşaat, madencilik, tarım, enerji, sağlık, lojistik, taşımacılık ve son kullanıcı çözümlerinde sis bilişim altyapıları kullanılabilir.

2.2. Sis Bilişim Yetenekleri

Sis bilişim çözümleri, veri taşıma, veri yönlendirme, sis düğümü keşfi, uygulama taşıma, hata yönetimi, kaynak ayırma gibi birçok bileşenden oluşmaktadır. Bu bileşenler Şekil 2.2’de gösterilmektedir.



Şekil 2.2. Sis Bilişim Yetenekleri

2.2.1. Veri taşıma

Nesnelerin interneti uygulamalarının ürettiği verileri büyük veri çözümleriyle yönetebilmek mümkündür. Ancak bu üretilen verinin büyüklüğü ve yoğunluğu zaman içerisinde değişkenlik gösterir. Birçok farklı konumdaki veri merkezinde bulunan sunucular üzerinde tutulan verilere erişirken ağ erişimi, kaynak tüketimini de optimize etmek gerekmektedir. Sis bilişim altyapıları farklı coğrafi konumlarda bulunacağından sis bilişimde saklanan verilerin bulut bilişim altyapılarına iletilmesi, analiz edilmesi, anlamlandırılması gerekmektedir. Veri taşıma özelliği sayesinde sis düğümlerinde toplanmış verilerin farklı sis düğümlerine aktarılması gerekebilir. Jiaman, Sichen, Yi ve Lianyin [8] çalışmasında bulut bilişim ortamında veri taşıma yöntemleri incelenmiştir. CloudSim kullanarak Veri dağıtım yöntemi ve dinamik veri taşıma algoritması önermişlerdir.

2.2.2. Uygulama taşıma

Sis bilişim mimarisinde düğümler farklı coğrafi konumlarda bulunduğu için sis bilişim altyapısı düğümler arası geçiş imkânı sunmalıdır. Sis bilişim platformunda çalışan uygulamaların farklı düğümlerde de çalıştırılması için literatürde çözümler üretilmiştir. Bellavista, Corradi, Foschini ve Scotece [9] tarafından yapılan çalışmada üç katmanlı uç hesaplama ortamı için hareketli düğümleri destekleyen bir çatı önerilmiştir. Önerilen model

uygulama duyarlı ve uygulama duyarsız iki farklı yöntemi desteklemektedir ve uç hesaplama ortamını yönetir. Sis bilişim yaklaşımları uç/sis düğümlerine uygulamaların dinamik olarak yüklenmesini, çalıştırılmasını destekler. Aynı zamanda servislerin uç/sis düğümlerinde yönetimi, değerlendirilmesini de sağlar. Ancak sis düğümleri arasında uygulamaların taşınması bir zorluğu da beraberinde getirir. Docker temelli bir platform oluşturulmuş örnek servislerin uç/sis düğümlerinde çalıştırılması hedeflenmiştir.

2.2.3. Sis düğümü keşfi

Sis bilişim ile sis düğümleri verinin üretildiği yere çok yakın noktalara konumlanmıştır. Tasarlanan sis bilişim mimarilerinde birçok sis düğümü birlikte çalışır. Uç cihazın sis düğümünün seçilmesi için bulut katmanında kayıtlı sis düğümleri bilgisini coğrafi olarak elde etmesi gerekmektedir. Bu aşamada sis düğümünün sadece uzaklığa göre seçilmesi düğümler arasındaki yük dağılımının orantısız olmasına neden olur. Akıllı cihazların kendisine en yakın ve en uygun sis düğümünün tercih etmesi sağlanır. Bu çalışma kapsamında düğüm seçimi ile ilgili detaylar 4. Bölümde açıklanmıştır. Sis düğümü seçimi aşamasında işlemci kaynakları, hafıza kaynakları ve ağ kaynakları, düğüm üzerinde işlem yapan cihaz sayısı ve işlenmiş komut sayısı kullanılarak bir düğüm puanı elde edilmiştir. Akıllı cihaz düğüm puanı ve gecikme süresi değerini kullanarak düğüm seçimi yapar.

2.2.4. Sis düğümü hata yönetimi

Sis bilişim coğrafi olarak farklı noktalarda bulunan birçok sis düğümünden oluşmaktadır. Bu düğümlerde oluşan sorunların yönetilmesi gerekmektedir. Düğümlerde oluşan sorunların gözlenmesi, müdahale edilmesi, uyarı sistemlerinin oluşturulması sistemin sağlıklı olarak işletilebilmesi için önemlidir. Sis düğümlerinin erişilebilirliğini kaybetmesi, istikrarlı çalışamama, internet bağlantısının kopması sorunuyla karşılaşıldığında platformun kusursuz bir şekilde bu durumu yönetmesi beklenir. Mobil uygulamalar ya da uç cihazların bağlantı kurduğu sis düğümü erişilemez duruma geldiğinde, uygulama ve uç cihazların kullanıcı davranışını etkilemeden farklı bir sis düğümü ile iletişime geçmesi gerekmektedir. Birden çok sis düğümü bulunan ortamda kendisine en yakın ve en uygun sis düğümünü seçmesi beklenir. Mohamed, Al-Jaroodi ve Jawhar [10] tarafından yapılan çalışmada şu yöntemler önerilmiştir.

1. Watch-Dog: Bir sis düğümün farklı bir sis düğümü tarafından izlenmesi.
2. ServiceReplication: Bir sis düğümünün üzerinde çalışan sis servisinin farklı bir sis düğümüne kopyalanması
3. ReplaceMainBroker: Uç cihazın bağlanacağı nodun değiştirilmesi.
4. FindFogNodes: Bir sis düğümüne komşu düğümlerin listesinin bulunması
5. FindIntermediateNodes: çok katmanlı sis servisleri için iki sis düğümü arasındaki sis düğümlerinin listesi
6. FindCommunicationTime: iki sis düğümü arasındaki erişim zamanının kontrolü
7. FindResourceStatus: bir sis düğümünde kullanılan kaynakların durumunun kontrolü

2.2.5. Veri yönlendirme

Sis düğümlerindeki kaynakların sınırlı olması nedeniyle toplanan verilerin bulut katmanında saklanması ihtiyacı bulunmaktadır. Sis katmanında toplanan verilerinin bulut katmanında arşivlenmesi, aramalara ve raporlara dâhil edilmesi gerekmektedir.

2.2.6. Kaynak ayırma-paylaştırma

Sis bilişim mimarilerinde sis düğümleri üzerindeki kaynağın yönetilmesi ve zamanlı görevlerin çalıştırılması ihtiyacı bulunmaktadır. Naha, Garg, Chan ve Battula [11] tarafından yapılan çalışmada akıllı fabrika senaryosunda gerçek zamanlı bir örnek geliştirilmiştir. Üretim hattında ortaya çıkan sorunları tespit etme hedeflenmiştir. Sanallaştırma yöntemiyle sis düğümlerinde görevlerin çalıştırılması sağlanmıştır. Görev çalışmalarının gecikmesini önlemek için kaynak yönetimi geliştirilmiştir.

2.3. Sis Bilişim Gerekliliği

Bulut bilişim üzerinden hizmet veren servislere erişen mobil cihazların sayısı artmaktadır. Bulut bilişim üzerinden bu uygulamaları kullanmak gecikme, kaynak kısıt problemi yoğunluk ve bant genişliği problemlerine neden olmaktadır. Sis bilişim ile üretilen verinin kaynağında işlenmesi sağlanarak bu sorunların etkilerinin en aza indirilmesi amaçlanmaktadır.

Bulut bilişim sektörde olgunluğa erişmiş ürünlere sahip olduğundan farklı iş modellerine hem kullanıcıları hem de sağlayıcıları için uygun modeller sunmaktadır. Uygun maliyet, kullandığın kadar ödeme seçeneği, ölçekleme imkânları, iş fırsatları anlamında girişim şirketlerine uygun çözümler sunmaktadır. Verilere ve uygulamalara istedikleri zaman istedikleri yerden erişilebilir olması kullanıcılar açısından avantaj sağlamaktadır. Ancak bu avantaj gerçek zamanlı etkileşim gerektiren uygulamalarda dezavantaja dönüşmektedir. Büyük veri merkezlerinde bulunan uygulamalar ve veriler coğrafi olarak sınırlı bölgeden hizmet vermek zorundadır. Amazon dünyadaki en büyük bulut hizmet sağlayıcısı olmasına rağmen Avrupa'da 5 konumda (Paris, Frankfurt, İrlanda, Londra, Stockholm) hizmet vermektedir.

Uygulamaların ve verilerin kullanıcıya daha yakın bir noktaya taşınması gerekmektedir. Veri merkezleri ve kullanıcı arasında bir köprü oluşturmak için sis bilişim altyapıları mimarileri önerilmiştir. Çizelge 2.1'de nesnelerin interneti uygulamaları açısından bulut bilişim ve sis bilişim karşılaştırması yapılmıştır.

Çizelge 2.1. Bulut Bilişim ve Sis Bilişim Gerekliliği

	Bulut Bilişim	Sis Bilişim
Anlık Gecikme(delay)	Yüksek	Düşük
Toplam gecikme(latency)	Yüksek	Düşük
Servis konumu	İnternet Erişimi ile	Uç Cihaza Yakın
Konum Farkındalığı	Yok	Var
Hareketlilik Desteği	Sınırlı	Desteklenir
Gerçek zamanlı etkileşim	Desteklenir	Desteklenir

2.3.1. Ağ gecikme problemi

"Akıllı fabrika, akıllı şebeke, akıllı gaz dağıtım sistemleri, paketleme sistemleri, akıllı ev sistemleri ağları algılayıcılardan kontrol birimlerine kadar çok hızlı iletişim altyapısına ihtiyaç duyar" [12]. Birçok nesnelerin interneti uygulaması akıllı ev, akıllı trafik, akıllı tarım, uçan ulaşım aracı kontrol uygulamaları, oyun uygulamaları, akıllı şebekeler kullanıcı dostu

deneyim için milisaniyeler düzeyinde gereksinim duyarlar. Çizelge 2.2'de Amazon EC2 servislerinin farklı coğrafi noktalardaki sunucularına Yozgat'tan erişim süreleri paylaşılmıştır. Bu süreler hız gereksinimi bulunan bir nesnelerin interneti uygulaması için yeterli değildir.

2.3.2. Kaynak kısıt problemi

Birçok nesnelerin interneti cihazı kısıtlı hesaplama gücüne sahiptir. Örneğin algılayıcılar, kontroller, kameralar, etkinleştiriciler, araçlar, trenler gömülü sistemler ile kontrol edilmektedir. Bu cihazlar üretilen verileri işleyebilecek hesaplama gücüne sahip değildir. Bu nesnelerin doğrudan bulut altyapılarına bağlanması da uygun bir çözüm değildir. Bir sıcaklık algılayıcısının ürettiği veriyi bulut altyapısına göndermesi teknik olarak mümkün değildir. Aynı zamanda bulut altyapısı ile etkinleştiricileri tetiklemek, doğrudan iletişim kurmak mümkün değildir.

Çizelge 2.2. Amazon EC2 Servisine Erişim Süreleri

SAWS Hizmet Bölgesi	HTTP Ping
N. Virginia	812ms
Oregon	935ms
N. California	862ms
Frankfurt	465ms
Ireland	843ms
London	875ms
Singapore	738ms
Tokyo	1433ms
São Paulo	1321ms

2.3.3. Bağlantı sürekliliği problemi

Bulut altyapıları hizmet sürekliliği çözümleri sunmuş olsalar da nesnelerin interneti uygulamaları için bağlantı sürekliliği çok önemlidir. Çizelge 2.3'te Türk Telekom'a ait planlı kesintiler paylaşılmıştır. Örneğin akıllı trafik uygulamaları düşünüldüğünde anlık üretilen verinin hızlıca işlenmesi ve ilgili etkinleştiricilerin tetiklenmesi gerekmektedir. Akıllı ev uygulamalarının da anlık olarak erişilebilir olması kullanıcı dostu deneyim için gereklidir.

2.3.4. Bant genişliği problemi

"Nesnelerin internetine bağlanan cihaz sayısı üstel olarak artmaktadır" [8]. Akıllı bir ev, gaz algılayıcıları, ışık algılayıcıları, sıcaklık algılayıcıları, ısı kameraları, güvenlik kamera görüntüleri, kullanıcı alışkanlıklarının analiz verileri olmak üzere gigabaytlarca veri üretmektedir. Tüm bu verileri bulut altyapılarına göndermek yüksek bant genişliğine ihtiyaç duyar. Bu çoğu zaman gereksizdir veya kullanıcı gizliliği düzenlemeleri nedeniyle yasaklanmıştır. "ABI Araştırma üretilen verinin %90'ının bulut altyapıları yerine uç noktalarda saklanıp işleneceğini öngörmektedir" [13].

Çizelge 2.3. Türk Telekom Planlı Altyapı Kesintileri

Planlanan Başlangıç Tarihi	Planlanan Bitiş Tarihi	Etkilenecek İl	Hizmet Türleri
11.01.2019 02:00:00	11.01.2019 06:00:00	SAMSUN BÖLGE	Ses, İnternet
11.01.2019 02:00:00	11.01.2019 06:00:00	KAYSERİ BÖLGE	Ses, İnternet
11.01.2019 00:00:00	11.01.2019 06:00:00	ANKARA BÖLGE	Ses, İnternet
11.01.2019 00:00:00	11.01.2019 06:00:00	ADANA BÖLGE	Ses, İnternet

2.3.5. Veri gizliliği

Tren yolları, otobanlar, yüksek güvenlik gerektiren binalarda üretilen verilerin gizliliği çok önemlidir. İşletilen verilerin bir kısmı kapalı devre sistemler içerisinde işlenmek zorunda olabilir. Bu sebeple kendi yönetimlerinde bir kenar/sis düğümü ile verileri işlenebilmelidir.

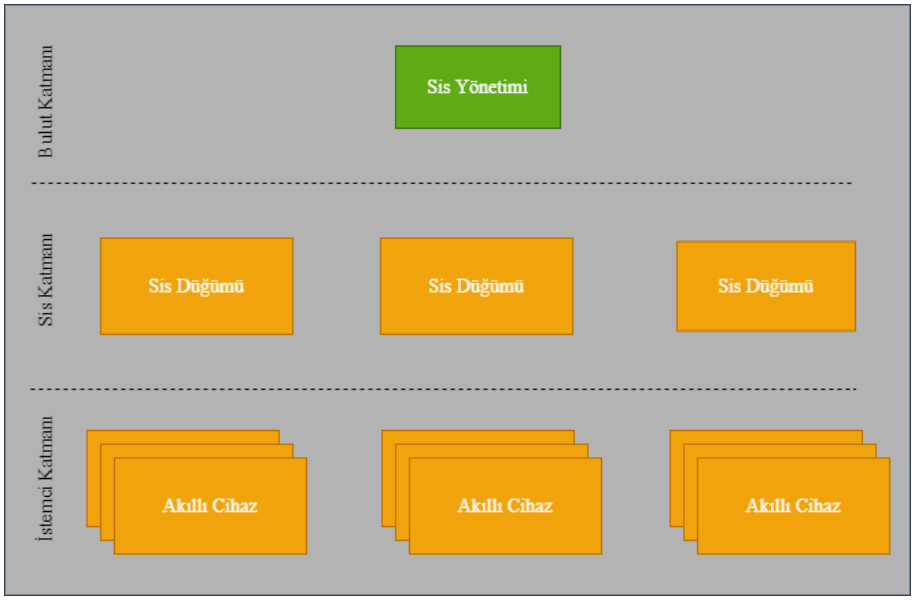
2.4. Sis Bilişim Katmanları

Sis Bilişim altyapılarının mimarisi çoklu katmanlar üzerinde çalışmaktadır. Genel itibari ile bulut katmanı, sis katmanı ve istemci katmanı olarak tanımlanmıştır. Sis katmanı kendi içerisinde birden çok katmana bölünebilir ancak sorumluluk ve kapsamı değişmemektedir. Şekil 2.3'de sis bilişim mimarisinin katmanları bulunmaktadır.

Bulut Katmanı, sis düğümlerinin sistem içerisindeki sağlıklı şekilde çalıştırılmasından ve uygulamanın çalışan düğümlere bağlanmasını sağlamakla yükümlüdür. Yeni düğümler eklenmesi, çıkarılması, düğümlerin sağlık değerlerinin kontrolü bulut katmanının sorumluluğundadır. Aynı zamanda sis düğümlerinin döküm bilgilerini de bulut katmanı yönetir. Envanter bilgisi, sis düğümlerinde hangi akıllı cihazların aktif olarak bağlandığı bilgisini içerir.

Sis katmanı, birden sis düğümünün birlikte çalışmasıyla oluşur. Sis düğümü içerisinde veri tabanı, kuyruk, önbellek, asenkron iletişim servisi, xmpp servisi gibi uygulamanın çalışabilmesi için gerekli servisler bulunur. Sis düğümü 2 türde özel ve genel olarak tanımlanabilir. Sistem tarafından sağlanan genel sis düğümü herkesin kullanabildiği/bağlanabildiği bir düğümdür. Belirli veri merkezlerinde bulunur. Özel sis düğümü ise kullanıcıların kendi tercihlerine bağlı olarak çalıştırdıkları veri merkezlerinde ya da özel sunucularında sadece kendi kullanımlarına özel çalıştırılan düğümlerdir. Akıllı cihazların bağlantı kurma, komut işleme, cevap gönderme gibi işlemleri sis düğümleri aracılığı ile gerçekleştirir.

