T.C.
SELÇUK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BLOK ZİNCİR TEKNOLOJİSİNİ KULLANARAK NESNELERİN İNTERNETİ AĞINDA VERİ GÜVENLİĞİNİ SAĞLAMA: AKILLI EV ÖRNEĞİ

Kadriye Nur ERMAN

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

Haziran-2023
KONYA

TEZ KABUL VE ONAYI

Kadriye Nur ERMAN tarafından hazırlanan "Blok Zincir Teknolojisini Kullanarak Nesnelerin İnterneti Ağında Veri Güvenliğini Sağlama: Akıllı Ev Örneği" adlı tez çalışması 04/07/2023 tarihinde aşağıdaki jüri tarafından oy birliği / oy çokluğu ile Selçuk Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Prof. Dr. Şakir TAŞDEMİR	……..
Danışman Dr. Öğr. Üyesi Ahmet Cevahir ÇINAR	……..
Üye Dr. Öğr. Üyesi Sedat KORKMAZ	……..

Yukarıdaki sonucu onaylarım.

Prof. Dr. Ömer Faruk YÜKSEL
FBE Müdürü

.

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Kadriye Nur ERMAN

Tarih:

ÖZET

YÜKSEK LİSANS TEZİ

BLOK ZİNCİR TEKNOLOJİSİNİ KULLANARAK NESNELERİN İNTERNETİ AĞINDA VERİ GÜVENLİĞİNİ SAĞLAMA: AKILLI EV ÖRNEĞİ

Kadriye Nur ERMAN

Selçuk Üniversitesi Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Ahmet Cevahir ÇINAR

2023, 85 Sayfa

Jüri
Dr. Öğr. Üyesi Ahmet Cevahir ÇINAR
Prof. Dr. Şakir TAŞDEMİR
Dr. Öğr. Üyesi Sedat KORKMAZ

Teknolojik olarak dünyanın hızla gelişmesi, gizlilik ve güvenlik gibi tehditler bakımından açıkların oluşmasına zemin hazırlamıştır. Veri akışının güvence altında olabilmesi için farklı yöntemler geliştirilmiştir, ancak bu yöntemlerin çoğu merkezidir ve hala kaynak kısıtlı olan nesnelerin interneti cihazları bakımından uygun olmayan işlem gücüne sahiptirler. Günümüzde nesnelerin interneti teknolojisinin gelişmesiyle birlikte akıllı ev sistemleri giderek daha popüler hale gelmektedir. Akıllı ev otomasyon sistemleri, insanların yaşamını kullanılabilirlik açısından elverişli hale getirmesi ve hayat standardını üst düzeylere getirmesinden dolayı günümüzde sıklıkla tercih edilmektedir. Akıllı evlerde kullanılan, nesnelerin interneti cihazlarının çeşitliliği de artmaktadır. Bu artan talep ve çeşitlilikle, işlem gücünde ve depolama miktarında da artış gerçekleşmiştir ve dolasıyla çok sayıda güvenlik endişesi ortaya çıkmıştır. Kullanıcı verilerinin şeffaflığını, güvenliğini ve mahremiyetini sağlamak için blok zincir teknolojisi ile heterojen kullanımı önerilmiştir. Blok zincir teknolojisinin, ağda dağıtık bir şekilde bulunan fiziksel bir veri tabanı olması sebebi ile nesnelerin interneti teknolojisine homojen bir ağ yapısı sağlar. Blok zincir teknolojisinin değişmezlik, şeffaflık, yetkilendirme, kimlik doğrulama ve verilerin dağıtık veri tabanı şeklinde kaydedilmesi özellikleri sayesinde, nesnelerin interneti cihazlarından elde edilip zincire eklenen veriler güvenli bir şekilde depolanır hale gelmektedir. Bu tez, blok zincir teknolojisi ve IoT'nin entegrasyonu hakkında kapsamlı bir araştırma sunmaktadır. Bu tez çalışmasında, blok zincir teknolojisi kullanılarak nesnelerin interneti ağında güvenliği, şeffaflığı ve mahremiyeti sağlamak amacıyla akıllı ev ağı üzerinde örnek bir uygulama gerçekleştirilmiştir. Bu uygulamada 1 adet ağ geçidi, 1 adet DHT11 sıcaklık ve nem sensörü, 1 adet MQ2 gaz sensörü ve 3 adet NodemCU ESP8266 kartı kullanılmıştır. Ağ geçidi olarak NodemCU ESP8266 kartı tercih edilmiştir. Ağ geçidi, sensörlerden alınan verileri blok zincire iletmek için kullanılmıştır. Blok zincirde depolama için lokal depolama ve bulut depolama kullanılmıştır.

Anahtar Kelimeler: Akıllı ev, blok zincir, depolama, gizlilik, güvenlik, nesnelerin interneti, şeffaflık

ABSTRACT

MS THESIS

ENSURING DATA SECURITY ON THE INTERNET OF THINGS BY USING BLOCKCHAIN TECHNOLOGY: THE SMART HOME EXAMPLE

Kadriye Nur ERMAN

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE OF SELÇUK UNIVERSITY
THE DEGREE OF MASTER OF SCIENCE
IN COMPUTER ENGINEERING

Advisor: Asst. Prof. Dr. Ahmet Cevahir ÇINAR

2023, 85 Pages

Jury
Asst. Prof. Dr. Ahmet Cevahir ÇINAR
Prof. Dr. Şakir TAŞDEMİR
Asst. Prof. Dr. Sedat KORKMAZ

The rapid development of the world in technology has paved the way for the emergence of vulnerabilities in terms of threats such as privacy and security. Different methods have been developed to secure the data flow, but most of these methods are centralized and still have unsuitable processing power for resource-constrained IoT devices. Today, with the development of IoT technology, smart home systems are becoming more and more popular. Smart home automation systems are frequently preferred today because they make people's lives convenient in terms of usability and bring the standard of living to higher levels. The variety of internet of things devices used in smart homes is also increasing. With this increased demand and variety, there has been an increase in processing power and amount of storage, thus raising a number of security concerns. Heterogeneous use with blockchain technology is proposed to ensure transparency, security and privacy of user data. Since the blockchain technology is a physical database distributed in the network, it provides a homogeneous network structure to the internet of things technology. Thanks to the immutability, transparency, authorization, authentication and recording of data in the form of a distributed database of blockchain technology, data obtained from IoT devices and added to the chain becomes securely stored. This thesis presents extensive research on the integration of blockchain technology and IoT. In this thesis, an exemplary application has been carried out on the smart home network in order to ensure security, transparency and privacy in the internet of things network using blockchain technology. In this application, 1 gateway, 1 DHT11 temperature and humidity sensor, 1 MQ2 gas sensor and 3 NodemCU ESP8266 cards are used. NodemCU ESP8266 card is preferred as gateway. The gateway was used to transmit the data received from the sensors to the blockchain. Local storage and cloud storage are used for storage in the blockchain.

Keywords: Blockchain, internet of things, privacy, security, smart home, storage, transparency

ÖNSÖZ

Blok zincir teknolojisi, oluşma sırasına bağlı olarak meydana gelmiş bloklar zinciridir. Günümüzde akıllı evlerde yaşanan güvenlik açıklıkları ve mahremiyet problemlerine çözüm olarak, ayrıca akıllı evlerde kullanılan cihaz ve sensörlerin yetersiz depolama kabiliyetlerinden kaynaklı yaşanan problemlere çözüm olarak blok zincir teknolojisi tercih edilmektedir. Bu tez çalışmasında bunun deneysel bir alternatifi detaylı olarak gerçekleştirilmiş ve anlatılmıştır.

Bilgi ve tecrübeleri ile sabır ve istekle beni yönlendiren, desteğini esirgemeden akademik hayatımda gelişmeme katkıda bulunan değerli danışman hocam Dr. Öğr. Üyesi Sayın Ahmet Cevahir ÇINAR hocama, çalışmalarım boyunca engin bilgi ve tecrübeleri ile lisans ve yüksek lisans dönemlerimde her koşulda yanımda olup beni aydınlatan değerli hocam Prof. Dr. Sayın Şakir TAŞDEMİR hocama, tezimi okuyarak değerli görüşleriyle tezin içeriğinin daha kaliteli olmasına katkı sunan kıymetli jüri üyem Dr. Öğr. Üyesi Sayın Sedat KORKMAZ hocama, ayrıca hayatım boyunca maddi ve manevi her durumda yanımda olan aileme en derin duygularla teşekkür ederim.

Kadriye Nur ERMAN
KONYA-2023

İÇİNDEKİLER

ÖZET ..... **iv**

ABSTRACT ..... **v**

ÖNSÖZ ..... **vi**

İÇİNDEKİLER ..... **vii**

ŞEKİLLER LİSTESİ ..... **ix**

ÇİZELGELER LİSTESİ ..... **x**

SİMGELER VE KISALTMALAR ..... **xi**

1. GİRİŞ **1**

2. LİTERATÜR TARAMASI **5**

3. BLOK ZİNCİR TEKNOLOJİSİ **8**

3.1. Blok Zincir Yapısı 8
3.2. Blok Zincir Çözümlerinin Özellikleri 12
3.3. Blok Zincir Jenerasyonları 13
3.4. Blok Zincir Türleri 14
3.5. Özet Fonksiyonları 16
3.6. Akıllı Sözlemeler 19
3.7. Blok Zincir Katmanları 20
3.8. Şifreleme Yöntemleri 21
3.9. Dijital İmzalar 23
3.10. Merkeziyetsiz Mutabakat Algoritmaları 24
3.10.1. Proof-of-Work (POW) 25
3.10.2. Proof-of-Stake (POS) 25
3.10.3. Pratik Bizans Hata Toleransı (PBFT) 26

4. IoT MİMARİSİ **28**

4.1. IoT Merkezileştirilmiş Modelin Zorlukları ve Sorunları 28

5. BLOK ZİNCİR VE IoT ENTEGRASYONU **30**

5.1. Blok Zincir Mimarisi ile IoT 33
5.2. Blok Zincir ve IoT Entegrasyon Şemaları 35

6. BLOK ZİNCİR VE AKILLI EV UYGULAMA ÇERÇEVESİNİN ARKA PLANI **39**

6.1. Akıllı Ev Nedir? 40
6.2. Önerilen Ağ Yapılandırması 41

6.3. Verileri Ağ Geçidi İçinde Ön İşleme 42
6.4. Ağ Geçidi Cihazlarını Tanıma ve Veri Toplama 43
6.5. Akıllı Ev Ağ Geçidi Güvenlik Hususları 44
6.6. Bulut Bilişime Genel Bakış 45
6.6.1. Bulut bilişim ve blok zincir entegrasyonu 46
6.7. Güvenlik Değerlendirmesi 47

7. AKILLI EV PROTOTİPİ 50

7.1. Akıllı Ev Prototipinde Kullanılan Sensörler, Donanım Kartları ve Özellikleri 50
7.2. Donanım Şeması 52
7.3. MQ-2 Gaz Sensörü Çalışma Mantığı ve Kodları 53
7.4. DHT11 Sıcaklık ve Nem Sensörü Çalışma Mantığı ve Kodları 54
7.5. NodeMCU ESP8266 Ağ Geçidi Çalışma Mantığı ve Kodları 55
7.6. Blok Zincir Kodları 57

8. SONUÇLAR VE ÖNERİLER 61

8.1. Sonuçlar 61
8.2. Öneriler 63

KAYNAKLAR 65

ÖZGEÇMİŞ 73

ŞEKİLLER LİSTESİ

Şekil 3.1. Blok zincir ağ yapısı 8
Şekil 3.2. Tipik bir blok zincir sisteminin bileşenleri 10
Şekil 3.3. Blok zincir yapısı 11
Şekil 3.4. Özet fonksiyonlarının çalışma mantığı 17
Şekil 3.5. Akıllı sözleşme akış diyagramı 20
Şekil 3.6. Simetrik şifreleme 22
Şekil 3.7. Asimetrik şifreleme 22
Şekil 3.8. Dijital İmza Algoritması (Digital Signature Algorithm - DSA) 24
Şekil 4.1. 3 katmanlı IoT mimarisi 28
Şekil 5.1. IoT'ye genel bakış 30
Şekil 5.2. IoT ve blok zincir platformlarının bir arada kullanıldığı kavramsal şema örneği 33
Şekil 5.3. IoT'de blok zincir kullanımında katmanlı sistem 34
Şekil 5.4. Blok zincir IoT ağ geçidi entegrasyonu 35
Şekil 5.5. Blok zincirin uç noktaları olarak ağ geçidi cihazları şeması 36
Şekil 5.6. Blok zincirine işlem verenler olarak IoT uç cihazları şeması 37
Şekil 5.7. Blok zincirin uç noktaları olarak birbirine bağlı IoT cihazları şeması 37
Şekil 5.8. IoT uç cihazları ile bulut-blok zinciri hibriti şeması 38
Şekil 6.1. Akıllı ev sisteminin geleneksel yapısının tasviri 40
Şekil 6.2. Önerilen ağın tasarımına genel bakış 41
Şekil 6.3. Akıllı ev ağ geçidi için blok zincir tabanlı bulut özellikli mimarinin metodolojik akış şeması 42
Şekil 6.4. Ağ geçidi verileri ön işleme 43
Şekil 6.5. Ağ geçidi cihazlarını tanıma ve veri toplama şeması 44
Şekil 6.6. Blok zincirin bulut bilişime sağladığı faydalar 47
Şekil 7.1. DHT11 Sıcaklık ve Nem Sensör Modülü 50
Şekil 7.2. MQ-2 Gaz Sensör Modülü 51
Şekil 7.3. NodeMCU ESP8266 Wifi Modülü 51
Şekil 7.4. NodeMCU ESP8266 ve DHT11 sensör bağlantısı 52
Şekil 7.5. NodeMCU ESP8266 ve MQ-2 sensör bağlantısı 52
Şekil 7.6. NodeMCU ESP8266 kartı ile DHT11 ve MQ-2 sensör bağlantısı ve ağ geçidi olarak NodeMCU ESP8266 kartı 53
Şekil 7.7. MQ-2 gaz sensörü ve bağlı olduğu NodeMCU ESP8266 kartı kodları 54
Şekil 7.8. DHT11 sıcaklık ve nem sensörü ve bağlı olduğu NodeMCU ESP8266 kartı kodları 55
Şekil 7.9. NodeMCU ESP8266 ağ geçidi kodları 56
Şekil 7.10. "Transaction.cs" sınıfı kodları 57
Şekil 7.11. "Block.cs" sınıfı kodları 57
Şekil 7.12. "Blockchain.cs" sınıfı kodları 58
Şekil 7.13. "Program.cs" sınıfı kodları 59

ÇİZELGELER LİSTESİ

Çizelge 3.1. Özel, hibrit ve genel blok zincirlerinin karşılaştırılması 16
Çizelge 3.2. Özet fonksiyonları karşılaştırması 19
Çizelge 3.3. Blok zincir katmanları ve içerikleri 20
Çizelge 3.4. Mutabakat algoritmalarından bazılarının kriter karşılaştırması..... 27

SİMGELER VE KISALTMALAR

Simgeler

Kısaltmalar

AES:	Gelişmiş Şifreleme Standardı (Advanced Encryption Standard)
BFT:	Bizans Hata Toleransı (Byzantine Fault Tolerance)
DES:	Veri Şifreleme Standardı (Data Encryption Standard)
DLT:	Dağıtılmış Defter Teknolojisi (Distributed Ledger Technology)
DDOS:	Dağıtılmış Hizmet Reddi Saldırısı (Distributed Denial of Service Attack)
DOS:	Hizmet Reddi (Denial of Service)
DSA:	Dijital İmza Algoritması (Digital Signing Algorithm)
D2D:	Cihazdan Cihaza (Device-to-Device)
ECDSA:	Eliptik Eğri Dijital İmza Algoritması (Elliptic Curve Digital Signature Algorithm)
ETH:	Ether
EVM:	Ethereum Sanal Makinesi (Ethereum Virtual Machine)
IBPAS:	Kimlik Tabanlı Vekil Toplu İmza (İdentity Based Proxy Aggregate Signature)
IoT:	Nesnelerin İnterneti (Internet of Things)
MAC:	Medya Erişim Kontrol Adresi (Media Access Control Address)
MD5:	Mesaj Özeti 5 (Message Digest 5)
NIST:	Ulusal Standartlar ve Teknoloji Enstitüsü (National Institute of Standards and Technology)
PBFT:	Pratik Bizans Hata Toleransı (Practical Byzantine Fault Tolerance)
PoW:	İş Kanıtı (Proof of Work)
PoS:	Alan Kanıtı (Proof-of-Stake)
P2P:	Eşler Arası (Peer to Peer)
RC5:	Rivest Şifresi 5 (Rivest Cipher 5)
RFID:	Radyo Frekansıyla Tanımlama (Radio Frequency Identification)
RIPEMD:	Çeşit Bütünlüğü İlkel Değerlendirme Mesaj Özeti (Race Integrity Primitive Evaluation Message Digest)

RSA:	Rivest, Shamir, Adelman
RTS-DELM:	Kaynaşmış Gerçek Zamanlı Sıralı Derin Ekstrem Öğrenme Makinesi (Fused Real-Time Sequential Deep Extreme Learning Machine)
SHA:	Güvenli Özet Algoritması (Secure Hash Algorithm)
TEM:	Geçişken Enerji Yönetimi (Transactive Energy Management)

1. GİRİŞ

Nesnelerin İnterneti (IoT) teknolojisi insan yaşamını doğrudan etkileyen, gelişime açık ve yenilikçi bir teknolojidir. Bu teknoloji sayesinde insan gücü kullanımı azalmaya ve dolayısıyla insan yaşamı kolaylaşmaya, verimli hale gelmeye ve dijitalleşmeye hızla devam ediyor.

Nesnelerin internet üzerinden iletişim halinde olup, insan yardımı olmadan gerekli işlevleri yerine getirmesi fikri, 1999'da Kevin Ashton'un yaptığı bir sunumda "Nesnelerin İnterneti (IoT)" kavramını kullanılması ile ortaya çıktı (Ashton, 2009). O zamanlarda internetin kullanım amaçlarından farklı olarak, cihazların birbirleri ile etkileşimini internet üzerinden sağladı.

Statista'dan elde edilen verilere göre (Statista, 2022) 2021 yılında dünya çapında 11.3 milyar IoT cihazı kullanılmıştır. 2022 yılında bu sayı 13.1 milyar, 2030 yılına kadar ise 29.4 milyar olarak beklenmektedir. Akıllı evler, akıllı arabalar, akıllı fabrikalar, akıllı şebekeler ve akıllı şehirler gibi hayatın her alanında kullanılan Nesnelerin İnterneti (IoT), insan müdahalesi olmadan farklı sensörlerin ve objelerin internet aracılığı ile bir ağ üzerinden iletişim kurmasıdır. IoT teknolojisinin süreçleri otomatikleştirmesi, zaman ve güçten verimliliği artırıp maliyeti düşürmesi, iş süreçlerinin takibini kolaylaştırması gibi sayısız faydası olmasına rağmen küresel pazar talebindeki artışla beraber kullanımdaki ve üretimdeki yaygınlık bazı sorunları da beraberinde getirdi (Xiang, Jianlin ve ark., 2012). Bunlardan bazılarına örnek olarak; büyük miktarda veri oluşması, yüksek enerji tüketimi, merkezi bir otorite tarafından yönetildiği için sistemin manipüle edilebilmesi veya tamamen çalışamaz hale getirilmesi, tek bir merkezde verileri depolama özelliğinden dolayı merkezde oluşacak herhangi bir hatanın tüm sistemi etkilemesi sorunları verilebilir (Borgohain, Kumar ve ark., 2015). Tüm bu sorunlar da ölçeklenebilirliğe ve IoT'nin geniş çapta benimsenmesine engel olmaktadır. Bunları takiben değiş tokuş edilen/toplanan verilerin güvenliği, güvenilirliği, mahremiyeti ve doğrulanması sorunları da beraberinde gelir. Bu derece kullanımı yaygın ve gelişmiş sistemlerde, kontrolü kaybetmemek ve olası sorunlar meydana geldiği durumlarda kurtarabilmek adına, merkezi sunucu kullanmaktansa dağıtık sistemler tercih edilmektedir (Huckle, Bhattacharya ve ark., 2016). Örneğin, merkezi sunucuya yapılan hizmet reddi saldırısı (DOS - Denial of Service) veya her düğüme yapılabilecek olan dağıtılmış hizmet reddi saldırısı (DDOS The Distributed Denial of Service) sonucunda sistem iş yapamaz hale gelebilir

(Kajwadkar ve Jain, 2018). IoT teknolojisinde bu potansiyel zorlukları ve sorunları çözmek için tek hata noktası olmayan blok zincir teknolojisi ile entegre kullanımı önerilmiştir (Huckle, Bhattacharya ve ark., 2016).

Akıllı ev otomasyonları konut sahiplerine sağladığı avantajlar ve kolaylıklar sayesinde mevcut IoT uygulamaları arasında giderek daha popüler hale gelmektedir. Akıllı ev teknolojisinin yakın gelecekte birçok evde bulunması bekleniyor (Stojkoska ve Trivodaliev, 2017). Akıllı evler, insanlara güvenlik, konfor ve artan yaşam kalitesi sağlayan bir ağa bağlı çeşitli sensörler, cihazlar, monitörler ve arayüzlerle otomatik ve akıllı hizmetler sağlayan konutlardır (Fisk, 2001; Kim, Park ve ark., 2017; Wilson, Hargreaves ve ark., 2017). Akıllı evler; akıllı telefonlar ve akıllı sayaçlar gibi çeşitli akıllı cihazları birbirine bağlandığı bir ağdır ve IoT ise bu ağın temel platformudur. Akıllı evler gerçek zamanlı olarak verileri gönderir ve alır. Kullanıcılar, evin ağ yapılandırmasına ve kullanıcı ayarlarına bağlı olarak çeşitli işlevleri izlemek ve kontrol edebilmek için birden fazla akıllı ev ürününün kullanımını gerçekleştirirler. Akıllı evin ağ yapısı, internet üzerinden haberleşen cihazlardan ve gömülü bilgisayarlardan oluşur. Bu ağ yapısı kabloludan kablosuza kaymaktadır (Wang, Gao ve ark., 2019). Sensörler, ev sahipleri tarafından cihazlara ulaşabilmek ve çalıştırabilmek için kablosuz bağlantı yoluyla uzaktan erişimde kullandıkları ev aletlerinde kullanılmaktadır. Akıllı ev pazarına talep gittikçe yükselmektedir. Google, Samsung ve Amazon gibi önde gelen Bilişim Teknolojileri (BT) şirketleri, bu artan talebe karşılık yeni oluşan rekabetçi pazara girerek, akıllı ev hizmetlerini ve ürünlerini genişletmek için hızlı hareket etmişlerdir (Pal, Funilkul ve ark., 2018; Sanguinetti, Karlin ve ark., 2018; Shin, Park ve ark., 2018). Küresel akıllı ev aletlerinin penetrasyon oranının 2020'de %10,62'den 2025'te %21,09'a çıkacağı tahmin edilmektedir (Statista, 2022). Akıllı evlerde kullanılan birçok IoT cihazı büyük miktarda veri üretmektedir. Örneğin video bilgilerinin kamerada depolanması için büyük miktarda depolama alanı gerekmektedir. Günümüzde geleneksel bilgi sistemleri, üretilen bu akıllı ev verilerini yönetme ve sürdürmede farklı sorunlar ile karşılaşmaktadır. Bulut depolama, veri depolama konusunda büyük bir rol oynamıştır. Depolama maliyetini azaltmış ve depolama kapasitesi konusuna çözüm olmuştur. Bulut teknolojisi, akıllı evler alanında yaygın olarak tercih edilmektedir (Han ve Lim, 2010; Zhao, Liu ve ark., 2018). Tüm bu kolaylıklara rağmen ve akıllı evler sahiplerine birçok fayda sağlarken, kötü niyetli siber saldırılar için riskler oluşturmaktadır (Alam, St-Hilaire ve ark., 2019). Saldırganlar akıllı evlere yetkisiz olarak erişim sağlayabilirler, kullanıcı verilerine ulaşabilir ve kullanıcı verilerini

değiştirebilirler. Örneğin sıcaklık verisine erişip kullanıcının can ve mal güvenliğini tehlikeye atabilirler. Bu durum mahremiyet ve güvenlik konusunda risk teşkil etmektedir. Bu risklere çözüm olarak sunulan geleneksel yöntemler, merkezileştirilmiştir ve kötü amaçlı saldırılara karşı savunmasızdırlar. Blok zincir teknolojisinin veri depolama alanında şeffaf ve güvenilir bir teknoloji olarak yükselişi akıllı evlerdeki ciddi sorunlardan olan veri gizliliğini, güvenliğini ve bütünlüğünü çözmeye yönelik yeni potansiyellerin önünü açmaktadır. Son zamanlarda blok zincir teknolojisi çoğu yeni nesil IoT uygulamalarında güvenliği sağlamak için tercih edilmektedir (Sharma, Rathore ve ark., 2018).

Blok zincir, üçüncü bir taraf olmadan verilerin dağıtık ve merkeziyetsiz bir ağda güvenli ve değiştirilemez şekilde tutulduğu ve yönetildiği bir dağıtık defter teknolojisidir. Mahremiyeti koruması, dağıtık olması, güvenilirlik sağlaması ve geniş çaplı ölçekte çalışabilmeye uygun yapıya sahip olması özellikleriyle IoT için bir tamamlayıcı olmuştur. Blok zincir teknolojisinin çalışma mantığındaki fikrin ilk adımı 1991 yılına kadar uzanmaktadır. 1991 yılında kriptograf ve bilgisayar bilimci Stuart Haber ve bilimsel araştırmacı ve fizikçi Wakefield Scott Stornetta dijital belgelerin değiştirilememesi amacıyla zaman damgası ve kriptografik yöntemleri kullanan bir çözüm sundular (Haber ve Stornetta, 1991). Ardından anonim bir kişi veya grup olarak bilinen Satoshi Nakamoto, 2008'de merkeziyetsiz, eşler arası, elektronik nakit sistemi olan Bitcoin'i tanıtan bir yazı yayınladı (Nakamoto, 2008). Blok zincir tabanlı çalışan bu sistemi tanıtan yazının ardından 2015 yılında "blok" ve "zincir" kelimelerinin bir arada kullanılmasıyla blok zincir teknolojisi popüler hale gelmeye başladı (Ali, Vecchio ve ark., 2018). Sağlık (Belotti, Božić ve ark., 2019), endüstri (Guo ve Liang, 2016), lojistik (He, Wang ve ark., 2022), finans (Nakamoto, 2008), tarım (Hang, Ullah ve ark., 2020) gibi alanlarda güvenlik konusunda tercih edilen teknoloji haline geldi. Yoğun bir şekilde kriptografik fonksiyonlar kullanan ve dijital bir veri tabanı olan blok zincir teknolojisi sayesinde, normalde uzun süreli ve insan yardımından faydalanılarak çözülen süreçler daha hızlı, daha şeffaf ve dijital hale geldi. Dağıtık ağda karşılıklı güven gerekmeksizin fikir birliği algoritmaları sayesinde işlemler yürürken, yerleşik işlem-yönetim sistemlerinin aksine merkezi otoriteye gerek duyulmaz. Merkezi sunucu veya otorite olmadığı için yüksek özellikli sunucular gibi donanım altyapısına ve bakıma gerek duyulmaz. Böylelikle hem performansta artış olur hem de maliyetle ilgili sorunlar minimum seviye iner. Ayrıca ağda bulunan her katılımcı blok zincirin bir kopyasını tuttuğundan zincirde olan veriler değiştirilemez şekilde korunur. Eklenen her

blok ise asimetrik şifreleme algoritmaları, dijital imza ve özet fonksiyonları gibi belirli kriptografik işlemlerden geçtikten sonra eğer doğru ise blok zincire eklendiği için veri bütünlüğü ve veri doğruluğu garantilenmiş olur. Blok zincire veriler eklenirken zaman damgası ile eklendiği için, bütün katılımcılar tarafından izlenilebilir.

Bu tez çalışmasının ana katkıları şu şekildedir; akıllı ev mimarisi ve blok zincir teknolojisinin uygulama metodolojisine yer vermektedir ve özel blok zincir kullanarak bir akıllı ev mimari tasarımı gerçekleştirip ve uygulamaktadır. Uygulamada yaygın olarak bulunan IoT sensörlerini kullanarak basit, güvenli ve izlenebilir bir akıllı ev mimarisi için deneysel donanım test tasarımı uygulamaktadır. Bir akıllı ev ağı içinde bulunan cihazlar, yalnızca ağ geçidinde onaylanıp kaydedilen cihazlardır.