İstemci Katmanı, gerçek dünya ile algılayıcılar ve etkinleştiriciler aracılığıyla iletişime bu katmanda geçilir. Raspberry Pi, Arduino, LattePanda, ASUS TINKER BOARD gibi bir gömülü sistem üzerinde çalışan istemcilerden oluşur.



Şekil 2.3. Sis Bilişim Mimarisi Katmanları

2.5. Yapılan Çalışmalar

Mevcut çalışmalar uluslararası taranan dergiler ve YÖK sistemi üzerinden taranmıştır.

2.5.1. Dünyada sis bilişim çalışmaları

Dünyada sis bilişim ile ilgili yapılan çalışmaların büyük bir çoğunluğu Çin üniversiteleri bünyesinde yayınlanmıştır. Finlandiya ve Amerika'da da bu konuda çalışmalar yapılmaktadır. Yapılan çalışmaların genel itibariyle yoğunluğu çoklu uygulama destekli sis platformu geliştirmeye yöneliktir.

Premsankar, Francesco ve Taleb [14] tarafından yapılan çalışma ile bulut bilişim ve uç bilişimin birlikte kullanıldığı mimariye uygun bir model geliştirmişler ve uç bilişimin düşük gecikme gerektiren uygulamalarda zorunlu olduğunu ortaya koymuşlardır.

Montero, Huedo ve Llorente [15] tarafından yapılan çalışmada farklı coğrafi konumlarda bulunan veri merkezlerinin arasındaki özel bir iletişim ağı kurulmuştur. Kullanıcıların uzak konumdaki sunuculara erişimlerinin kendilerine en yakın nokta üzerinden sağlanması üzerine çalışılmıştır. Uzak bir noktaya kendilerine en yakın erişim noktasından bağlanmanın doğrudan uzak noktaya bağlanmaya göre daha hızlı olacağı sonucunu elde etmişlerdir.

Zahra ve diğerleri [16] tarafından yapılan çalışma ile sis düğümleri ve akıllı cihazlar arasındaki güvenlik maddeleri üzerine çalışılmış, sis düğümü ve akıllı cihaz arasında güvenli ve iyileştirilmiş bir erişim kontrol protokolü geliştirilmiştir.

Carrega ve diğerleri [17] tarafından yapılan çalışma ile farklı sunucu tiplerinde uygulama çalıştırılmasına olanak sağlayan bir orta katman geliştirilmiştir. Uç bilişimin gerekliliğiyle ilgili bir gözlem yapılmıştır.

Verba ve diğerleri [7] tarafından yapılan çalışma ile java diline özel olarak bir altyapı geliştirilmiş, intranet üzerinde uygulamaların çalıştırılacağı bir altyapı sunulmuştur.

Suganuma ve diğerleri [18] tarafından yapılan çalışma ile esnek bir uç bilişim modeli önerilmiştir. Önerilen model ile katmanlar arası etkileşimin yönetilmesi mümkün hale gelmiştir.

Dang ve Hoang [19] tarafından yapılan çalışma ile sis bilişim için veri güvenliği üzerine çalışılmıştır. 3 yeni model önerisi getirmişlerdir. Bölgesel düğümler arası güvenilirlik modeli, sis temelli gizlilik için geliştirilmiş rol tabanlı erişim kontrol modeli ve değişken konumlu akıllı cihazlar için erişim yönetimi modeli geliştirmişlerdir.

Fan ve diğerleri [20] tarafından yapılan çalışma ile çalışma alanları arasında veri paylaşımına imkân sağlayan bir uç bilişim modeli geliştirilmiştir. Önerilen model ile uç hesaplama düğümleri yönetimi sağlanmıştır. Veri gizliliğinin sağlanması için RSA algoritması ve şifreli metin özellik tabanlı şifreleme yöntemi (CP-ABE) geliştirilmiştir.

Yi ve diğerleri [21] tarafından yapılan çalışma ile mevcut yapılan çalışmalar incelenmiş, sis bilişim modelinin zorlukları ve hedefleri analiz edilmiş örnek bir model geliştirilerek test edilmiştir.

Bonadio ve arkadaşları [22] yaptıkları çalışmada mobil uç hesaplama sistemi(MEC) geliştirmişlerdir. MEC ile kullanıcılara yakın noktada hesaplama ve saklama alanı oluşturmayı hedeflemişlerdir. Mobil cihazlar ile MEC arasındaki erişim kablosuz iletişim üzerinden sağlandığından gecikme zamanını düşürmüşlerdir. Çoklu sunucu sistemi modeli

ile performans artışı dışında potansiyel uç düğümlerinin kayıplarını yönetebilecek yapıyı elde etmişlerdir.

Hussain ve arkadaşları [23] yaptıkları çalışma ile sis platformunda bulunan zararlı düğümlerin tespitini amaçlamışlardır. Güven hesaplaması yapılan düğümlerin durumlarını dikkate alan bir model önermişlerdir. Kurulan geribildirim sistemi ile zararlı olabilecek düğümlerin tespiti hedeflenmiştir.

Rocha Filho ve arkadaşları [24] yaşam alanlarının yönetimi için ImPrRIum çalışmasını önermişlerdir. ImPrRIum çalışması, yaşam alanlarını gözlemleyerek, akıllı nesnelerin kendi aralarında iletişim kurmasıyla karar alabilen bir model önermişlerdir. Hem gerçek hem de similasyonlar üzerinde çalışarak düşük gecikme zamanı, enerji tasarrufu sağlamayı hedeflemişlerdir.

Chekired ve arkadaşları [25] elektrikli araçlar şarj istasyonlarının yönetimi sağlamak için merkezi olmayan bir sis bilişim mimarisi önermişlerdir. Önerilen mimari ile şarj istasyonlarının elektrik talebini planlamayı amaçlamışlardır. Markov zinciri ile öncelikli kuyrukların matematiksel modellerini geliştirmişler ve elektrikli araç bağlantısı süresini azalttıklarını belirtmektedirler. Öncelik belirlemek için farklı üç algoritma öne sürmüşlerdir. Sis bilişim temelinde geliştirdikleri merkeiz olmayan dağıtık mimari ile elektrik ağının yönetimi üzerine çalışmışlar, planlama ve öngörü yetenekleri ile elektrik şebekesinin kararlığını sağlamaya yönelik önermişlerdir.

Baucas ve arkadaşları [26] şehirlerdeki ses seviyesini ölçmek için bulut ve sis bilişim mimarisine uygun iki uygulama geliştirerek sonuçları karşılaştırmışlardır. Şehir ses sınıflandırmasını yaparken mimarinin daha etkin olduğunu raporlamışlardır.

Liu ve arkadaşları [27] nesnelerin interneti için bulut-sis bilişim mimarilerine karma bir çözüm önermişler ve yüksek gecikme sorununu çözmeyi hedeflemişlerdir. Bulut katmanıyla uç katman arasında sis katmanı ekleyerek bir altyapı geliştirmişlerdir. 2 GB büyüklüğünde bir verinin iletimi için %14 ile %34 arasında gecikme sürelerinde iyileşme raporlamışlardır. Gecikmeyi azaltmak için yeni sis düğümlerinin eklenmesini önermişlerdir.

Naha ve arkadaşları [11] zaman içerisinde değişen kaynak ihtiyaçlarının çözümüne yönelik önerilerde bulunmuşlardır. Kaynak ihtiyacı değiştiğinde sis düğümlerinde dinamik ve zamanlı kaynak tahsisi önermişlerdir. Gerçek bir sis bilişim ortamını simülasyon araçları ile modelleyerek önerilen algoritmaları değerlendirmişlerdir. Önerilen algoritmanın mevcut algoritmalara göre çok daha etkin olduğunu raporlamışlardır. İşlem zamanı ve maliyetlerde mevcut çözümlere göre %12 ve %15 oranında iyileştirme gözlemlemişlerdir.

2.5.2. Ülkemizde sis bilişim çalışmaları

Ülkemizde sis bilişim konusuyla ilgili yapılan çok sınırlı sayıda çalışma bulunmaktadır. Kavramsal olarak bu konunun gelecekte karşımıza daha çok çıkacağı düşünüldüğünde yapılan çalışmaların sayısı ve olgunluğu çok yetersizdir.

Erdal [28] tarafından yapılan çalışma ile sis bilişim altyapısına uygun imza doğrulama yöntemi geliştirilmiştir. Bankalar için geliştirilen örnek senaryo ile güvenlik değerlendirilmesi yapılmıştır. Geliştirilen yöntemin bulut bilişime göre daha güvenli bir yöntem olduğu doğrulanmıştır.

Güven [29] tarafından yapılan çalışmada kenar bilişim üzerinde çalışan ve Nesnelerin İnterneti cihazlarının güvenliğini sağlayan Kılıç Kenar Bilişim Güvenlik Uygulaması önerilmektedir. Tehdit seviyelerini algılayarak farklı seviyelerde güvenlik yaklaşımını değiştiren Kılıç ve parçaları detaylı olarak açıklanmıştır.

Hasan [30] tarafından yapılan çalışmada, artırılmış gerçeklik, sanal gerçeklik, çevrimiçi oyun veya video görüşme gibi bir multimedya uygulamasının video kodlama isleri ürettiği varsayılmıştır. Tek kenar sunuculu bir sistemde işlerin öncelikle işlenmesi sağlanmıştır.

Eşrefoğlu [31] tarafından yapılan çalışma ile hareketli kullanıcıların bu ağlardaki servis kalitesi üzerine çalışılmıştır. Mininet ve OpenFlow tabanlı Yazılım Tanımlı Ağlar kurulup testleri yapılmıştır.

Karataş [32] doktora tezinde bulut ve sis bilişim temelli bir nesnelerin interneti mimarisi önermiştir. Bu mimari ile veri saklama ve işleme altyapısı önermiştir. Önerilen mimari ve

tekniklerin gecikmeyi bant genişliği tüketimini arttırmadan gecikmeyi düşürdüğü raporlanmıştır.

Okay [33] doktora çalışmasında sis hesaplama ile birlikte çalışabilen Yazılım Tanımlı Ağlar (Software Defined Networking / SDN) teknolojisinden yararlanarak yönlendirme kararlarının daha etkin ve optimize bir şekilde alınması amaçlamıştır. İlk çalışmada, SDN kontrolcü merkezi olarak konumlandırılıp kaynak ve hedef düğüm arasında en kısa yol algoritması çalıştırılarak örnek bir senaryo üzerinde performans analizi yapmıştır. Bir diğer çalışmada ise, SDN kontrolcüler dağıtık olarak yerleştirilip sis kontrolcüleri ve bulut kontrolcüsü olarak hiyerarşik hale getirmiştir. Önerilen hiyerarşik SDN modelinin performansı gecikme, iş çıktısı ve iletim ek yükü bakımından benzetim çalışmalarıyla analiz etmiştir.

3. ARAÇLAR

Sis bilişim platformlarının oluşturulabilmesi için kullanılan teknolojiler bu bölümde tanımlanmıştır. Sis bilişim altyapıları bulut, sis ve uç katmanı olmak üzere üç ana katmandan oluşmaktadır. Bu katmanlar arasındaki iletişim çözümleri, sis düğümlerinin uygulamaları çalıştırabileceği bir sanallaştırma yöntemi, sis katmanında saklanacak verilerin saklanması, test edilmesi, ölçeklenebilmesi için kullanılan teknolojiler belirtilmiştir.

3.1. Sanallaştırma

Sis bilişim altyapıları birden çok nesnelerin interneti uygulamasını çalıştırabilmelidir. Sis düğümlerinin uygulamaları ön hazırlık gerektirmeden çalıştırabilmesi için biz sanallaştırma yöntemi kullanmak zorunludur. Nesnelerin interneti uygulamaları ölçeklenebilir, sürekli hizmet verebilir olma zorunluluğu bulunmaktadır. Hızlı ayağa kaldırma, ölçeklenebilir olma, otomasyon gibi özelliklerin sağlanması için sanallaştırma teknolojileri kullanılmaktadır. Sanallaştırma teknolojisi ölçeklenebilirlik, sürekli açıklık gibi özelliklerin sağlaması için gereklidir.

Konteynır temelli sanallaştırma işletim sisteminin çekirdeği üzerine kurulu birbirinden izole edilmiş kaynak kullanımına imkân sağlamaktadır. Docker en yaygın olarak kullanılan sanallaştırma aracıdır.

3.1.1. Linux cgroups

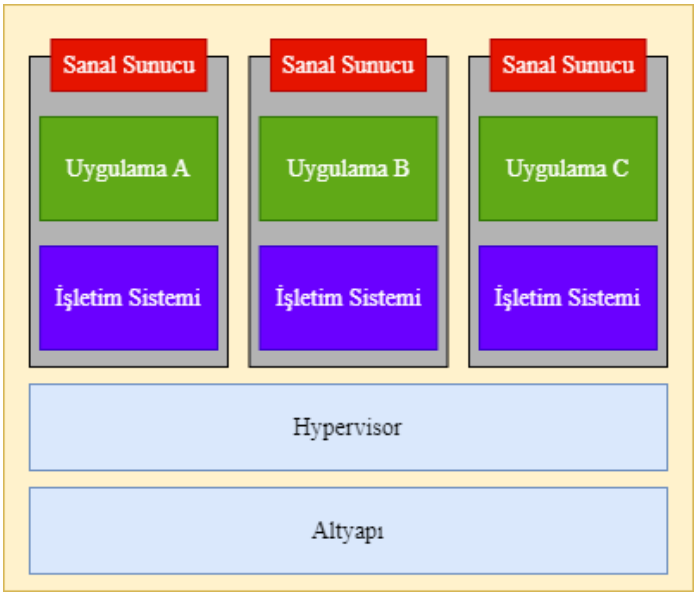
Cgroups, iş parçacıklarını gruplama, izole etme ve sınırlandırma gibi yetenekleri olan bir linux çekirdeği özelliğidir. 2006 yılında Google mühendisleri tarafından geliştirilmeye başlamış, 2017 yılında da ilk yayını yapılmıştır.

Bir kontrol grubu bir küme iş parçacığının aynı ölçütlerle sınırlandırılmasıdır. Bu gruplar hiyerarşik olarak düzenlenebilir, üst gruplardan özellikleri katılım yoluyla alabilirler. İşletim sistemi çekirdeği, işlemci, hafıza, disk kontrolü gibi birden çok kontrol aracı sağlar. Böylece aynı işletim sisteminde çalışan işler (process) birbirinden habersiz kendi kaynaklarıyla izole bir şekilde çalışabilir.

Linux çekirdeğindeki cgorups özelliğini kullanmak ilk dönemler karmaşık olduğundan yaygınlaşması da zaman almıştır. Docker ve diğer konteyner teknolojileri cgroups üzerine geliştirilmiş teknolojilerdir. Cgroups ile yapılan karmaşık işlemleri basitleştirmiş, kontaynır oluşturmayı kolaylaştırmışlardır.

3.1.2. Docker

Docker, konteyner yapısıyla uygulamaları çalıştırmak, ölçeklemek için tasarlanmış bir araçtır. Konteynerler, yazılım geliştiriciler gerekli kütüphane ve diğer bağımlılıklarıyla birlikte uygulamalarını tek paket haline getirebilirler. Bu yöntem ile yazılım geliştiriciler işletim sistemi ayarlarından bağımsız olarak uygulamaların farklı sunuculardaçalışmalarını kolaylaştırır.

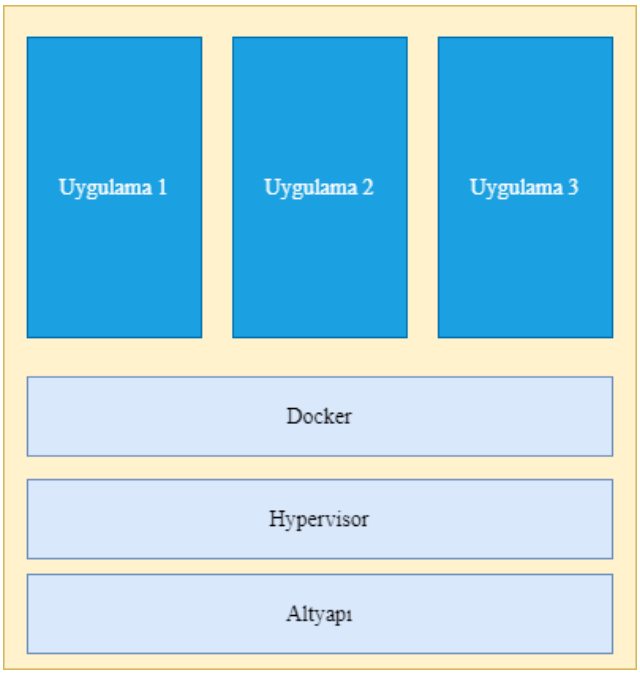


Şekil 3.1. Hypervisor ile Sanallaştırılmış Uygulamalar

Her uygulamanın kullandığı kütüphane, çalıştırılabilir dosyalar konteynerlerde birbirinden izole edilmiştir. Aynı işletim sisteminde bulunmalarına rağmen her bir konteyner izole edilmiş işlemci, hafıza, ağ ve dosya kaynaklarına sahiptir. Bu sayede birçok konteyner aynı işletim sistemi üzerinde 8birbirinden bağımsız olarak çalıştırılabilir. [26]'da yapılan çalışmada docker ile uç hesaplama altyapısı testleri yapılmıştır. Docker ile konteynerlar VM teknolojilerine göre 26 kat daha hızlıdırlar [26]. Docker sanal işletim sistemine benzer ama işletim sisteminden farklı olarak tüm işletim sistemini ayağa kaldırmak yerine, aynı linux

çekirdeğini kullanır. Bu yöntem ile önemli bir performans artışı sağlanır ve kaynak tüketimi azaltır. Şekil 3.1'de hypervisor ile sanallaştırma yöntemi görünmektedir. Şekil 3.2'de uygulamaların docker ve sanallaştırılması görülmektedir. Bu

Sanal Sunucular, fiziksel donanım katmanı üzerine kurulmaktadır. Hypervisor tek sunucuda çeşitli sanal sunucular çalıştırılmasına imkân sağlamaktadır. Her bir sanal sunucu işletim sisteminin bir kopyası ve gerekli çalıştırılabilir dosyalar ve kütüphaneleri içermektedir. Bu yüzden sanal sunucu çalıştırılırken ayağa kalkması zaman almaktadır.



Şekil 3.2. Docker ile Sanallaştırılmış Uygulamalar

3.1.3. Docker imaj ve katmanları

Bir docker imajı bir katmanlar serisinden oluşur. Her katman bir içerisindeki Dockerfile ile temsil edilir.

Dockerfile dosyası katmanı oluşturan 4 komuttan oluşur. "FROM" komutu ubuntu:15.04 imajının temel alınacağını belirler. "COPY" komutu imaj içerisine istenilen dosyaları eklemek için kullanılır. "RUN" komutu imaj oluşturulurken çalıştırılacak komutları çalıştırır. Örnekte uygulamanın derlenmesi için make komutu çalıştırılmıştır. Son olarak "CMD" komutu docker imajının hangi komut ile çalıştırılacağını belirler.

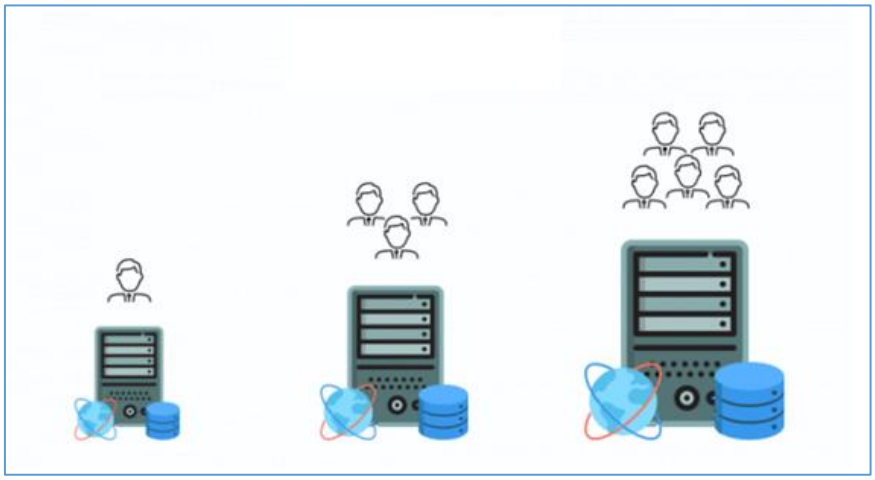
Docker imajının oluşturulabilmesi için Dockerfile dosyasının bulunduğu klasör içerisinde şu komut çalıştırılmalıdır. “-t” parametresi imajın etiketlenmesi için kullanılmaktadır.

Her katman sadece temel alınan katmandan farklılıkları içerir. Yeni bir konteyner tanımlandığında var olan katmanlara yeni bir katman eklenmiş olur. Bu katman genellikle konteyner katmanı olur. Çalışan konteynerdeki tüm değişiklikler, yeni dosya oluşturulması, var olan dosyaların değiştirilmesi, silinmesi bu ince katmana yazılır.

3.1.4. Docker ambarları

Uygulamaların docker üzerinde çalıştırılabilmesi için oluşturulan docker imajlarının bir ambarda/sunucuda saklanması gerekmektedir. Herkese açık ambarlar kullanılabilir veya sadece belirli kullanıcıların erişebileceği özel docker ambarları tanımlanabilir. Docker Hub kullanıcılara, müşterilere veya docker topluluğuna docker imajlarını paylaşabilecekleri ücretsiz bir ambar sunar.

3.2. Ölçekleme



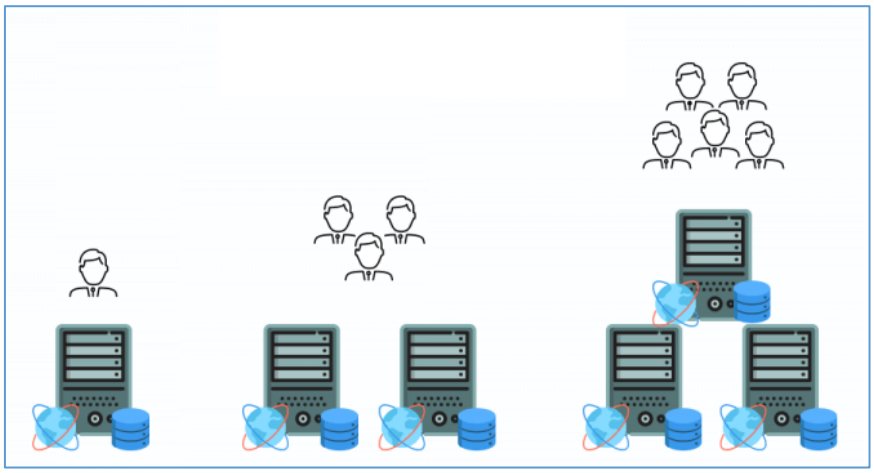
Şekil 3.3. Dikey Ölçekleme

Uygulama yazılımları üzerinde barındıkları sunucuların kaynaklarını kullanarak hizmet verirler. Kullanım artması ile birlikte uygulama yazılımlarının hizmet verebilirliğinin devamı için farklı modeller geliştirilmiştir. Tek bir bilgisayar üzerinde çalışmak için geliştirilmiş yazılımlar ihtiyaçları karşılamamış, farklı sunucular üzerinde çalıştırılacak yeni

mimariler ortaya çıkmıştır. Ölçekleme kavramı bu noktada ortaya çıkmıştır. Ölçekleme uygulama yazılımının yük altında kaldığında yükü nasıl kaldıracağı ile ilgili yöntemin kendisidir. Dikey ve yatay olmak üzere 2 tür ölçekleme yöntemi bulunmaktadır.

3.2.1. Dikey ölçekleme

Dikey ölçekleme, ihtiyaç halinde sunuculara CPU, RAM, DISK gibi kaynakların eklenmesini ifade eder. Dikey ölçekleme genellikle orta küçük ölçekli şirketlerin ürün ve uygulamalarında kullanılır. Şekil 3.3, dikey ölçekleme yöntemi ile kaynak artışı göstermektedir. En yaygın kullanım örneği pahalı bir donanım satın alıp üzerine ESX veya VMWare gibi sanallaştırma yazılımları kurmaktır. Dikey ölçekleme, sunucu donanımını yükseltmek anlamına gelir. IOPS artışı, CPU/RAM kapasite artırımı, disk kapasitesi artırımı gibi nedenlerle bu yönteme başvurulur. Sanallaştırılma kullanılıyor olsa bile, istenilen performans artışı sağlansa dahi, arıza süresi riski yatay ölçeklemeye göre dikey ölçeklemede daha yüksektir.



Şekil 3.4. Yatay Ölçekleme

3.2.2. Yatay ölçekleme

Yatay ölçekleme, fiziksel makinelerin sunucu altyapısı ve veri tabanına eklenmesi ile sağlanır. Şekil 3.4, dikey ölçekleme yöntemi ile kaynak artışını göstermektedir. Sunucu kümesine yeni düğümler eklenir. Her bir düğümün sunucu kümesindeki sorumluluğu azalmış olur. Birçok internet sitesi ve web servisler yatay ölçekleme ile hızlandırılmıştır.

Gmail, Youtube, Yahoo, Facebook, EBay, Amazon gibi servisler yatay ölçekleme yöntemini kullanırlar.

3.3. Test Yöntem ve Araçları

Performans testleri uygulamanın yük altındaki davranışlarını, performans sınırlarını, darboğazları tespit etmek için farklı sunucular üzerinde çeşitli araçlar kullanılarak yapılmaktadır. Yapılan testler ile darboğazlar tespit edilerek performans artışı sağlanır. İzleme Uygulama içerisinde hafıza kullanımı gözlemlenerek hafıza kaçakları tespit edilir, çöp toplayıcısı yapılandırma ayarları en uygun haline getirilir.

Performans testi sistemdeki yavaşlıkları tespit etmek için yapılır. Yapılan testler sonucunda uygulama seviyesinde iyileştirmeler yapılır ya da veri tabanı üzerinde düzenlemeler yapılarak performans artışı sağlanır. Performans testi yapılabilmesi için uygulamanın tutarlı bir halde çalışması sağlanır. Uygulamanın normal koşullar altında cevap sürelerini belirlemeyi amaçlar. İhtiyacı karşılayıp karşılamadığının tespiti bu test ile öğrenilir.

Yük Testi, uygulamanın normal koşulların üzerinde bir yoğunluk oluştuğundaki cevap sürelerinin hesaplandığı testtir. Uygulama yük altındayken yavaşlayacaktır ancak bu testin amacı artan yük ile birlikte uygulamanın sağlıklı kalacağı eşik değerleri tespit etmektir. Yük testi sırasında gerçek işlemleri simülasyonu yapılır. Yük testi düşük bir yük ile başlatılır ve istenilen yüke kadar artırılarak test edilir.

Stres testi, yük testine yöntem olarak benzese de asıl amacı uygulamanın aşırı yük altında davranışlarını gözlemlemektir. Yük artışı çok fazla olduğunda sistem hataları, kaynak problemleri, beklenmedik sorunları tespit etmeyi amaçlamaktadır.

3.3.1. Test ve izleme araçları

Apache JMeter uygulaması, yük testi, fonksiyon testi, performans ölçümü için tasarlanmış açık kaynaklı, java uygulamasıdır. Web servisleri test etmek için geliştirilmiş olsa da diğer fonksiyonel testlerin yapılması için de kullanılmaktadır.

VisualVM uygulaması, jre üzerinde çalışan uygulamaların hafıza tüketimini, nesne havuzlarını işlemci ve thread kullanımını gözlemlemek için kullanılır. Performans ve hafıza kullanımını hem canlı ortamda hem de test ortamlarında izlemek için geliştirilmiştir.

Sorunların analiz edilebilmesi için detaylı çalışma zamanı bilgisi sağlayan ve alt seviye incelemelere imkân tanıyan Java Mission Control uygulaması ve Java Flight Recorder uygulaması birlikte kullanılır. Java Flight Recorder, Oracle JDK içerisinde bulunur ve izleme, olay toplama altyapısı sunar. JFR'ın sunduğu bilgiler ile geliştiriciler JVM içerisinde neler olduğunu, nasıl davranıldığını detaylı alt seviye bilgileri edinmiş olurlar. JMC ise JFR tarafından sağlanan verilerin detaylı şekilde incelenmesi, analiz edilmesi amacıyla kullanılır.

3.4. Sunucu ve İstemci Arasındaki İletişim - Server-Sent Events

Akıllı cihazlar ağ mimarisi gereği bir network cihazının arkasında dış dünyadan doğrudan erişilemeyecek noktada çalışırlar. Akıllı cihazların da kendilerine gönderilen komutları çok hızlı bir şekilde ele alıp işlemesi beklenmektedir. Sis düğümleri ve akıllı cihazlar arasında hızlı bir haberleşme yapısı oluşturulması gerekir. Nesnelerin interneti uygulamalarında xmpp, mqtt, server-sent events gibi protokol ve yöntemler kullanılır. Bu tez kapsamında geliştirilen uygulamada server-sent events kullanılmıştır.

Server-sent events tek yönlü iletişim imkânı sağlayan bir teknolojidir. Html 5 ile standart haline gelmiştir. Server-sent events sunuculardan istemcilere doğru veri iletişimi sağlamak için kullanılır. Birçok yazılım dili tarafından desteklenir.

3.5. Büyük Veri

Nesnelerin interneti uygulamaları çok büyük miktarda veri üretmektedir. Üretilen büyük miktardaki verinin işlenmesi, özetlenmesi, anlamlı sonuçlar çıkarılması ancak büyük veri işleme yöntemleri ile mümkündür [34]. Sis noktalarında dalgalanan veri akışını yönetebilecek kapasitededir. Mevcut nesnelerin interneti mimarileri düşük ölçeklenebilir yapıdadırlar [35]. Sis noktalarında üretilen dalgalı veri akışı ölçeklenebilir bir altyapı ile işlenebilir.

3.5.1. MongoDB

MongoDB basit, açık kaynak kodlu, doküman tabanlı, ölçeklenebilir bir nosql veri tabanı uygulamasıdır. Verileri JSON ve BSON formatında saklamaktadır. Yüksek hacimli ve yapısal olmayan verileri saklamak için geliştirilmiştir. Geleneksel kolon ve satır yapısının yerine veriler koleksiyonlar içerisinde ayrı dokümanlar olarak saklanır. Geleneksel Veri tabanı ile NoSQL veri tabanı kavram eşleştirilmesi aşağıdaki gibidir. Doküman temelli, yüksek performanslı, ölçeklenebilir, dinamik, sabit bir şeması yok, esnek, farklı tip verileri saklayabilir, birleştirme yok, dağıtık mimariye uygun olması gibi avantajları bulunmaktadır. ACID ve işlem (transaction) özellikleri yoktur. Bu yüzden yoğun işlem gerektiren uygulamalarda kullanılması uygun değildir. Gerçek zamanlı veri işleme, günlük, içerik yönetim sistemleri, yapılandırma yönetimi, önbellekleme, sosyal medya platformlarında sıklıkça kullanılabilir. Çizelge 3.1'de İlişkisel veritabanındaki varlıkların ve mongodb veritabanındaki karşılıkları gösterilmiştir.

Büyük hacimli verileri veya yüksek çıktı bekleyen uygulamalarının ihtiyaçları, tek bir sunucuda çalışan veritabanı sistemleri ile karşılanamaz. Yüksek sorgu oranı olan bir uygulama CPU üzerinde yük oluşturacaktır. Büyük hacimli veriler ile çalışmak sistemin I/O performansını etkileyecektir. Büyüyen sistemlerdeki bu sorunları adreslemek için 2 yöntem vardır.

Çizelge 3.1. SQL ve MongoDB Kavram Karşılıkları

SQL	MongoDB
Database	Database
Table	Collection
Index	Index
Row	Document
Column	Field
Joining	Linking & Embedding
Partition	Sharding (Range Partition)
Replication	ReplSet

Dikey ölçekleme tek bir sunucunun işlemci, hafıza, disk kapasitesini arttırmaktır. Mevcut teknolojilerin limitleri tek bir sunucu üzerinde yeterli donanım artışlarını kısıtlamaktadır. Bu yüzden dikey ölçeklemenin bir limiti vardır.

Yatay ölçekleme sistemdeki yükü ve veriyi sunuculara parçalamak üzerine kuruludur. Yeni kaynakları mevcut sistemin kapasitesin arttırarak değil yeni sunucular ekleyerek arttırmaktadır. Her bir sunucu iş yükünün bir kısmını ele almaktadır. Bu yaklaşımda ise yazılım mimarisi ve yayılma (deployment) karmaşıklığı artmaktadır.

Sharding, verileri sunucular üzerinde dağıtmakta kullanılan yöntemdir. Mongodb, büyük hacimli verileri ve yüksek çıktı beklentisi olan uygulamaları desteklemek için sharding yapısını kullanmaktadır.

MongoDB Sharding router (mongos), config servers (mongod) ve shards (mongod) olmak üzere 3 ana parçadan oluşmaktadır. Shard, Her bir shard, parçalanmış verinin bir kısmını saklamaktadır. Her bir shard bir kopya kümesi olarak yüklenebilir. Mongos, sorgu yönlendiricisi gibi davranır. İstemci uygulama ve kopya kümesi arasında bir arayüz sunmaktadır. Config servers, dağıtık yapıya ait ek bilgileri ve konfigürasyonları saklamaktadır.

Çizelge 3.2. Raspberry PI 3 B+ Özellikleri

Özellik	Değer
İşlemci	Cortex-A53 64-bit 1.4GHz işlemci
Hafıza	1 GB LPDDR2 SDRAM
Wireless LAN/Bluetooth	2.4GHz ve 5GHz IEEE 802.11.b/g/ n/ac, Bluetooth 4.2, BLE
Ethernet	USB 2.0 üzerinden Gigabit Ethernet (maksimum çıkış 300 Mbps)
GPIO	Genişletilmiş 40 pinli GPIO başlığı
USB	4 USB 2.0 bağlantı noktası
Görüntü/Ses	Video ve ses çıkış portu (4'lü jack çıkışı)
Hafiza	Micro SD port

3.6. Nesnelerin İnterneti Akıllı Cihazı

Raspberry Pi, dünyada çok yaygın olarak kullanılan bir gömülü sistemdir. Bu tez çalışması kapsamında Raspberry Pi 3 B+ kullanılmıştır. Teknik özellikleri şu şekildedir. Debian üzerine geliştirilmiş Raspbian Linux dağımı ile kullanılabilmektedir. Üzerinde bulunan grafik, usb, ses, hdmi, ethernet, wifi, hafıza kartı ile avantaj sağlamaktadır. Bu tez kapsamında yapılan çalışmada geçit (gateway) olarak Raspberry Pi kullanılmıştır. Çizelge 3.2'de Raspberry PI 3 B+'a ait donanım özellikleri gösterilmiştir.