2. LİTERATÜR TARAMASI

Günümüzde akıllı evlerin kullanımı, insan yaşamını kolaylaştırmasından ötürü yaygınlaşmıştır. Bu yaygınlaşmayla beraber depolanacak veri miktarında artış gözlenmiştir. Ayrıca akıllı ev ağlarının yapısı gereği ve akıllı ev ağlarında kullanılan cihazlar gereği, akıllı ev ağları farklı saldırılara açık durumdadır. Bu gibi sorunlardan ötürü blok zincirin eşler arası merkeziyetsiz yapıda depolama gerçekleştirmesi ve kimlik doğrulama mekanizması gibi özelliklerinden akıllı ev ağlarıyla heterojen kullanımı tercih edilmektedir.

Zhu, Loke ve ark. (2019), akıllı evde IoT cihazları için, kontrol erişimi konusunda iki temel zorluk olan bütünlük ve gizlilik üzerine bir incelemeyi Dağıtık Defter Teknolojisi (DLT – Distributed Ledger Technology) ile gerçekleştirdiler. Ek olarak DLT kullanan akıllı evlerde, kullanıcıların, insan konforunun yanı sıra kaynakların optimize edilmesini sağlamak için sıcaklık, nem, aydınlatma ve hava kalitesi gibi çeşitli verileri izleyebilecekleri şekilde tasarladılar.

Lin, He ve ark. (2019), akıllı bir evde kullanıcılar ve ev ağ geçidi arasında karşılıklı kimlik doğrulama sağlamak için blok zincirin diğer tekniklerle nasıl kullanılabileceğini göstermişlerdir. Sistem ayrıca, daha sonra uygunsuz davrandığı tespit edilen herhangi bir kullanıcının verimli bir şekilde izlenmesine olanak tanır. Akıllı evlerde ve diğer uygulamalarda uygulanabilecek yeni bir güvenli karşılıklı kimlik doğrulama sistemi oluşturmuşlardır.

Lee, Rathore ve ark. (2020) mevcut akıllı ev ağ geçidi ortamı ve IoT için blok zinciri tabanlı veri kurcalamaya karşı korumalı bir ağ geçidi mimarisi önermişlerdir. Akıllı ev ağ geçidi ve heterojen IoT'de ortaya çıkan gizlilik ve kimlik doğrulama sorunlarını çözmek için SHA2 şifreleme algoritmasını kullanmışlardır. Ağ geçidinde depolanan verilerin bütünlüğünü korumak için blok zincir teknolojisini kullanmışlardır.

Yang ve Wang (2021), akıllı evlerdeki enerji yönetimi üzerine bir çalışma yaptılar. Transaktif Enerji Yönetimi (TEM - Transactive Energy Management) sistemi ve blok zincir teknolojisini bir arada kullanarak, TEM teknolojisinin dezavantajlarını en aza indirdiler. Bu çalışma sonucunda IoT tabanlı akıllı evler üzerinde yaptıkları sistemin, önceki sistemin toplam maliyetini %25 oranında düşürdüğünü gördüler.

Ren, Leng ve ark. (2021), akıllı bir evin veri depolaması, sakinlerinin güvenlik gereksinimlerini karşılayamadığı için, güvenliği güçlendirmek adına birden fazla bulut

sağlayıcısı altındaki blok zincirlere dayalı bir çalışma yaptılar. Ayrıca imza doğrulama verimliliğini artırmanın yanı sıra depolama alanını sıkıştırmak ve iletişim bant genişliğini azaltmak için Kimlik Tabanlı Proxy Toplu İmza (IBPAS - Identity Based Proxy Aggregate Signature) şeması önermişlerdir. Yaptıkları deneye göre, IBPAS şemasının iletişim maliyeti, sıradan bir imza şemasının sadece %12 ila %39'unu oluşturmuştur.

Qashlan, Nanda ve ark. (2021) akıllı ev sistemlerinde IoT cihazları için güvenli bir çerçeve oluşturmak adına öznitelik tabanlı erişim kontrolünü akıllı sözleşmeler ve uç bilgi işlemle birleştiren kimlik doğrulama şeması önermişlerdir. Önerdikleri çerçeve, istenen güvenlik ve gizlilik hedeflerine ulaşır ve değişiklik, DoS saldırıları, veri madenciliği ve bağlantı saldırılarına karşı dirençlidir.

Tchagna Kouanou, Tchito Tchapga ve ark. (2022), IoT mimarisinde verileri güvence altına almak için bir blok zincir yöntemi önermişlerdir. Akıllı ev için bir IoT ekosistemi oluşturmak üzere Arduino, Raspberry Pi ve sensörleri birleştirip, ölçeklenebilirlik ve esneklik özelliklerinden dolayı EOS blok zinciri kullanmışlardır. Elde ettikleri sonuçlarda, önerdikleri blok zinciri yönteminin, blok zincirinde milisaniye başına depolanan veri sayısı ve ölçeklenebilirlik açısından ilgili çalışmalardan daha iyi performans gösterdiğini görmüşlerdir.

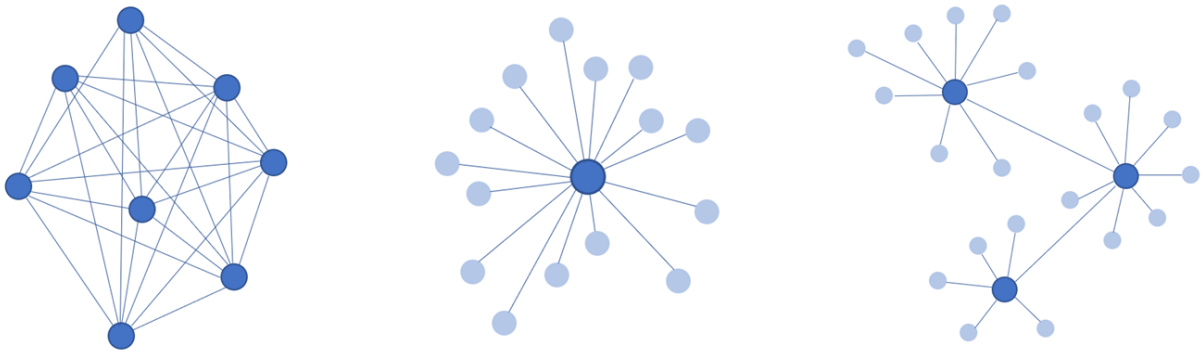
Farooq, Khan ve ark. (2022) “Kaynaşmış Gerçek Zamanlı Sıralı Derin Ekstrem Öğrenme Makinesi” (Fused Real-Time Sequential Deep Extreme Learning Machine - RTS-DELM) sistem modeli ile güçlendirilmiş izinsiz giriş tespitini tahmin etmek için özel bir blok zincir tabanlı akıllı ev ağı mimarisi önermişlerdir. Herhangi bir kötü amaçlı etkinliği tespit etmek için blok zinciri tabanlı akıllı evlerde uygulanan RTS-DELM metodolojisini kullanmışlardır. Fused RTS-DELM tekniği, akıllı ev ağlarındaki herhangi bir izinsiz giriş etkinliği için düşük hata oranıyla çok önemli düzeyde kararlılık sağlar. RTS-DELM çözümü, %95,28 doğruluk oranıyla olağanüstü yüksek bir başarı oranı elde etmiştir.

Yakubu, Khan ve ark. (2023) akıllı evlerde kullanılan cihazların dışarıdan gelecek saldırılara karşı savunmasız olduğu fikrinden yola çıkarak cihazdan cihaza (D2D - Device-to-Device) etkileşimlerde kimlik doğrulama sistemi önermişlerdir. Ethereum blok zincir ağından ve akıllı sözleşmelerden faydalanmışlardır. DDoS saldırısını engellemek için yaptıkları bu çalışmanın tehditlere karşı dayanıklı olduğunu gösterdiler.

Liu, Wu ve ark. (2023) akıllı evlerde blok zincir tabanlı bir mimari önermişlerdir. Önerilen mimari, sistemin tüm üyeler arasında ölçeklenmesine izin veren parça tabanlı bir ağ oluşturur. Blok zincir teknolojisini akıllı evlerde kullanırken verim ve gecikmede bazı sorunlar oluşmaktadır. Bu sorunların üstesinden gelebilmek adına, bu çalışmada heterojen cihazlardan oluşan akıllı ev sistemlerine uygun yeni bir ölçeklenebilir blok zincir mimarisi olan "Communitychain" önerilmiştir. Yeni madenci düğümleri ve lider düğüm seçim algoritmalarını içeren, mimarinin düğüm sayısındaki artışa uyum sağlamasını sağlayan parçalamaya dayalı bir topluluk modeli tasarlamışlardır.

3. BLOK ZİNCİR TEKNOLOJİSİ

Blok zincir; gerçekleşen tüm işlemlerin güvenli, kronolojik ve değişmez bir şekilde kalıcı bir kaydını tutan sürekli büyüyen bir defter, fiziksel ve dijital bir veri tabanı, kısmen ilişkili veriler zinciri olarak tanımlanabilir (Nofer, Gomber ve ark., 2017). Dağıtılmış bir veri yapısıdır ve dağıtılmış defter teknolojisi olarak da bilinir (Tschorsch ve Scheuermann, 2016). Dağıtık bir ağ yapısının, merkezi ve çok merkezli ağ yapılarına göre farkı, Şekil 3.1.'de verilmiştir (Baran, 1964).



(a) Dağıtık Ağ (b) Merkezi (c) Çok Merkezli

Şekil 3.1. Blok zincir ağ yapısı

Her bir işlem blok adı verilen yapıda tutulur ve bloklar birbirlerine bağlanarak blok zinciri oluşturur. Blok zincirin yapısı gereği, eklenen işlemler daha sonradan değiştirilemez, eğer değiştirilmek istenirse tüm zincir etkilenecektir. İşlemler kronolojik olarak blok zincire eklendiği için zamana göre sıralıdırlar. Bundan dolayı banka ve hükümet gibi yüksek güvenlik isteyen uygulamalarda üçüncü tarafa ihtiyaç duyulmadan mülk, sözleşme gibi ögelerin transferi gerçekleşir.

Blok zincir teknolojisinin üçüncü bir taraf olmadan birçok katılımcının bulunduğu dağıtık ağ yapısında, merkeziyetsizliği güvenli bir şekilde nasıl sağladığını öğrenmek önemlidir. Bu bölümde blok zincirin ana özellikleri, yapısı ve çalışma mantığı anlatılmıştır.

3.1. Blok Zincir Yapısı

Veri Ambarı (Datastore): Blok zincirde bulunan bilgileri ve bilgilerin tutulmasında kullanılan veri yapısını ifade eder. Her bir blok, kendinden bir önceki bloğun özet değerini tutar ve bloklar bu şekilde birleşir. Bloklar ileriye doğru birbirleri ile bağlıdırlar

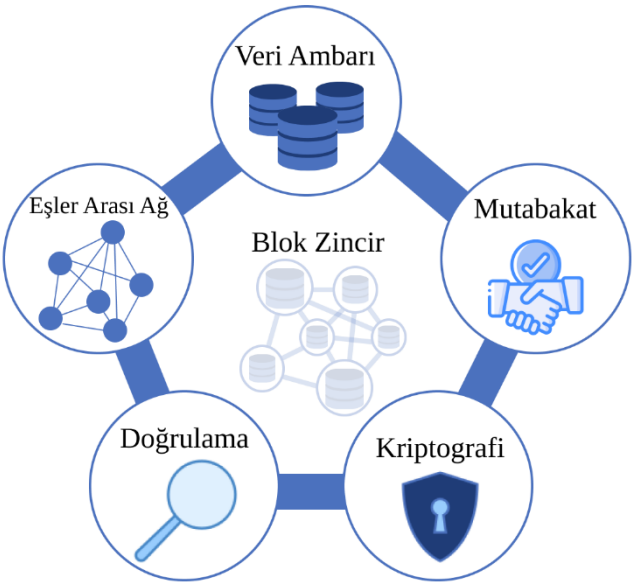
ve bu nedenle herhangi bir blokta yapılacak değişiklik ile bloğun özet değeri değişecektir ve zincir kopacaktır. Blok zincirde bulunan verilere sıralı erişim söz konusudur ve bu yüzden veriler bloklarda tutulurken Merkle ağaç yapısı (Szydlo, 2004) kullanılır. Merkle ağaç yapısı tipik olarak ikili ağaç yapısındadır. Her bir işlemin özet değeri bulunur. Daha sonra bu özet değerleri ikili grup olarak özeti oluşturulur ve yeni bir özet değeri bulunur. Bu şekilde devam ederek kök özet değeri bulunur. Şekil 3.3.'te blokta bulunan örnek bir Merkle ağacı görülebilir. Herhangi bir işlemde oluşacak değişiklik özet değerlerinde değişikliğe sebep olacaktır ve böylece kolay tespit edilebilirlik sağlanır. Bloklarda işlemler tutulurken Merkle ağaç yapısına ek olarak grafik tabanlı yaklaşımlar da önerilmiştir (Sompolinsky, Lewenberg ve ark., 2016; Popov, Saa ve ark., 2019).

Mutabakat (Consensus): Blok zincir, karşılıklı güvene gerek duyulmayan bir yapıdadır. Blok zincire veriler eklenirken düğümler tarafından tek bir veri değeri üzerinde dinamik bir şekilde anlaşmaya varılır. Düğümler tarafından anlaşmaya varılırken veri bütünlüğünü ve sistemin tutarlı olmasını sağlama amacıyla belirli mutabakat algoritmalarından faydalanılır. Asenkron sistemlerde stokastik süreçler söz konusu olduğu zaman mutabakat algoritmaları için karar vermek zor bir iştir (Fischer, Lynch ve ark., 1985). Bu nedenle mutabakat algoritmaları bazı basitleştirici işlemleri ön plana alır. Mutabakat algoritmaları günümüzde gelişmeye ve yenilenmeye devam etmektedir.

Doğrulama (Validation): Blok zincirde doğrulama, tüm düğümlere dağıtılır. Tüm bilgileri, ağdaki herkes okuyabilir. Doğrulama sürecinde, blok zincire çifte harcama gibi yanlış bir veri aktarılması engellenir. Bitcoin'de doğrulama işlemini madenciler üstlenirken, Ethereum ağında bu işlemler akıllı sözleşmeler ile sağlanır. Madenciler; işlem içeriğini doğrulamak, gönderici ve alıcıyı kontrol etmekle görevlidirler. Akıllı sözleşmeler ise, işlem içeriğini doğrulayıp gönderici ve alıcı arasındaki anlaşmayı sağlayan bir dizi kod parçasıdır.

Eşler Arası Ağ (Network): Blok zincir, dağıtılmış bir ortamda, üçüncü bir taraf olmadan eşler arası ağda geçerli işlem verilerinden oluşan bloklar kümesi olarak tanımlanabilir. Bu sistemde, herhangi bir düğüm bir işlem başlatabilir ve oy çokluğu ile işlem, zincire eklenir. Bu şekilde işleyen bir sistemde bilgi yayılımının kararlılığı önemlidir. Eşler arasında veriler iletilirken, karşılıklı kimlik doğrulaması asimetrik şifreleme mekanizması ile sağlanır. Ağda başlatılan işlemler ise madenciler tarafından doğrulanır.

Kriptografi (Cryptography): Blok zincirde verilerin kriptografik biçimde tutulması gizlilik ve güvenlikte önemli rol oynar. Blok zincir işlemler için asimetrik şifreleme mekanizması kullanır ve bunun önemli örneği dijital imzalardır. Asimetrik şifreleme genel ve özel anahtar çiftlerine dayanan bir şifreleme yöntemidir. Genel anahtar, herkese açık bir anahtardır. Özel anahtar ise sadece sahibi tarafından bilinen anahtardır. Bir işlemi gerçekleştirdiğinizde, özel anahtarınız ile işlemi imzalamış olursunuz. Dijital imzaların sağladığı özellikler; inkâr edilemezlik, taklit edilemezlik ve kolay doğrulamadır.

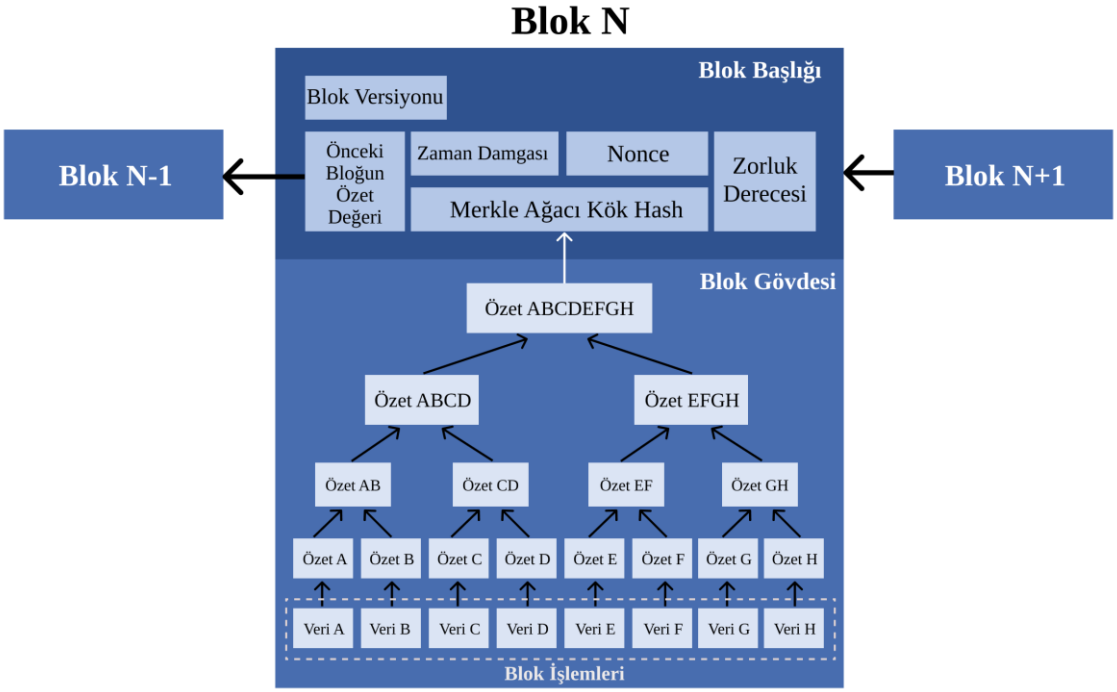


Şekil 3.2. Tipik bir blok zincir sisteminin bileşenleri

Blok zincir teknolojisi, birbirine özet değerleri ile bağlı bloklardan oluşur. Blok zincirde bulunan bloklar, ağda olan katılımcıların yaptıkları bir veya daha fazla işlemin ayrıntılı bilgisini içerir. Blok tamamladığında blok zincire kalıcı olarak girer. Blok yapısı genel olarak Şekil 3.2.'deki gibi olsa da ağın tasarımına ve kullanıcı isteğine göre şekillendirilebilir.

Her blok, blok başlığı ve blok gövdesi olmak üzere mantıksal olarak 2 yapıdan oluşur. Blok gövdesinde, yapılan işlemlerin Merkle root değerini elde edebilmek için yapılmış aşamalar yer alır. Blok başlığında ise, bu bloktan önceki bloğa ait özet değeri, blok versiyonu, zaman damgası, nonce değeri, zorluk derecesi ve işleme ait Merkle root değeri vardır. Blok oluşturulurken otomatik olarak kendi oluşma zamanı yani bir diğer deyişle zaman damgası eklenir. Bu sayede blok, zincire eklendikten sonra üzerinde değişiklik yapmak mümkün olmaz.

Zincirde bulunan ilk bloğa “genesis blok” denir ve Bloklar Şekil 3.3.’te olduğu gibi birbirlerine bağlıdırlar. Her bloğun ana tanımlayıcısı vardır ve kriptografik özet fonksiyonu ile bloğun içerdiği tüm veriler kullanılarak elde edilir. Özet fonksiyonları tek yönlüdür ve aynı içeriğe uygulandığında her zaman aynı çıktıyı verir. Bu sebeple bloğun içeriğinde yapılacak herhangi bir değişiklik bloğun özet değerinin değişmesine sebep olur. Değiştiği zaman ise bloklar arasında olan ve bir önceki bloğun özet değeri ile sağlanan bağlantı kopar. Bir saldırının başarılı olabilmesi için, bir bloğunun içeriğinin değiştirilmesi yetersiz kalır, geri kalan ardışık bloklarında içeriğinin değiştirilmesi gerekir. Bu değişikliği ise ağda bulunan tüm düğümlerde uygulaması gerekir, böylece eşler bu konuda fikir birliğine varabilir.



Şekil 3.3. Blok zincir yapısı

Her blokta nonce değeri bulunur. Bu değer geçerli bir özet değeri elde etmek için kullanılır. Özet fonksiyonu aynı girdiye ait aynı çıktı değeri üretir. Çıktının değişebilmesi için içeriğinde değişmesi gerekir. Bu da nonce değeri ile sağlanır. Özet fonksiyonunun ürettiği çıktının değişmesi isteği, blok zincire blok eklerken geçen süreyi belirlemek ve böylelikle zorluk derecesini kontrol edebilmek için vardır. Örneğin Bitcoin’de geçerli bir özet değeri için ilk 4 hane sıfır ile başlamalıdır ve bu da yeni blok oluşturmak için yaklaşık olarak 10 dakikaya tekabül eder (Göbel ve Krzesinski, 2017; Ma, Ge ve ark., 2020).

Her blok başlığında, blok gövdesinde aşamaları yer alan işlem veya işlemlerin Merkle ağaç kökü vardır (Merkle, 1988). Merkle ağaç yapısının kullanılması, işlemlerin doğrulanması için yapılması gereken adımları kolaylaştırır. Bir bloktaki her işlemin kendine ait işlem kimliği vardır. Bu işlem kimlikleri, ilgili işlemin kriptografik özetini ifade eder (Wood, 2014; Ahram, Sargolzaei ve ark., 2017). Bu işlem kimlikleri çiftler halinde bir araya getirilir ve Şekil 3.3'te olduğu gibi özet ağacı meydana gelir. En son oluşan özet ise kök özet değeri olarak bilinir ve Merkle root olarak blok başlığına eklenir.

3.2. Blok Zincir Çözümlerinin Özellikleri

Blok zincir teknolojisinin kullanım avantajları sunmasını sağlayan belirli önemli özellikleri şunlardır:

- Değişmezlik (Immutability): Bir işlem blok zincir ağına kaydedildiğinde ağda bulunan tüm katılımcılara yayınlanır ve değiştirilemez. Blok zincir ağında kayda alınan bir işlem sonsuza kadar ağda kalır.
- Sahteciliğe Dayanıklı (Forgery Resistant): Merkezi olmayan özel, açık veya konsorsiyum blok zincir ağları farklı türde saldırılar için uygundur. Değerli verileri içerebilecek olan blok zincir ağında sahtecilik saldırıları olağandır. Bunun önüne geçebilmek için kriptografik özet fonksiyonlarından ve dijital imzalardan faydalanılır. Ağda bulunan her katılımcının kendine ait dijital bir imzası vardır. Asimetrik şifreleme algoritmalarının yapısı gereği bir katılımcının imzasını taklit etmek hesaplama açısından imkansızdır. İşlemler dijital imzalarla gerçekleşir ve bir katılımcı işlemi yapıp imzaladıktan sonra işlem değiştirilip, katılımcının farklı bir işlemi imzalandığı söylenemez veya işlemi imzalayan katılımcı imzalamadığını söyleyemez. Burada katılımcı, bir cihaz olarak da düşünülebilir. Örneğin bir IoT cihazından diğer IoT cihazına gönderilecek veri, cihazların özel ve genel anahtarları ile asimetrik şifreleme yöntemi kullanılarak gönderilir, bir diğer ifadeyle dijital imza yönteminden faydalanılır.
- Demokratik (Democratic): Eşler arasında dağıtık bir şekilde olan yapı demokratik olmalıdır, otoriter olmamalıdır. Karar verilirken oy birliği göz önüne alınır, oy birliği ise iş kanıtı algoritmaları veya akıllı sözleşmeler ile sağlanır. Blok zinciri, ağdaki tüm katılımcıların, blok zincirin aynı kopyasını tutacak şekilde düzenlenir. Blok zincir ağında bulunan tüm katılımcılar, blok zincirin aynı nüshasını tutarlar. Dolayısıyla, bir cihazın blok zinciri verilerini kötü bir

amaçla değiştirmesi durumunda, sistem bunu anlayabilir ve oy çokluğu ile reddeder, bu şekilde blok zincirin durumu değiştirilmeden kalır.

- Çift Harcamaya Dayanıklı (Double-Spend Resistant): Bir dijital varlığın, aynı anda birden fazla kez kullanılmasına çift harcama denir. Parasal veya parasal olmayan dijital verilerde çift harcama saldırıları yaygındır (Iqbal ve Matulevičius, 2019). Dağıtık ağ yapısında çalışan blok zincir teknolojisinde, yapılan her işlem bir onaydan geçerek zincire eklenir. Bitcoin blok zincirinde bu onaylar madenciler tarafından yapılmaktadır. Ethereum ve Ethereum mantığında çalışan blok zincirlerinde ise bu onaylar için akıllı sözleşmelerden faydalanılır. Bir kripto para blok zincirinde hesabınızda bulunan miktarı, iki katılımcıya birden göndermeye çalışırsanız işlemleriniz gerçekleşmez. Çünkü size ait miktardan fazlasını kullanmaya çalışıyorsunuz anlamına gelir.
- İzlenilebilirlik/Denetlenebilirlik (Auditability): Blok zincirde bulunan her bir blok kendinden önce olan bloğun özet değerini tutar ve bu şekilde zincir oluşur. Birbirine bağlı bu yapı da denetlenebilirlik sağlanmış olur. İşlemin gerçekleşip gerçekleşmediği hızlı bir şekilde kontrol edilebilir.
- Şeffaflık: Blok zincirde veriler açık bir şekilde her katılımcıda tutulur. Her katılımcı, zinciri tutarlı hale getirmek için bilgiyi sorgulayabilir. Her veri açık ve güvenilirdir. Blok zincir teknolojisinin kullanıldığı yerlerde yapı gereği şeffaflık özelliği söz konusudur, blok zincirde verilerin güvenli denetimi için bu şeffaflık temel gereksinimdir.
- Merkeziyetsizlik (Decentralization): Merkezi sistemlerde, merkezi otorite tüm işlemlere hakimdir ve ona uğramadan herhangi bir işlemin gerçekleşmesi mümkün değildir. Ancak blok zincir, eşler arası ağ teknolojisi olduğu için, veriler üçüncü tarafa ihtiyaç duymadan katılımcılar arasında otomatik olarak dağıtılır (Crosby, 2022).

3.3. Blok Zincir Jenerasyonları

İlk olarak Bitcoin (Nakamoto, 2008) ile birlikte bilişim dünyasına adım atan blok zincir teknolojisinin kullanım alanları zaman içerisinde çeşitlenmeye başlamıştır ve blok zincir teknolojisi zamanla gelişim göstermiştir. Swan (2015), blok zincir teknolojisinin gösterdiği bu gelişime Bitcoin ile başladığını (blok zincir 1.0), akıllı sözleşmeler ile bu gelişimin devam ettiğini (blok zincir 2.0) ve kripto para dışında

uygulamalarda kullanılmasıyla gelişimine katkıda bulunduğunu 2015 yılında öne sürmüştür.

Blok zincir jenerasyonları 3 kategori altında incelenebilir (Bashir, 2017). Efanov ve Roschin (2018) ilk nesil olan "Blok zincir 1.0" 'ı, dijital para birimi (digital currency) olarak adlandırmıştır. İkinci nesil "Blok zincir 2.0" 'ı, dijital ekonomi (digital economy) olarak ve üçüncü nesil "Blok zincir 3.0" 'ı dijital toplum (digital society) olarak adlandırmıştır.

Blok zincir 1.0: Blok zincir teknolojisi dünyaya ilk olarak Bitcoin adı verilen dijital para birimi aracılığıyla tanıtılmıştır. Bu jenerasyon; madenciliği, özetleme işlemini, muhasebe hesaplamalarını, işlemleri sağlayan yazılımı ve kripto para birimlerini temsil eder (Efanov ve Roschin, 2018).

Blok zincir 2.0: Bitcoin'in kullanımının yaygınlaşması ile birlikte, blok zincir teknolojisinin basit para transferlerinden farklı olarak kullanılabileceği görüldü. İkinci nesil blok zincir, hisse senetleri ve ipotekleri içeriyor (Burgess ve Colangelo, 2015). Ayrıca birinci nesil blok zincirde mutabakat protokolü olarak Proof of Work (PoW) kullanılır. PoW, hesaplaması zor ve doğrulaması kolay bir mutabakat protokolüdür. Madenci, blok zincir ağına eklenecek olan bloğu teklif etmeden önce belirli kurallara uygun olarak blok özeti üretmeye çalışır. Bu aşama PoW olarak değerlendirilir. PoW mutabakat protokolünün; işlem yükü, tüketilen elektrik miktarı, yüksek gecikme süresi ve düşük verim dezavantajlarından dolayı akıllı sözleşmeler ortaya atıldı ve ikinci nesilde değerlendirildi. Dijital para biriminin kullanımının genişlemesi ile birlikte dijital ekonomiye doğru yol alan bu jenerasyonda akıllı sözleşmeler yer alıyor.