3.7. Rest Servis Teknolojileri

REST (Representational State Transfer) 2000 yılında Roy Fielding tarafından doktora tezinde tanımlanmış ve tanıtılmıştır. REST dağıtık sistem tasarımı için tanımlanmış mimari yaklaşımdır. Kısıtlar, istemci-sunucu ilişkisi, durumsuzluk gibi özellikleri tanımlar. REST http ile doğrudan ilişkili değildir fakat genellikle http ile ilişkilendirilir[19]. Rest altyapısı genel olarak http protokolü ile kullanılır. Http durum kodları ve anlamları Çizelge 3.3'de bulunmaktadır.

Idempotent Http metodu, aynı parametrelerle birden çok kez çağrıldığında cevapta farklılık oluşturmayan web servislerdir. 1 kez ya da 10 kez çağrılması sonucu değiştirmez.

Http Post metodu genellikle sunucuda yeni dosya/varlık oluşturmak için kullanılır. Aynı post servisi n kez çağrılırsa n tane dosya/varlık oluşacaktır. Post servisler idempotent bir servis değildir.

Http Get, Head, Options ve Trace metotları sunucuda hiç bir değişiklik yapmazlar. Sadece sunucudan dosya/varlık bilgilerini çekmek için kullanılırlar. Bu metotlara ait servisleri defalarca çağırmak sunucuda bir değişiklik yaratmayacağı için bu tür servisler idempotent servislerdir.

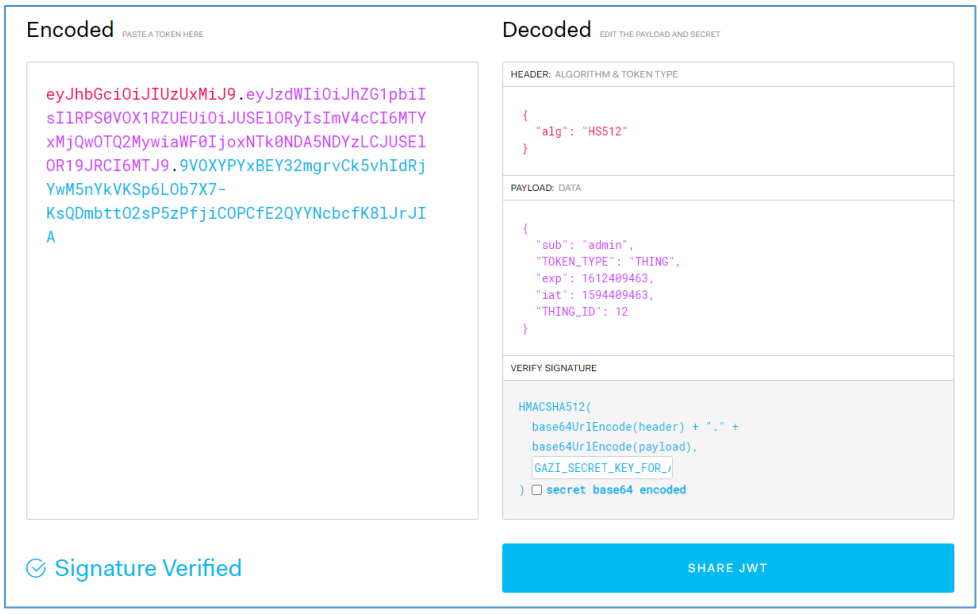
Http Put Metotlu servisler genellikle sunucudaki dosyayı/varlığı güncellemek için kullanılır. N kez bu servisin çağrılması aynı dosya/varlık değişikliğini defalarca kez yapacağı için sonuç değişmez. PUT metotlu servisler idempotent servislerdir.

Http Delete metotlu servisler dosya/varlık silmek için kullanılır. N defa çağrılması sonucu, ilk çağrıda dosyayı/varlığı siler. Diğer çağrılarda 204 (No Content) ya da 404 (Not Found) kodu döner. Delete metodu servisler idempotent değildir.

Çizelge 3.3. HTTP Durum Kodları ve Anlamları

HTTP Durum Kodu	Açıklaması
1XX	bilgi kodları
2XX	başarılı
3XX	yönlendirme
4XX	istemci hataları
5XX	sunucu hataları

Rest servisler geliştirilirken veri taşınmasında JSON veya XML yaygın olarak kullanılır. Mesajlar Http metotları (Get, Post, Put, ve delete) ile iletilir. Durumsuzluğu (Stateless) sağlamak için sunucu tarafında istemciye özel bir bağlam oluşturulmaz. Sunucuda herhangi bir durum saklanmaz.



Şekil 3.5. JWT Örneği

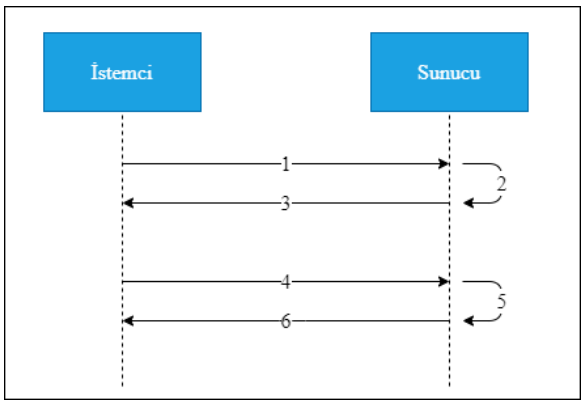
3.8. JSON Web Anahtarı

JSON Web anahtarları, iki parça arasında güvenli veri paylaşımına izin veren açık bir standarttır. JWT 3 parçadan oluştmaktadır ve nokta ile bu parçalar birbirinden ayrılır. Başlık

(Header), Gövde (Payload) ve imza (Signature). xxxxx.yyyyy.zzzzz şeklinde bir görüntüsü bulunmaktadır. Şekil 3.5, geliştirilen uygulamada akıllı cihazlar için üretimiş bir jwt anahtarnı göstermektedir.

1. Başlık: Başlık temel olarak kullanılan algoritma ve tercih edilen anahtar tipinden oluşmaktadır. JWT'nin ilk parçası olan bölüm bu json'ın base64 formatına dönüştürülmüş halidir. Örneği şu şekildedir.
2. Gövde: JWT anahtarın ikinci bölümü olan gövde içerisinde veriler taşınabilmektedir. JWT'nin gövde parçası olan bölüm bu json'ın base64 formatına dönüştürülmüş halidir.
3. İmza**:** JWT anahtarının bütünlüğünü doğrulamak için kullanılan bölümdür. Örneğin HMAC SHA256 algoritması tercih edilmişse, imza alanı şu fonksiyon ile oluşturulur. İmza ile mesajın ve anahtarın iletim sırasında değişmediğini kontrol edilmektedir. Ayrıca mesajın kaynağının da doğrulanmasını sağlar.

JWT Kullanımı, web uygulamalarında ve mobil uygulamalarda geleneksel yaklaşım bir oturum başlatılması ve bu oturumun kullanıcı işlem yaptığı sürece açık bırakılmasıyla sunucularda korunması sağlanıyordu. Sunucuların her kullanıcı için veri tutmasının önüne geçilmek için kullanıcı kimliklendirme işlemi sonrasında bir JWT anahtara sahip olur ve bu anahtar içerisinde oturuma ait bilgiler saklanmaktadır. Kullanıcı gizlenmiş bir kaynağa her erişmek istediğinde bu anahtarı istek başlığında iletmesi beklenmektedir. Bu sayede sunucu üzerinde kullanıcının kimliğiyle ilgili bir veri saklanmamaktadır. İletilen JWT anahtarı sunucu tarafında açılarak içerisindeki kullanıcıya ait bilgiler kullanılmaktadır. Geçerliliği devam eden bir anahtar ise ve kullanıcının bu kaynaklara erişimi varsa kullanıcı erişimine izin verilmektedir. Şekil 3.6, istemci ve sunucu arasındaki JWT akışını göstermektedir.



Şekil 3.6. JWT Kullanım akışı

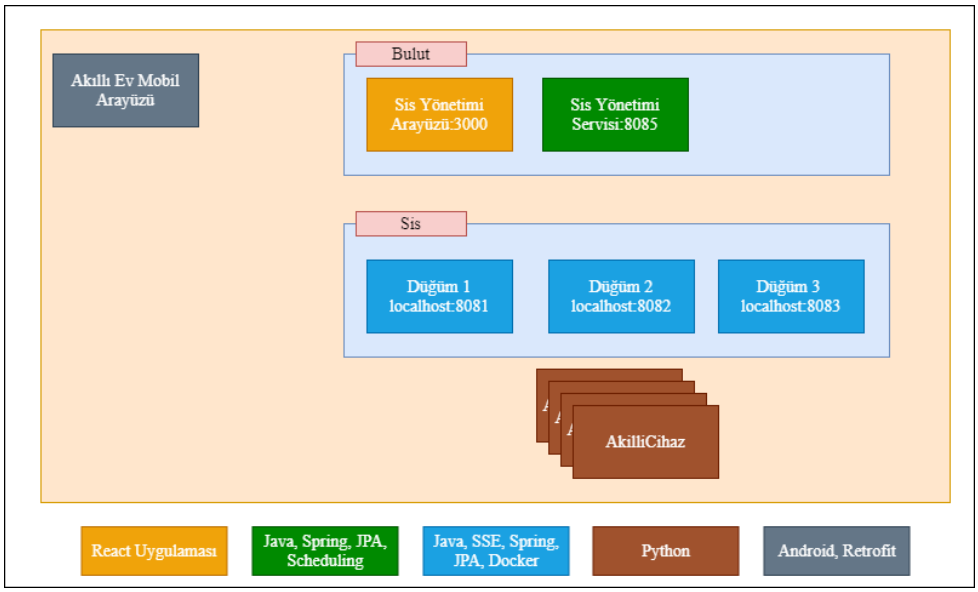
1. Kullanıcı adı ve şifre ile birlikte giriş yapma isteği iletilir.
2. Anahtarın içerisine saklanmak istenen veriler gizli bir kelime ile şifrelenir.
3. Oluşturulan JWT anahtarı istemciye iletilir.
4. Kullanıcı gizli bir kaynağa erişmek istediğinde isteğin başlık kısmında Authorization: Baerer {JWT_ANAHTAR} şeklinde sunucuya iletir.
5. Sunucu JWT anahtarın doğruluğu kontrol eder ve kullanıcı bilgilerini JWT anahtarından alır.
6. Erişim izni varsa istemciye talep edilen kaynak iletilir.

4. SİS BİLİŞİM ALTYAPISI İLE AKILLI EV UYGULAMA ÖRNEĞİ

Bulut bilişim ile sis bilişim birbirlerine alternatif değil birbirlerini tamamlayan mimari çözümlerdir. Bir nesnelerin interneti uygulamasının sis noktalarına dağıtılması ve aygıtların bu sis noktalarından birine bağlanarak tüm işlemlerini yapabilmesi gerekmektedir. Bir akıllı ev uygulaması bir coğrafi bölgede hizmet verebilir. Yeni bölgelerde de hizmet vermeye başladığında yeni bölgelerde sis noktaları tanımlanarak uygulamalar kendilerini bu sis noktalarına dağıtabilirler.

Bu çalışma kapsamında bir sis bilişim altyapısı geliştirilmiştir. Geliştirilen sis bilişim altyapısı üzerinde çalışabilmesi için örnek bir akıllı ev uygulaması geliştirilerek geliştirilen sis bilişim altyapısının testi yapılmıştır.

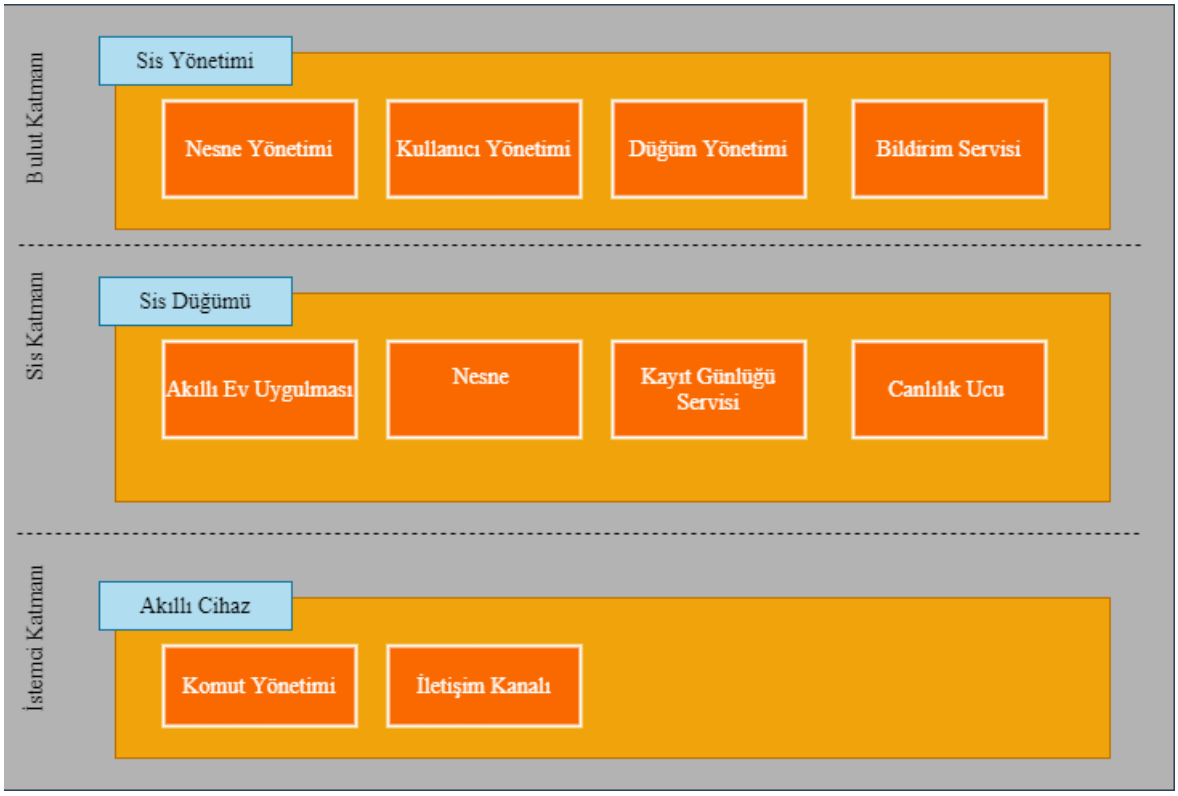
Akıllı sistemlerde veri gizliliği ön plana çıkmaktadır. Uygulamaların veri gizliliği ön plana çıktığı durumlarda çalıştırılan uygulamanın kullanıcıların kendi sunucularında çalışması istenebilir. Mevcut çalışmalar incelendiğinde kullanıcıların kendi düğümlerini ekleyebilecek bir altyapı görülmemiştir. Bizim çalışmamızda kullanıcıların genel sis düğümlerini kullanabildikleri gibi kendi sunucularına ait sis düğümlerini ekleme imkânına sahiptir. Mevcut bulut bilişim altyapılarından kaynaklı kısıtlardan kurtulmak için Sis Bilişim Altyapısı kullanarak akıllı ev örnek uygulamasını geliştirilmiştir.



Şekil 4.1. Sis Bilişim Altyapısı için Geliştirilen Uygulamalar

4.1. Sistem Bileşenleri

Bu tez kapsamında geliştirilen uygulamalar ve kullanılan teknolojiler Şekil 4.1'de gösterilmiştir. Bu kapsamda sistem yönetimi arayüzü, sistem yönetimi servisi, düğüm yönetimi servisi, akıllı ev örneği için mobil uygulama ve akıllı cihaz yönetimi için Raspberry pi kodlaması yapılmıştır. Web servisler ve sistem yönetimi için java, spring, jpa, spring security, sis düğümü yönetimi uygulamasının sanallaştırması için docker kullanılmıştır. Raspberry pi üzerinde python ile akıllı ev uygulaması istemcisi geliştirilmiştir. Android üzerinde java ile bir mobil uygulama geliştirilmiştir. Şekil 4.2'de üst düzey sistem mimarisi gösterilmiştir.



Şekil 4.2. Sis Bilişim Altyapısı Sistem Parçacıkları

Bulut Katmanı, düğümler ve akıllı cihazları kullanıcı yönetimi, bildirim servisi, düğüm yöneticisi ve nesne yöneticisi bileşenleri aracılığıyla yönetmektedir. Sis yönetim arayüzü aracılığıyla sisteme yeni kullanıcıların dâhil olabilmesi, yeni cihazların eklenebilmesi ve yeni düğüm kayıtlarının tanımlanabilmesi için geliştirilmiş bir web arayüzüdür. React, saga, redux, axios kütüphaneleri kullanılarak geliştirilmiştir.

Sis Katmanı, sistemin yatay olarak ölçeklendiği katmandır. Özel ve genel sis düğümlerinden oluşmaktadır. Sis düğümleri kayıt günlüğü servisi, nesne erişilebilirliği ve nesnelerin interneti uygulamasından oluşmaktadır. Docker ile sanallaştırılmış akıllı ev uygulaması bu katmanda yeni düğümlere dağıtılarak sis ortamına dâhil edilir. Bu çalışma kapsamında Java, Spring, JPA kullanılarak akıllı ev uygulaması geliştirilmiştir.

Uç Katmanı, Bulut katmanı ve sis katmanı ile iletişime geçerek uygun sunucuyla bağlantı kurar. Akıllı cihazın tüm yönetimi bu katmanın sorumluluğundadır. Bu tez kapsamında Raspberry Pi kullanılarak python ile bir akıllı ev istemcisi geliştirilmiştir. Anlık bildirim (SSE) yöntemiyle sunuculardan gelen komutların akılı ev istemcisi üzerinde işlenmesi sağlanmıştır.

4.1.1. Kullanıcı yönetimi

Sis bilişim platformuna dâhil olan uygulamaların kullanıcı yönetimi bu bileşen ile yapılır. Kullanıcı kayıt, güncelleme, silme ve düğüm tercihlerini yöneten bileşendir. Bu bileşen akıllı cihazların kullanacağı gizli anahtarları oluşturur ve düğümlerden gelen doğrulama isteklerine cevap verir.

4.1.2. Bildirim servisi

Sis bilişim platformunda farklı coğrafi konumlarda sis düğümleri bulunur. Sis düğümlerinin erişilebilir olduğu düzenli olarak kontrol edilir. Düğümlerde oluşabilecek sorunların kontrol edilmesi, izlenmesi, gerekli durumlarda düğüm sahibine bildirimlerde bulunulması gerekmektedir. Bildirimler düğüm sahiplerinin tercihlerine bağlı olarak yapılmaktadır.

4.1.3. Düğüm yönetimi

Sis bilişim platformu içerisinde bulunan düğümlerin yönetimi bu bileşen ile sağlanır. Düğüm yönetimi bileşeni ile platforma yeni düğümler eklenmesi, çıkarılması ve düğümlerin sağlık kontrolleri yapılmaktadır. Düğüm yönetimi aynı zamanda düğümlerin üzerindeki yükleri, kaynak tüketimlerini takip etmektedir.

Baek ve arkadaşları [36] tarafından yapılan çalışma ile yük dağıtım algoritmalarını kullanarak sis ağlarını yönetmeyi amaçlamışlardır. Çalışma kapsamında MDP adı verilen bir pekiştirmeli öğrenme algoritması (Q-learning) geliştirmişlerdir. Çalışma kapsamında yük dağıtımı için sisteme ait şu parametreler kullanılmıştır. Düğümlerin yük dağılımı, düğümlerde bekleyen komut sırası uzunluğu, komşu düğüme gönderilme ihtimali olan işlem sayısı, komşu düğüm komut uzunluğu, işlemcinin anlık işlem kapasitesi, düğümlere işlem gönderilme oranı, yük oluşma ihtimali parametre olarak kullanılmıştır.

Akıllı cihazlar, her düğüm için hesaplanmış düğüm puanını kullanarak kendilerine en uygun düğüm ile iletişime geçerler. Bu çalışma kapsamında düğümlerin seçilmesi düğüm puanı ve akıllı cihaz ile düğüm arasındaki uzaklık olmak üzere iki parametreye bağlıdır. Düğüm puanı işlemci, hafıza, ağ kullanım oranları ile düğüme bağlı kullanıcı ve akıllı cihaz sayısı arasındaki ilişki ile elde edilir.

Düğüm puanı D_P ile düğümün işlemci puanı P_{CPU} ile hafıza puanı P_{MEM} ile ve ağ puanı P_{NET} ile ifade edilir. Düğüm puanı Eş. 4.1'deki gibi ifade edilir.

$$D_P = (P_{CPU} + P_{MEM} + P_{NET}) / 3 \quad (4.1)$$

Eş. 4.2'de işlemci puanı, Eş 4.3'te hafıza puanı, Eş 4.4'te ağ puanı denklemleri görülmektedir. C düğümdeki cihaz sayısını, K düğümdeki komut sayısını, C_{Yo} bir cihazın oluşturduğu ortalama yükü, D_{CPI} düğümün işlem kapasitesi, D_{MPI} düğümün hafıza kapasitesi, D_{NPI} düğüm ağ kapasitesi ifade eder.

$$P_{CPU} = CY_o * C / D_{CPI} \quad (4.2)$$

$$P_{MEM} = CY_o * C / D_{MPI} \quad (4.3)$$

$$P_{NET} = CY_o * C / D_{NPI} \quad (4.4)$$

Bu denklemler ile sis düğümlerinin mevcut kaynaklarının kullanım oranları hesaplanmıştır. Her bir düğümde bir cihazın oluşturduğu yük Eş 4.5'te gösterilmiştir.

$$CY_o = K / C \quad (4.5)$$

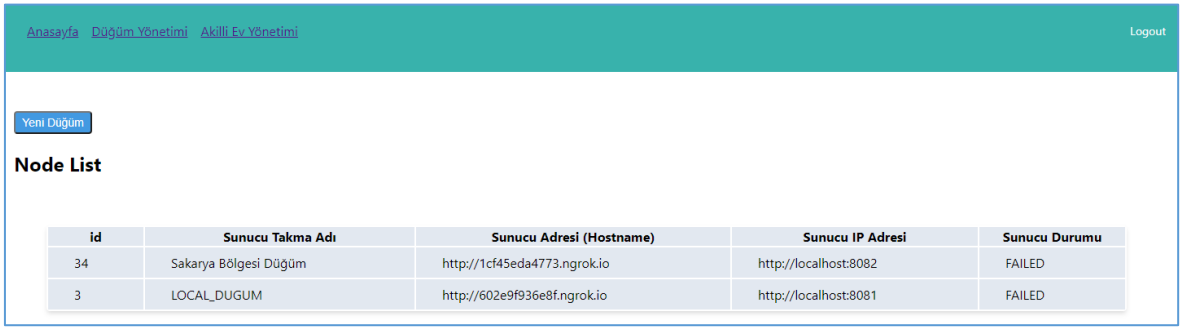
K_{CS} bir komutun işlenmesi için gerekli işlemci kaynağı sabitini, K_{MS} bir komutun işlenmesi için gerekli hafıza kaynağı sabitini, K_{NS} bir komutun işlenmesi için gerekli ağ kaynağı sabitini, D_{CF} düğümün işlem kapasitesini, D_{MC} düğümün hafıza kapasitesi, D_{BW} düğümün ağ kapasitesi ifade eder. Eş. 4.6, Eş. 4.7 ve Eş. 4.8'de komut işlem sabiti ile düğümlerin işleyebilecekleri ortalama komut sayısı belirtilmiştir.

$$D_{CPI} = D_{CF} / K_{CS} \quad (4.6)$$

$$D_{MPI} = D_{MC} / K_{MS} \quad (4.7)$$

$$D_{NPI} = D_{BW} / K_{NS} \quad (4.8)$$

Her bir düğümün düğüm puanı değeri hesaplanarak veritabanında saklanır. Sistem yöneticisi tarafından belirlenen zaman dilimlerinde düğüm puanı değeri güncellenir. Akıllı cihazlar bu değerleri kullanarak en uygun düğüme bağlanma isteği gönderirler. Şekil 4.3'te düğüm yönetimi arayüzü gösterilmiştir. Sistem yöneticisi tarafından erişilebilen arayüzde düğümlerin listesi ve mevcut durumları gösterilmiştir.

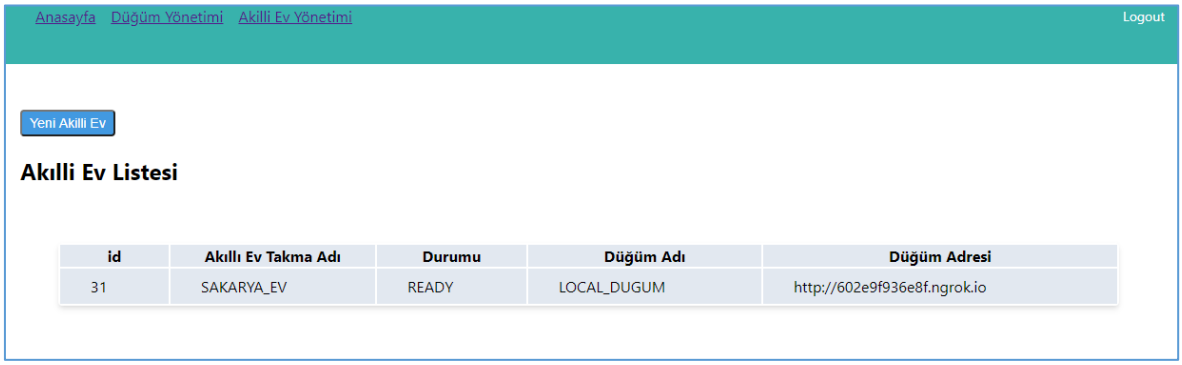


Şekil 4.3. Düğüm Yönetimi Arayüzü

4.1.4. Nesne yönetimi

Nesnelerin interneti uygulamaları içinde bulunan nesneleri yöneten bileşendir. Nesnelerin erişim denetimlerini ve aktif bağlantılarını takip eder. Binlerce düğüm içerisinde hangi düğümde hangi nesnenin olduğu bu bileşen ile yönetilir. Bir nesneye nesne yönetimi birimi üzerinden erişilebilir. Şekil 4.4'te nesne yönetimi arayüzü bulunmaktadır. Tüm kullanıcılara

açık olan bu arayüzde kullanıcılar kendilerine ait tüm akıllı cihazları ve bağlı oldukları düğümleri görüntüleyebilirler.



Şekil 4.4. Akıllı Cihaz Yönetimi Arayüzü

4.1.5. Kayıt günlüğü yönetimi

Sis bilişim platformunda yapılan işlemlerin takip edilmesi, izlenmesi, yönetilmesi zorunludur. Sistemde gerçekleşen her olayın geçmiş işlemlere ait kayıtların saklandığı, oluşturulduğu bir bileşendir.

4.1.6. Nesne erişilebilirliği

Sis platformunda birçok sis düğümü bulunmaktadır. Sis platformuna bağlantı kurmak için nesnelerin kendilerine en yakın ve en uygun sis düğümüne bağlanması gerekmektedir. Bunu sağlamak için nesnelere ait geçmiş işlemler ile hesaplanmış dinamik erişilebilirlik değerinin güncel tutulmasından sorumlu bir bileşendir. Nesneler bu bileşen sayesinde her düğümün erişilebilirliğini temin edip kendilerine en uygun bağlantıyı kurmaktadır.

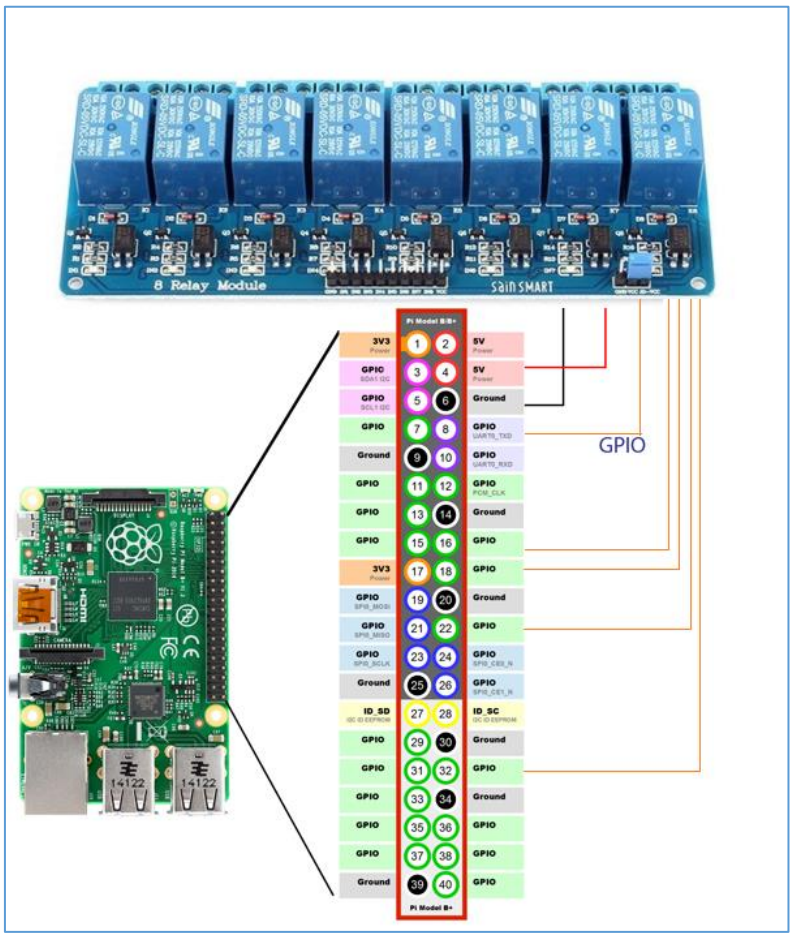
4.1.7. Nesne veritabanı

Sis bilişim platformunu kullanan nesnelerin interneti uygulamalarını sis düğümlerinde ürettikleri verileri saklayabilecekleri bir veritabanına ihtiyaç duymaktadır. Veritabanı servisi ile sis düğümlerinde uygulamalar verilerini saklamaktadır.

4.1.8. Akıllı ev uygulaması

Nesnelerin interneti için özel geliştirilmiş akıllı ev uygulamasıdır. Akıllı ev uygulamasıyla röle kontrolü, algılayıcılardan veri alma gibi yetenekler bulunmaktadır. Akıllı ev uygulamasının çalışır durumda olduğunun kontrolü canlılık kontrol ucu ile sağlanır.

Canlılık kontrol ucu, Sis düğümlerinin sağlık durumlarını kontrol etmek için tanımlanan bağlantı noktalarıdır.



Şekil 4.5. Raspberry Pi Röle Bağlantı Şeması

Yapılan çalışmada bulut ve sis katmanı için Java kullanılmıştır. Sis katmanında uygulamalar docker ile sanallaştırılmıştır. İstemci katmanında Raspberry pi 3 b+ kullanılmıştır. Python ile akıllı nesne istemcisi geliştirilmiştir. Şekil 4.5'te Raspberry pi bağlantı şeması gösterilmiştir.

4.2. Sis Bilişim Katmanları

4.2.1 Bulut katmanı servisleri

Çizelge 4.1'de bulut katmanında geliştirilmiş rest servisleri ve bu servislerin sağlayıcı ve tüketicileri paylaşılmıştır.

Düğüm Listeleme Servisi, akıllı cihazlar tarafından potansiyel bağlanılabilecek düğümlerin listesini almak için tüketilir. Akıllı cihazların sahibine ait özel bir sis düğümü varsa, sadece özel sis düğümü listede görünür. Bu yöntem ile akıllı cihazların sahibine ait sis düğümü ile iletişim kurulması zorunlu hale gelir.

Yeni düğüm kayıt servisi, yeni düğümlerin sisteme eklenmesi işlemini yürütür. Yeni düğüm kayıt servisini docker imaj içerisindeki uygulama tüketir. İki farklı sis düğüm tipi bulunmaktadır. Genel tipte olan sis düğümleri sadece sistem yöneticisi tarafından eklenebilir. Özel tipte olan sis düğümleri ise tüm kullanıcılar tarafından eklenebilmektedir. Özel tipte olan sis düğümleri sadece o kullanıcıların hesabına bağlı akıllı cihazlar tarafından kullanılabilir.

Çizelge 4.1. Bulut Katmanı Web Servisleri

Servis Adı	**Erişim Adresi**	**Sağlayıcı**	**Tüketici**
Düğüm Listeleme Servisi	@GetMapping("/thing/edge") getEdgeList	Düğüm Yönetimi	İstemci Katmanı
Yeni Düğüm Kayıt Servisi	@PostMapping("/edge")Response registerNewEdge(@RequestBody Edge edgeInformation)	Düğüm Yönetimi	Sis Katmanı
Anahtar Doğrulama Servisi	@GetMapping("/edge/token/{tempToken}")Response validateToken(String tempToken)	Kullanıcı Yönetimi	Sis Katmanı
Yeni Nesne Kayıt Servisi	@PostMapping("/edge/{ip}/thing")Response registerNewThing(String ip,Thing thingDto)	Nesne Yönetimi	Sis Katmanı
Nesne Listesi Sorgulama Servisi	@GetMapping("/mobile/thingList")Iterable<Thing> getThingList()	Nesne Yönetimi	Mobil Arayüz
Oturum Açma Servisi	@GetMapping("/authenticate") login(Object loginData)	Kullanıcı Yönetimi	Mobil Arayüz

Sis düğümüne bağlantı kurmak isteyen akıllı cihazların sis düğümüne gönderdiği gizli anahtarın doğrulaması sis düğümü tarafından bulut katmanında Anahtar Doğrulama Servisi ile yapılır.

Sis düğümüne yeni bağlantı kurmak isteyen akıllı cihazın bulut katmanında nesne yönetimi birime kaydedilmesi gerekmektedir. Sis düğümü bağlantı kurmasına izin verdiği tüm nesneleri bulut katmanında nesne yönetimi birimine ait yeni nesne kayıt servisi aracılığıyla kaydeder.

Mobil uygulamalarda işlem yapılacak nesneler ve nesnelerin bağlı olduğu sis düğümleri bilgileri nesne listesi sorgulama servisi aracılığıyla elde edilir. Her kullanıcı kendisine ait hesaba bağlı akıllı cihazları görüntüleyebilir.

Mobil uygulama ve web arayüzünde oturum açmak kullanıcı giriş servisi kullanılır. Kullanıcı tipine göre farklı JWT anahtarlar üretilir. Gönderilen kullanıcı adı ve şifre doğrulaması yapılır.

4.2.2 Sis katmanı servisleri

Çizelge 4.2'de sis katmanına ait servisler ve bu servislerin tanımları paylaşılmıştır.