Blok zincir 3.0: Blok zincir teknolojisi finansal hizmetler sektörü dışında gizliliği, güvenliği ve izlenebilirliği sağlamak amacıyla çeşitli alanlarda kullanılmıştır. Üçüncü nesil blok zincir sağlık, sanat, adalet, endüstri gibi birçok alanda yapılan çalışmaları kapsar (Burgess ve Colangelo, 2015; Cheng, Lee ve ark., 2018).

3.4. Blok Zincir Türleri

Blok zincir teknolojisinin kullanım alanlarının yaygınlaşmasıyla beraber farklı yerlerde farklı işlevsel amaçlarla kullanılan blok zincir ağında olan bağlantı türleri de birbirlerinden farklılaşmışlardır (Xu, Weber ve ark., 2017). Blok zincir türleri, ağda bulunan katılımcıların birbirleriyle bağlanma şekline göre ayrılır. IoT uygulamalarında bu türler, blok zincirin nasıl kullanıldığına ve kullanılırken hangi özellikleri taşıdığına

göre sayısız türe ayrılabilir. Bu kısımda, IoT uygulamalarına yardımcı olabilecek farklı blok zincir türlerine değinilmiştir.

- **Genel Blok Zincir:** Blok zincir teknolojisi ilk defa Bitcoin ile birlikte kullanıldı ve açıklandı. Dolayısıyla ilk kullanılan blok zincir teknolojisi genel blok zincir olarak kullanıldı. Genel blok zincir kullanan yapılarda, ağa isteyen herkes katılabilir. Halka açık bir yapıdadır. Doğrulama işlemi, tüm düğümlere dağıtılır. Mutabakat protokollerine uyarak herkes verileri okuyabilir ve blok oluşturabilir. Bu nedenle tamamen dağıtık bir yapıdadır, tamamen şeffaftır ve çok sayıda anonim katılımcılar içerir. Bitcoin ve Ethereum en belirgin örnekleridir. Genel blok zincirlerde güvenliği artırmak amacıyla, zincire yeni blok eklemek isteyen katılımcılar pahalıya mal olacak bir takım işlem yapmak zorundadırlar. Ekledikten sonra ise bir işlem ücreti kazanırlar. Genel blok zincirde, yüksek hesaplama maliyetinin doğası gereği, IoT ile pratik uygulaması zordur. Genel blok zincirler, halka açık olarak işlediğinden ve daha fazla düğüm bulunduğundan dolayı fikir birliğinin oluşması, diğer blok zincir ağ türlerine göre daha uzun zaman almaktadır. Dolayısıyla elektrik tüketim miktarı da doğru orantılı olarak artmaktadır.
- **Özel Blok Zincir:** Özel blok zincir ağlarında, ağa katılmak için izin almak gerekir ve katılımcıların kimliği bellidir. Özel blok zincirler tek bir kurum tarafından yönetilecek olan kurumsal çözümler için genel olarak tercih edilir. Bu kuruluş, verileri okuyabilmek ve ağa katılabilmek için gerekli kuralları isteğine göre düzenler. Özel blok zincirler aynı kurumda bulunan bireyler veya departmanlar arasındaki veri aktarımında kullanmak için tercih edilir. Bilgi ticaretini izlemeyi amaçlayan senkronize çalışan bir veri tabanı görevi görür. Kuruluş tarafından seçilen belirli katılımcılardan bazıları daha çok yetkiye sahiptir. Yazma işlemi merkezileştirilmiş olarak yapıldığı için herkese açık değildir. Genel blok zincirde olduğu gibi doğru yapılmış bir blok yayınlama işlemi sonunda işlem ücreti yer almaz. Sınırlı katılımcı sayısı nedeniyle blok yapım süresi daha kısadır (Dinh, Wang ve ark., 2017) ancak kurcalanmalara karşı genel blok zincir kadar dayanıklı değildir. GemOS (GemOS, 2022), Multichain (MultiChain, 2022) ve Kadena zinciri (Kadena, 2022) örnek olarak verilebilir. IoT ile çalışma açısından özel blok zincir daha uygundur. Küçük ve orta ölçekli ağlar söz konusu olduğu zaman özel blok zincir önerilmektedir (Sabry, Kaittan ve ark., 2019).

- **Hibrit Blok Zincir:** Hibrit blok zincir ağları da, özet blok zincir ağları gibi izin alınarak dahil olunan ağ türüdür. Birden fazla kuruluş arasında veri işlemlerinin şeffaflığı ve denetlenebilirliği için kullanılır. Tamamen merkezi olmayan bir yapıda değildir. Hem tamamen merkezi olmaması sebebiyle hem de bazı izin ve yetkilerin taraflarda olması sebebiyle tamamen sansüre dayanıklı bir yapıda değildir (Zheng, Xie ve ark., 2018). Örnek olarak Hyperledger Fabric ağı verilebilir (Xu, Weber ve ark., 2017). Endüstri ve alan tabanlı (field-based) IoT uygulamalarında hibrit blok zincir ağı kullanılması daha verimli olur (Atzei, Bartoletti ve ark., 2017). Katılımcılara belirli izinler verilebilmektedir veya okuma izni halka açık olarak düzenlenebilir. Özel ve genel blok zincir ağlarının sentezlenmiş hali gibi düşünülebilir.

Çizelge 3.1. Özel, hibrit ve genel blok zincirlerinin karşılaştırılması

	Özel Blok Zincir	**Hibrit Blok Zincir**	**Genel Blok Zincir**
Mutabakatta Olan Katılımcılar	Tek bir organizasyon	Birden çok kuruluştaki düğümler arasında	Bütün düğümler
Merkeziyetsizlik	Merkezi	Kısmen dağıtık	Merkeziyetsiz
Erişim	Kısıtlanabilir	Kısıtlanabilir	Herkese açık okuma ve yazma
Değişmezlik	Değiştirilebilir	Kısmen değişmez	Değişmez
İşlem Süresi	Kısa	Kısa	Uzun
İzinsiz Olup Olmama Durumu	Hayır	Hayır	Evet
İzlenebilirlik	İzlenebilir	Kısmen izlenebilir	İzlenebilir
Yeterlik	Yüksek	Yüksek	Düşük
Okuma Yetkisi	Herkese açık veya kısıtlı	Herkese açık veya kısıtlı	Herkese açık
Süreklilik	Kurcalanmış olabilir	Kurcalanmış olabilir	Neredeyse imkansız
Örnekler	GemOS, Multichain, Kadena zinciri	Hyperledger Fabric	Bitcoin, Ethereum

3.5. Özet Fonksiyonları

Özet fonksiyonları, değişken uzunluktaki girdilerde sabit uzunlukta çıktı veren ve şifreleme işlemlerinde kullanılan bir dizi matematiksel işlemdir. Girdi ve çıktı değerleri arasında benzersiz bir ilişki oluşturur. Deterministiktir ve tek yönlü şifreleme fonksiyonudur. Özet fonksiyonundan elde edilen çıktılara "özet değeri (hash value)" veya "girdi özeti (message digest)" denir.

Şekil 3.4. Özet fonksiyonlarının çalışma mantığı

İşlemler (transactions), işlem ayrıntılarını ve oluşturulma zamanını tutan zaman damgası değerinden oluşur. Bu işlemler tutulurken, özet fonksiyonundan geçirilerek tutulur. İyi bir özet fonksiyonu için; düşük hesaplanabilirlik maliyeti, determinizm, yayılma, rastgelelik, düzenlilik ve tekdüzelik özellikleri gereklidir. Şekil 3.4.'te özet fonksiyonlarının çalışma mantığına yer verilmiştir. Özet fonksiyonlarının sahip olduğu belirli işlevsellikler vardır. Bunlar:

- Özet fonksiyonunda girdi, çeşitli boyutlarda olabilir. Çıktı ise sabit uzunluktadır. Çok büyük uzunlukta bir girdiyi, sabit uzunlukta çıktıya dönüştürebilir.
- Özet fonksiyonlarında, verilen girdiye ait çıktıyı bulmak nispeten verimli olmalıdır. Çünkü işlemleri doğrulama aşamasında fazla işlem gücüne ihtiyaç duyulmamalıdır.
- Her farklı girdi değerinde farklı sabit uzunlukta bir çıktı vermelidir.
- Aynı girdi değerlerinde her zaman aynı çıktıyı vermelidir. Aynı girdi değerlerinde farklı çıktılar vermemelidir. Aynı özet değerine sahip başka bir girdi olmamalıdır. Bu da kırılmalara dayanıklı olması gerektiği anlamına gelir.
- Özet fonksiyonları tek yönlüdür. Yani çıktı değerinden girdi değeri elde edilemez olmalıdır.
- Özet fonksiyonlarında güvenliği sağlayan diğer önemli etken, girdi de yapılacak ufak bir değişiklik ile çıktı değerinden büyük bir değişiklik görülmesidir. Bu özellik sayesinde, blok işlemlerinde yapılacak herhangi bir değişiklik hash değerinin değişmesine sebebiyet verir ve bu değişiklik ile zincirde kopukluk oluşacaktır.

Bir girdi ve bu girdiye ait şifrelenmiş çıktının olduğu durumda, orijinal girdiye şifrelenmiş metinden ulaşılamaz. Bir örnek olarak parola depolama verilebilir. Parolalar, veri tabanında şifrelenmiş olarak tutulur ve şifrelenmiş metinden parolaya tekrar ulaşılamama konusu deterministiktir. Veri tabanına ulaşan biri parolayı

okuyamamalıdır ve şifreli metinden parolaya ulaşamamalıdır. Doğrulama yapılırken ise, oturum açılacağı zaman girilen parola tekrar özet değeri oluşturulur ve veri tabanında olan özet ile karşılaştırılır. Eğer özetler aynı ise giriş başarıyla gerçekleştirilir.

Blok zincire yeni kayıt eklendiği zaman (yeni blok eklendiği zaman), bloğun özeti tüm ağa yayınlanır. Ağda olan her düğüm blok zincirin tamamının kopyasını tutmaz, belirli bazı düğümlerin tutması yeterlidir. Ağda olan her düğüm, son özet değerini bildiğinden dolayı doğrulama işlemini yapabilirler. Verilerde değişiklik yapmanın tek yolu aynı özet değerlerini farklı bloklarda elde etmek gerektiğidir, bu da pratik olarak imkansızdır. Çünkü her blok, oluştuğu zamana ait zaman damgası değerini içerir ve bu değer ile özetlenir.

MD5 (Message Digest 5) (Rivest, 1992) özet algoritması, Ron Rivest tarafından geliştirilen bir algoritmadır. Girdi olarak değişken uzunlukta veri alır ve 128 bitlik özet çıktı üretir (Pittalia, 2019). MD2, MD3 ve MD4 versiyonları da vardır (Damasevicius, Ziberkas ve ark., 2012). Bu versiyonlara göre MD5, daha yavaş çalışır ve daha güvenilirdir. Genel olarak veri bütünlüğü kontrol etmekte kullanılır (Damasevicius, Ziberkas ve ark., 2012). Benzersiz özet üretme konusunda bazı sorunlara sahiptir, bu nedenle güvenlik açıkları bulunmaktadır.

The Secure Hash Algorithm (SHA), ilk olarak 1993 yılında National Institute of Standards and Technology (NIST) tarafından yayınlanmıştır. SHA256 ve SHA512, günümüzde güvenli olarak kabul edilen ve kullanılan özet algoritmalarıdır. Bitcoin, şu anda özet algoritması olarak SHA256'yı kullanmaktadır (Courtois, Grajek ve ark., 2014).

Keccak-256 algoritması, SHA256 ile benzer bir algoritmadır. Ethereum'da kullanılan özet algoritmasıdır.

Race Integrity Primitive Evaluation Message Digest (RIPEMD), 1996 yılında Hans Dobbertin tarafından geliştirilmiştir. Bu özet algoritması, MD5'te bulunan güvenlik açıklıklarının yerini doldurmuştur. Günümüzde güvenli bir özet algoritması olarak kabul görmektedir. RIPEMD-128, RIPEMD-160, RIPEMD256 ve RIPEMD-320 gibi varyasyonları vardır (Boyd ve Nieto, 2009).

Çizelge 3.2. Özet fonksiyonları karşılaştırması

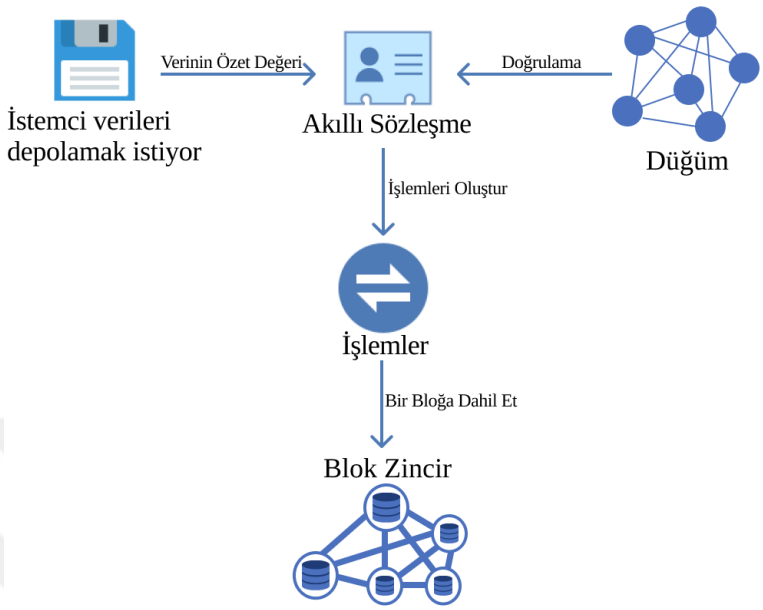
Özet Fonksiyonu	Özet Uzunluğu	Sembol Sayısı	Güvenlik
MD5	128 bit	32 sembol	Hayır
SHA1	160 bit	40 sembol	Hayır
SHA256	256 bit	64 sembol	Evet
SHA512	512 bit	128 sembol	Evet
Keccak-256	256 bit	64 sembol	Evet
Ripemd 1	160 bit	40 sembol	Evet

3.6. Akıllı Sözlemeler

Akıllı sözleşme, dijital ortamda tanımlanan ve bir dizi ön koşulun olduğu karşılıklı taraflar arasındaki işlemleri yürütme mantığını (execution logic) içeren programlanabilir anlaşmadır. Akıllı sözleşmeler blok zincir içinde Ethereum Sanal Makinesi'nde (EVM - Ethereum Virtual Machine) çalışan yürütülebilir kod parçalarıdır. Bu fikir ilk olarak 1994 yılında Nick Szabo tarafından "kamu ağlarında ilişkilerin güvenliğini sağlama" amacıyla ortaya atılmıştır (Szabo, 1997). Akıllı sözleşmede koşullar sağlandığı müddetçe dinamik bir biçimde yürütme mantığı ilerler ve uygun bilgiler elde edilir (Christidis ve Devetsikiotis, 2016). Akıllı sözleşmeler Blok Zincir 2.0 jenerasyonunun temelini oluşturur. Belirlenen koşullar sağlandığı müddetçe veriler ve akıllı varlıklar işlenir ve yönetilir (Luu, Chu ve ark., 2016). Otomatik olarak çalışır ve programlanabilir olması özelliği ile karmaşıklığı azaltır. Akıllı sözleşmeler kullanıcıların ve sistemlerin güvenliğine katkıda bulunsa da dijital bir platformda bulunduğu için potansiyel güvenlik tehditleri de bulunmaktadır (Singh, Parizi ve ark., 2020). Akıllı sözleşmeler yürütülürken yapılan işlemler için belirli ücretleri alınır, bu da olası kötü niyetli saldırıları engeller ve yeni blok yayınlamaya teşvik eder.

Akıllı sözleşmeler, blok zincir programlama dilinden farklı bir programlama dili ile yazılmaktadır ve yazıldıktan sonra blok zincire entegre edilmektedir. Akıllı sözleşmelerin çalışması 3 aşamada gerçekleşmektedir. Birinci adım, akıllı sözleşmenin oluşturulmasıdır. Bu aşamada, Solidity (Ethereum blok zincir için) veya Chaincode (Hyperledger blok zincir için) gibi yazılımlar kullanılır. Solidity, üst düzey programlama dilidir. İkinci adımda akıllı sözleşme, eşler arasında dağıtılır. Akıllı sözleşmeler blok zincire dağıtıldıktan sonra değiştirilemez veya üzerinde düzenleme yapılamaz. Blok zincirde saklanır ve bir adreste bulunur. Üçüncü adımda ise akıllı sözleşme yürütülür. Akıllı sözleşmeler otomatik olarak belirli yürütme koşulları ve belirli yürütme mantığı sağlandığında uygulanır. Tüm işlemler, izlenebilir işlemler

olarak blok zincire kaydedilir (Li, Li ve ark., 2020). Herhangi bir zamanda herhangi bir katılımcı, akıllı sözleşmeyi çağırabilir. Şekil 3.5.'te akıllı sözleşmelerin akış diyagramına yer verilmiştir.



Şekil 3.5. Akıllı sözleşme akış diyagramı

3.7. Blok Zincir Katmanları

Blok zincir teknolojisi katmanlı bir mimariye sahip değildir ancak yapı gereği matematik, bilgisayar bilimleri, ekonomi ve kriptoloji gibi farklı alanları içeren bir teknolojidir. Ayrıca nesnelerin interneti, bulut bilişim ve yapay zekâ gibi farklı teknoloji alanlarıyla beraber kullanılmaktadır. Blok zincir birçok düğüm arasında birçok cihazla kullanılabilmektedir. Bu sebeplerden dolayı katmanlı bir yapıda değerlendirilmesi ve ortak bir yapıda klasikleştirilmesi faydalı olacaktır. Blok zincir mimarisi 6 katmandan oluşmaktadır (Yuan ve Wang, 2018; Lu, 2019). Bu katmanlar ve içerdikleri bazı bileşenler Çizelge 3.3.'te verilmiştir.

Çizelge 3.3. Blok zincir katmanları ve içerikleri

KATMANLAR	**BİLEŞENLER**
Veri Katmanı	Blok zincir verileri, zincir yapısı, zaman damgası, Merkle ağacı, özet fonksiyonu, şifreleme, kriptografik algoritmalar, cihazlar, araç, kaynaklar, dijital imza
Ağ Katmanı	P2P ağı, doğrulama mekanizması, yayılma teknikleri
Mutabakat Katmanı	PoW-Proof of Work, PoS-Proof of Stake, PoB-Proof of Burn, pBFT-Practical Byzantine Fault Tolerance
Kontrat Katmanı	Akıllı sözleşme, komut dosyası kodlama, teşvik mekanizması, sorun mekanizması
Servis Katmanı	Ethereum, Hyperledger
Uygulama Katmanı	Kripto para, lojistik kontrol, üretim, sağlık, bulut hizmeti

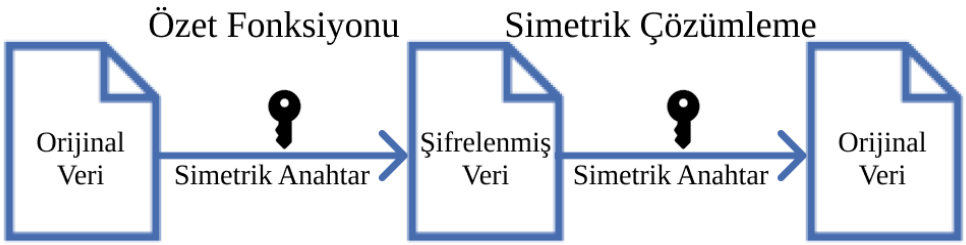
Katmanların farklı faaliyetlerini daha net açıklayabilmek için, Lojistik firmalarının bir yerden başka bir yere taşıyacağı ürünlerin kontrolü ve kayıtları blok zincir ile uygulanırken, işlemlerin hangi katmanlarda yer aldığı verilecektir. Bu durumda ürünlerin nerede olduğu ve durumları hakkında bilgi sahibi olmak isteyen düğümler, son kullanıcılardır. Lojistik firmaları ise, potansiyel ürün sağlayıcılardır. Bu sebeple, hibrit blok zincir yapısı ve Hyperledger blok zinciri ile uyumlu çalışacak bir mekanizmadır. Bu kısım servis katmanında yer almaktadır. Gerekli bilgiler, talepler, ödemelerin koordine edilmesinin kaydedilmesi ve doğrulanması kısmı ise veri katmanında yer almaktadır. Ağ katılımcıları üçüncü bir taraf olmadan, doğrudan sağlanan verilerle eşler arası ağda iletişime geçebilmektedir ve bu kısım ağ katmanında yer almaktadır. Katılımcılar belirli bir fikir birliği algoritmasına göre lojistik firmaları tarafından sağlanan verileri işlerler. Bu kısım mutabakat katmanında yer almaktadır. Lojistik firmasının bildirdiği bilgiler uygun akıllı sözleşmelere göre blok zincire eklenir ve bu kısım kontrat katmanındadır. Bu uygulamanın bir emsali farklı uygulamaların blok zincir ile entegre aşamasında benzer olarak kullanılabilir. Bu kısım ise uygulama katmanında yer almaktadır.

3.8. Şifreleme Yöntemleri

Şifreleme işlemleri kriptografik işlemler kullanılarak yapılır. Kriptografi, taraflar arasından veri alış-verişi yapılırken, istenmeyen tarafların ham verilere ulaşmasını engelleyen matematiksel yöntemlerin bütünüdür. Verilerin bütünlüğünü ve gizliliğini sağlar. Verilerin gizliliği ve bütünlüğü için kullanılan yöntemler simetrik şifreleme ve asimetrik şifreleme olarak ikiye ayrılır. Bu yöntemlerde verileri şifrelemek ve şifreyi çözmek için anahtar olarak isimlendirilen bir şifreleme elemanı kullanılır.

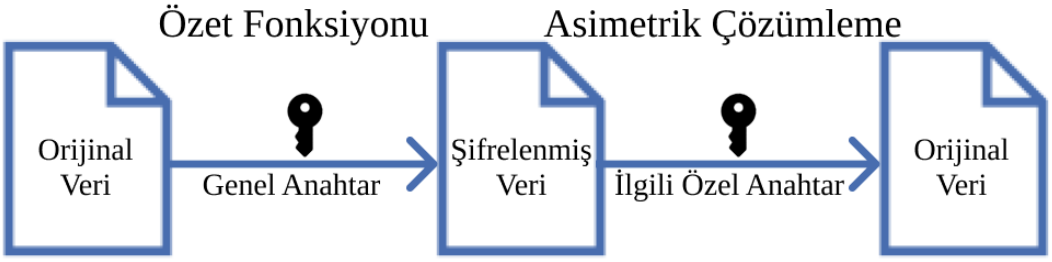
Simetrik şifreleme (Symmetric Encryption): Simetrik kriptografi de verileri şifrelemek ve verilerin şifresini çözmek için kullanılan anahtar aynıdır. Bu sebeple paylaşılan anahtar şifrelemesi ismiyle de kullanılır. Şifreleme ve şifre çözme işlemleri kullanılacak olan anahtarın ne olacağına dair anlaşılarak dağıtılmalı veya taraflar arasında olacak veri alış-verişinden önce oluşturulmalıdır. Daha az işlem gücü gerektirdiğinden hızlı bir kriptografi türü olsa da kullanılan anahtarın aynı olması sebebiyle güven konusunda bazı riskler teşkil etmektedir. Örneğin paylaşılan anahtarın üçüncü taraflar tarafından ele geçirilmesiyle gönderilen verilere ulaşılabilir. RC5 (Rivest Cipher 5), DES (Data Encryption Standard) ve AES (Advanced Encryption

Standard) yöntemleri simetrik şifrelemeye örnek olarak gösterilebilir (Fontaine ve Galand, 2007). Şekil 3.6.'da simetrik şifreleme çalışma mantığına yer verilmiştir.



Şekil 3.6. Simetrik şifreleme

Asimetrik şifreleme (Asymmetric encryption): Asimetrik kriptografi de, verileri şifrelemek ve şifreyi çözmek için kullanılan anahtarlar farklıdır. Şifreleme için kullanılan anahtar, şifre çözümleme işlemi için kullanılmaz. Bu anahtarlara genel (açık) anahtar ve özel anahtar isimleri verilmiştir. Genel anahtar, kimlik doğrulama ve şifreleme için kullanılır. Özel anahtar ise imzalama ve şifre çözme için kullanılır. Genel anahtar herkes ile paylaşılırken, özel anahtar sadece sahibi tarafından bilinmelidir. Genel anahtar ile şifrelenmiş bir veri ancak ilgili özel anahtar ile çözülebilir. Özel anahtar ile şifrelenmiş bir veri ise yalnızca ilgili genel anahtar ile çözülebilmektedir. Asimetrik şifreleme yöntemleri nispeten daha karmaşık anahtarlama sistemlerinden dolayı daha fazla işlem gücü gerektirmektedir. Simetrik şifreleme sistemlerine göre daha yavaş çalışırlar. Asimetrik şifreleme sistemleri, yalnızca ilgili alıcının okuması gerekli olduğu durumlarda kullanılır. Asimetrik şifrelemenin kullanıldığı durumlarda, gönderilen mesaj ele geçirilse dahi şifreli olduğundan ve ilgili özel anahtara sahip olmadan çözülemeyeceğinden dolayı içeriğe ulaşılamayacaktır. DSA (Digital Signing Algorithm) ve RSA (Rivest-ShamirAdleman) asimetrik şifreleme şemaları örnek olarak verilebilir. Şekil 3.7.'de asimetrik şifreleme çalışma mantığına yer verilmiştir.

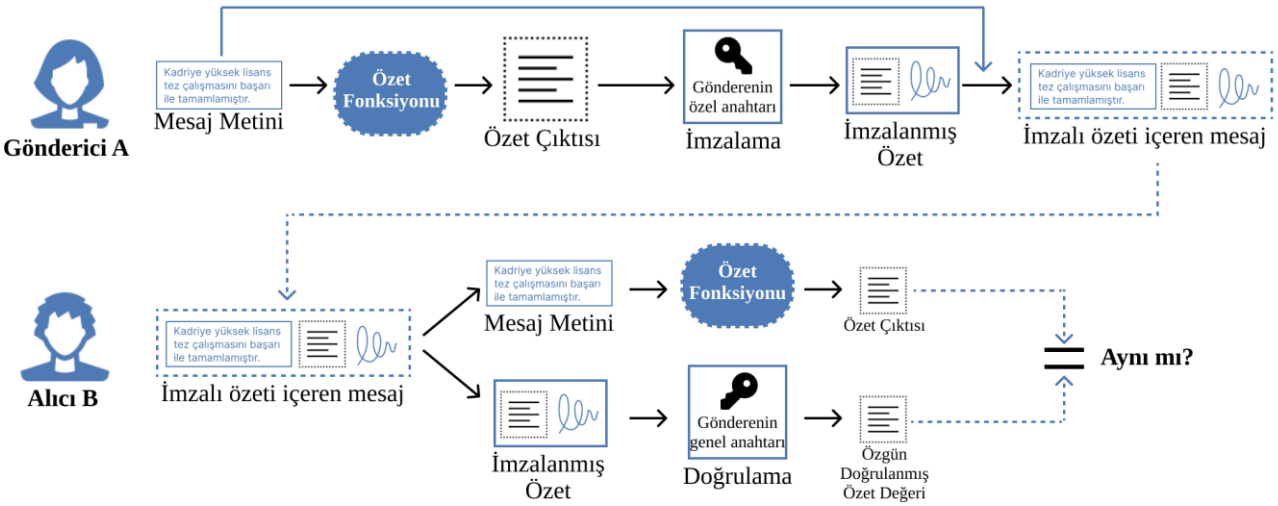


Şekil 3.7. Asimetrik şifreleme

3.9. Dijital İmzalar

Blok zincir teknolojisi, güvenliği ve gizliliği sağlayabilmek için temel olarak kullandığı işlem, asimetrik şifrelemedir. Asimetrik şifreleme, açık anahtar şifrelemesi (public key cryptography) olarak da bilinir. Asimetrik şifreleme özel anahtar ve genel anahtar olmak üzere iki adet anahtar çiftinden oluşur (Buldas, Laanoja ve ark., 2017). Özel anahtar, yalnızca sahibi tarafından bilinir ve açık anahtar ise herkesle paylaşılır. Bu anahtar çifti için bir login ekranında gelen kullanıcı adı ve şifre benzetmesi yapılabilir. Kullanıcı adını herkes bilebilir, ancak şifre sadece sahibi tarafından bilinmelidir. Blok zincir ağında bulunan her düğüm bu anahtar çiftine sahip olmak zorundadır. Asimetrik şifreleme blok zincirde veri şifreleme ve dijital imzalar olma üzere iki görev için kullanılmaktadır (Yli-Huumo, Ko ve ark., 2016). Blok zincirde veri şifreleme ile, verinin güvenliğinden ve verinin bütünlüğünün bozulmadığından emin olunur. Bu işlem verileri, blok zincir ağı üzerinden iletilir. Gönderen ve alıcı birbirlerinden emin olabilmeleri için dijital imzalar kullanılır.