Çizelge 4.2. Sis Katmanı Servisleri

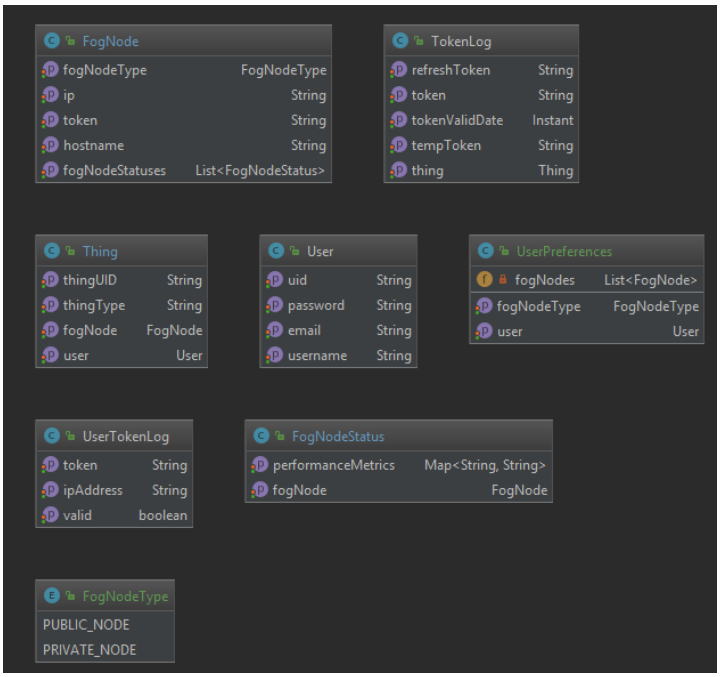
Servis Adı	**Erişim Adresi**	**Sağlayıcı**	**Tüketici**
Sağlık Sorgulama Servisi	@RequestMapping(path = "/healthCheck", method = RequestMethod.GET) Map<Object, Object> healthCheck(HttpServletResponse response)	Liveness Probe	Bulut Katmanı - Düğüm Yönetimi
Bağlantı Oluşturma Servisi	Response<ConnectResponseDTO> connect(ConnectionRequestDTO connectionDTO) {	Erişilebilirlik Yönetimi	İstemci Katmanı- Akıllı Cihaz

Sağlık sorgulama servisi bulut katmanı tarafından sis düğümünün sağlık durumunu kontrol etmek için kullanılır. Sağlık sorgulama servisi, veri tabanı bağlantısını, disk durumunu, işlemci yükünü kontrol eder.

Bağlantı oluşturma servisi akıllı cihazlar bağlantı oluşturma servisi ile seçilen sis düğümüne bağlanırlar. Bağlantı isteği sırasında gönderilen gizli anahtar gönderilir ve doğrulanır. Bağlantı oluşturma servisi bu işlem sonrasında bulut katmanında bulunan nesne yönetimi birimine yeni eklenen akıllı cihaz bilgilerini gönderir.

4.2.3. Bulut katmanı veritabanı modeli

Şekil 4.6'da bulut katmanına ait veritabanı modeli bulunmaktadır.



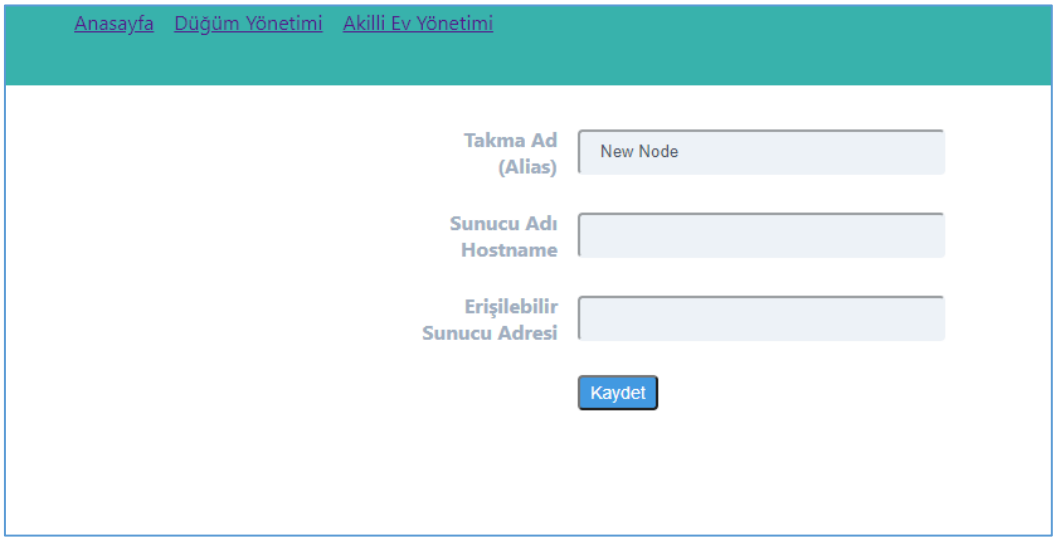
Şekil 4.6. Bulut katmanı veri tabanı modeli

1. *FogNode Tablosu:* Sis Bilişim platformuna bağlı düğümlerin bilgileri bu tabloda tutulmaktadır. Bu tabloda düğümlere ait performans parametreleri de saklanmaktadır.
2. *TokenLog Tablosu*: Sis Bilişim Platformuna yeni bir düğüm eklenmek istendiğinde TokenLog tablosunda bir kayıt oluşturulur. Yeni Düğüm Kayıt işlemi sırasında gönderilen gizli anahtarın doğrulaması bu tablo üzerinden yapılır.
3. *Thing Tablosu*: Düğümlere akıllı cihazlar bağlandıktan sonra sis düğümünün nesne yönetimi birimine bildirimi sonrası Thing tablosu üzerinde varsa akıllı cihaza ait kayıt güncellenerek bağlı bulunduğu düğüm değiştirilir yoksa kayıt oluşturularak akıllı cihazın takibi yapılır.

4. *User Tablosu*: Kullanıcılara ait bilgilerin tutulduğu tablodur. Kullanıcı giriş ve yetkilendirme işlemleri bu tablo aracılığıyla yapılır.
5. *UserPreferences Tablosu*: Kullanıcılara tercihlerin saklandığı tablodur. Kullanıcıların kendilerine bağlı cihazların özel sis düğümlerine ya da herkese açık sis düğümlerine bağlanma tercihleri UserPreferences tablosunda saklanır.
6. *UserTokenLog Tablosu*: Kullanıcılar sisteme giriş yaptıklarında oluşturulan gizli anahtarın saklandığı tablodur. UserTokenLog tablosu ile kullanıcı erişimleri kontrolü sağlanır.

4.2.4. Yeni sis düğümü tanımlama

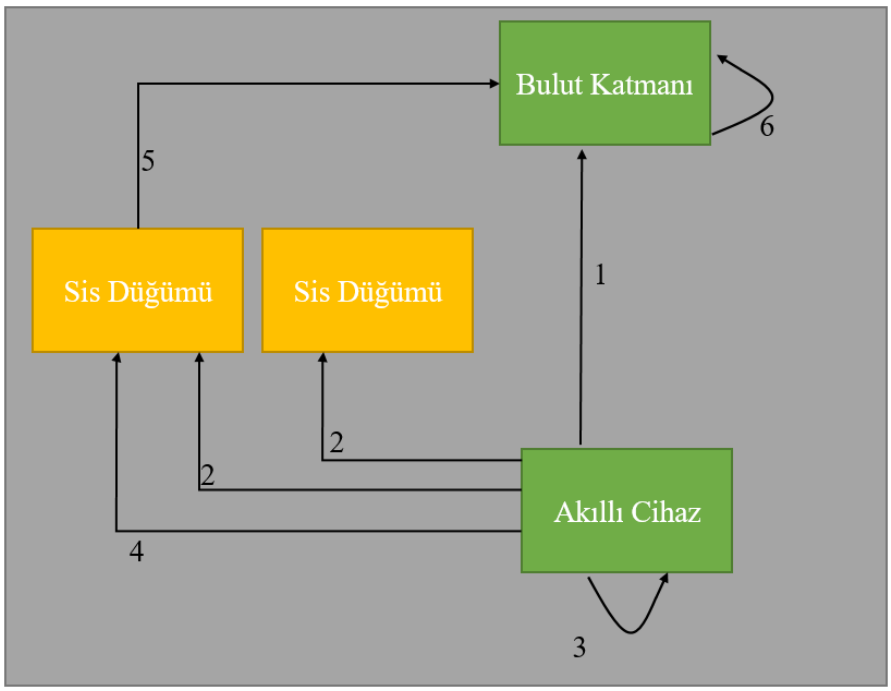
Sistem yöneticisi ve kullanıcıları tarafından yeni sunucuları sis altyapısına eklenebilir. Sis Düğümleri docker ile sanallaştırılmıştır. Bu yüzden sis altyapısına eklenmek istenen sunucuda docker kurulu olmalıdır. Yönetim arayüzü üzerinden girilen parametreler ile yeni düğümün tanımlanması yapılmaktadır. Sistem tarafından bir docker komutu oluşturulur ve bu komut eklenmek istenen sunucuda çalıştırılarak yeni düğüm eklenmiş olur. Bu yöntem hem sistem yönetimi hem de kullanıcılar için aynıdır. Kullanıcılar ekledikleri özel düğümlerin kullanımını zorunlu kılarak kendilerine ait akıllı nesnelerin özel düğümlere bağlanmalarını sağlarlar. Sistem güvenliği kullanıcı sorumluluğundadır. Şekil 4.7'de yeni sis düğümü tanımlama ekranı gösterilmiştir. Sis düğümü eklendikten sonra ekranda gözüken docker komutu yeni sis düğümünde çalıştırılarak sisteme dâhil edilir.



Şekil 4.7. Yeni Düğüm Tanımlama Ekranı

4.2.5. Cihazların sis düğümü ile iletişimi

Sis Bilişim altyapısında farklı konumlarda çalışan birçok sis düğümü olabilir. Düğümler üzerindeki yüklerin yoğunlukları da farklı olabilir. Akıllı cihazlar bağlanacakları düğümü kendileri seçerler. Akıllı cihazlar erişebileceği sis noktalarının listesini bulut katmanından temin edip erişebildiği en uygun sis noktasıyla bağlantıya geçerek tüm işlemlerini bu sis noktası üzerinden ilerletir. Herhangi bir sis noktası erişilemez duruma geldiğinde aygıtlar tekrar bağlantı isteğinde bulunurlar ve kendilerine en uygun sis noktasına bağlanırlar. Şekil 4.8'de akıllı cihazların sis düğümü seçim akışı gösterilmiştir.



Şekil 4.8. Akıllı Cihazların Sis Düğümüne Bağlantı Akışı

1. Akıllı cihaz bağlanabileceği düğümlerin listesini bulut katmanı servisinden alır.
2. Düğüm listesindeki tüm düğümlere erişim talebinde bulunur. Düğümler ortalama komut işleme zamanını akıllı cihaza bildirirler.
3. Akıllı cihaz, tüm düğümlerden kendisine en yakın ve en hızlı düğümü seçip ona bağlanır.
4. Bağlantı isteği düğüme ulaştığında gönderilen gizli anahtar bulut katmanındaki erişim kontrolü servisine gönderilir.
5. Akıllı cihazın düğüme bağlanmaya yetkili olup olmadığını sorgular.

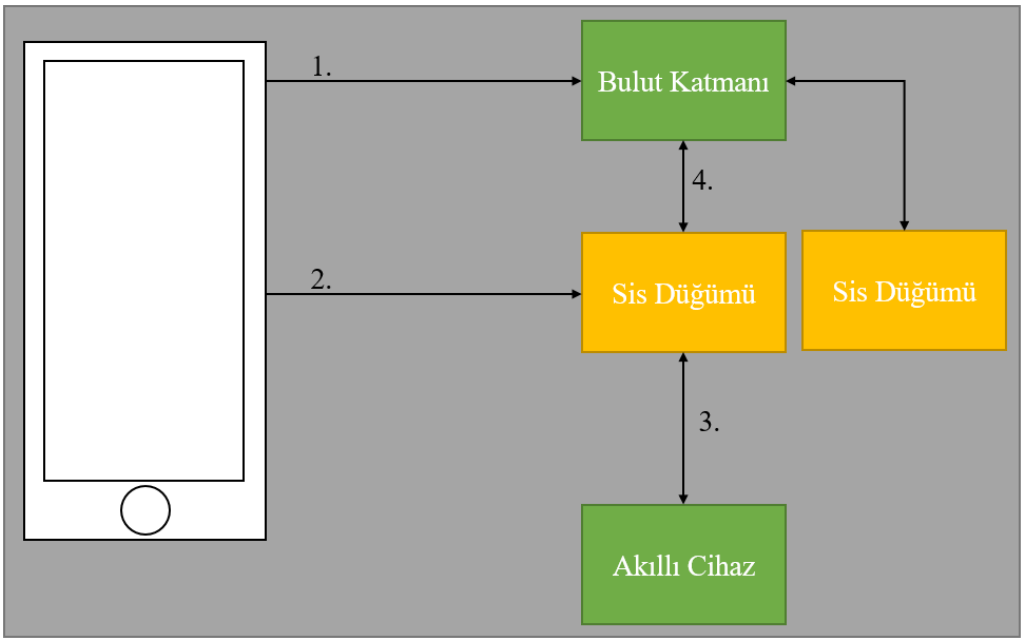
6. Başarılı cevap dönülürse bağlantı onaylanır ve Nesne yönetimi servisi akıllı cihazın bilgilerini günceller.

Sis ağlarının yük dağılımı (load balancer), gelen yükün etkili bir şekilde düğümler arasında dağıtılmasını ifade eder. Dinamik veya sabit yöntem ile yük dağıtımı yapılabilir. Sabit yük dağıtım yöntemi ile sis düğümleri sırayla seçilerek yük dağıtımı yapılır. Akıllı cihazların en yakın düğüme bağlanmalarını sağlamak sis bilişimin ana amacıdır ancak en yakın sis düğümü üzerindeki yükün dağıtılması ve bazı düğümler üzerinde yük oluşmaması gerekmektedir. Sadece bir bölgede akıllı cihazların yoğun olarak kullanıldığını düşünürsek o bölgeye ait düğümde yoğunluk oluşması muhtemel bir durumdur. Bu çalışma kapsamında dinamik yük dağıtımı yapılmıştır. Akıllı cihazlar bağlantı kuracakları aktif sis düğümleri listesini bulut katmanından elde ederler. Sis düğümü listesinde sis düğümlerine ait düğüm puanları bulunur. Düğüm puanı ile ilgili detaylar düğüm yönetimi bölümünde aktarılmıştır. Sis düğümü ile akıllı cihaz arasındaki gecikme zamanı G_z olarak ifade edilir. Düğüm puanı ve gecikme zamanı kullanılarak düğüm seçme puanı (D_{SP}) kullanılarak elde edilir. Düğüm puanı sistem yöneticisi tarafından belirlenen en yüksek puandan fazla ise düğüm seçim listesine dâhil edilmez. Elde edilen değerler, sistem yönetimi tarafından belirlenen gecikme zamanı ve düğüm puanı aralıkları içerisinde ise düğüm seçimi gerçekleşir.

4.2.6. Akıllı cihaz erişilebilirliği

Sis bilişim ağında birçok düğüm bulunmaktadır. Dağıtık olarak hizmet veren sis bilişim altyapısında düğümler ile akıllı cihazlar arasında bir bağlantı vardır ve sis bilişim ağında akıllı cihazlara erişim nesne yönetimi servisi aracılığıyla sağlanır. Şekil 4.9'da akıllı ev uygulamasına mobil cihazdan işlem gönderme akışı gösterilmiştir.

1. Bir akıllı cihaza erişilmek istendiğinde bulut katmanındaki nesne yönetimi servisinden akıllı cihazlar ve ilişkili düğümleri sorgulanır.
2. Akıllı cihazlara mesaj gönderilmek istendiğinde düğüm üzerinden mesaj iletilir.
3. Akıllı nesneler istemcilere bir mesaj iletmek istediğinde bağlı oldukları düğüm üzerinden iletişim kurar.
4. Bulut katmanı tarafından sis düğümünün düzenli olarak sağlık kontrolü yapılır.



Şekil 4.9. Akıllı Cihaz Erişilebilirliği

4.2.7. Canlılık kontrol ucu

Sis bilişimde kayıtlı birçok düğüm bulunur ve bu düğümlerin canlılık kontrollerinin yapılması gerekir. Düğümlerin bazıları kapanmış ya da hizmet veremez duruma gelmiş olabilir. Kesintilerin veya performans sorunlarının önceden tespiti sistem sağlığı açısından çok önemlidir. Sistemin çalışır ve hazır bir durumda olduğunun tespitini yapmak için sis düğümlerinde Canlılık kontrol ucu tanımlanmıştır. Canlılık kontrol ucu sis düğümünde veri tabanı, disk, işlemci durumu gibi bilgileri kontrol eder. Sis düğümünün çalıştığını garanti eder. Bir sorun ile karşılaşıldığında bu düğüm erişilebilir olmaktan çıkarılır ve yeni akıllı cihazların bu düğüme bağlanması engellenir. Sis düğümünde sağlanan bu servis bulut katmanından düzenli olarak çağrı yapılarak kontrol edilir.

Sis bilişim platformu sis düğümlerini kontrol etmek için iki farklı servisi kontrol etmektedir. Canlılık ve kullanıma hazırlık servisleri sis düğümlerinin hizmete devam edebilmesi için düzenli olarak kontrol edilir.

Kullanıma hazır servisi, sis düğümünün sisteme dâhil edilirken çalıştırıldığı ve tam olarak kullanıma hazır olup olmadığını kontrol etmektedir. Bulut katmanı tarafından düğümün kullanıma hazır durumuna geçmesinden sonra bu kontrol tekrarlanmaz. Bu servisten başarılı cevap dönülmez ise bu düğüm sis bilişim altyapısına dâhil edilmez.

Canlılık servisi, sis bilişim platformu tarafından düzenli olarak çağrı yapılarak kontrol edilir. Canlılık servisi çağrım sıklığı ayarlama dosyalarından ayarlanabilir. Http servis çağrıları yapılarak sis düğümünün sağlık durumu kontrol edilir.

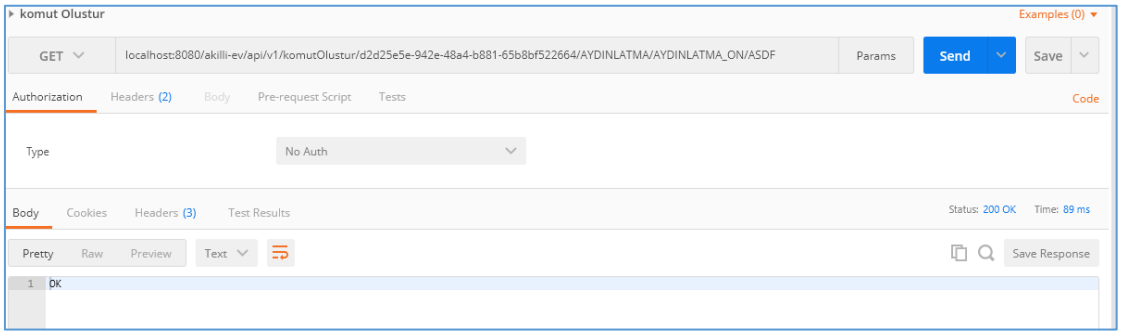
4.3. Akıllı Ev Uygulaması Akışları

Bu bölümde geliştirilen akıllı ev uygulamasına ait rest servis çağrıları ve akışları tanımlanmıştır. Akıllı telefon üzerinden tanımlı komutlarla uç cihazda algılayıcı, röle kontrolü mümkündür. Akıllı ev uygulamasında bulunan nesneler ve nesnelerin alabileceği komutlar Çizege 4.3'te gösterilmiştir.

Çizelge 4.3. Nesne Özellikleri

Nesne Tipi	Aydınlatma	Gaz	Elektrik	Kombi	Şohben	Klima
Açma/Kapatma	Var	Var	Var	Var	Var	Var
Komut Parametreleri	Yok	Yok	Yok	Sıcaklık Değeri	Yok	Sıcaklık Değeri

4.3.1. Android uygulaması üzerinden cihazı yönetme



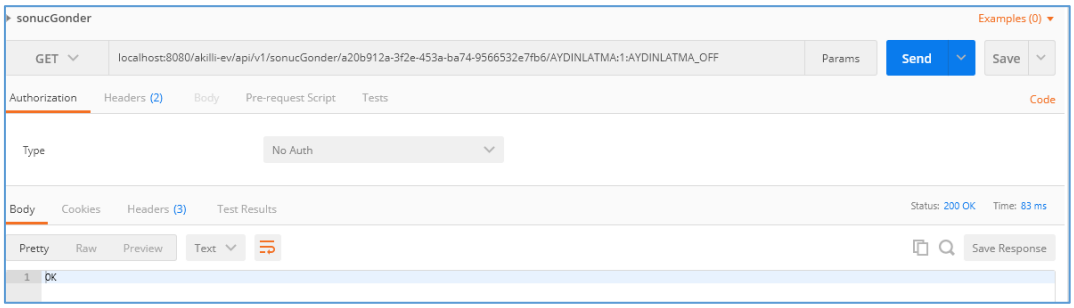
Şekil 4.10. Komut gönderme servisi

Sis bilişim altyapısını kullanan akıllı ev uygulamasına ait Android işletim sistemi için geliştirilmiş mobil uygulama üzerinden akıllı cihazlar yönetilebilmektedir. Mobil uygulama üzerinden oluşturulan bir komutun akıllı cihaza iletilmeden önce sis düğümüne iletilmesi

gerekmektedir. Öncelikle komut gönderilmek istenen uç cihazın hangi sis düğümüne bağlı olduğu bulut katmanı servislerinden elde edilir ve şekildeki istek sis düğümüne iletilir. Şekil 4.10'da komut gönderme servisinin istek ve cevabı görünmektedir.

4.3.2. Cihazın sonuçları sis düğümüne iletmesi

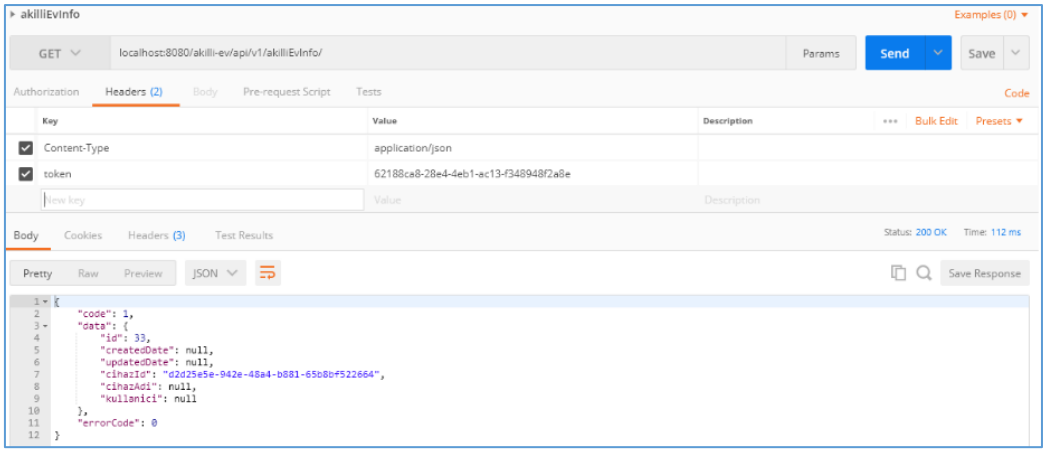
Akıllı cihaz kendisine iletilen komutları düzenli olarak kontrol ederek işleme almaktadır. İşlediği komutların sonuçlarını da bağlı olduğu sis düğümüne iletmektedir. Şekil 4.11'de görünen istek ile hangi nesne için hangi komut talimatını işlediğini belirterek işlem sonucunu sis düğümüne iletir.



Şekil 4.11. Komut cevaplama servisi

4.3.3. Sonuçların android uygulamasına aktarılması

Sis bilişim altyapısında birden çok düğüm bulunmaktadır. Bulut katmanı içerisinde bulunan nesne servisi ile uç cihazların hangi düğüme bağlı olduğu takip edilmektedir. Mobil uygulama üzerinden verilen bir komut işleme talimatının sonucu sis düğümleri üzerinde saklanmaktadır. Mobil uygulama üzerinden verilen komut işleme talimatının sonucu bulut katmanından ilgili uç cihazının bağlı olduğu sis düğümü sorgulanarak elde edilmektedir. Şekil 4.12'de görünen istekte uç cihazın tüm nesnelerine ait bilgiler görülmektedir.



Şekil 4.12. Uç sistem durum servisi

5. BULGULAR VE DEĞERLENDİRME

Mevcut bulut bilişim altyapıları tüm işlemleri veri merkezlerindeki sunucular ile yapmaktadır. Ancak nesnelerin interneti uygulamalarının gelişmesi, kullanım alanlarının yaygınlaşması ve gömülü sistemlerin daha ulaşılabilir olmasıyla üretilen veri miktarında artış beklenmektedir. Üretilen tüm verilerin bulut sistemlere taşınması mevcut internet altyapıları ile mümkün değildir.

Bulut Bilişim altyapıları ölçeklenebilir servisler sunmaktadır. Web Servis uygulamalarını sadece veri merkezlerinde çoğaltmak yeterli değildir. Uygulamaların farklı veri merkezlerinde, intranette problemsiz çalışmasını sağlayacak bir altyapıya ihtiyaç vardır.

Bu çalışma ile çözmeye odaklandığımız problemlerden ilki ölçeklenebilir bir akıllı ev mimarisi ortaya koymaktır. Yatay ölçekleme yöntemini benimsediğimiz bu çalışma ile farklı veri merkezlerinde, internette veya intranette bulunan sunucular sis hesaplama platformuna dâhil edilebilir. Kullanıcıların tercihlerine bağlı olarak tanımlı sunucular üzerinde işlemlerini yürütebilirler.

Mevcut bulut bilişim sağlayıcıları bile, Amazon gibi, beklenmedik kesintilerle karşılaşabiliyor. Bulut bilişim altyapılarında yaşanan kesintiler hem itibar hem de maddi kayıplara sebep olmaktadır. Bu gibi kesintileri dikkate alındığında tek bir bulut bilişim altyapısında hizmet vermek beklenmedik hizmet kesintilerine neden olmaktadır.

Bu çalışma ile sunucuların ve sis düğümlerinin farklı bulut bilişim platformlarında birlikte çalıştırılabilmesi amaçlanmıştır. Farklı bulut bilişim altyapılarında aynı anda çalışan sis düğümleri sayesinde, platformlardan 1 tanesi erişilemez hale gelse dahi sistem çalışmaya devam edecektir. Aynı zamanda platforma kullanıcıların kendilerine özel sis düğümü ekleyebilmeleri sayesinde bulut altyapılardan bağımsız bir çalışma imkânı ortaya çıkmıştır.

Bulut bilişim altyapıları üzerinde yapılan operasyonlar sanal sunucu seviyesinde, uygulama sunucusu seviyesindedir. Ancak bulut bilişim altyapılarında uygulama özelinde bir yönetim söz konusu değildir. Yapılan bu çalışma ile yeni sis düğümlerinin bulut bilişim

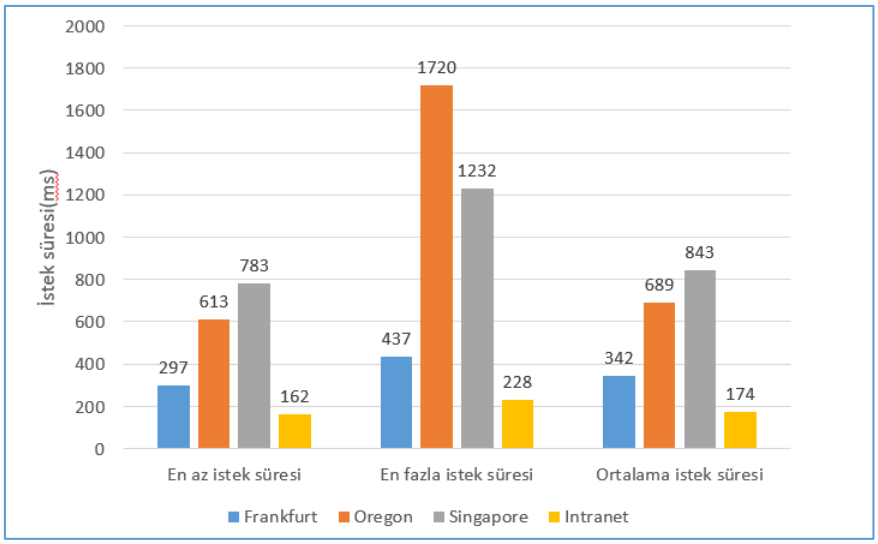
platformlarından bağımsız hale getirilmesi sağlanmıştır. Sunucu kümeleri bulut katmanında bulut bilişim altyapısından bağımsız bir şekilde dağıtık olarak yönetilmektedir.

Mevcut nesnelerin interneti uygulamalarının sunucuları bulut bilişim altyapıları üzerinde çalıştırılmaktadır. Veri gizliliği içeren uygulamalarda ya da bulut bilişim altyapısını kullanmak istemeyen şirket, kurum ve kuruluşlar kendi sunucuları üzerinde bu operasyonları yürütmektedir. Elle yürütülen bu süreç bu çalışma ile sis bilişim altyapısı içerisinde otomatik hale getirilmiştir. Kullanıcıların kendilerine ait olarak eklenmiş sis düğümleri istenilen nesnelerin interneti uygulamasının çalıştırılmasına imkân sağlar.

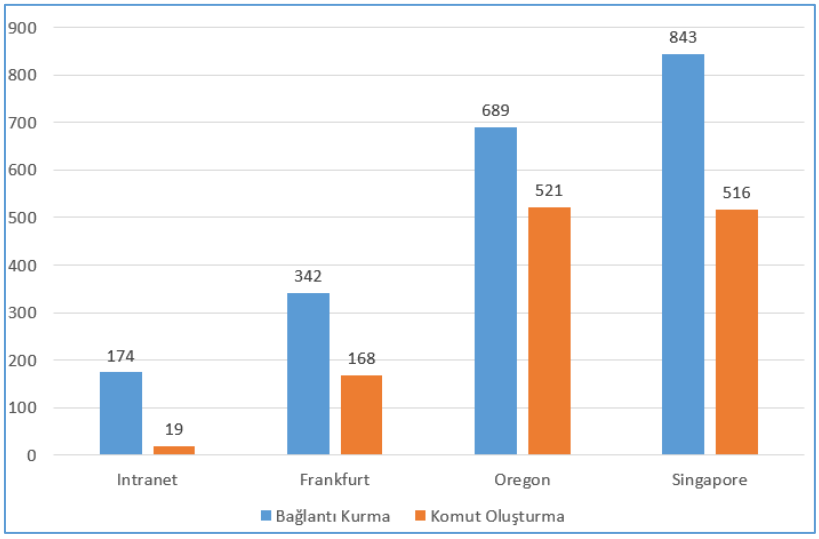
Performans değerlendirmesi için yapılan çalışmada bulut katmanı ve sis katmanı için Amazon Elastic Cloud Computing (EC2) c4.2xlarge (8vCPU, 15GB RAM, EBS, High Network-1Gbit) kullanıldı. Bulut katmanı yazılımı N. Virginia bölgesinde c4.2xlarge sunucu üzerinde çalıştırıldı. Amazon EC2 servisinin bulunduğu 3 farklı noktaya da sis düğümü eklendi. 1 sis düğümü de intranet üzerinde çalıştırıldı. API performans testlerinin yapılabilmesi için Jmeter kullanıldı. Testler i56600 işlemci, 16GB RAM, 240GB SSD, 50Mbps hızında internet bağlantılı bir istemcide çalıştırıldı. Testler sırasında 4 farklı yoğunlukta 2 senaryo çalıştırıldı. Saniyede 100, 200, 400, 800 istek gönderildi. Sunucu ve istemci bağlantı limitlerine ulaştığı ve bağlantı hatası aldığı için saniyede 800 istek gönderilen testler göz ardı edildi. Performans sonuçları Çizelge 2.2'de elde edilen sonuçlar ile uyumlu sonuç vermiştir. Çizelge 5.1'de sis düğümüne gönderilen istek ve işlem süreleri bulunmaktadır. Bu istek bulut katmanı ile etkileşime girmektedir.

Akıllı cihazların bağlantı kurma isteğine ait performans veriler Şekil 5.1'de görülmektedir. Çizelge 5.1'de ham veriler gösterilmiştir. Testler sırasında saniyede 100-200-400-800 istek gönderilmiştir. Bağlantı sınırlarına ulaşıldığı ve bağlantı hataları oluşmaya başladığı için saniyede 800 istek senaryosu göz ardı edilmiştir. Bağlantı kurma isteği sırasında sis düğümleri bulut katmanında bulunan kullanıcı yönetimi ve nesne yönetimi birimiyle etkileşime geçmiştir. Bu yüzden sis düğümü ile bulut katmanı arasındaki mesafe de önem kazanmıştır. İstemci-Sis Düğümü arasındaki gecikmeye ek olarak Sis düğümü-Bulut katmanı arasındaki gecikme de eklenmektedir. İstemci Yozgat'ta çalıştırılmıştır. Veriler incelendiğinde doğal sonuç olarak intranet sis düğümü, en iyi hız değerlerini vermiştir. Sonrasında en yakın konum olan Almanya-Frankfurt gözükmektedir. Oregon, Singapur'a göre daha yakın olduğu için daha hızlı cevap vermiştir.

Mobil cihazların akıllı cihazlarda çalıştırılmak üzere çağırdığı komut oluşturma servisine isteğine ait performans veriler Şekil 5.2'de görülmektedir. Intranet verileri beklendiği gibi ağ gecikmesi yaşanmadığı için en iyi sonuçlardır. Frankfurt diğer sis düğümlerine göre konum olarak Yozgat'a daha yakın olduğu için gecikme daha düşüktür.

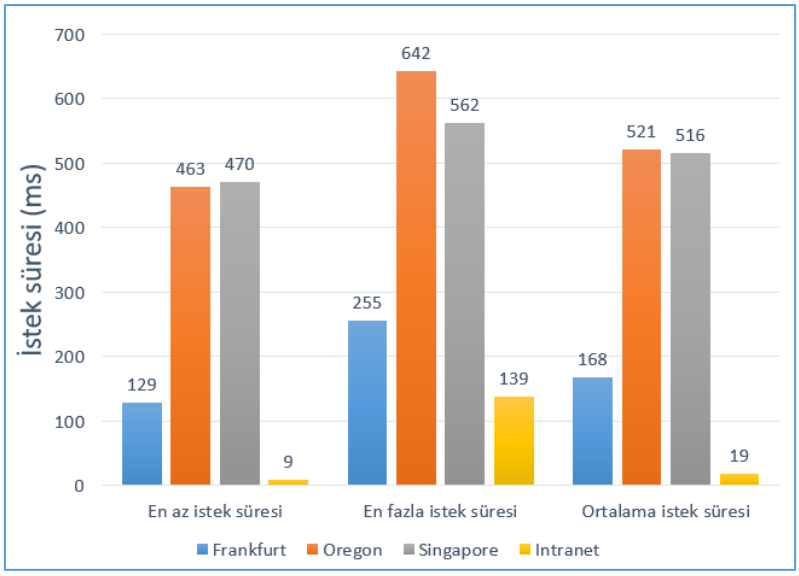


Şekil 5.1. Bağlantı kurma isteği süreleri



Şekil 5.2. Düğümlerin cevap süreleri

Şekil 5.2'de iki farkı istek türünün ortalama cevaplama sürelerini göstermektedir. Şekil 5.3'de komut oluşturma isteğine ait cevap süreleri bulunmaktadır. Komut oluşturma servisi ile bağlantı kurma arasındaki farkın nedeni sis düğümünün bağlantı kurma isteği için bulut katmanıyla iletişime geçmesidir. Intranet için bu gecikme yaklaşık 150ms, Frankfurt için 160ms, Oregon için 170ms ve Singapur için yaklaşık 330ms bir gecikmeye neden olmuştur.