Blok zincirde, özel anahtar ve genel anahtar çiftleri veri şifreleme dışında dijital imzalar için de kullanılır. Bir kullanıcı, öncelikle göndermek istediği belgenin özetini elde eder ve ardından kendi özel anahtarı ile bu özeti şifreler. Aslında burada dijital bir imza atmış olur. Tüm işlem, şifrelenmiş verilerden ve dijital imzadan oluşur. Şifrelenen karakter dizisini ve belgenin aslını gönderir. Herhangi bir alıcı gönderenin genel anahtarını kullanarak bu belgenin gerçekten gönderen tarafından imzalanıp imzalanmadığını kontrol edebilir. Unutulmamalıdır ki, genel anahtar ile şifrelenmiş bir metin özel anahtar ile çözülebilir, aynı şekilde özel anahtar ile şifrelenmiş metin genel anahtar ile çözülebilir. İşlem devamında şifrelenen karakter dizisi ve belgenin aslı, alıcının genel anahtarı ile tekrar özet fonksiyonundan geçiriliyor ve gönderiliyor. Alıcı ise kendi özel anahtarı ile çözer. Alıcının genel anahtarı ile tekrar şifrelenmesi ve özel anahtarı ile çözülmesi işlemi, ortak anahtar şifrelemesi olarak adlandırılmaktadır. Belgenin aslına ve belgenin göndericinin özel anahtarı ile özet fonksiyonundan geçirilmiş haline erişiyor. Göndericinin genel anahtarı ile gelen belgenin aslını özet fonksiyonundan geçiriyor ve gelen özet ile karşılaştırıyor. Eğer aynı ise belgenin bütünlüğünden emin oluyor. Dijital imzanın arkasındaki mantık buradan gelmektedir. Dijital imza, kripto para birimi dışında blok zincir teknolojisi içeren uygulamalarda alış-verişi yapılan verilerin güvenliğini sağlamak amacıyla kullanılmaktadır. Şekil 3.8'de dijital imza algoritmasının çalışma mantığına yer verilmiştir.



Şekil 3.8. Dijital İmza Algoritması (Digital Signature Algorithm - DSA)

3.10. Merkeziyetsiz Mutabakat Algoritmaları

Blok zincir ağında birden fazla düğüm bulunur. Fikir birliği algoritmaları, blok zincir ağının doğru ve düzgün bir şekilde çalışması için gereklidir. Bir veya daha fazla kötü niyetli düğümün var olduğu blok zincirde, geçerli bir işlem geçersiz olarak veya geçersiz bir işlem geçerli olarak belirtilebilir. Bu istenmeyen bir durumdur. Bu tür sorunların çözümü olarak uzlaşma algoritmalarına ihtiyaç duyulur.

Her düğüm, blok zincirin bir kopyasına sahiptir ve karar verici merkezi bir otorite yoktur. Mutabakat sürecinde, herhangi bir düğüm blok önerir ve geri kalan diğer düğümler, önerilen blok konusunda hemfikir olurlar. Hemfikir olduktan sonra, her düğüm yeni bloğu kendi blok zincirine ekler. Aslında mutabakat, diğer düğümlerin blok zincire eklenecek olan blok konusunda hemfikir olması için sağlanması gereken koşulların belirlenmesidir (Zheng, Xie ve ark., 2017).

Anlatılan şekilde bir tasarım planında, yeni blok eklenmeden önce belirli bir süre geçmesini istersiniz. Çünkü aynı anda birden fazla bloğun gelme ihtimali olmamalıdır. Bu durumda blok zincire işlemlerin nasıl ekleneceği ve sıralanacağı fikir birliği algoritmaları ile belirlenir. Merkezi bir otorite olmadığı için, eklenecek yeni blok üzerinde her düğümün hemfikir olması beklenir.

Mutabakat algoritması blok zincirde bulunan tüm işlemlerin orijinal ve geçerli olmasını sağlar, ayrıca defterdeki kayıtların tutarlı olmasını sağlar. Günümüzde bir çok farklı mutabakat algoritması bulunurken PoW ve PoS algoritmaları en çok oranda tercih edilmektedir (Zheng, Xie ve ark., 2017). Mutabakat algoritmalarında amaç, çeşitli saldırılara karşı ağı dirençli ve güvenilir bir hale getirmektir. Bu bölümde mutabakat

algoritmalarından bahsedilecektir ve amaç, blok zincir mutabakat algoritmalarının IoT uygulamalarıyla birlikte kullanılırken, uygulamaya uygunluğunun farkında olabilmektir.

3.10.1. Proof-of-Work (POW)

Bu algoritma ilk olarak Bitcoin blok zincirinde kullanılmıştır. Bu algoritmada kullanılan mantık, bir blok ağa teklif edilmeden önce belirli işlerin yapılması gerekir. Bu iş, zaman alan ve hesaplama açısından maliyetli bir iştir. Belirtilen bu süreci tamamlamak zordur ancak doğrulaması kolaydır. Blok zincire eklenen işlemlerin güvenilir ve benzersiz olduğunun kanıtıdır. POW algoritması, hesaplama işlemini kriptografik olarak yapar. Blok içeriği aynı kaldığı düşünüldüğü zaman, bloğun kök özet değeri değişen bir nonce değerine göre ayarlanır. Bölüm 3.1.'de anlatıldığı üzere nonce değeri rastgele bir sayıdır ve sayıya göre bloğun kök özet değeri değiştirilebilir. POW algoritmasının mantığı ise, bloğun kök özet değerinin belirli sayıda sıfır ile başlama koşuludur. Bitcoin'de bu koşul, bloğun kök özet değerinin ilk dört hanesinin sıfır ile başlamasıdır. Bu değeri bulmak zaman alıcıdır ve önemli miktarda işlemci gücüne ihtiyaç duyulur.

Herhangi bir düğümün önerdiği bloğun blok zincire eklenebilmesi için, iş kanıtı algoritmasının koşuluna uyması gerekir. Blok eklemeyi başaran düğüm, bunun sonucunda ödül alır. Bu ödülü almak için düğümler yarış halindedirler. Yeni blok ekleme sürecine madencilik adı verilmektedir ve yeni bu görevi yapan düğümlere ise madenci denir. Ağın büyüklüğü ve düğüm sayısı arttıkça hem ağın güvenliği artmaktadır hem de iş kanıtı algoritmasının zorluğu da artmaktadır. Günümüzde Bitcoin ve Ethereum madenciliği için büyük donanım havuzları gerekmektedir. POW algoritmasının gerektirdiği işlem gücü, zaman alıcı olması, işlem sonucunda verilen ödül ve tükettiği elektrik miktarının fazla olması (Lin ve Liao, 2017) ağı saldırılara karşı korur. POW algoritmasının sahip olduğu dezavantajlardan dolayı, IoT uygulamalarında dikkatli bir şekilde kullanılması önerilmektedir. Gerektirdiği hesaplama gücü ve maliyeti sebebiyle, kullanılan IoT uygulamalarında verimsiz hale gelebilir.

3.10.2. Proof-of-Stake (POS)

POW algoritmasında, harcanan enerji ve oluşan masrafların önüne geçebilmek adına geliştirilmiştir. Harcanan elektrik tüketimini azaltmayı amaçlar (O'Dwyer ve Malone, 2014). POW algoritmasında yer alan madencilik teriminin yerini, stake etmek terimi almaktadır. Madenciler teriminin karşılığı ise, doğrulayıcılardır. Bu algoritmada

işlem blokları, doğrulayıcılar tarafından doğrulanır. Madencilik işlemi için gereken aşırı güç tüketimi ve güçlü donanım havuzlarına gerek yoktur. Ödül kazanılması için, belirli bir ücret ortaya konulmalıdır. Bankaya, mevduatınızı faizli bir şekilde yatırmak benzetmesi yapılabilir. Belirli bir süre boyunca verilen ücret kullanılamaz. Süre sonunda verilen ücret miktarı serbest bırakılır ve sistem belirtilen miktarı öder.

Doğrulayıcılar rastgele seçilirler ve rekabet etmezler. Bundan dolayı hesaplama gücüne ihtiyaç duyulmaz ve yüksek elektrik tüketimi gerçekleşmez. Seçilme şansını arttırmak için stake edilen ücret miktarı, stake etme süresi, ağda toplamda stake eden doğrulayıcı sayısı ve enflasyon oranı bu algoritmada önemlidir. İlk aşama için belirli bir ücret almak gerekir ve sisteme koymak gerekir. Seçildikleri zaman ise, bloğu oluşturmaları gerekir ve eğer başka bir blok önerilirse onaylamaları gerekir (Leonardos, Reijsbergen ve ark., 2020). Peercoin (Kaur, Chaturvedi ve ark., 2021) ve Ethereum bu iş kanıtı algoritmasını kullanmaktadır. Ayrıca POW algoritmasında olduğu gibi işlem gücü gerektirmediğinden, POW algoritmasına göre hızlı çalışır (Solat, 2018). POS algoritması, parasal bir mantığa dayandığı için IoT uygulamaları için uygun değildir.

3.10.3. Pratik Bizans Hata Toleransı (PBFT)

Bu algoritma Bizans Generalleri Problemi'nin (Byzantine Generals Problem) (Vukolić, 2016; Maull, Godsiff ve ark., 2017) çözümüne yönelik eş zamansız ortamlar için bir algoritmadır. Ağda bulunan düğümlerin üçte birinden daha az miktarının kötü niyetli olabileceği varsayılır. Ağın tüm düğümleri tarafından oylama gerçekleştirilir. Bu oylamanın daha pratik olabilmesi adına şifreli mesajlaşma içerir. İşlemin gerçekleşebilmesi için, ağda bulunan düğümler tarafından en az 2/3 oranıyla oy çokluğu ile fikir birliğine varmış olmak gerekir.

Blok ağa yayınlamadan önce, ağdaki doğrulayıcı düğümler bloğun kök hash değerini hesaplarlar. Bir dizi işlem "lider" adı verilen düğüm tarafından bir bloğa toplanır ve ağa yayınlanır. Ağın geri kalan düğümlerinden elde edilen oylama sonucunun 2/3'ü, ağa yayınlanacak aday bloğun lehineyse, düğümler bunu kendi blok zincir kopyalarına eklerler.

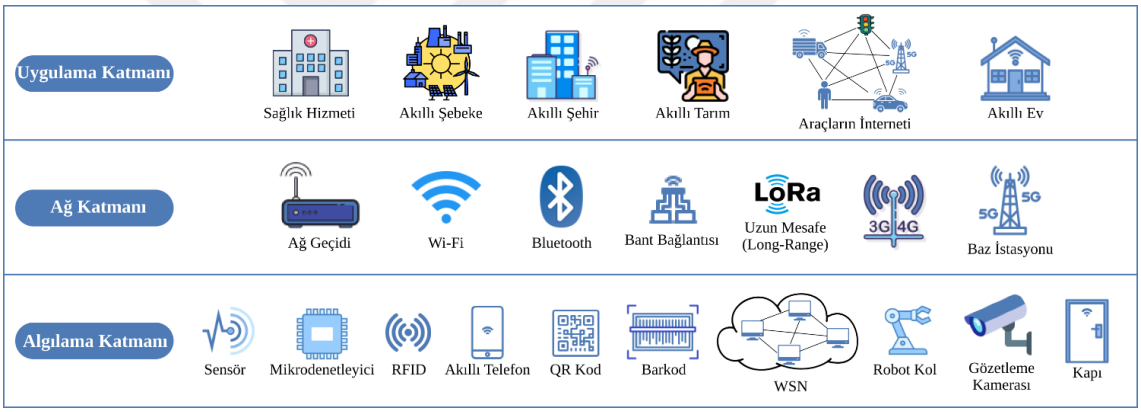
Bu algoritma, düşük gecikme gücü, yüksek derecede verim ve işlem hızının hızlı olmasını sağlar. Bu özelliklerinden dolayı IoT uygulamalarında tercih edilebilir. Ancak oylamadan kaynaklanan ek yük, algoritmayı ölçeklenemez hale getirir. Bu sebeple, büyük IOT ağlarında kullanım için uygun değildir.

Çizelge 3.4. Mutabakat algoritmalarından bazılarının kriter karşılaştırması

Kriterler	**PoW**	**PoS**	**PBFT**
Enerji harcaması	Yüksek	Orta derecede	Düşük
Güvenilir Model	Güvenilmeyen	Güvenilmeyen	Yarı güvenilir
Eşler arası ağ ölçeklenebilirliği	Yüksek	Yüksek	Düşük veya yüksek
Verim	Düşük	Düşük	Yüksek
Tip	İzinsiz	İzinsiz	İzinli
Örnek	Bitcoin	Peercoin, Ethereum	Hyperledger Fabric

4. IoT MİMARİSİ

Bir dizi düğümü kontrol edebilmek ve yönetebilmek için bir mimari tasarıma ihtiyaç duyulur. Bu düğümler bilgisayarlar, cep telefonları veya sensörler gibi çeşitli işlevleri yerine getirebilen aletler olabilir. Bu mimari tasarımda popüler bir yaklaşım olan merkezi mimari yer alır. Merkezi mimari istemci-sunucu mimarisi olarak da adlandırılır. Merkezi mimari yaklaşımında merkezi sunucu, yönetici görevindedir (Hugoson, 2009). Düğümler arasında görev dağılımını yapar ve düğümlerden gelen isteklere cevap verir. Merkezi mimarinin en yaygın örneklerinden biri olan IoT sistemi; üç, dört veya beş katmandan oluşabilir (Tewari ve Gupta, 2020). Ancak IoT mimarisi, temel olarak 3 katmandan oluşur (Fernández-Caramés ve Fraga-Lamas, 2018). Bunlar; algılama katmanı, ağ katmanı ve uygulama katmanıdır. Şekil 4.1.'de 3 katmanlı IoT mimarisine yer verilmiştir.



Şekil 4.1. 3 katmanlı IoT mimarisi

4.1. IoT Merkezileştirilmiş Modelin Zorlukları ve Sorunları

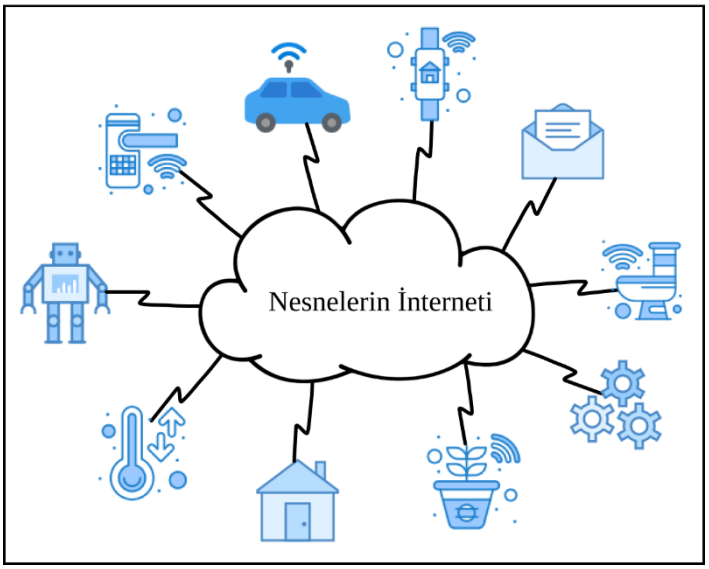
Nesnelerin interneti için, bulundukları fiziksel ortamdaki verileri toplayan, işleyen ve ileten sensörlerin elektronik yetenekleri olduğu söylenebilir. Çok fazla miktarda olan bu cihazlar fazlasıyla veri üretir. Üretilen bu verilerin çokluğu hem avantajları hem de dezavantajları beraberinde getirir. Örneğin günümüz bilimsel çalışmaları için gerekli verileri sensörler otomatik toplayabilir. Bu verilerin güvenli bir şekilde tutulması da güvenlik, gizlilik ve ölçeklenebilirlik konusunda kritik bir sorun teşkil etmektedir. IoT'de üretilen bu veriler, kimi zaman cihazlarda (sunucular, akıllı telefonlar gibi) kimi zaman da internet aracılığı ile bulutta tutulabilmekte veya duran veri (atıl veri) olarak adlandırılabiliyor. IoT sisteminin, günümüzde kullanılan merkezi mimarisi, kullanılan cihazların sayısının fazla olması ve bu cihazların kuracakları iletişim konusunda belirli avantajlar sağlar. Ancak cihazlar açık bir ağda çalıştığı için ve

iletim halinde oldukları için çeşitli güvenlik açıkları bulunmaktadır. Bu bölümde merkezi IoT modelinin sahip olduğu bazı sorunlar verilmiştir.

- IoT sistemleri heterojen bir yapıdadır, farklı şekilde çalışan ve farklı biçimde veriler üreten cihazlar içerir. Bu cihazların iletişim protokolleri farklı yapıda olabilir ve heterojen bir şekilde çalışmayı gerektirir. Ağ ve güvenlik koordinasyonun sağlanması zorlaşır. Bu özellik, sorunların temelini oluşturmaktadır.
- Tüm işlemler merkezi bir sunucuda depolanır, işlenir ve ağda gideceği yere yönlendirilir. Merkezi sunucuda oluşacak herhangi bir hata tüm ağı etkiler.
- Merkezi sunucuda depolama sağlandığından dolayı, düzenlenecek herhangi bir saldırıda tüm verilerin mahremiyeti ve güvenliği tehdit altına girer. Verilerin tek bir yerde tutulması veri ihlaline sebebiyet verebilir (Conoscenti, Vetro ve ark., 2017). Ayrıca kötü amaçlı bir düğüm eklenerek tüm sistemin etkilenmesi sağlanabilir.
- Merkezi sunucuda depolama yapılması, fiziksel olarak büyük miktarda depolama yerlerine ihtiyaç duyulur. Bu durumda maliyete sebebiyet verir (Fernández-Caramés ve Fraga-Lamas, 2018).
- Merkezi sunucu, IoT ağında olan cihazlar arasında olan tüm veri iletişimini ve veri işleme işlemlerini kontrol etmekle yükümlüdür. Bu durum, merkezi sunucuya büyük bir iş yükü oluşturur. Dolasıyla ağda bazı gecikmeler yaşanabilir ve kullanıcıların esnekliğinde sınırlamalar meydana gelebilir (Atlam, Walters ve ark., 2018).
- Merkezi IoT mimarisinde, ağ iyi ölçeklenemeyebilir. Cihaz sayısı arttıkça merkezi sunucunun görevi ve işlevleri artar, dolayısıyla ağın ölçeklenebilirliği düşer (Atlam ve Wills, 2019).
- IoT ağında çok sayıda cihaz ağa giriş çıkış yapabilir. Bu durumda merkezi sunucunun verimi düşebilir ve ağ tıkanıklığı, kimlik doğrulamada hata gibi sorunlar oluşur. IoT'de hizmetlerin çoğuna gerçek zamanlı olarak ihtiyaç duyulur. Yerine getirilecek işlev, gerçek zamanlı olarak tamamlanamazsa, yeniden planlama söz konusu olamaz.
- Çok fazla cihazın yer aldığı IoT ağlarında, sistem karmaşıklığı oluşabilir ve bu durum veri kaybına sebep olabilir.

5. BLOK ZİNCİR VE IoT ENTEGRASYONU

IoT'de ki nesneler kavramı, makinelerde bulunan sensörler veya insan yaşamındaki verileri toplayan fiziksel cihazların hepsini içerir (Yan, Zhang ve ark., 2014). İnsan yaşamını kolaylaştırmak ve insan gücünü en aza indirmek avantajıyla süre gelen bu teknoloji ile insanlar, nesneler ve sensörler sürekli bağlantı halinde bulunmaya başlamıştır. Özellikle akıllı telefon ve akıllı saat teknolojilerinin gelişimi ile, sayısız cihaz IoT'de yer almaya başlamıştır. Fiziksel ortamı algılama, iletişim kurma ve harekete geçirme yeteneğine sahip IoT teknolojisinde temel amaç, kişisel bilgisayarın, sensörlerin ve akıllı cihazların; buzdolabı, çamaşır makinesi, arabalar gibi ögelerle birlikte bir ağ üzerinden uyum içerisinde çalışabilmesidir (Aazam, St-Hilaire ve ark., 2016). Şekil 5.1.'de nesnelerin internetine genel bakış açısı verilmiştir.



Şekil 5.1. IoT'ye genel bakış

Uygulamalarının çoğunun sensör verilerine dayanan IoT heterojen ortamının, veri manipülasyonuna ve yetkisiz paylaşıma karşı dirençli olması gerekir. Ayrıca güvenlik kameraları ve nem, sıcaklık, gaz miktarı gibi verileri sensörlerden toplayan IoT cihazları fiziksel tehlikeye açıktır (Balamurugan ve Biswas, 2017). IoT'nin mevcut topolojisinden ve kısıtlı kaynaklarından ötürü günümüzde kullanılan güvenlik teknolojileri, IoT teknolojisine tam olarak uygulanamaz (Dorri, Kanhere ve ark., 2017; Lu ve Da Xu, 2018). Bu sebeple, IoT cihazlarında yüklü olan kodların ve cihazlar arası paylaşılan verilerin bütünlüğü tehlikededir. IoT sistemleri, gerçek zamanlı veri aktarımı sağladığı için düşük bellek, düşük güç tüketimi ve düşük hesaplama yeteneği gibi sistem

performans verimliliği faktörleri de göz önünde bulundurulmalıdır. Bulut bilişim ve IoT teknolojilerinin entegre kullanımı sayesinde cihazlar tarafından üretilen büyük miktardaki veri işlenebilir hale gelir. Bulut bilişim, kullanıcılar arasında paylaşılan internet tabanlı bilgisayar kaynakları sağlayan ve zamana bağlı olmadan erişilebilen dağıtılmış bilgi işlemdir. Yüksek bilgi işlem gücü (high computing power), yüksek performans (high performance), cihaz erişilebilirliği (device accessibility), düşük maliyetli hizmetler (low-cost services), geniş ağ erişimi özellikleri sayesinde (Botta, De Donato ve ark., 2016) IoT teknolojisinin sorun teşkil eden özelliklerine çözüm sunar.

Bulut bilişim teknolojisi ve IoT'nin sentez kullanımı sayesinde IoT büyük bir ilerleme kaydetmiştir. Ancak bu entegre kullanımda, bulut servis sağlayıcısının bulutta ve ilgili servislerde depolanan veriler üzerinde kontrolü olduğu için veri manipülasyonunda güvenlik konusunda açıklar bulunmaktadır. Ayrıca fiziksel güvenlik, veri erişim kontrolleri ve şifreleme yöntemleri gibi güvenlik tehditleri de vardır (Horrow ve Sardana, 2012). Bulut servis sağlayıcısı verilere ulaşabilir ve verilere müdahalede bulunabilir (Ravishankar, Kulkarni ve ark., 2020).

IoT'de cihazların senkron çalışması, cihazların yetkilendirilmesi, veri işleme ve veri aktarımı gibi tüm görevler merkezi bir model üzerinden yönetilir. Bu sistemde, görevleri aksatmadan yerine getirebilecek üst düzey sunucular gerekir. Sürekli artan IoT cihazları göz önüne alındığında, sunucuların karşılaması gereken işlevler de artmıştır. Güvenli bir şekilde bağımsız olarak veri alış-verişi yapması gereken bir sistem için bu modelleme uygun değildir.

Blok zincir teknolojisi hem nesnelerin interneti cihazları için hem de dağıtılmış bulut bilişim (distributed cloud computing) ile entegre kullanımının dezavantajları için güvenli bir dağıtılmış sistem sunar. IoT sistemlerinin heterojen yapıda olması, cihazların kısıtlamaları, yetkilendirme problemleri, bilgi yönetimi ve verilerin depolanması, sistem yapılandırması, erişim kontrolü gibi sorunlar ana sorun olarak görülmektedir (Jing, Vasilakos ve ark., 2014). Bu sorunlara dayanarak blok zincir teknolojisi ve IoT'nin bir arada kullanılması sentezinin sunduğu avantajlar şunlardır:

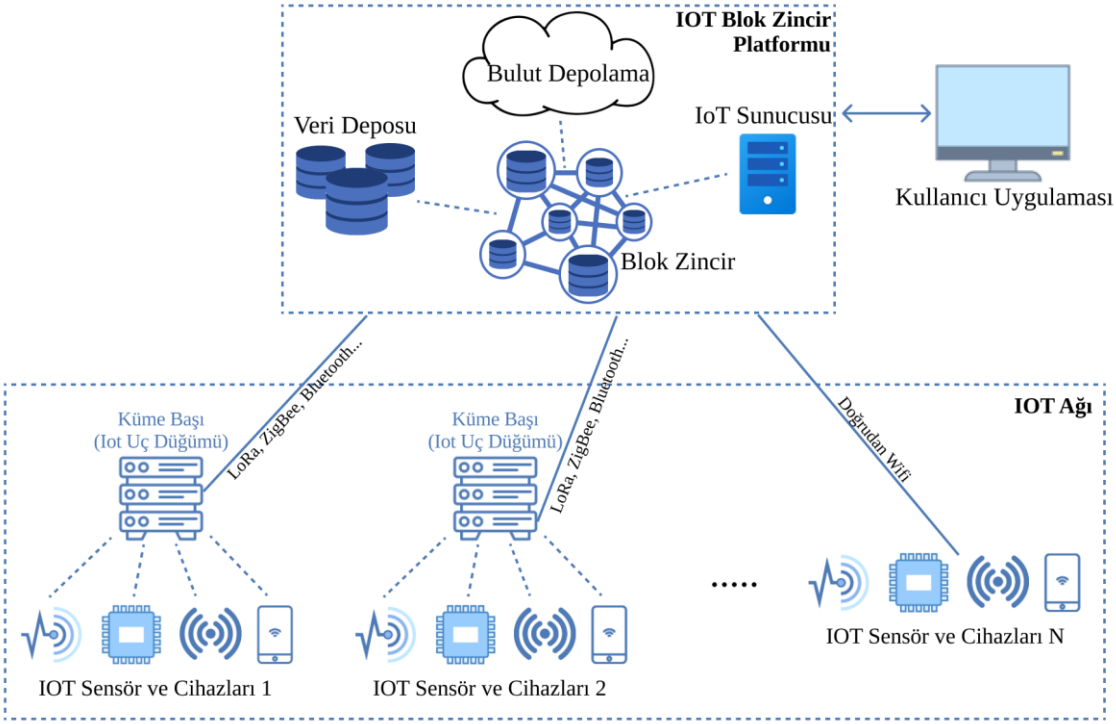
- IoT cihazlarının bulunduğu ağın dağıtık ve eşler arası yapıda olması özerklik (autonomy) sağlar ve verilerin merkezi bir sunucudan geçmesine gerek kalmaz. Blok zincir ağında bulunan katılımcılar, veriyi gönderen katılımcının kimlik doğrulamasını yapabilirler ve verinin değişime uğrayıp uğramadığının kontrolünü sağlayabilirler (Alharby ve Van Moorsel, 2017). Üçüncü bir tarafa veya cihaza ihtiyaç duymadan her katılımcı kendi arasında iletişimde

bulunabilir. Güven, karşılıklı iki taraf arasında doğrudan sağlanır. Bu nedenle gelen veriye olan güvenilirlik vardır.

- Blok zincir ağları, sunucusuz kayıt tutma fikrinin temelidir, katılımcılar üzerinde yedek kayıtlarını depolar. Böylelikle kırılmalara karşı dayanıklılık sağlar. IoT çerçevelerinin hem birlikte çalışabilirlik özelliğini hem de veri sızıntı ve kırılmalarına karşı dirençli olmasını sağlar.
- Blok zincire eklenen her blokta, "zaman damgası" olarak adlandırılan, bloğun oluşma zamanını içeren bir bilgi vardır. Böylelikle IoT'de izlenebilirlik ve garantili hesap verebilirlik sağlanır.
- Blok zincir teknolojisi, dağıtılmış bir ağda mutabakat protokolleri sayesinde tutarlı olmayı ve hataları tolere edebilmeyi sağlar. IoT'de herhangi bir saldırının başarılı olabilme ihtimali veya yapılan bir hatanın kalıcı hale gelmesi böylelikle imkânsız hale gelir.
- Blok zincirde eşler arası ağda veriler gönderilirken asimetrik şifreleme, dijital imzalar ve özet fonksiyonları kullanıldığı için IoT cihazlarının içerebileceği gizli ve özel verilerin mahremiyeti korunur. Blok zincir dayanıklı kayıt tutma mekanizmasıdır. IoT'nin karşı karşıya olduğu en önemli sorunlardan olan güvenlik ve gizliliktir. IoT'de güvenlik konusunda fazla miktarda tehdit ve zayıflık vardır (Alaba, Othman ve ark., 2017). Blok zincire eklenen herhangi bir verinin değiştirilmesi mümkün değildir. Böylelikle DDoS ve DOS saldırılarına karşı IoT ağı güvenli hale gelmiş olur.
- Blok zincir teknolojisinde kullanılan akıllı sözleşmeler, eşler arası ağda işlemler sağlanırken karşılıklı güvene gerek kalmamasını sağlar. Akıllı sözleşmeler, gerekli şartlar sağlanmadığında işlemin gerçekleşmeyeceğini garanti eder. IoT'de kimlik doğrulamasına ve erişim kontrolüne yardımcı olur.
- Genel blok zincir ağ türü özel sunucular kullanmayı gerektirmez, katılımcıların hesaplama ve depolama yeteneği kullanılır. Katılımcılar blok zincire yaptıkları katkılar için ödül aldığından IoT'de demokratikleşmekte önemli rol oynar.

IoT cihazları sensörler ve aktüatörler olarak iki gruba ayrılırlar. Sensörler, fiziksel çevreden verileri toplar ve topladıkları bu verileri sunuculara iletirler. Aktüatörler ise, gerçekleştireceği belirli eylemleri kullanıcılardan alınan komutlara göre yapar. Şekil 5.2.'de IoT ve blok zincir teknolojilerinin bir arada kullanıldığı bir örnek

şema verilmiştir. Şekilde verilen "Kullanıcı Uygulaması", blok zincir ağından veri okuyabilecek veya veri yazabilecek bir cihaz olabilir. Örneğin, akıllı bir evde ev kullanıcıları, evin blok zincir ağında bulunan cihazların durumunu okuyabilir veya yeni veri yazabilir. IoT blok zincir ağında, depolama üniteleri sabit disk gibi donanımsal ve bulut gibi yazılımsal olabilir.



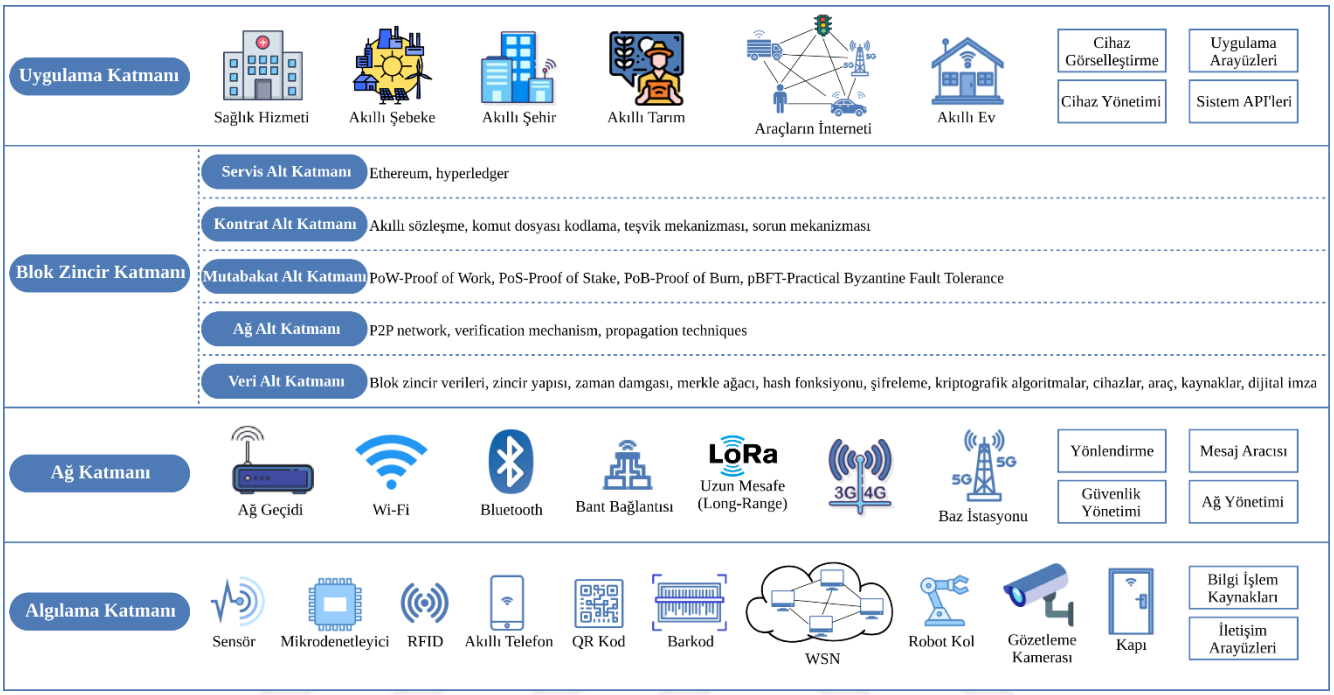
Şekil 5.2. IoT ve blok zincir platformlarının bir arada kullanıldığı kavramsal şema örneği

5.1. Blok Zincir Mimarisi ile IoT

IoT teknolojisi; heterojen yapısı, kaynaklardan dolayı olan kısıtlamalar, mahremiyet ve güvenlik gibi sorunlar ile karşı karşıyadır. Blok zincir teknolojisi bu sorunlara; gelişmiş gizlilik ve güvenlik, birlikte çalışabilirlik gibi özellikler ile çözüm sunmaktadır.

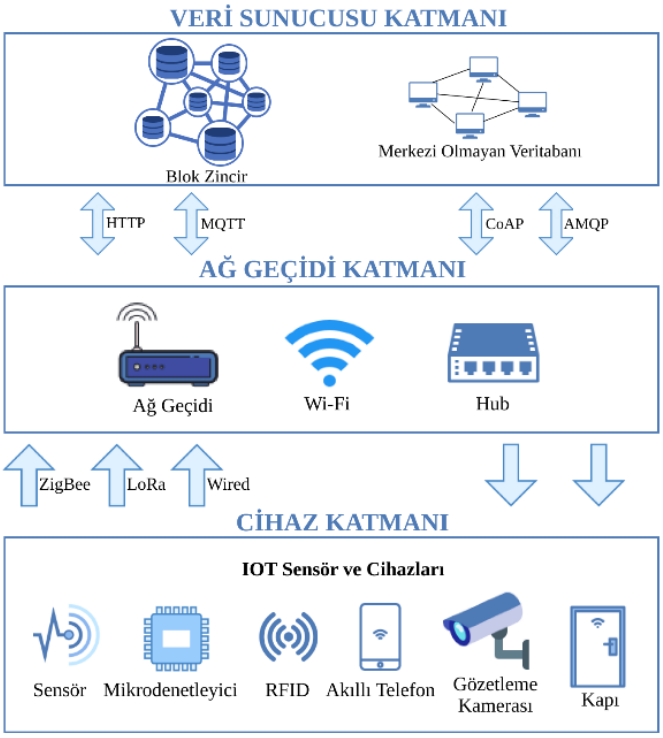
Şekil 5.3'te veri üretiminden IoT cihazlarının sorumlu olduğu ve veri depolama da blok zincir teknolojisinin kullanıldığı bir mimari verilmiştir. Veriler depolandıktan sonra değişiklik kesinlikle yapılamaz, çünkü blok zincirde veriler kalıcı bir şekilde depolanır. Blok zincire kaydedilecek veri onaylandığında, bloğa kaydedilir ve blok kurcalanmalara karşı dayanıklı hale gelir. Çünkü zincir herhangi bir değişikliğe açık halde değildir. Yapılacak herhangi bir değişiklik zincirde kırılmaya sebebiyet verir. Ayrıca ağda yapılan her kayıt, kolayca ulaşılabilirdir. Bu şekilde izlenilebilirlik sağlanır. Ağın türüne göre (genel, özel veya hibrit blok zincir) yetkili düğüm veya düğümler tüm

işlemlere bakabilir. Ayrıca farklı IoT cihazlarında eşler arası bir ağ kurabilmek için protokoller gereklidir. API arayüzü, IoT uygulamalarının blok zincir hizmetlerine erişmesini sağlar.



Şekil 5.3. IoT'de blok zincir kullanımında katmanlı sistem

Fiziksel bir katman olan algılama katmanı, veri iletişimi ve veri depolaması özelliklerini barındıran ve çeşitli bağlantılar içeren cihazlardan oluşur. Ağ katmanının birincil görevi cihazlar arasında yönlendirme yönetimini gerçekleştirmektir. Bunun sebebi fiziksel cihazların küresel internet protokolleri bulunmamasıdır. Diğer görevleri ise güvenlik yönetimi, mesaj aracısı ve ağ yönetimidir. Blok zincir katmanı, eşler arası ağ ve veri tutarlığının sağlanabilmesi için gerçekleştirilen mutabakat algoritmaları gibi blok zincir teknolojisinin çeşitli özelliklerini IoT teknolojisinde sağlamak amacıyla ortak hizmetleri içerir. Ağda bulunan her düğüm, blok zincirin bir kopyasına sahip olabilir. Bu katmanda, fiziksel cihazlar tarafından toplanan verilerin güvenli depolanması sağlanır. Ağda yapılacak herhangi bir değişiklik, kopyalanan her blok zincire saniyeler içinde yansıtılır. Burada akıllı sözleşmeler, defterdeki erişimi yöneten kod parçasıdır ve harici bir uygulama tarafından çağrılır. IoT cihazları arasında kimlik doğrulama ve mutabakat gibi işlemler eşler arası blok zincir ağında devam ettirilir. En üst katman olan uygulama katmanı, cihazları kontrol etmek için farklı arayüzlerin barındığı ve cihazlardan sağlanan verilerin görselleştirildiği katmandır.



Şekil 5.4. Blok zincir IoT ağ geçidi entegrasyonu

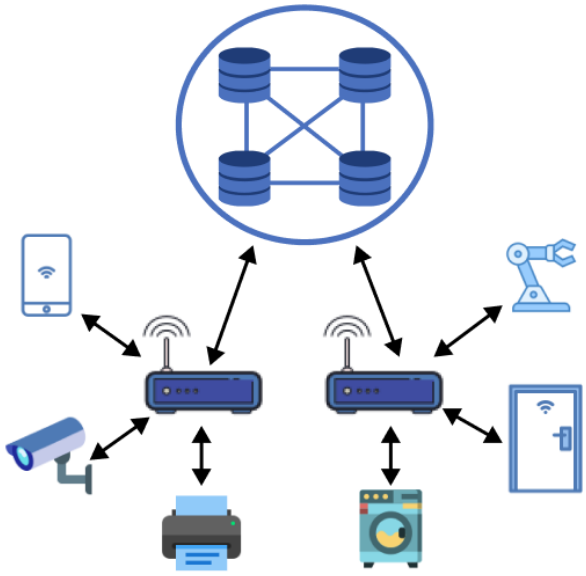
IoT teknolojisi ve blok zincir teknolojisinin bir arada kullanıldığı uygulamalarda bant genişliği ve aktarılan veri miktarında artış gözlenmektedir. Şekil 5.4.'te blok zincir teknolojisinin kullanıldığı bir IoT uygulamasının etkileşimi gösterilmektedir. Ağ geçidi, blok zincir tabanlı IoT mimarilerinde veri yönetimi ve ağ yönetimi olarak önemli rol oynar.

Blok zincir tabanlı IoT ağlarında, tüm IoT cihazları aynı ağda çalışır. IoT cihazları bir karma algoritma gerçekleştirerek ağa yeni blok yayınlayacak kazanan cihazı (düğümü) seçerler. Tüm cihazlar bloğu doğrular. Ardından ağda bulunan her bir IoT cihazı, bloğun kopyasını kendi yerel defterine kaydeder.

5.2. Blok Zincir ve IoT Entegrasyon Şemaları

IoT cihazlarının kaynak kısıtlamalarını göz önünde bulundurursak bir blok zincir ağına katılımlarında bazı önemli hususları göz önünde bulundurmak gerekir. Örneğin birçok IoT cihazının kriptografik işlem gücü yoktur, bundan dolayı blok zincirde bulunan mutabakat algoritmalarına katılma ve depolama işlevlerini karşılayamazlar. Blok zincir teknolojisi IoT ağına farklı şekillerde entegre edilebilir. Blok zincir tabanlı bir IoT ağı geliştirmeye yönelik 4 farklı yaklaşımdan söz edilmektedir.

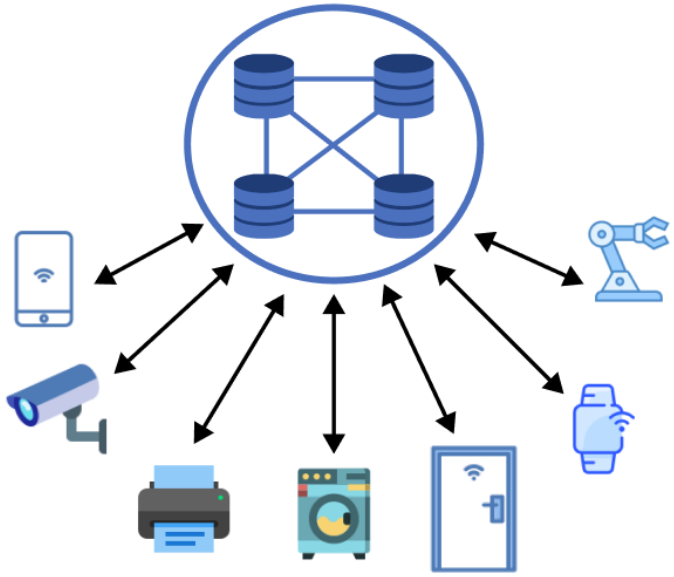
Blok zincirin uç noktaları olarak ağ geçidi cihazları: Bu entegrasyon şemasında bütün işlemler blok zincirden geçer. Tüm iletişim mekanizmaları blok zincir aracılığıyla gerçekleşir. IoT ağ geçitleri, blok zincir ağına uç noktalar olarak bağlıdırlar. IoT cihazları ise, IoT ağ geçitlerine bağlıdırlar ve kayıtlıdırlar. Ağ geçitleri, IoT cihazlarından aldıkları işlemleri blok zincire iletirler. Böylece tüm işlemler için izlenebilirlik sağlanmış olur. Bu entegrasyon şeması, farklı blok zincir ağ geçitlerine bağlı cihazlar arasında olan iletişimin kimlik doğrulaması içinde kullanılabilir (Cha, Chen ve ark., 2018). Bu entegrasyon yaklaşımında, aktarılan tüm veriler blok zincirinde depolanmak zorunda değildir. Aktarılan tüm verilerin blok zincire kaydedilmesi, bant genişliğini ve depolama gereksinimini artırır. Bu ölçeklenebilirlik sorunu, blok zincir teknolojisi ve IoT entegrasyonu ile ilgili önemli bir araştırma sorunudur. Şekil 5.5., bu entegrasyona ait şemayı göstermektedir. Bu tez çalışmasında bu entegrasyon şeması kullanılmıştır.



Şekil 5.5. Blok zincirin uç noktaları olarak ağ geçidi cihazları şeması

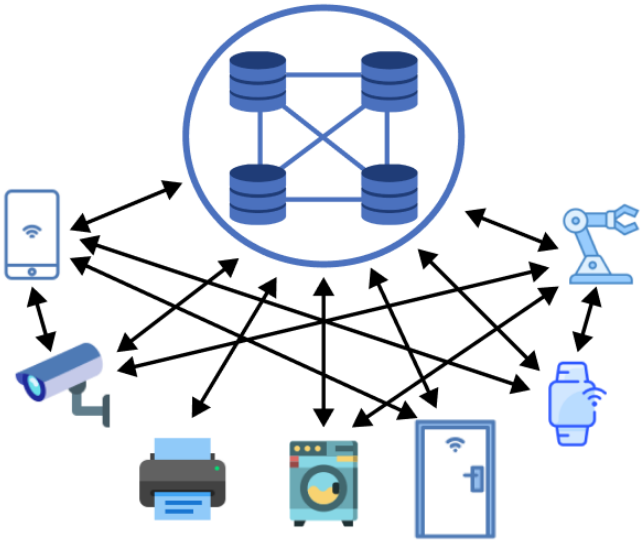
Blok zincirine işlem verenler olarak IoT uç cihazları: Bu yaklaşımda, tüm etkileşimler blok zincirinden geçer ve değişmez bir etkileşim kaydı sağlar. Bu yaklaşımda tüm etkileşimler blok zincire kaydedildiği için, bütün detaylar sorgulanabilir ve tüm etkileşimler izlenebilir. Güvenli hesap verilebilirlik sağlanmış olur. Tüm etkileşimlerin kaydedilmesi bant genişliğinde artışa sebep olacaktır. Ancak IoT cihazlarında blok zincirin bir kopyası saklanmaz. Şekil 5.6.'da gösterildiği gibi basitçe

blok zincire işlemler düzenler. Bu yaklaşımda IoT cihazları daha fazla karmaşık hesaplamalara maruz kalır ancak özerkliği de artar.



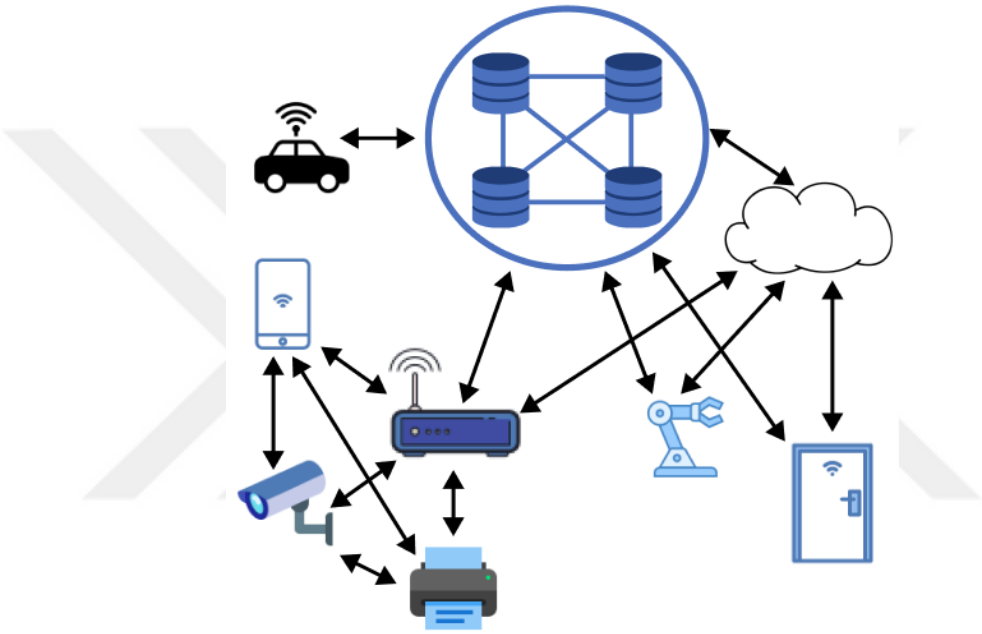
Şekil 5.6. Blok zincirine işlem verenler olarak IoT uç cihazları şeması

Blok zincirin uç noktaları olarak birbirine bağlı IoT cihazları: Bu yaklaşım Şekil 5.7.'de verilmiştir. IoT cihazları blok zincir ile doğrudan işlem yaparlar ve birbirleriyle blok zincir dışında da iletişim kurabilirler. Çevrimdışı çalışabilir. Düşük gecikme süresi sağlar. Blok zincire her etkileşimin kaydedilme zorunluluğu yoktur. Belirli etkileşimler blok zincirde saklanır ve IoT etkileşimleri blok zincir kullanılmadan gerçekleşir. Bu yaklaşım genel olarak IoT etkileşimleri sayısının yüksek olduğu, düşük gecikme ve yüksek güven istendiği durumlarda kullanılır.



Şekil 5.7. Blok zincirin uç noktaları olarak birbirine bağlı IoT cihazları şeması

IoT uç cihazları ile bulut-blok zinciri hibriti: Bu yaklaşımda verilerin yalnızca bir kısmı blok zincirde saklanır, geri kalan işlemler ise doğrudan IoT cihazları arasında gerçekleşen yaklaşımın bir uzantısıdır (Reyna, Martín ve ark., 2018). Bu yaklaşımda olan zorluklardan biri, hangi etkileşimlerin blok zincirde depolanacağını seçmektir. Bu yaklaşımda blok zincir ve IoT'nin sınırlamalarını tamamlamak için bulut bilişim kullanılabilir. Diğer zorluk olarak, gerçek zamanlı olarak meydana gelen etkileşimler ve blok zincirden geçen işlemler ayrımının optimize edilmesidir. Şekil 5.8., bulut tabanlı blok zincir IoT yaklaşımını göstermektedir.



Şekil 5.8. IoT uç cihazları ile bulut-blok zinciri hibriti şeması

Hangi entegrasyon şemasının kullanılacağına IoT uygulamasının gereksinimlerine göre karar verilir. Değiştirilemez kayıt tutma gereksinimi olduğunda ve nispeten daha az etkileşim gerçekleştiğinde, ilk iki etkileşim şemasını kullanmak daha mantıklıdır. Daha yüksek performans gerektiren uygulamalarda tek başına blok zincir kullanmak yeterli olmayabilir, hibrit bir entegrasyon şeması kullanmak daha mantıklı olacaktır.

6. BLOK ZİNCİR VE AKILLI EV UYGULAMA ÇERÇEVESİNİN ARKA PLANI

Akıllı ev tüketicilere; güvenlik, sağlık, konfor ve iyileştirilmiş yaşam standardı gibi özellikler sunan IoT entegre konutudur. Akıllı ev sistemleri, insanların hayatlarını ve bağımsız yaşamlarını daha kolay ve daha iyi hale getirme yeteneğine sahiptir. Akıllı ev kullanıcılarının ve akıllı ev cihaz geliştiricilerinin dikkatini çeken hareketleri izleme ve hareketlerin güvenlik değerlendirmesini yapma gibi kullanışlı özellikler sağlar. Akıllı evler, ev sahiplerine ve ilgili taraflara büyük faydalar sağlasa da, kullanıcıların güvenliğini ve mahremiyetini tehlikeye atabilecek kötü niyetli siber saldırılar için potansiyel olarak risk altındadır. Bu tür tehditler, merkezi ve büyük saldırılara karşı savunmasız olan geleneksel çözümlere sahiptir. Ayrıca çok yönlülük ve ölçeklenebilirlik bakımından eksiklik barındırmaktadır. Birçok akıllı teknoloji insanların hayatını kolaylaştırmaktadır (Abbas, Khan ve ark., 2020). Aynı zamanda bu tür teknolojiler büyük miktarda veri üretir ve sürekli gelişen bu teknolojilerin verileri, geleneksel depolama yöntemleri ile saklanması güvenlik endişeleri yaratır.

Blok zincir teknolojisi, uzaktan bağlantı ve veri iletimi gibi çeşitli akıllı ev teknolojilerinde siber güvenlik altyapısının temel taşı olarak olağanüstü performans elde etmiştir. Blok zincir teknolojisi ve merkeziyetsiz depolama ağları bu sorunları çözmek için kullanılabilir. Merkezi olmayan sistemler tarafından kontrol edilebilen, zaman damgası içeren, kötü niyetli saldırılara karşı dayanıklı belgeler topluluğu olan blok zincir teknolojisi aynı zamanda esnek bir yapıya sahiptir.

Akıllı ev IoT ekosistemi, bilgi ve iletişim teknolojilerindeki gelişmelerin tetiklediği bir geçiş sürecinden geçmektedir. Akıllı ev uygulamalarındaki en önemli zorluklardan biri güvenlik ve mahremiyetin sağlanmasıdır. Blok zincir teknolojisi potansiyel olarak akıllı ev uygulamaları için uygundur. Blok zincir teknolojisi, bu sistemdeki potansiyel tehdidi ortadan kaldırmaya yardımcı olur ve kullanıcıların güvenini kazanır.

Blok zincir teknolojisi, akıllı ev sahiplerinin kritik bilgilerini, herhangi bir üçüncü taraf hizmet sağlayıcı tarafından mahremiyetleri ihlal edilmeden bankalar ve çevrimiçi pazarlar gibi diğer kurumlara güvenli bir şekilde göndermelerine olanak tanımaktadır (Moniruzzaman, Khezr ve ark., 2020). Sonuç olarak, blok zincir akıllı evlerde giderek daha popüler hale gelmektedir.

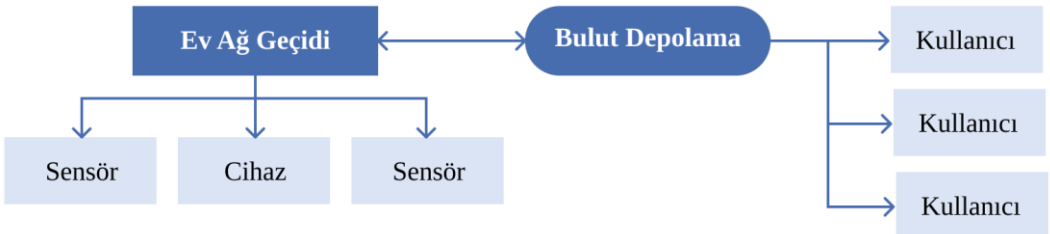
Blok zincir sistemlerinin güvenli ve merkezi olmayan olduğu kanıtlansa da, ev aletleri ve cihazlar arasındaki iletişim ve işbirliği miktarı, akıllı ev sistemlerini siber saldırılara ve tehditlere karşı daha savunmasız hale getiren veri sızıntısına yol açabilir. Blok zincir teknolojisi, kullanıcılar için güvenliği sağlamak ve veri gizliliğini korumak için temel bir çözüm olarak kullanılmıştır. Bununla birlikte, IoT uygulamalarında blok zincir teknolojisinin kullanımına ilişkin bazı zorluklar vardır (Dorri, Kanhere ve ark., 2016; AbuNaser ve Alkhatib, 2019). Bunlar:

- Veriler, zaman içinde artan düğüm sayısı nedeniyle büyük miktarda yer kaplama eğilimindedir, bu da defter boyutunda bir artışa yol açar.
- Ölçeklendirmede yaşanan zayıflık sebebiyle, ağdaki artan düğüm sayısıyla başa çıkma yeteneği düşüktür. Bu nedenle blok zincirinin temel özelliği değişebilir ve yaklaşımını daha merkezi bir kontrol tarzına kaydırabilir.
- IoT cihazları, yüksek oranda kısıtlanmış kaynaklar içerir.
- Blok madenciliği uzun zaman alabilmektedir.

6.1. Akıllı Ev Nedir?

Akıllı evlerin nasıl çalıştığını anlamak için, akıllı evi oluşturan çeşitli cihazların olduğunu ve bu cihazların uzaktan kullanılabileceğini, kontrol edilebileceğini ve düzenlenebileceğini bilmek gerekir. Akıllı evler aslında iletişimsel ağlardır ve bu iletişimsel ağlar akıllı ev ağında bulunan bireylerin yaşam tarzlarına uyan kişiselleştirilmiş bir ortam kurulmasına yardımcı olurlar ve cihazlar arasında veri gönderip alma görevinde yer alırlar.

Araştırmacılar akıllı evi, "iletişim ağında bulunan sensörler ve cihazları ev bireylerinin uzaktan izlemelerine ve uzaktan kontrol etmelerine izin veren, iletişimin sık ve düzenli olduğu hizmetler bağlamında bulunan bir ev" olarak tanımlamaktadır (Gram-Hanssen ve Darby, 2018).