Şekil 5.3. Komut oluşturma isteği süreleri

Çizelge 5.1. Performans sonuçları

Bağlantı Kurulan Şehir/Servis adı	100 istek/saniye			200 istek/saniye			400 istek/saniye		
	En kısa (ms)	En uzun (ms)	Ortalama (ms)	En kısa (ms)	En uzun (ms)	Ortalama (ms)	En kısa (ms)	En uzun (ms)	Ortalama (ms)
Frankfurt / komutOlustur	129	255	168	139	1071	555	127	2832	1619
Frankfurt / Connect	297	437	342	317	9726	2408	360	17017	8243
Oregon / komutOlustur	463	642	521	453	1400	879	468	3306	1934
Oregon / Connect	613	1720	689	683	10695	2944	635	16905	8695
Singapore / komutOlustur	470	562	516	486	1423	884	463	3158	1807
Singapore / connect	783	1232	843	809	10217	2925	783	16610	8667
Intranet-connect	162	228	174	165	349	181	164	723	209
Intranet-komutOlustur	11	62	26	9	139	19	9	139	19

Nesnelerin interneti ağının mevcut büyüklüğü ve gelecek öngörüleri düşünüldüğünde üretilecek verilerin işlenmesi bulut bilişim altyapıları ile mümkün değildir. Bu çalışmada

bulut bilişim kullanan nesnelerin interneti uygulamaları için genişletilmiş bir sis bilişim mimarisi geliştirilmiştir.

Nesnelerin interneti ile ilgili yapılmış tüm çalışmalar bazı öngörüler sunmaktadır. Tüm bu öngörülerin ortaklaştığı tek nokta üretilecek verinin büyüklüğü ve bağlanacak cihaz sayısının büyüyeceği yönündedir. Mevcut bulut bilişim altyapılarında çalışan nesnelerin interneti uygulamalarının veri iletim hızı, bant genişliği, akıllı cihazların kaynak kısıtı, internet sürekliliği gibi problemler bulunmaktadır. Bu problemler üretilen verilerin tamamını bulut altyapılarına taşıdığımızda daha çok büyüyecektir. Yapılan çalışma ile bulut bilişim ile sis bilişim altyapısı geliştirilmiştir. Akıllı cihazların en yakın sis düğümüne bağlantı kurması hedeflenmiş ve sonuçları gözlemlenmiştir.

Tüm veriyi bulut altyapısına taşımak yerine daha yakın noktaya taşımanın sonuçları paylaşılmış ve başarılı sonuç elde edildiği görülmüştür. Tüm işlemlerin sis düğümü üzerinden yapılması ile performans artışı sağlamıştır. Sadece bağlantı kurma isteğiyle ilgili özel bir durum bulunmaktadır. Sis katmanında bulunan bağlantı kurma servisinin bulut katmanı ile iletişime geçmesi nedeniyle performans artışı daha azdır.

Üretilen çözüm ile farklı noktalara hem sistem yöneticisi tarafından genel hem de kullanıcıların kendilerine özel sis düğümlerinin eklenmesine imkân sağlanmıştır. Kullanıcıların tercihlerine bağlı olarak akıllı cihazların genel veya özel sis düğümlerine bağlanabilmesi sağlanmıştır. Farklı konumlarda bulunan sis düğümlerinin yönetimi, sağlık durumlarının kontrolü için bir bulut katmanında sis düğüm yönetimi eklenmiştir. Sis düğümlerinin veri tabanı erişimi, disk erişimi, işlemci yükü, hafıza kullanımı verilerinin düzenli kontrolü sağlanmıştır.

Devlet kurumları ve özel şirketler nesnelerin interneti uygulamalarını yoğunlukla kullanmaktadırlar. Kurum ve kuruluşlar nesnelerin interneti uygulamalarını hem maliyet hem de gizlilik endişeleri nedeniyle kendi veri merkezlerinde çalıştırmak isteyebilirler. Bizim önerdiğimiz mimari ve yöntem ile bu tür talepleri karşılamak mümkün hale gelmiştir.

6. SONUÇ

Nesnelerin interneti çözümlerinde sis bilişim mimarilerinin uygulanması kaçınılmaz görünmektedir. Bu çalışmada sis bilişimin özellikleri, mevcut mimarilerin kısıtları, akıllı cihazlar ve uç sistemlerin etkileşimi hakkında araştırmalar yapılmıştır. Örnek bir model geliştirilmiş ve yatay ölçeklemeye imkân tanıyan bir mimari model önerilmiştir. Akıllı cihazlar ve sis düğümleri arasındaki güvenlik kontrolleri, akıllı cihazların uygun sis düğümünü seçebilme yeteneği, mobil istemcilerin akıllı cihazlara erişimleri üzerine çalışılmıştır. Dinamik sis düğümü yönetimi sayesinde kullanıcıların kendi düğümlerini ekleyebilmeleri sağlanmıştır. Dinamik sis düğümü yük dağıtımı yöntemiyle düğümlere gelen yükün dağıtılması amaçlanmıştır. Düğümlere yük dağıtımı yapılırken düğümlerdeki cihaz sayısı, komut sayısı, düğümün işlem kapasitesi, hafıza kapasitesi, ağ kapasitesi, bir komutun işlenmesi için gerekli donanım kaynağı ihtiyacı parametreleri ile bir algoritma geliştirilmiştir. Kullanıcıların akıllı cihazlarını özel düğümlere bağlanmasına imkân verilmiştir. Sistem performans odaklı test edilmiş ve önerilen mimarinin sis düğümlerinin yakınlık ve yoğunluk durumuna göre %60 ile %200 performans artışı gözlemlenmiştir. Bu çalışmada sis düğümlerinin erişilemez duruma geldiğinde verilerin yeni düğümlere taşınması kapsamın dışında tutulmuştur.

KAYNAKLAR

1. Plakhteyev, A., Perepelitsyn, A. and Frolov, V. (2018). Edge computing for IoT: An educational case study. *2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies*. 130-133.

2. Yi, S., Hao, Z., Qin, Z. and Li, Q. (2015). Fog computing: Platform and applications. *2015 Third IEEE Workshop on Hot Topics in Web Systems and Technologies*. 73-78.

3. İnternet: Fog Computing: Fog and Cloud along the Cloud-to-Thing Continuum, i-Scoop, 26 July 2020, URL: www.i-scoop.eu/internet-of-things-guide/fog-computing-cloud-internet-things/. Son Erişim Tarihi: 01.07.2020.

4. İnternet: Meulen, R. (October). What Edge Computing Means for Infrastructure and Operations Leaders. URL: https://www.gartner.com/smarterwithgartner/what-edge-computing-means-for-infrastructure-and-operations-leaders. Son Erişim Tarihi: 01.07.2020.

5. Fan, Q. and Ansari, N. (2018). Towards workload balancing in fog computing empowered IoT. *IEEE Transactions on Network Science and Engineering.* 1, 253-262

6. Kaul, L. and Goudar, R. H. (2016). Internet of things and Big Data-challenges. *2016 Online International Conference on Green Engineering and Technologies*. 1-5.

7. Verba, N., Chao, K. M., Lewandowski, J., Shah, N., James, A. and Tian, F. (2019). Modeling industry 4.0 based fog computing environments for application analysis and deployment. *Future Generation Computer Systems*. 91, 48-60.

8. Jiaman, D., Sichen, W., Yi, D. and Lianyin, J. (2016). A Dynamic Migration Method for Big Data in Cloud. *2016 17th International Conference on Parallel and Distributed Computing, Applications and Technologies*. 95-98.

9. Bellavista, P., Corradi, A., Foschini, L. and Scotece, D. (2019). Differentiated Service/Data migration for edge services leveraging container characteristics. *IEEE Access.* 7, 139746-139758.

10. Mohamed, N., Al-Jaroodi, J. and Jawhar, I. (2019). Towards fault tolerant fog computing for IoT-based smart city applications. *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*. 0752-0757.

11. Naha, R. K., Garg, S., Chan, A. and Battula, S. K. (2020). Deadline-based dynamic resource allocation and provisioning algorithms in Fog-Cloud environment. *Future Generation Computer Systems*. 104, 131-141.

12. Weiner, M., Jorgovanovic, M., Sahai, A. and Nikolié, B. (2014). Design of a low-latency, high-reliability wireless communication system for control applications. *2014 IEEE International Conference on Communications (ICC)*. 3829-3835.

13. Kelly, R. (2016). Internet of things data to top 1.6 zettabytes by 2022. *Campus Technology*. 9, 1536-1233.

14. Premsankar, G., Di Francesco, M. and Taleb, T. (2018). Edge computing for the Internet of Things: A case study. *IEEE Internet of Things Journal*. 5(2), 1275-1284.

15. Moreno-Vozmediano, R., Montero, R. S., Huedo, E. and Llorente, I. M. (2017). Cross-site virtual network in cloud and fog computing. *IEEE Cloud Computing*. 4(2), 46-53.

16. Zahra, S., Alam, M., Javaid, Q., Wahid, A., Javaid, N., Malik, S. U. R. and Khan, M. K. (2017). Fog computing over Iot: A secure deployment and formal verification. *IEEE Access*. 5, 27132-27144.

17. Carrega, A., Repetto, M., Gouvas, P. and Zafeiropoulos, A. (2017). A middleware for mobile edge computing. *IEEE Cloud Computing*. 4(4), 26-37.

18. Suganuma, T., Oide, T., Kitagami, S., Sugawara, K. and Shiratori, N. (2018). Multiagent-based flexible edge computing architecture for iot. *IEEE Network*. 32(1), 16-23.

19. Dang, T. D. and Hoang, D. (2017). A data protection model for fog computing. *2017 Second International Conference on Fog and Mobile Edge Computing (FMEC)*. 32-38.

20. Fan, K., Pan, Q., Wang, J., Liu, T., Li, H. and Yang, Y. (2018). Cross-Domain Based Data Sharing Scheme in Cooperative Edge Computing. *2018 IEEE International Conference on Edge Computing*. 87-92.

21. Yi, S., Hao, Z., Qin, Z. and Li, Q. (2015). Fog computing: Platform and applications. *2015 Third IEEE Workshop on Hot Topics in Web Systems and Technologies*. 73-78.

22. Bonadio, A., Chiti, F. and Fantacci, R. (2019). Performance Analysis of an Edge Computing SaaS System for Mobile Users. *IEEE Transactions on Vehicular Technology*. 69(2), 2049-2057.

23. Hussain, Y., Zhiqiu, H., Akbar, M. A., Alsanad, A., Alsanad, A. A. A., Nawaz, A. and Khan, Z. U. (2020). Context-Aware Trust and Reputation Model for Fog-Based IoT. *IEEE Access*. 8, 31622-31632.

24. Rocha Filho, G. P., Meneguette, R. I., Maia, G., Pessin, G., Gonçalves, V. P., Weigang, L. and Villas, L. A. (2020). A fog-enabled smart home solution for decision-making using smart objects. *Future Generation Computer Systems*. 103, 18-27.

25. Chekired, D. A., Khoukhi, L. and Mouftah, H. T. (2019). Fog Computing Based Energy Storage in Smart Grid: A Cut-off Priority Queuing Model for plug-in Electrified Vehicles Charging. *IEEE Transactions on Industrial Informatics*. 16(5), 3470-3482.

26. Baucas, M. J. and Spachos, P. (2020). A scalable IoT-fog framework for urban sound sensing. *Computer Communications*. 103, 18-27.

27. Liu, W., Huang, G., Zheng, A. and Liu, J. (2020). Research on the optimization of IIoT data processing latency. *Computer Communications*. 151, 290-298.

28. Erdal, E. (2018) Sis Bilişim Tabanlı İmza Doğrulama: Senaryoya Dayalı Bir Yaklaşım. *Uluslararası Mühendislik Araştırma ve Geliştirme Dergisi*. 11(1), 64-76.

29. Güven, E. Y. (2018). *Kenar Bilişim İçin Siber Saldırıları Tespit ve Önleme Yöntemleri*. Yüksek lisans tezi, Fatih Sultan Mehmet Vakıf Üniversitesi Mühendislik ve Fen Bilimleri Enstitüsü

30. Hasan, K. (2018). *An Adaptive Offloading Decision Scheme in Two-Class Mobile Edge Computing Systems*. Yüksek lisans tezi, İstanbul Teknik Üniversitesi Bilişim Enstitüsü, İstanbu

31. Eşrefoğlu, R.M. (2018). *Realistic Performance Evaluation Of Edge Computing Scenarios Over A Hybrid Testbed Using Software-Defined Networking.* Yüksek lisans tezi, Boğaziçi Üniversitesi Fen Bilimleri Enstitüsü, İstanbul

32. Karataş, F. (2019). *Nesnelerin interneti uygulamaları için hibrit sis-bulut tabanlı veri dağıtımı.* Doktora tezi, İhsan Doğramacı Bilkent Üniversitesi / Mühendislik ve Fen Bilimleri Enstitüsü, Ankara

33. OKAY, F.Y. (2019*). Nesnelerin interneti için sis hesaplama tabanlı veri kümeleme ve yönlendirme modelleri*. Doktora tezi, Gazi Üniversitesi / Fen Bilimleri Enstitüsü, Ankara

34. Combe, T., Martin, A. and Di Pietro, R. (2016). To Docker or Not to Docker: A Security Perspective. *IEEE Cloud Computing*. 3(5), 54-62.

35. Ahlgren, B., Hidell, M. and Ngai, E. C. H. (2016). Internet of things for smart cities: Interoperability and open data. *IEEE Internet Computing*. 20(6), 52-56.

36. Baek, J. Y., Kaddoum, G., Garg, S., Kaur, K. and Gravel, V. (2019, April). Managing fog networks using reinforcement learning based load balancing algorithm. *2019 IEEE Wireless Communications and Networking Conference* . 1-7.

37. Marković, D., Koprivica, R., Pešović, U. and Randić, S. (2015). Application of IoT in monitoring and controlling agricultural production. *Acta Agriculturae Serbica*. 20(40), 145-153.

38. Bittencourt, L., Immich, R., Sakellariou, R., Fonseca, N., Madeira, E., Curado, M. and Rana, O. (2018). The Internet of Things, fog and cloud continuum: integration and challenges. *Internet of Things*. 3, 134-155.

39. Negash, B., Westerlund, T. and Tenhunen, H. (2019). Towards an interoperable Internet of Things through a web of virtual things at the Fog layer. *Future Generation Computer Systems*. 91, 96-107.

40. Elmroth, E., Leitner, P., Schulte, S. and Venugopal, S. (2017). Connecting fog and cloud computing. *IEEE Cloud Computing*. 4(2), 22-25.

41. Petersen, B., Bindner, H., You, S. and Poulsen, B. (2017). Smart grid serialization comparison: Comparision of serialization for distributed control in the context of the Internet of Things. *2017 Computing Conference*. 1339-1346.

42. Fielding, R. T., & Taylor, R. N. (2000). *Architectural styles and the design of network-based software architectures* (Vol. 7). Irvine: University of California, Irvine.

43. Richardson, L. and Ruby, S. (2008). RESTful web services. *O'Reilly Media, Inc*.

44. Fielding, R. T. and Taylor, R. N. (2002). Principled design of the modern Web architecture. *ACM Transactions on Internet Technology (TOIT)*. 2(2), 115-150.

45. Bonomi, F., Milito, R., Zhu, J. and Addepalli, S. (2012). Fog computing and its role in the internet of things. *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*. 13-16.

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı	: KANYILMAZ, Aykut
Uyruğu	: T.C.
Doğum tarihi ve yeri	: 01.11.1988, Sakarya
Medeni hali	: Evli
Telefon	: 0 (554) 786 76 54
e-mail	: aykutkanyilmaz@gmail.com

Eğitim

Derece	Eğitim Birimi	Mezuniyet Tarihi
Yüksek lisans	Gazi Üniversitesi / Bilgisayar Mühendisliği	Devam ediyor
Lisans	Gazi Üniversitesi / Bilgisayar Sist. Öğretmenliği	2011
Lise	Yunus Emre Çok Programlı Lisesi	2005

İş Deneyimi

Yıl	Yer	Görev
2018-Halen	İnnova Bilişim Çözümleri A.Ş.	Yönetici
2014-2018	Comodo Yazılım	Takım Lideri
2011-2014	İnnova Bilişim Çözümleri A.Ş.	Yazılım Geliştirici

Yabancı Dil

İngilizce

Yayınlar

Kanyılmaz A. and Çetin A. (2019). Fog Based Architecture Design for IoT with Private Nodes: A Smart Home Application. *2019 7th International Istanbul Smart Grids and Cities Congress and Fair*. 194-198.

GAZİ GELECEKTİR..