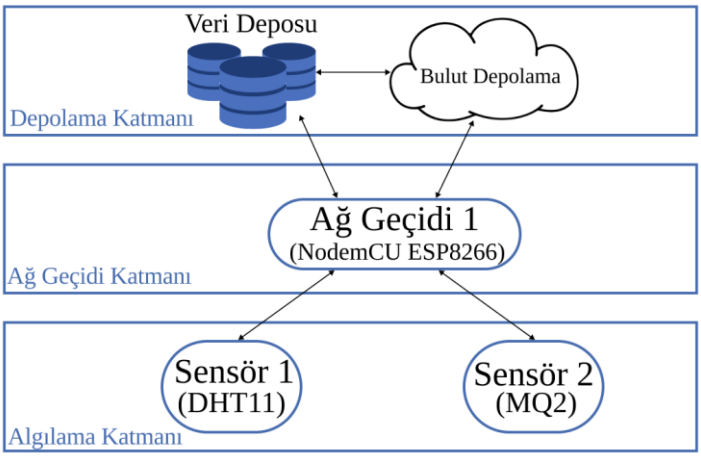


Şekil 6.1. Akıllı ev sisteminin geleneksel yapısının tasviri

Şekil 6.1.'de görüldüğü gibi, akıllı ev sistemlerinin temel teknik özelliği, cihaz ve sensörler ile akıllı ev bireyleri arasında aracı unsur rolünü merkezi bir kontrol merkezinin oluşturmasıdır. Bu merkezi kontrol merkezi "ev ağ geçidi" olarak adlandırılır. Harici ağı ev ağına bağlayan bir dizi iletişim protokolü kullanır (Li, Gu ve ark., 2018).

6.2. Önerilen Ağ Yapılandırması

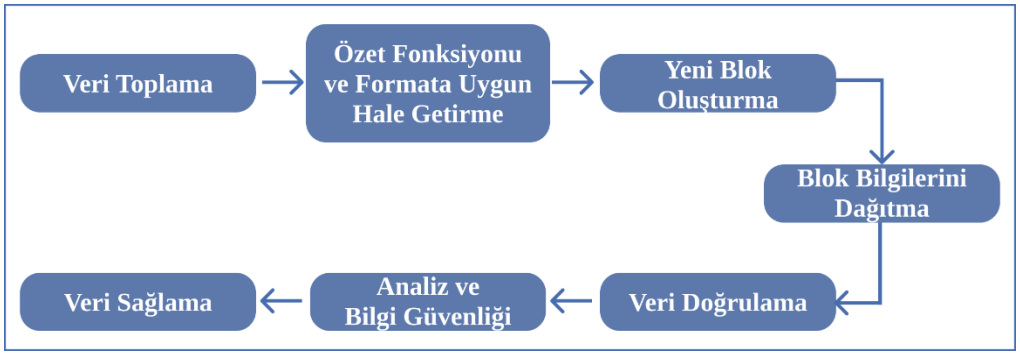
Akıllı ev ağ geçidi depolamasına uygulanan blok zincir; sensörler, IoT cihazları ve diğer ortamlar arasında bütünlük ve gizlilik ile veri aktarımı yapmanın önemli bir parçasıdır. Akıllı ev ağı, her akıllı evde merkezi bir ağ tipine sahip olmasına rağmen, depolama katmanında blok zincir kullanılarak dağıtık bir ağa merkezi olarak uygulanmıştır. Önerilen blok zincir tabanlı akıllı ev, üç katmana sahiptir. Bunlar; "algılama katmanı", "ağ geçidi katmanı" ve "depolama katmanı" 'dır. İlk katman olan algılama katmanı akıllı bir evde bulunan çeşitli IoT cihaz ve sensörlerini içerir. İkinci katman olan ağ geçidi katmanı cihaz katmanı tarafından üretilen verileri saklar ve gerektiği zaman kullanıcılara sunar. Üçüncü katman olan depolama katmanı, ağ geçidi kimliğini ve blok zincirdeki her ağ geçidi tarafından işlenen verileri kaydeder. Bloklar, kullanıcılara her zaman ve her yerde bilgi sağlanabilmesi için paylaşılır. Bu, Şekil 6.2.'de gösterildiği gibi ifade edilebilir.



Şekil 6.2. Önerilen ağın tasarımına genel bakış

Şekil 6.3., uçta bulunan cihaz ve sensörlerden veri toplanmasına, blok zincire kaydedilmesine ve kullanıcılara uygun şekilde sunulmasına izin veren mimarinin metodolojik akış şeması göstermektedir. Toplanan veriler, özet değeri oluşturma ve

formata uygun hale getirme işleminden geçer. Kullanıcılara yalnızca gerekli bilgileri sağlamak için veri analizi ve kalite bakımı sürekli olarak yapılmalıdır.

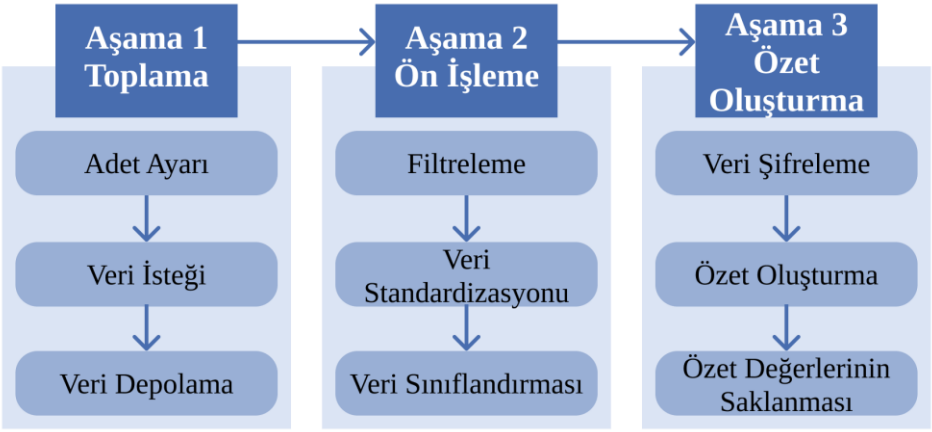


Şekil 6.3. Akıllı ev ağ geçidi için blok zincir tabanlı bulut özellikli mimarinin metodolojik akış şeması

6.3. Verileri Ağ Geçidi İçinde Ön İşleme

Akıllı evde bulunan heterojen IoT cihazlarından üretilen veriler, farklı veri türlerinden oluşan ve farklı boyutta olabilen ağ geçidine iletilir. Önerilen mimarinin ağ geçidi, IoT ağını doğru bir şekilde kontrol edebilmesi ve kullanıcının isteğine göre verileri işleyip üst katmana iletebilmesi gerekmektedir. IoT'den akıllı ev ağ geçidine veri aktarım sürecinde veri işleme 3 aşamaya ayrılır. Bunlar: "toplama", "ön işleme" ve "özet oluşturma" 'dır. Şekil 6.4.'te bu 3 kategoriye ait aşamalar verilmiştir.

- **Aşama 1 – Toplama:** Sensörlerden elde edilen veriler belirli bir adet boyunca iletilir. Ağ geçidinde yeni veriye ihtiyaç duyulduğunda veya bir işlem meydana geldiğinde sensörden veriler talep edilir. Daha sonra ham veriler gönderilir ve ağ geçidinde bulunan depolama aygıtında saklanır.
- **Aşama 2 – Ön İşleme:** Sensörlerden veya IoT cihazlarından gönderilen ham veriler, ağ geçidi içinde önceden işlenir. Depolama alanının verimliliği için, sensör veya cihaz kimliğine bağlı olarak yalnızca ihtiyaç duyulan veriler filtrelenir ve depolanır. Depolama işlemi standardizasyon ve sınıflandırmaya göre yapılır.
- **Aşama 3 – Özet oluşturma:** Akıllı evde üretilen veriler kullanıcıların hassas verilerini içerir. Blok zincir tabanlı akıllı evlerde bu hassas veriler, şifreleme yolu ile yönetilirler. Özet fonksiyonları sayesinde veriler şifrelenir ve saklanırlar.



Şekil 6.4. Ağ geçidi verileri ön işleme

6.4. Ağ Geçidi Cihazlarını Tanıma ve Veri Toplama

Akıllı evlerde yapılandırılan IoT cihazları veya sensörleri bir ağ geçidine bağlanır ve her elemanın eşsiz bir MAC (Media Access Control Address) adresi vardır. MAC adresi bir ağda bulunan her donanım cihazını tanımak için kullanılan kimliktir. Ağ geçitleri ve cihazlar sabit MAC adreslerine sahiptir ve genel anahtar ve SHA2 ile şifreleme ve kod çözme algoritmalarını çalıştırabilen bilgi işlem gücüne sahiptirler. Cihazlar ile ağ geçitleri arasında ara bağlantı protokolleri ve veri depolama işlem adımları Şekil 6.5.'te gösterildiği gibidir.

1-2-3. Bir ağ geçidine MAC adresleri tarafından tanımlanan sensörler, her işlemde doğrulanmalıdır. Algılama katmanındaki sensörler doğrudan ağ geçidine kaydedilir ve otomatik olarak ağ geçidine bağlanır. Ağ geçidi, bağlandığı sensörden bir kimlik alabilir veya bağlı sensörden ilgili ayrıntıları alabilir.

4. Bir sensöre MAC adresleri tarafından tanımlanan ağ geçitleri, her işlemde doğrulanmalıdır. Ağ geçidi bilgisi sensöre, sensör bilgisi de ağ geçidine kayıtlı olduğu için iletişimde sorun olmaz.

5-6. Ağ geçidi, doğru ve kayıtlı sensörlerin bağlandığından emin olmak için şifreyi çözer.

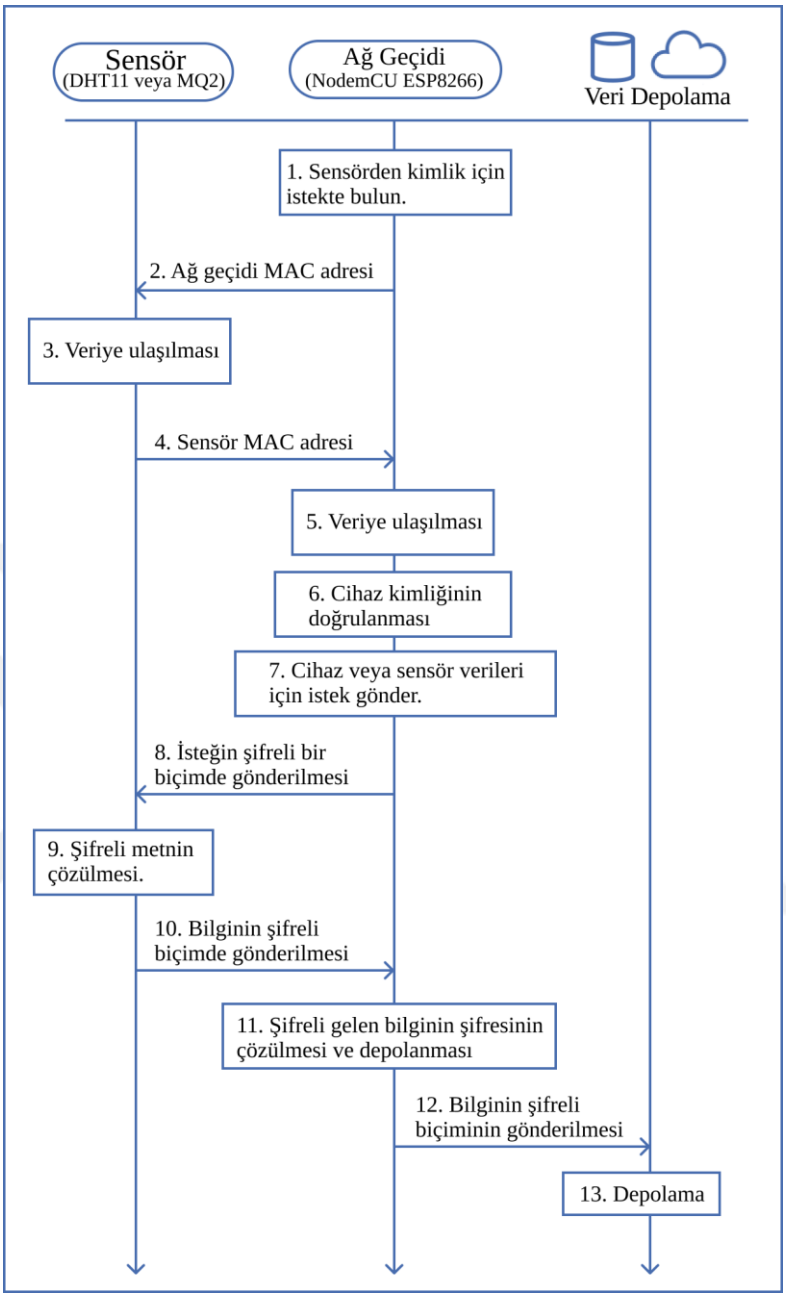
7. Ağ geçidi, sensörler tarafından üretilen verileri elde etmek için bir talep mesajı oluşturur ve bunu sensöre iletir.

8. Veri istek mesajları sensöre şifreli olarak gönderilir.

9-10. Şifrelenmiş veri istek mesajları sensörde çözülür, istek yerine getirilir ve bilgi ağ geçidine şifreli olarak gönderilir.

11-12. Ağ geçidi, alınan ham verilerin şifresini çözerek depolama birimlerine ve depolama katmanında bulunan bulut veya bulutlara iletir.

13. Veri ambarında sensörlerden alınan bilgiler şifreli olarak saklanmaktadır.



Şekil 6.5. Ağ geçidi cihazlarını tanıma ve veri toplama şeması

6.5. Akıllı Ev Ağ Geçidi Güvenlik Hususları

Tipik bir IoT ağında kaynak kısıtlamasına sahip cihazlar, sensör verilerini bulut veya sunucu bulunan üst katmanlara iletmekten sorumlu ağ geçidi ile iletişim kuran uç düğüm olarak kullanılır. Önerilen blok zincir tabanlı akıllı ev ağında, akıllı evi oluşturan birden fazla cihaz toplu olarak ağ geçidi üzerinden haberleşerek kontrol edilir ve izlenir. Bu şekilde olan ağ yapılandırmalarında evde olan veriler ifşa edilebilir, mahremiyet ihlallerine sebep olabilir ve cihaz arızaları oluşabilir ve kullanıcılar zarar görebilir.

Akıllı evler ve güvenlik standartlarının olmadığı ve heterojen cihazların bulunduğu bir ortamda cihazların bir araya getirilmesinde zorluklar yaşanmaktadır. Bu nedenle farklı hizmetler kullanıcılara sorunsuz bir şekilde sağlanamamaktadır. Ağ geçitleri için güvenlik önemli bir husustur. Akıllı evlerde bulunan ağ geçitleri için güvenlik gereksinimleri şu şekildedir:

- **Bütünlük:** Her cihaz ve ağ geçidi arasında veri alış-verişi gerçekleştiği zaman, herhangi bir yanlışlık meydana gelmemelidir. Özet fonksiyonundan geçirme işlemi, alış-verişi yapılan bu verilerin yanlış olma olasılığını azaltır. Aynı zamanda hangi verilerin istendiğinin ve kaydedildiğinin izlenebilirliğini sağlar.
- **Gizlilik:** Akıllı ev ağları, konut sakinlerinden hassas veriler de dahil olmak üzere birçok veriyi toplar ve depolar. Bu veriler yalnızca yetkili olan kullanıcılar tarafından erişilebilir olmalıdır. Akıllı evin özelliklerini gizli tutmak için, bir şifreleme algoritması ile bir anahtar kullanılmıştır.
- **Kimlik Doğrulama:** Akıllı ev ağında olan kimlik doğrulama işlemi, saldırganların dışarıdan kötü niyetli bir şekilde ağ içerisinde hareket etmesini engeller. Blok zincir teknolojisi üyelerin doğrulanmasında kullanılır.

6.6. Bulut Bilişime Genel Bakış

Büyük depolama odaları ve büyük miktarda elektrik kullanımı teknoloji tarihinde önemli bir rol oynamaktadır. Küçük boyutta ve daha verimli bilgisayarlar geçmişte olan büyük boyutlu ve verimsiz bilgisayarların yerini almıştır. Günümüzde verilere olan talep ve çevrimiçi kullanıcı sayısı önemli ölçüde artmıştır. Geleneksel bilgi işlem altyapısı pahalıdır ve yönetimi zordur. Ayrıca geleneksel bilgi işlem ile verilere her yerde ve her zaman erişim mümkün değildir ve artan çevrimiçi kullanıcı sayısını kaldıramaz. Küresel internet kullanımı önemli miktarda arttığı için ve hizmetlerin kullanım hacmi ile kullanılabilirliğinden dolayı bulut bilişim adı verilen yeni bir kavram ortaya çıkmıştır. Bulut bilişim, dünyanın herhangi bir yerinden veri depolamak, işlemek ve yönetmek için uzak sunuculardan oluşan bir ağı kullanmaktır. Yerel bir sunucu veya kişisel bilgisayar yerine kullanılır.

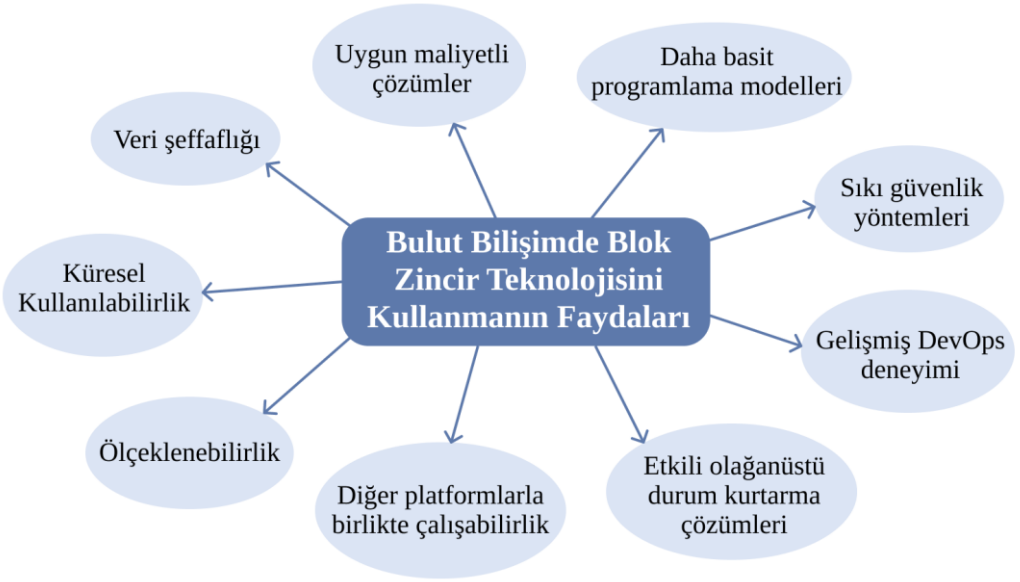
Bulut bilişim, son yıllarda artan taleplerle büyük ilgi görmüştür. Bulut tabanlı veri depolama çözümlerine yönelen kuruluşların çeşitli avantajları vardır. Bunlara örnek olarak; basitleştirilmiş bilişim teknolojileri alt yapısı ve yönetimi, kesintisiz internet bağlantısı ile dünyanın her yerinden etkili bir şekilde uzaktan erişim ve bulut bilişim sayesinde azalan hizmet maliyeti verilebilir. Hizmetler, herhangi bir yerde ağa dahil

olan bireyler arasında iş birliğini geliştiriyor. Yapılan herhangi bir değişiklikte bulutta bulunan yazılımlar otomatik olarak güncellenir. Bu sayede bulut kolaylıkla yönetilebilir.

Bulut bilişim, kaynakları yerel sunucular arasında paylaşarak güçlendirir. Bulutun yapısı dağıtım modeli ile ilişkilidir. Dağıtım modeli genel bulut, özel bulut ve hibrit bulut olmak üzere üç farklı tür içerir (Bhamare, Samaka ve ark., 2017). Özel bulutta, kuruluşun veri merkezi önemlidir. Bu modelde altyapı aynı kuruluşa aittir ve aynı kuruluş tarafından kullanılır. Kuruluşun veri merkezinde fiziksel olarak bulunabilir veya üçüncü taraf olan hizmet sağlayıcı tarafından bulundurulabilir. Güvenlik riski tespiti daha kolaydır. Genel bulutta bulut kaynakları, üçüncü taraf olan bulut hizmet sağlayıcılarına aittir. Yazılım, donanım ve diğer altyapılar hizmet sağlayıcısına aittir. Bu modelde kuruluşlar internet üzerinden açık erişim sağlayabilirler. Kaynakları çeşitli saldırılara karşı korumak zordur. Üçüncü dağıtım modeli olan hibrit bulut avantajlar bakımından diğer dağıtım modellerine göre daha avantajlıdır. Genel bulut hizmeti ve şirket içi özel bulutun kombinasyonunu oluşturur. Birbirine bağlı iki ortam arasında veri alışverişi sağlanabilir. Şirketin istediği esnekliği veri dağıtımında sağlar. Gizli olan veriler güvenliğin sağlanması amacıyla özel bulutta, diğer veriler ise verimli kullanım ve depolama için genel bulutta saklanabilir.

6.6.1. Bulut bilişim ve blok zincir entegrasyonu

Blok zincir teknolojisi, sektör alanlarını farklı hizmetlerle beraber kullanmaya teşvik etmiştir. Bulut bilişim ve blok zincir teknolojisinin entegre kullanımında çeşitli faydalar vardır. Bulut bilişimde veriler merkezi bir sunucuda tutulur, bulut bilişimde blok zincir teknolojisi kabul edilerek merkeziyetsizlik sağlanabilir. Blok zincirde sistemde arıza oluşma olasılığına karşı aynı bilgilerin farklı kopyaları çok sayıda bulut merkezine konulur. Ayrıca, farklı merkezlerde bilgilerin çok sayıda kopyası tutulduğu için bilgi eksikliği konusunda bir problem yaşanmaz. Bulut bilişim ile blok zincirin entegre kullanımı kullanıcılara tam gizlilik sunabilir. Çok yönlü ve yönetilebilir sistem sağlanabilir. Bulut bilişim sayesinde uzaktan izlenebilirlik özelliği sağlanabilir. Bulut depolama blok zincir teknolojisinin kullanılması, korumayı artırabilir ve bir kuruluşun güvenlik açıklarını azaltabilir. Bulut bilişim ve blok zincir teknolojileri sağlık (Mamdouh, Awad ve ark., 2021), tedarik zinciri (Mehta, Tanwar ve ark., 2021), finans (Xiaoping, Tao ve ark., 2021), akıllı şehirler (Samuel, Javaid ve ark., 2022) ve tarım (Demestichas, Peppes ve ark., 2020) olarak birçok farklı alanda entegre olarak kullanılmaktadır.



Şekil 6.6. Blok zincirin bulut bilişime sağladığı faydalar

Bulutta depolanan veriler çok yapılandırılmamış bir şekildedir. Blok zincirinde depolanan veriler çok yapılandırılmış bir şekildedir. Veriler, her blok için oluşturulan hash anahtarı kullanılarak izlenebilir. Her blok, önceki bloğun hash anahtarını içerir ve bu, ağı takip etmenin anahtarıdır. Bloktaki veriler doğrulanır ve ağda bulunan düğümler tarafından erişilebilir. Bulut, esnekliği destekler ve gerektiğinde hesaplama yüklerindeki dalgalanmaların üstesinden gelebilir. Blok zincir ayrıca kullanıcının anonimliğini sağlar ve kullanıcının bilgilerine üçüncü şahısların erişimini önlemek için kullanıcının kaydı sistemden güvenli bir şekilde kaldırılabilir. Bulutun blok zincir ile entegrasyonu aynı zamanda birçok işletmenin güven duymasını sağlayacak ve isteğe bağlı bir hizmet haline gelecektir. Blok zincir, veri şeffaflığı ve yetkilendirme gibi bulut bilişime birçok temel avantaj sağlar ve ayrıca uygun maliyetli çözümler sunar. Blok zincirin bulut hizmetleri ile kullanımında elde edilen ana faydalardan bazıları Şekil 6.6.'da gösterilmektedir. Blok zincir platformunu benimseyerek buluttaki verilerin daha güvenli olacağı sonucuna varılabilir (Murthy, Shri ve ark., 2020).

6.7. Güvenlik Değerlendirmesi

IoT, araştırma ve endüstride üstel bir büyüme yaşamaktadır, ancak yine de gizlilik ve güvenlik açıklarından muzdariptir. Bitcoin kripto para biriminin temelini oluşturan blok zincir, son zamanlarda IoT'ye benzer topolojilere sahip eşler arası ağlarda güvenlik ve gizlilik sağlamak için kullanılmıştır.

Sadece blok zincire kayıtlı ağ geçitleri veri gönderebilmelidir, ayrıca sadece ağ geçitlerine kayıtlı sensörler veya cihazlar veri gönderebilmelidir. Güvenilir olmalıdır, veriler değiştirilememelidir. Blok zincir teknolojisi merkezi olmayan dağıtık bir veri tabanıdır. Dolayısıyla merkezi olarak yönetilmez, veriler birden fazla veri tabanında saklanır. Blok zincirde bloklar, bir önceki blok başlığının özet değerini tutarak oluşturulurlar. Bu durum, yeni blok ekleneceği zaman blok zincirde hiçbir verinin değiştirilememesini sağlar. Yapılacak herhangi bir değişiklikte, zincir bozulur ve madenci veya madenciler tarafından kabul edilmez. Özet fonksiyonundan geçen her veride yapılacak en ufak bir değişiklik, verinin özetinde büyük bir değişikliğe sebep olmaktadır. Buda zincir yapısından değişikliğe sebep olur ve kabul edilemez durum ortaya çıkmış olur.

Her veri anonim olmayarak kaydedilir. Veriyi isteyen kullanıcı kim ise ona göre blok zincire kaydedilir.

Saldırılara karşı dayanıklı olmalıdır. Mevcut akıllı ev ağ geçitlerine yönelik saldırı teknikleri, ortam ağı ve IoT değiştikçe sürekli olarak değişmektedir. Özellikle IoT cihazları sınırlı bilgi işlem gücüne ve pil kapasitesine sahiptir. Bu ağ ortamında saldırgan, hedef cihaza göre çeşitli saldırı senaryoları belirleyebilir.

- **Hizmet Reddi Saldırısı (Denial-of-Service Attack (DoS)):** DoS saldırıları, sistem kullanılabilirliğini azaltarak yasal kullanıcıların sisteme erişimini engellemeye yönelik açık girişimlerdir. Örneğin, bir sistemin elektrik gücünün kasıtlı olarak kesilmesini fiziksel bir DoS saldırısı olarak değerlendirebiliriz. Bir DoS saldırısında, saldırgan kullanıcının hizmete veya verilere ulaşmasını engellemeye çalışır. Blok zincirde, bir saldırgan ağa sahte işlemler ve bloklar göndererek saldırıyı başlatabilir. Kullanıcılar tarafından gerçekleştirilen belirli bir dizi işlevin normal özelliklerini değiştirme, bunları kullanılamaz ve tamamen arızalı hale getirme fikri etrafında döner, böylece kullanıcı artık bu sistemin hizmetlerini veya işlevlerini kullanamaz ancak blok zincir de asimetrik şifreleme yönteminde kullanılan genel ve özel anahtar çifti sayesinde saldırının etkisi azaltılır.
- **Modifikasyon Saldırısı (Modification Attack):** Bu saldırıyı başlatmak için, düşmanın bulut depolama güvenliğini tehlikeye atması gerekir. Saldırgan daha sonra belirli bir kullanıcı için saklanan verileri değiştirmeye veya silmeye çalışabilir. Kullanıcı, buluttaki verilerin özet değerini kendi yerel blok zincirinde

saklanan özet değeri ile karşılaştırarak depolanan verilerindeki herhangi bir değişiklik varsa tespit edebilecektir.

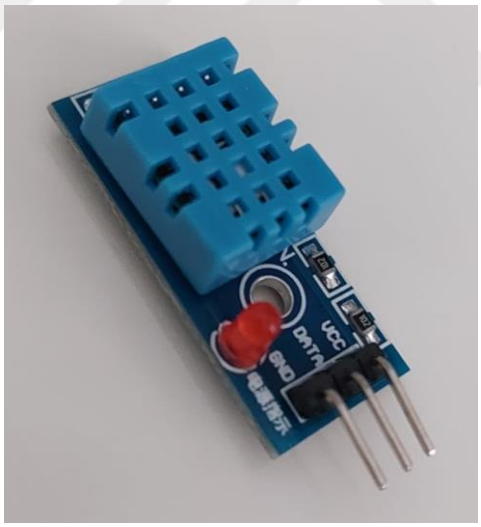
- **Çıkarım Saldırısı (Inference Attack):** Çıkarım saldırısı başlatmanın amacı, kullanıcılar hakkında yararlı bilgiler toplamaktır. Örneğin, çıkarım saldırısı, kullanıcının günlük rutini ve alışkanlıkları ile ilgili faaliyetleri hakkında veri elde etmeyi amaçlar. Önemsiz gibi görünen bu veriler, saldırganın oda sıcaklığı verileri gibi saldırıya uğrayabilecek ve değiştirilebilecek diğer önemli ayrıntıları bulması açısından oldukça faydalıdır. Evin sıcaklık verileri sayesinde, saldırgan kullanıcının nerede olduğu, hangi odada bulunduğu çıkarımını yapabilir. Bu nedenle araştırmacılar, çıkarım saldırılarının esas olarak veri madenciliği uygulamalarına ve tekniklerine dayandığını söylerler (Torre, Koceva ve ark., 2016).
- **Blok Zincir %51 Saldırısı:** Blok zincirde bir saldırganın, madencilik ve bilgi işlem gücünün %50'den fazlasını kontrol ettiği durumlardır. Tüm ağın %50'den fazlasını kontrol etmesi anlamına gelir. Saldırgan, yeni işlemlerin onaylanmasını engelleyebilir. Ancak önerilen blok zincir tabanlı mimaride %51 saldırısının başarılı olabilmesi için blok zincir ağına katılan tüm düğümlerin özet değeri gücünün, özet değeri hesaplama gücünün toplamından büyük olması gerekmektedir. Bu nedenle, önerilen blok zinciri tabanlı mimariye blok zinciri %51 saldırısı imkânsızdır.

7. AKILLI EV PROTOTİPİ

7.1. Akıllı Ev Prototipinde Kullanılan Sensörler, Donanım Kartları ve Özellikleri

Bu bölümde akıllı ev prototipinde kullanılan sensörler ve sensörlerin haberleşmesinde kullanılan donanımsal kartların özelliklerine yer verilmiştir.

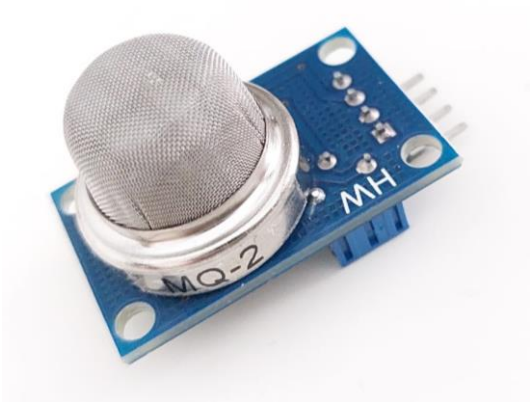
- **DHT11 Sıcaklık ve Nem Sensörü:** Bu modül, kalibre edilmiş bir dijital sinyal çıkışına sahip bir nem ve sıcaklık kompleksine sahiptir, yani DHT11 sensör modülü, kalibre edilmiş bir dijital çıkış sinyali veren nem ve sıcaklığı algılamak için birleşik bir modüldür. DHT11'in güç tüketimi oldukça düşüktür, 5 V güç kaynağı voltajı ve maksimum ortalama akım yaklaşık 0,5 mA'dır. DHT11 bize çok hassas nem ve sıcaklık değeri verir ve yüksek güvenilirlik ve uzun vadeli kararlılık sağlar. Bu sensör, hızlı yanıt veren, uygun maliyetli ve 4 pinli tek sıralı pakette bulunan dahili 8 bitlik bir mikro denetleyiciye sahip dirençli tip nem ölçüm bileşenine sahiptir. DHT11 modülü, seri iletişim yani tek telli iletişim üzerinde çalışır. Şekil 7.1.'de DHT11 sıcaklık ve nem sensör modülüne yer verilmiştir.



Şekil 7.1. DHT11 Sıcaklık ve Nem Sensör Modülü

- **MQ-2 Gaz Sensörü:** Çeşitli gaz türleri vardır ve modül tarafından tanımlanabilen gaz sensörleri de değişiklik gösterir. Ev gazlarını algılamak için Metan, Bütan, LPG ve Sigara Dumanını algılamak için MQ gaz sensörleri kullanılır. Düşük maliyetlidir ve farklı uygulamalar için uygundur. Gaz sensörünün çalışma şekli, havadaki gaz içeriğine duyarlı bir elektronik devre kullanmaktır, bu nedenle havadaki gaz konsantrasyonu arttıkça ortaya çıkan

voltaj da yükselir. Uzun bir çalışma ömrü ve kararlılığa sahiptir. Şekil 7.2.'de MQ-2 gaz sensör modülüne yer verilmiştir.



Şekil 7.2. MQ-2 Gaz Sensör Modülü

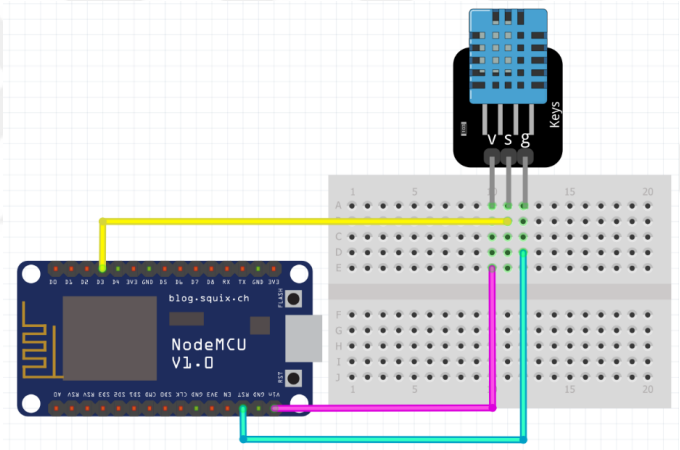
- **NodeMCU ESP8266:** NodeMCU açık kaynaklı bir platformdur, donanım tasarımını düzenlemeye, değiştirmeye ve oluşturmaya açıktır. NodeMCU geliştirme kartı, ESP8266 wifi özellikli çipten oluşur. ESP8266, Espressif Systems tarafından TCP/IP protokolü ile geliştirilmiş düşük maliyetli bir Wi-Fi çipidir. NodeMCU, Arduino benzeri bir cihazdır. Ana bileşeni ESP8266'dır. Programlanabilir pinlere sahiptir. WiFi yerleşiktir. Mikro-usb portu üzerinden güç alabilir. Maliyeti düşüktür. Birden çok programlama ortamı aracılığıyla programlanabilir. NodeMCU ESP8266'ya giden güç, yerleşik Mikro USB konektörü aracılığıyla sağlanır. ESP8266 NodeMCU, çeşitli geliştirme platformları kullanılarak programlanabilir. Ürün boyutları, 2.9 x 5.7 cm'dir. Şekil 7.3.'te NodeMCU ESP8266 Wifi modülüne yer verilmiştir.

Şekil 7.3. NodeMCU ESP8266 Wifi Modülü

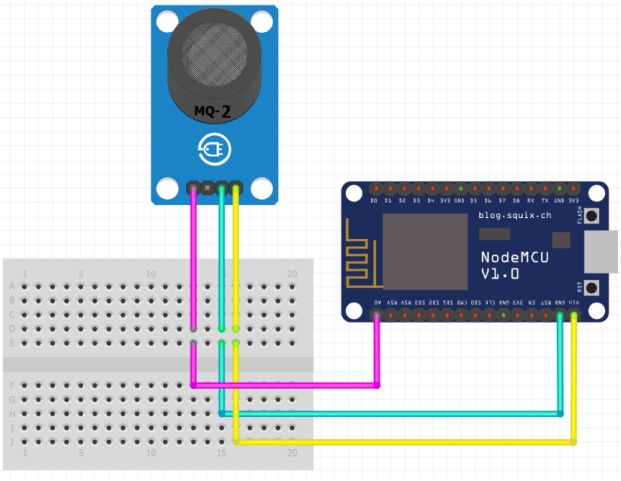
7.2. Donanım Şeması

Bu bölümde akıllı ev prototipinde sensörlerin ve donanımsal kartların beraber nasıl kullanıldıklarına ve devre şemasına yer verilmiştir.

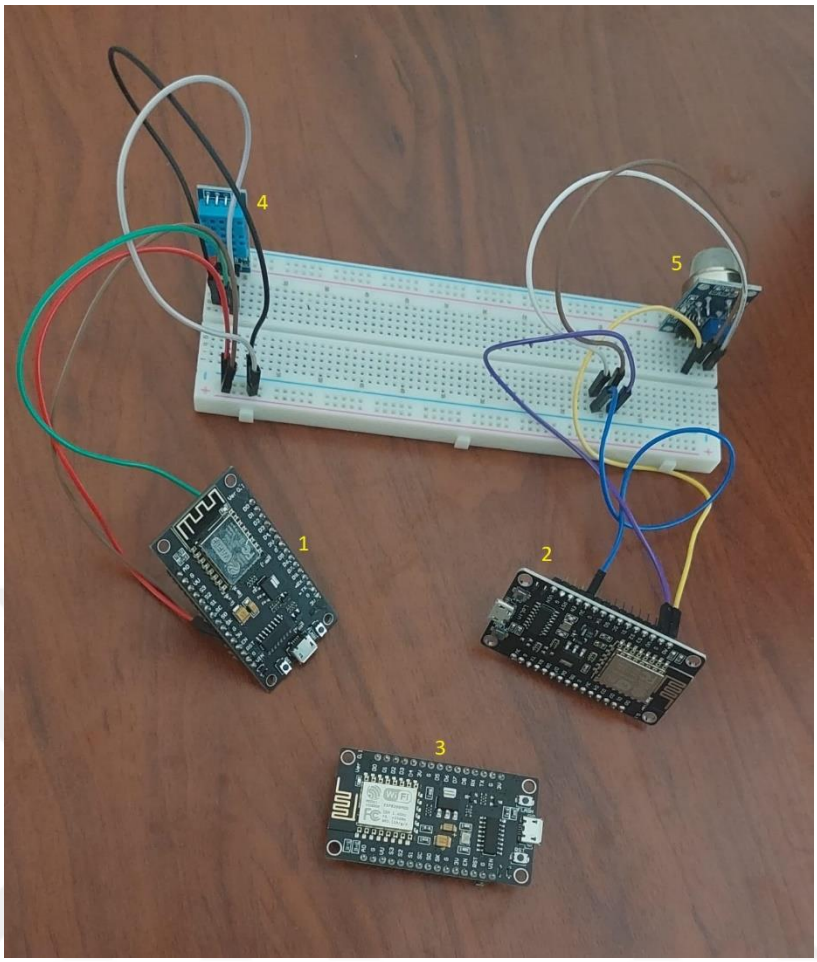
Şekil 7.4.'te MQ-2 gaz sensörü ve NodeMCU ESP8266 kartının bağlantı şeması verilmiştir. Şekil 7.5.'te DHT11 sıcaklık ve nem sensörünün NodeMCU ESP8266 kartı ile nasıl bağlandığıyla ilgili bağlantı şeması verilmiştir. Şekil 7.6'da DHT11 sıcaklık ve nem sensörü, MQ-2 gaz sensörü ve 3 adet NodeMCU kartı bulunmaktadır. 1 numaralı NodeMCU kartı ile 4 numaralı DHT11 sıcaklık ve nem sensörü, Şekil 7.5.'te verilen bağlantı şemasında olduğu gibi birbirine bağlıdır. 2 numaralı NodeMCU kartı ile 5 numaralı MQ2 gaz sensörü Şekil 7.5.'te bulunan bağlantı şemasında olduğu gibi birbirine bağlıdır. 3 numaralı NodeMCU kartı ise ağ geçidi olarak kullanılmıştır. Arduino yazılım editörü kullanılarak kodlama yapılmıştır ve MAC adresleri yardımı ile sensör ve kartların haberleşmesi sağlanmıştır.



Şekil 7.4. NodeMCU ESP8266 ve DHT11 sensör bağlantısı



Şekil 7.5. NodeMCU ESP8266 ve MQ-2 sensör bağlantısı



Şekil 7.6. NodeMCU ESP8266 kartı ile DHT11 ve MQ-2 sensör bağlantısı ve ağ geçidi olarak NodeMCU ESP8266 kartı

7.3. MQ-2 Gaz Sensörü Çalışma Mantığı ve Kodları

Şekil 7.7.'de verilen kod parçasında MQ-2 sensörü ve bağlı olduğu NodeMCU ESP8266 kartının kodlarına yer verilmiştir. NodeMCU ESP8266 geliştirme kartı Arduino IDE'si üzerinden kodlanabilmektedir ve bu tez çalışmasında Arduino IDE'si tercih edilmiştir. 1, 2 ve 3 numaralı kod satırlarında gerekli kütüphaneler tanımlanmış olup ardından kodda kullanılacak olan değişkenler tanımlanmıştır. Gaz sensöründen ağ geçidine veriler MAC adresi yolu ile aktarılacağı için ağ geçidi olan NodeMCU ESP8266 kartının MAC adresi de tanımlanmıştır. 12-16 kod satır aralığında bir struct tanımlanmıştır. Bunun sebebi aktarılacak verilerin belirli bir veri yapısında aktarılması gerektiğidir. 18. ve 19. kod satırlarında ise oluşturulan struct'tan değişken üretilmiş olup iki adettir. Bunun sebebi gönderilecek verilerin ve gelen verilerin farklı değişkenler üzerinde tutulması gerektiğidir. 21 ve 29 kod satırı aralığında ise "VerilerGonderildiginde()" fonksiyonu tanımlanmıştır. Bunun sebebi ise sensörden ağ geçidine verilerin sağlıklı bir biçimde aktarılıp aktarılmadığı bilgisinin öğrenilmesidir. 31 ve 36 kod satırı aralığında "VerilerAlindiginda()" fonksiyonu tanımlanmıştır. Bu

fonksiyonun işlevi gaz sensör verisinden veriler alındığı zaman ağ geçidine veriler aktarılır. 38 ve 48 kod satırı aralığında bulunan “sensorOku()” fonksiyonunun işlevi ise MQ-2 gaz sensöründen verileri okumaktır. 55. kod satırında “setup()” kod bloğu başlar. Program başladığı anda yalnızca bir kere çalışan bu kod bloğunda haberleşme başlatılır ve ağ geçidi ile bağlantı kurulur. 65 ve 69 kod satırı aralığında ağ geçidi olan NodeMCU ESP8266 kartı ile MAC adresi sayesinde bağlantı kurulur ve gerekli veriler yazılan fonksiyonlar kullanılarak aktarılır. 72. kod satırında “loop()” kod bloğu başlar. Bu kod bloğu bir döngüdür. Sensörden sürekli verilerin okunması, verilerin şifrelenmesi ve ağ geçidine aktarılması işlemleri yürütülür. Bu şekilde bir cihazdan diğer cihaza istek mesajı gönderilmektedir, ardından isteği alan cihaz sensörden okuduğu verileri isteği yapan cihaza göndermektedir.

```
#include <base64.hpp>
#include <ESP8266WiFi.h>
#include <espnow.h>

int gazDeger;
uint8_t alici_macadresi[] = {0x84, 0xCC, 0xA8, 0xB0, 0x1A, 0x48};
bool alinanistekmesaji=false;
char normal_text[3];
char base64_text[3];
String str;

typedef struct struct_messageGaz {
    bool istekgonder;
    String gaz;
    bool cevap;
} struct_messageGaz;

struct_messageGaz Gazverileri;
struct_messageGaz gelenveriler;

void VerilerGonderildiginde(uint8_t *mac_addr, uint8_t sendStatus) {
  Serial.print("Son verinin gonderilme durumu: ");
  if (sendStatus == 0){
    Serial.println("Veri Gonderme Başarılı");
  }
  else{
    Serial.println("Veriler Gönderilmedi!!!");
  }
}

void VerilerAlindiginda(uint8_t * mac, uint8_t *incomingData, uint8_t len) {
  memcpy(&gelenveriler, incomingData, sizeof(gelenveriler));
  Serial.print("Alınan veri boyutu: ");
  Serial.println(len);
  alinanistekmesaji = gelenveriler.istekgonder;
}

void sensorOku(){
  gazDeger=analogRead(A0);
  if (isnan(gazDeger)){
    Serial.println("Gaz sensör gaz verisi okunamadı!!!");
    gazDeger = 0.0;
  }
  str=String(gazDeger);
  str.toCharArray(normal_text,3);
  encode_base64((unsigned char *) normal_text,3,(unsigned char *) base64_text);
  Serial.print("Base64 Text: ");Serial.println(base64_text);
}

void gelenverileriyazdir(){
  Serial.println(" Gelen mesaj: ");
  Serial.println(alinanistekmesaji);
}

void setup() {
  Serial.begin(9600);
  delay(100);
  WiFi.mode(WIFI_STA);
  WiFi.disconnect();
  if (esp_now_init() != 0) {
    Serial.println("ESP-NOW Başlatılamadı !!!");
    return;
  }

  esp_now_set_self_role(ESP_NOW_ROLE_COMBO);
  esp_now_register_send_cb(VerilerGonderildiginde);
  esp_now_add_peer(alici_macadresi, ESP_NOW_ROLE_COMBO, 1, NULL, 0);
  esp_now_register_recv_cb(VerilerAlindiginda);
  delay(2000);
}

void loop() {
  sensorOku();
  delay(100);
  gelenverileriyazdir();
  if (alinanistekmesaji) {
    Gazverileri.gaz = (String)base64_text;
    esp_now_send(alici_macadresi, (uint8_t *) &Gazverileri, sizeof(Gazverileri));
   alinanistekmesaji=false;
  }
  delay(10000);
}
```

Şekil 7.7. MQ-2 gaz sensörü ve bağlı olduğu NodeMCU ESP8266 kartı kodları

7.4. DHT11 Sıcaklık ve Nem Sensörü Çalışma Mantığı ve Kodları

Şekil 7.8.’de verilen kodlar DHT11 sıcaklık ve nem sensörünün bağlı olduğu NodeMCU ESP8266 kartına ait kodlardır. Bu kodlar DHT11 sensöründen verileri alır ve ağ geçidine aktarır. 1 ve 4 numaralı kod satır aralığında uygulama için gerekli kütüphaneler tanımlanmıştır. 6. satırda ağ geçidi olan NodeMCU ESP8266 kartına ait MAC adresi girilmiştir. 8-19 kod satır aralığında gerekli değişkenler tanımlanmıştır. 21 ve 26 kod satır aralığında struct tanımlanmıştır. Bu struct sayesinde veriler belirli bir

yapıda aktarılır. Ağ geçidinde ise gelen veriler struct içerisinden kullanılır. 48 ve 73 numaralı kod satır aralığında DHT11 sensöründen veriler okunmuştur ve gerekli değişkenlere şifreli bir biçimde aktarılmıştır. DHT11 sensörü hem sıcaklık hem nem verilerini verdiği için iki veri için ayrı ayrı işlemler uygulanmıştır. 80. kod satırında "setup()" kod bloğu başlamıştır ve bu bölümde ağ geçidi ile haberleşme başlatılmıştır. Gerekli ayarlamalar bu bölümde yapılmıştır. 97. satırda başlayan "loop()" kısmında ise verilerin ağ geçidine gönderimi gerçekleşmiştir. Bu kısım bölüm 7.3.'te bahsedildiği üzere bir döngüdür ve tekrarlanır. Dolayısıyla veriler sürekli aktarım halindedir.

```
#include <base64.hpp>
#include <DHT.h>
#include <ESP8266WiFi.h>
#include <espnow.h>

uint8_t alici_macadresi[] = {0x84, 0xCC, 0xA8, 0xB0, 0x1A, 0x48};

#define DHTPIN D4
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);
float temperature;
float humidity;
bool alinanistekmesaji=false;
char normal_text[3];
char base64_text[3];
String str;
char normal_text1[3];
char base64_text1[3];
String str1;

typedef struct struct_message {
    bool istekgonder;
    String temp;
    String hum;
    bool cevap;
} struct_message;

struct_message DHTverileri;
struct_message gelenveriler;

void VerilerGonderildiginde(uint8_t *mac_addr, uint8_t sendStatus) {
  Serial.print("Son verinin gonderilme durumu: ");
  if (sendStatus == 0){
    Serial.println("Veri Gonderme Başarılı");
  }
  else{
    Serial.println("Veriler Gönderilmedi!!!");
  }
}

void VerilerAlindiginda(uint8_t * mac, uint8_t *incomingData, uint8_t len) {
  memcpy(&gelenveriler, incomingData, sizeof(gelenveriler));
  Serial.print("Alınan veri boyutu: ");
  Serial.println(len);
  alinanistekmesaji = gelenveriler.istekgonder;
}

void sensorOku(){
  temperature = dht.readTemperature();
  if (isnan(temperature)){
    Serial.println("DHT sensör sıcaklık verisi okunamadı!!!");
    temperature = 0.0;
  }
  Serial.print("Okunan Sıcaklık:");
  Serial.println(temperature);
  Serial.print("Okunan Nem:");
  Serial.println(humidity);
  if (isnan(humidity)){
    Serial.println("DHT sensör nem verisi okunamadı!!!");
    humidity = 0.0;
  }

  humidity = dht.readHumidity();
  str=String(temperature);
  str.toCharArray(normal_text,3);
  encode_base64((unsigned char *) normal_text,3,(unsigned char *) base64_text);
  Serial.print("Base64 Text: ");Serial.println(base64_text);
  str1=String(humidity);
  str1.toCharArray(normal_text1,3);
  encode_base64((unsigned char *) normal_text1,3,(unsigned char *) base64_text1)
  Serial.print("Base64 Text: ");Serial.println(base64_text1);
}

void gelenverileriyazdir(){
  Serial.println(" Gelen mesaj: ");
  Serial.println(alinanistekmesaji);
}

void setup() {
  Serial.begin(9600);
  dht.begin();
  WiFi.mode(WIFI_STA);
  WiFi.disconnect();
  if (esp_now_init() != 0) {
    Serial.println("ESP-NOW Başlatılamadı !!!");
    return;
  }

  esp_now_set_self_role(ESP_NOW_ROLE_COMBO);
  esp_now_register_send_cb(VerilerGonderildiginde);
  esp_now_add_peer(alici_macadresi, ESP_NOW_ROLE_COMBO, 1, NULL, 0);
  esp_now_register_recv_cb(VerilerAlindiginda);
  delay(2000);
}

void loop() {
  sensorOku();
  delay(100);
  gelenverileriyazdir();
  if (alinanistekmesaji) {
    DHTverileri.temp = (String)base64_text;
    DHTverileri.hum = (String)base64_text1;
    DHTverileri.cevap=1;
    esp_now_send(alici_macadresi, (uint8_t *) &DHTverileri, sizeof(DHTverileri));
   alinanistekmesaji=false;
  }
  Serial.println(temperature);
  delay(10000);
}
```

Şekil 7.8. DHT11 sıcaklık ve nem sensörü ve bağlı olduğu NodeMCU ESP8266 kartı kodları

7.5. NodeMCU ESP8266 Ağ Geçidi Çalışma Mantığı ve Kodları

Akıllı ev prototipinde ağ geçidi önemli bir rol oynamaktadır. Ağ geçidi, verilerin toplandığı ve C# programlama diline aktarıldığı ve dolayısıyla blok zincire aktarıldığı yerdir. Şekil 7.9.'da ağ geçidine ait kodlara yer verilmiştir. Ağ geçidine 2 adet NodeMCU ESP8266 kartından veri gelir. Bunlardan biri DHT11 sensörü, diğer ise MQ-2 sensörüdür. Dolayısıyla aktarma işlemleri hem DHT11 hem de MQ-2 sensörü için

yapılacaktır. Kodlarda öncelikler gerekli kütüphaneler tanımlanmıştır. Daha sonra kodlarda lazım olacak değişkenler tanımlanmıştır ve gerekli değişkenlere ilk değer ataması yapılmıştır. 18 ve 29 kod satır aralığında struct yapısı tanımlanmıştır. Bu yapılara gelen değerler aktarılır. Bundan dolayı DHT11 ve MQ-2 sensör kodlamalarında kullanılan struct yapısı ile aynı olmak zorundadır. 31. ve 32. kod satırlarında DHT11 ve MQ-2 sensörlerinin bağlı bulunduğu, haberleşmeleri sağlayan NodeMCU ESP8266 kartlarının MAC adresleri tanımlanmıştır. Veriler gönderildiğinde ve alındığında çalışacak fonksiyonlar hem sıcaklık ve nem sensörü için hem de gaz sensörü için yazılmıştır. “setup()” kısmında haberleşme başlatılmıştır. “loop()” kısmında ise isteğin gönderilmesi ve verilerin alınması işlemleri gerçekleştirilmiştir. Ağ geçidi kodlarında önemli olan “Serial.println()” kodudur. C# programlama dilinde blok zincir kodlanmıştır ve Arduino IDE’den veriler çekilirken “Serial.println()” kodu ile seri ekrana yazdırılan değerler çekilir. Bu yüzden seri ekrana yazdırılan değerler önemlidir.

```
#include <base64.hpp>
#include <ESP8266WiFi.h>
#include <espnow.h>

String gelenTemp;
String gelenHum;
bool gelencevap;
bool istekiletildi=1;
const long bekleme = 5000;
unsigned long oncekiMillis = 0;

String gelenGaz;
bool gelencevapGaz;
bool istekiletildiGaz=1;
const long beklemeGaz = 5000;
unsigned long oncekiMillisGaz = 0;

typedef struct struct_message {
    bool istekgonder;
    String temp;
    String hum;
    bool cevap;
} struct_message;

typedef struct struct_messageGaz {
    bool istekgonder;
    String gaz;
    bool cevap;
} struct_messageGaz;

uint8_t alici_macadresi[] = {0xA4, 0xCF, 0x12, 0xDA, 0x26, 0xE1};
uint8_t alici_macadresiGaz[] = {0x5C, 0xCF, 0x7F, 0x87, 0xE5, 0x48};

struct_message istekyolla;
struct_messageGaz istekyollagaz;
struct_message gelenveriler;
struct_messageGaz gelenverilergaz;

void VerilerGonderildiginde(uint8_t *mac_addr, uint8_t sendStatus) {
  if (sendStatus == 0){
    istekiletildi=1;
  }
  else{
    istekiletildi=0;
  }
}

void VerilerGonderildigindeGaz(uint8_t *mac_addr1, uint8_t sendStatus1) {
  if (sendStatus1 == 0){
    istekiletildiGaz=1;
  }
  else{
    istekiletildiGaz=0;
  }
}

void VerilerAlindiginda(uint8_t * mac, uint8_t *incomingData, uint8_t len) {
  memcpy(&gelenveriler, incomingData, sizeof(gelenveriler));
  gelenTemp = gelenveriler.temp;
  gelenHum = gelenveriler.hum;
  gelencevap = gelenveriler.cevap;

}

void VerilerAlindigindaGaz(uint8_t * mac1, uint8_t *incomingData1, uint8_t len1) {
  memcpy(&gelenverilergaz, incomingData1, sizeof(gelenverilergaz));
  gelenGaz = gelenverilergaz.gaz;
  gelencevapGaz = gelenverilergaz.cevap;
}

void gelenverileriyazdir(){
  Serial.print(gelenTemp);
  Serial.print("/");
  Serial.print(gelenHum);
  Serial.print("/");
}

void gelenverileriyazdirGaz(){
  Serial.println(gelenGaz);
}

void setup() {
  Serial.begin(9600);
  WiFi.mode(WIFI_STA);
  WiFi.disconnect();
  if (esp_now_init() != 0) {
    Serial.println("ESP NOW Başaltılamadı!!!");
    return;
  }
  delay(2000);
}



void loop() {
  istekyolla.istekgonder = 1;
  istekyollagaz.istekgonder=1;
  oncekiMillis=millis();
  oncekiMillisGaz=millis();

  esp_now_set_self_role(ESP_NOW_ROLE_COMBO);
  esp_now_register_send_cb(VerilerGonderildiginde);
  esp_now_add_peer(alici_macadresi, ESP_NOW_ROLE_COMBO, 1, NULL, 0);
  esp_now_register_recv_cb(VerilerAlindiginda);
  esp_now_send(alici_macadresi, (uint8_t *) &istekyolla, sizeof(istekyolla));
  delay(10000);

  if (gelencevap==true){

    unsigned long simdikiMillis = millis();
    if (simdikiMillis - oncekiMillis >= bekleme) {
      gelencevap=false;
      oncekiMillis = simdikiMillis;
    }
   gelenverileriyazdir();
  }
  else{
      if (istekiletildi==0){
          istekiletildi=1;
          delay(2000);
      }
  }

  esp_now_send(alici_macadresiGaz, (uint8_t *) &istekyollagaz, sizeof(istekyollagaz));
  esp_now_set_self_role(ESP_NOW_ROLE_COMBO);
  esp_now_register_send_cb(VerilerGonderildigindeGaz);
  esp_now_add_peer(alici_macadresiGaz, ESP_NOW_ROLE_COMBO, 1, NULL, 0);
  esp_now_register_recv_cb(VerilerAlindigindaGaz);
  gelenverileriyazdirGaz();
  delay(10000);
}
```

Şekil 7.9. NodeMCU ESP8266 ağ geçidi kodları

7.6. Blok Zincir Kodları

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BlockchainCoding
{
    11 başvuru
    public class Transaction
    {
        1 başvuru
        public string Temperature { get; set; }
        1 başvuru
        public string Humidty { get; set; }
        1 başvuru
        public string Gaz { get; set; }

        3 başvuru
        public Transaction(string temperature, string humidty, string gaz)
        {
            Temperature = temperature;
            Humidty = humidty;
            Gaz = gaz;
        }
    }
}
```

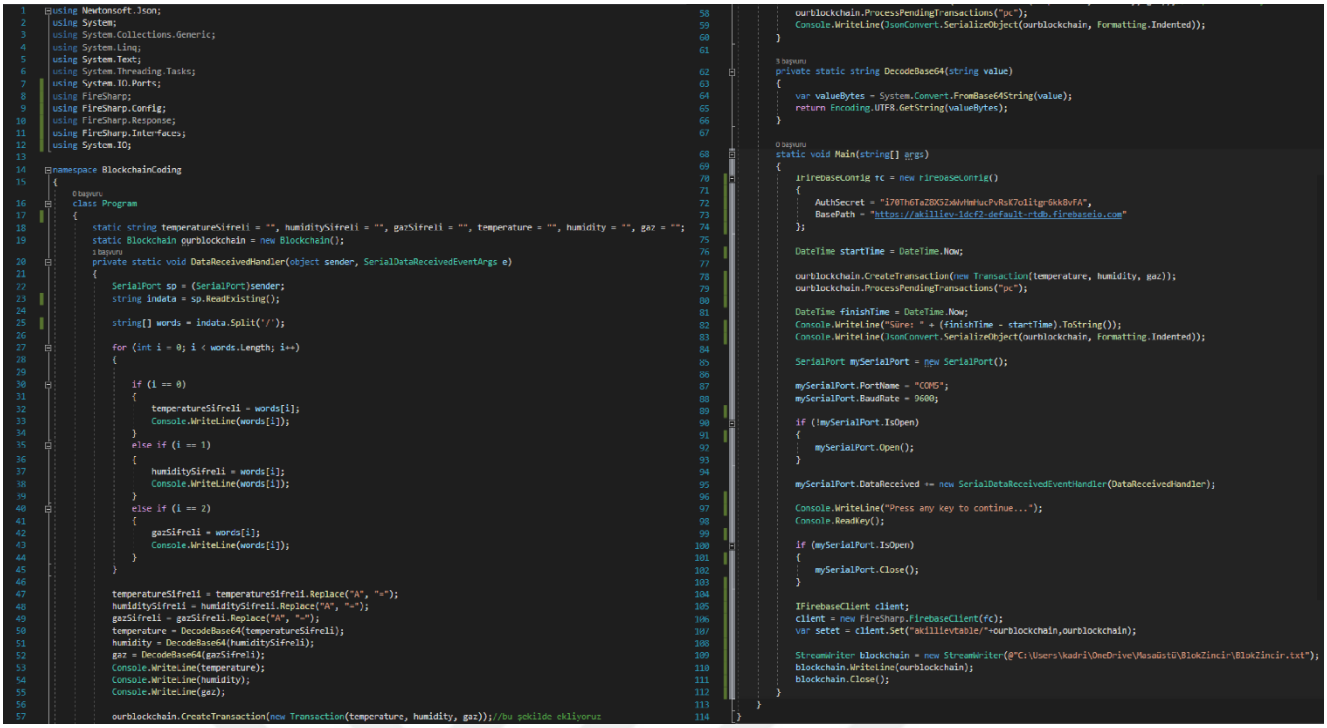
Şekil 7.10. "Transaction.cs" sınıfı kodları

```
using Newtonsoft.Json;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Security.Cryptography;
using System.Text;
using System.Threading.Tasks;

namespace BlockchainCoding
{
    13 başvuru
    public class Block
    {
        3 başvuru
        public int Index { get; set; }
        2 başvuru
        public DateTime TimeStamp { get; set; }
        4 başvuru
        public string PreviousHash { get; set; }
        8 başvuru
        public string Hash { get; set; }

        2 başvuru
        public IList<Transaction> Transactions { get; set; }
        2 başvuru
        public int Nonce { get; set; } = 0;

        2 başvuru
        public Block(DateTime timeStamp, string previousHash, IList<Transaction> transactions)
        {
            Index = 0;
            TimeStamp = timeStamp;
            PreviousHash = previousHash;
            Transactions = transactions;
        }

        3 başvuru
        public string CalculateHash()
        {
            SHA256 sha256 = SHA256.Create();
            byte[] inbytes = Encoding.ASCII.GetBytes($"{TimeStamp}-{PreviousHash ?? ""}-{JsonConvert.SerializeObject(Transactions)}-{Nonce}");
            byte[] outbytes = sha256.ComputeHash(inbytes);
            return Convert.ToBase64String(outbytes);
        }

        2 başvuru
        public void Mine(int difficulty)
        {
            var leadingzeros = new string('0', difficulty);
            while (this.Hash == null || this.Hash.Substring(0, difficulty) != leadingzeros)
            {
                this.Nonce++;
                this.Hash = this.CalculateHash();
            }
        }
    }
}
```

Şekil 7.11. "Block.cs" sınıfı kodları

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BlockchainCoding
{
    3 başvuru
    public class Blockchain
    {
        IList<Transaction> PendingTransactions = new List<Transaction>();
        8 başvuru
        public IList<Block> Chain { get; set; }
        2 başvuru
        public int difficulty { get; set; } = 2;
        1 başvuru
        public string Reward { get; set; } = "1";

        1 başvuru
        public Blockchain()
        {
            InitializeChain();
            AddGenesisBlock();
        }

        1 başvuru
        public void InitializeChain()
        {
            Chain = new List<Block>();
        }

        1 başvuru
        public Block CreateGenesisBlock()
        {
            Block block = new Block(DateTime.Now, null, PendingTransactions);
            block.Mine(difficulty);
            PendingTransactions = new List<Transaction>();
            return block;
        }

        1 başvuru
        public void AddGenesisBlock()
        {
            Chain.Add(CreateGenesisBlock());
        }

        2 başvuru
        public Block GetLatestBlock()
        {
            return Chain[Chain.Count - 1];
        }

        1 başvuru
        public void AddBlock(Block block)
        {
            Block latestblock = GetLatestBlock();
            block.Index = latestblock.Index + 1;
            block.PreviousHash = latestblock.Hash;
            block.Hash = block.CalculateHash();
            block.Mine(this.difficulty);
            Chain.Add(block);
        }

        3 başvuru
        public void CreateTransaction(Transaction transaction)
        {
            PendingTransactions.Add(transaction);
        }

        2 başvuru
        public void ProcessPendingTransactions(string minerAddress)
        {
            CreateTransaction(new Transaction(null, minerAddress, Reward));
            Block block = new Block(DateTime.Now, GetLatestBlock().Hash, PendingTransactions);
            AddBlock(block);
            PendingTransactions = new List<Transaction>();
        }

        0 başvuru
        public bool IsValid()
        {
            for (int i = 1; i < Chain.Count; i++)
            {
                Block currentBlock = Chain[i];
                Block previousBlock = Chain[i - 1];
                if (currentBlock.Hash != currentBlock.CalculateHash())
                {
                    return false;
                }
                if (currentBlock.PreviousHash != previousBlock.Hash)
                {
                    return false;
                }
            }
            return true;
        }
    }
}
```

Şekil 7.12. “Blockchain.cs” sınıfı kodları



Şekil 7.13. "Program.cs" sınıfı kodları

Blok zincir kodları C# programlama dilinde kodlanmıştır. Akıllı sözleşmeler, koşullar anlamına gelmektedir ve bu sebeple "if-else" blokları ile kodlanmıştır. "Transaction.cs" sınıfı işlemleri içermektedir. Şekil 7.10.'da verilmiştir. İşlemler belirli bir düzendedir ve birden fazla olabilir. Bu sebeple bir sınıf olarak tanımlanmıştır. Her işlem sıcaklık, nem ve gaz değerlerinden oluşur. Bloklar ise "Block.cs" sınıfında tanımlanmıştır. Şekil 7.11.'de "Block.cs" sınıfına ait kodlar verilmiştir. "Block.cs" sınıfı kullanılarak blok zincir oluşturulmuştur ve "Blockchain.cs" sınıfı kodlanmıştır. Her bir bloğun içerdiği nonce, zaman damgası, önceki bloğun özet değer, bloğun kendi özet değeri ve işlemler yer almaktadır. Özet değeri hesaplama işlemi de "Block.cs" sınıfında sağlanmıştır. Şekil 7.12.'de "Blockchain.cs" sınıfına ait kodlar verilmiştir. "Blockchain.cs" sınıfında bulunan "IsValid()" fonksiyonu ise güvenlik amacı ile yazılmış bir fonksiyondur. Blok zincire yeni bir blok eklendikten sonra blok zincirin herhangi bir değişikliğe uğrayıp uğramadığının kontrolünü sağlar. "Program.cs" sınıfında ise seri port yardımı ile Arduino IDE'sinden okunan verilerle birlikte blok zincir oluşturulur. Bulutta ve yerelde depolama "Program.cs" sınıfında sağlanmaktadır. Şekil 7.13.'te "Program.cs" sınıfına ait kodlar yer almaktadır.

Blok zincir ile depolama sağlanmadan önce akıllı ev verileri, saldırganlar tarafından erişilip değişikliğe uğrama ve farklı kötü amaçlar için kullanılma tehdidine açık haldedirler. Blok zincir kullanarak bu sorunlara çözüm getirilmiştir. Veriler

değişikliğe uğrama tehlikesinde değildirler. Hem lokalde hem de bulutta depolanan veriler çift taraflı kontroldedirler. Ayrıca blok zincirde bloklar birbirlerine bağlı oldukları için herhangi bir değişiklikte bağlantı kopmaktadır. “IsValid()” fonksiyonu devreye girmektedir. Gerekli saldırılarda böylelikle önlenmiş olur. Donanım kısmında ise veriler şifreli biçimde aktarılmaktadır ve MAC adresleri ile doğrudan gidecekleri adres bellidir.

8. SONUÇLAR VE ÖNERİLER

8.1. Sonuçlar

Günümüzde IoT güvenliği hem akademik çalışmalarda hem de endüstriyel uygulamalarda büyük ilgi görmektedir. Mevcut güvenlik çözümleri, yüksek enerji tüketimi ve işlem yükü nedeniyle IoT için uygun olmayabilmektedir. Bu tez çalışmasında, değişmez bloklar defteri olan blok zincir teknolojisinden yararlanarak bu zorlukları ele alan bir yöntem önerilmiştir. Temsili bir vaka çalışması olarak bir akıllı ev prototipi kullanılmıştır. Bu tez çalışması, akıllı evler bağlamında blok zinciri optimum olumlu şekilde kullanmayı amaçlayan bir çalışmadır. Özel blok zincir kullanarak güvenli bir mimariyi uygulamak için basit bir model sunarak örnek çalışma gerçekleştirilmiştir. Blok zincir ve IoT verilerini birlikte nasıl kullanabileceği gösterilmiştir. Bu yazıda açıklandığı gibi blok zincir, temel güvenlik uygulamalarının çok ötesine geçen güçlü bir araçtır. Blok zincir kullanmanın avantajı, iletişim modelinin alt katmanında ve uygulama katmanında çalışabilmeleri ve böylece mekanizmanın IoT ekosisteminin tüm katmanları ve etki alanlarında iş birliği içinde kullanılabilmesidir. Verim, teknolojinin temel unsurlarından ve ucuz yönlerinden biridir. Bu çalışma, ev verilerini güvence altına almak için bir iletişim hattı ve bunun uygulanması için önerilen mimariyi uygulamıştır.

Bu tez çalışması; blok zincir konsepti, mekanizması, veri yapısı ve blok zincir platformunun temel genel bakışını açıklayarak başlamıştır. Ardından, akıllı evin blok zincire uyum sağlaması için gerekenler açıklanmıştır. Katmanlı yapıya değinilmiştir. Akıllı ev uygulamaları için blok zincir platformunu uygulayarak, uygulanabilir bir prototip geliştirmek amacıyla ihtiyaç duyulacak gerekli bilgilere değinilmiştir. Ek olarak, literatürde olan veri pazarı için akıllı ev blok zinciri gibi akıllı ev uygulamasını içeren çalışmalar sunulmuştur. Spesifik olarak, akıllı ev IoT ekosisteminden üretilen büyük hacimli verilerle güvenilir bir akıllı ev veri pazarı oluşturmak için blok zincirin potansiyeli vurgulanmıştır. Son olarak, çalışmanın güvenlik değerlendirmesinde olan kriterlerine değinilmiştir.

Bu tez çalışmasında, spesifik olarak önerilen sistem akıllı evde bulunan DHT11 sıcaklık ve nem sensörü ve MQ-2 gaz sensörü NodeMCU ESP8266 kartlarına ayrı ayrı bağlıdırlar. Bunun sebebi sensörlerin blok zincirde birer düğüm olarak yer alması ve blok zincirde olacak gerekli işlemleri gerçekleştirebiliyor olması gerektiğidir. Ana ağ geçidi ve sensörler arasında olan güvenli bağlantı, mükemmel iletme gizliliği ile bilgi

sızdırmadan bir talep sahibinin kimliğini doğrulamak için MAC adresleri ile sağlanır ve aktarılan veriler şifreli bir şekilde aktarılır. Ağ geçidine MAC adresleri yardımıyla şifreli bir biçimde gelen veriler, C# programlama dili kullanılarak şifreli bir biçimde aktarıldıktan sonra, C# yazılım dilinde şifre çözülür ve ham veri özet fonksiyonundan geçirilerek özet değeri elde edilir. Ardından özet değeri; nonce, önceki bloğun özet değeri gibi blok içerisinde yer alan değerlerle blok zincirde tutulur. Ayrıca bu tez çalışmasında, mevcut akıllı ev ağ geçidi ortamı ve IoT için blok zincir tabanlı veri kurcalamaya karşı korumalı bir ağ geçidi mimarisi önerilmiştir. Bu çalışma, akıllı evi oluşturan heterojen IoT ve merkezi ağ geçitlerinin gizlilik, bütünlük ve kimlik doğrulama sorunlarını en aza indirmek için çözümler sağlamaktadır.

Blok zincir gelişmeye devam eden güçlü bir teknolojidir. Önemli avantajlara sahiptir ancak bunun yanında hem IoT ile kullanılması hem de geliştirilmesi konularında ölçeklenebilirlik, depolama kapasitesi ve mahremiyet gibi alanlarda zorluklara sahiptir.

Blok zincir aslında kurcalamaya karşı dayanıklı bir defterdir. Veriler değiştirilemez biçimde blok zincire kaydedilmektedir. Ama bir hata durumunda, geri getirilemez kayıplar oluşabilir. Veriler blok zincir içinde bozulmuş ise, o şekilde kalmaktadır.

Blok zincir teknolojisi IoT sistemlerinin güvenliğini artırmak için genel olarak tercih edilmektedir. Ancak blok zincir yazılımında ve kullanılan akıllı sözleşmelerde güvenlik açıklıkları bulunmuştur. Bu durum ciddi problemlere yol açmaktadır.

Blok zincirler denetlenebilirlik konusunda son derece verimli çalışırlar ancak bu durum genel blok zincirlerde gizlilik konusunda istenmeyen durum yaratmaktadır.

Blok zincirde ölçeklenebilirlik bir dezavantajdır. IoT uygulamalarında makineden makineye (machine-to-machine) birçok işlem aktarılır. Blok zincirlerde blok ekleme süresini göz önüne alındığında, otomatik ödeme platformları gibi trafiğin yoğun olduğu uygulamalarda ölçeklenebilirlik sorun haline gelmektedir.

Bu heterojen kullanımın karşılaştığı sorunlar arasında gerekli kanun düzenlemelerinin olmaması da yer almaktadır. Blok zincirin sağladığı özellikler arasında anonimlik yer alır ve kötü niyetli kullanıcılar için saldırıda iyi bir ortam haline gelir. Farklı ülkeler bu sorunla nasıl başa çıkılacağı ve uygun kanunların nasıl olacağı konusunda sorunlar yaşamışlardır.

Birçok IoT cihazı düşük enerjiye, sınırlı bellek miktarına ve düşük işlem gücüne sahiptir. Blok zincir teknolojisinde kullanılan mutabakat algoritmaları ise yüksek enerji

miktarına ve işlem gücüne ihtiyaç duyarlar. Bu nedenle IoT ve blok zincir teknolojilerinin entegre kullanımında kısıtlı kaynak sorunu vardır.

8.2. Öneriler

Blok zincir teknolojisi ve IoT'nin sentez kullanımı bir devrim yaratabilir ve blok zincir teknolojisinin kullanımı bu sayede hızla artacaktır. Belirtilen zorluklar göz önüne alınarak yapılan çalışmalara yön verilmelidir.

IoT ve blok zincir teknolojilerinin kullanımında güvenlik ve mahremiyet konularında daha fazla araştırmaya gereksinim vardır. Gerekli kanun düzenlenmeleri yapılmalıdır. Blok zincir verileri her blokta depolanır ve depolama pahalı bir işlemdir, bu sorunu çözebilmek için yeni yöntemler araştırılmalıdır.

Blok zincir ve IoT'nin birlikte kullanımında ölçeklenebilirlik sorunu için daha kullanışlı mutabakat algoritmaları konusunda çalışmalar yapılabilir. Ağda kullanılmak üzere farklı özel blok zincir yapıları üzerine çalışmalar yapılabilir.

Blok zincir büyüdükçe, her düğüm tarafından depolanması zorlaşır. Bu durumda IoT ağında verimli bir depolamanın nasıl sağlanacağı üzerine çalışmalar yapılmalıdır. IoT'de bulunan kısıtlı kaynak sorununun çözümü için bulut bilişim, blok zincir ve IoT teknolojilerinin entegre kullanımı konularında çalışmalar yapılmalıdır. Merkezi bulut depolamada blok zincir teknolojisinin kullanımı, çalışmalarda ana sorun olarak ele alınmalıdır.

Blok zincirde işlemlerin doğrulanması süreçlerinde, hız açısından iyileştirme yapılması, süreçlerin hızlanabileceği anlamına gelir. Bu yönde çalışmalar yapılabilir.

Ayrıntılı erişim kontrolü elde etmek için (ancak erişim politikasının gizliliğini tehlikeye atmadan) çalışmaya öznitelik tabanlı kriptografik yaklaşımları dikkate alınabilir. Ayrıca, diğer protokollerle paralel çalışırken güvenliği garanti altına almak için evrensel birleştirilebilirlik ayarında güvenli protokol tasarımı dikkate alınabilir. Ayrıca, gerçek dünyada daha yüksek fayda elde etmek için genişletilmiş sistemin bir prototipini uygulama ve değerlendirme işlemi yapılabilir. Farklı akıllı ev uygulamaları için blok zincir platformunu uygulayarak, uygulanabilir bir prototip geliştirmek amacıyla ve simülasyon kullanarak çalışma genişletilebilir. Bu tez çalışmasının bir sonraki aşamasında, bir sorunu çözmek için gereken enerji ile her bir düğüm için mevcut veya gerekli olan enerji arasındaki ilişkinin yanı sıra özet değeri, blok kuyruğa alma ve bekleme süresine yönelik hesaplama zorlukları incelenebilir. Gelecekteki çalışmalarda, blok zincir ağındaki IoT veri güvenliğini iyileştirmek için büyük veri

kavramıyla derin öğrenme ve makine öğrenimi kullanımı ile ilgili çalışmalar yapılabilir. Aynı akıllı ev uygulaması Ethereum ve Hyperledger platformları ile de oluşturularak simülasyonlar yardımı ile enerji tüketiminin hangi çalışmada nasıl yönlendiğine dair bir uygulama gerçekleştirilebilir.

KAYNAKLAR

Aazam, M., ve ark., 2016, PRE-Fog: IoT trace based probabilistic resource estimation at Fog. 2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC), IEEE.

Abbas, S., ve ark., 2020, "Modeling, simulation and optimization of power plant energy sustainability for IoT enabled smart cities empowered with deep extreme learning machine." Ieee Access **8**: 39982-39997.

AbuNaser, M. ve A. A. Alkhatib, 2019, Advanced survey of blockchain for the internet of things smart home. 2019 IEEE Jordan international joint conference on electrical engineering and information technology (JEEIT), IEEE.

Ahram, T., ve ark., 2017, Blockchain technology innovations. 2017 IEEE technology & engineering management conference (TEMSCON), IEEE.

Alaba, F. A., ve ark., 2017, "Internet of Things security: A survey." Journal of network and computer applications **88**: 10-28.

Alam, M. R., ve ark., 2019, "Peer-to-peer energy trading among smart homes." Applied energy **238**: 1434-1443.

Alharby, M. ve A. Van Moorsel, 2017, "Blockchain-based smart contracts: A systematic mapping study." arXiv preprint arXiv:1710.06372.

Ali, M. S., ve ark., 2018, "Applications of blockchains in the Internet of Things: A comprehensive survey." IEEE Communications Surveys & Tutorials **21**(2): 1676-1717.

Ashton, K., 2009, "That ‘internet of things’ thing." RFID journal **22**(7): 97-114.

Atlam, H. F., ve ark., 2018, "Fog computing and the internet of things: A review." big data and cognitive computing **2**(2): 10.

Atlam, H. F. ve G. B. Wills, 2019, "An efficient security risk estimation technique for Risk-based access control model for IoT." Internet of Things **6**: 100052.

Atzei, N., ve ark., 2017, A survey of attacks on ethereum smart contracts (sok). Principles of Security and Trust: 6th International Conference, POST 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings 6, Springer.

Balamurugan, B. ve D. Biswas, 2017, Security in network layer of IoT: Possible measures to preclude. Security Breaches and Threat Prevention in the Internet of Things, IGI Global**:** 46-75.

Baran, P., 1964, "On distributed communications networks." IEEE transactions on Communications Systems **12**(1): 1-9.

Bashir, I., 2017, Mastering blockchain, Packt Publishing Ltd.

Belotti, M., ve ark., 2019, "A vademecum on blockchain technologies: When, which, and how." IEEE Communications Surveys & Tutorials **21**(4): 3796-3838.

Bhamare, D., ve ark., 2017, "Optimal virtual network function placement in multi-cloud service function chaining architecture." Computer Communications **102**: 1-16.

Borgohain, T., ve ark., 2015, "Survey of security and privacy issues of internet of things." arXiv preprint arXiv:1501.02211.

Botta, A., ve ark., 2016, "Integration of cloud computing and internet of things: a survey." Future generation computer systems **56**: 684-700.

Boyd, C. ve J. G. Nieto, 2009, Information Security and Privacy: 14th Australasian Conference, ACISP 2009 Brisbane, Australia, July 1-3, 2009 Proceedings, Springer.

Buldas, A., ve ark., 2017, "Keyless signature infrastructure and PKI: hash-tree signatures in pre-and post-quantum world." International Journal of Services Technology and Management **23**(1-2): 117-130.

Burgess, K. ve J. Colangelo, 2015, "The promise of bitcoin and the Blockchain." Consumers' Research.

Cha, S.-C., ve ark., 2018, "A blockchain connected gateway for BLE-based devices in the internet of things." Ieee Access **6**: 24639-24649.

Cheng, J.-C., ve ark., 2018, Blockchain and smart contract for digital certificate. 2018 IEEE international conference on applied system invention (ICASI), IEEE.

Christidis, K. ve M. Devetsikiotis, 2016, "Blockchains and smart contracts for the internet of things." Ieee Access **4**: 2292-2303.

Conoscenti, M., ve ark., 2017, Peer to peer for privacy and decentralization in the internet of things. 2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C), IEEE.

Courtois, N. T., ve ark., 2014, Optimizing sha256 in bitcoin mining. Cryptography and Security Systems: Third International Conference, CSS 2014, Lublin, Poland, September 22-24, 2014. Proceedings 3, Springer.

Crosby, M., 2022, BlockChain Technology: Beyond Bitcoin. Applied Innovation Review (AIR). Issue No. 2 June 2016. Berkeley.

Damasevicius, R., ve ark., 2012, "Energy consumption of hash functions." Elektronika ir elektrotechnika **18**(10): 81-84.

Demestichas, K., ve ark., 2020, "Blockchain in agriculture traceability systems: A review." Applied Sciences **10**(12): 4113.

Dinh, T. T. A., ve ark., 2017, Blockbench: A framework for analyzing private blockchains. Proceedings of the 2017 ACM international conference on management of data.

Dorri, A., ve ark., 2016, "Blockchain in internet of things: challenges and solutions." arXiv preprint arXiv:1608.05187.

Dorri, A., ve ark., 2017, Blockchain for IoT security and privacy: The case study of a smart home. 2017 IEEE international conference on pervasive computing and communications workshops (PerCom workshops), IEEE.

Efanov, D. ve P. Roschin, 2018, "The all-pervasiveness of the blockchain technology." Procedia computer science **123**: 116-121.

Farooq, M. S., ve ark., 2022, "Blockchain-Based Smart Home Networks Security Empowered with Fused Machine Learning." Sensors **22**(12): 4522.

Fernández-Caramés, T. M. ve P. Fraga-Lamas, 2018, "A Review on the Use of Blockchain for the Internet of Things." Ieee Access **6**: 32979-33001.

Fischer, M. J., ve ark., 1985, "Impossibility of distributed consensus with one faulty process." Journal of the ACM (JACM) **32**(2): 374-382.

Fisk, M. J., 2001, The implications of smart home technologies. Inclusive housing in an ageing society, Policy Press**:** 101-124.

Fontaine, C. ve F. Galand, 2007, "A survey of homomorphic encryption for nonspecialists." EURASIP Journal on Information Security **2007**: 1-10.

GemOS (2022). "GemOS: The Blockchain Operating System." Ziyaret Tarihi February, 2023, URL https://enterprise.gem.co/.

Göbel, J. ve A. E. Krzesinski, 2017, Increased block size and Bitcoin blockchain dynamics. 2017 27th International Telecommunication Networks and Applications Conference (ITNAC), IEEE.

Gram-Hanssen, K. ve S. J. Darby, 2018, "“Home is where the smart is”? Evaluating smart home research and approaches against the concept of home." Energy Research & Social Science **37**: 94-101.

Guo, Y. ve C. Liang, 2016, "Blockchain application and outlook in the banking industry." Financial innovation **2**: 1-12.

Haber, S. ve W. S. Stornetta, 1991, How to time-stamp a digital document, Springer.

Han, D.-M. ve J.-H. Lim, 2010, "Design and implementation of smart home energy management systems based on zigbee." IEEE Transactions on Consumer Electronics **56**(3): 1417-1425.

Hang, L., ve ark., 2020, "A secure fish farm platform based on blockchain for agriculture data integrity." Computers and Electronics in Agriculture **170**: 105251.

He, M., ve ark., 2022, "T2L: A traceable and trustable consortium blockchain for logistics." Digital Communications and Networks.

Horrow, S. ve A. Sardana, 2012, Identity management framework for cloud based internet of things. Proceedings of the First International Conference on Security of Internet of Things.

Huckle, S., ve ark., 2016, "Internet of things, blockchain and shared economy applications." Procedia computer science **98**: 461-466.

Hugoson, M.-Å., 2009, Centralized versus decentralized information systems: A historical flashback. History of Nordic Computing 2: Second IFIP WG 9.7 Conference, HiNC2, Turku, Finland, August 21-23, 2007, Revised Selected Papers 2, Springer.

Iqbal, M. ve R. Matulevičius, 2019, Blockchain-based application security risks: a systematic literature review. Advanced Information Systems Engineering Workshops: CAiSE 2019 International Workshops, Rome, Italy, June 3-7, 2019, Proceedings 31, Springer.

Jing, Q., ve ark., 2014, "Security of the Internet of Things: perspectives and challenges." Wireless Networks **20**: 2481-2501.

Kadena (2022). "Kadena." Ziyaret Tarihi June, 2023, URL https://kadena.io/.

Kajwadkar, S. ve V. K. Jain, 2018, A novel algorithm for DoS and DDoS attack detection in Internet of things. 2018 Conference on Information and Communication Technology (CICT), IEEE.

Kaur, S., ve ark., 2021, "A research survey on applications of consensus protocols in blockchain." Security and Communication Networks **2021**: 1-22.

Kim, Y., ve ark., 2017, "A study on the adoption of IoT smart home service: using Value-based Adoption Model." Total Quality Management & Business Excellence **28**(9-10): 1149-1165.

Lee, Y., ve ark., 2020, "A blockchain-based smart home gateway architecture for preventing data forgery." Human-centric Computing and Information Sciences **10**(1): 1-14.

Leonardos, S., ve ark., 2020, "Weighted voting on the blockchain: Improving consensus in proof of stake protocols." International Journal of Network Management **30**(5): e2093.

Li, H., ve ark., 2020, "A blockchain-based traceable self-tallying E-voting protocol in AI era." IEEE Transactions on Network Science and Engineering **8**(2): 1019-1032.

Li, M., ve ark., 2018, "Smart home: architecture, technologies and systems." Procedia computer science **131**: 393-400.

Lin, C., ve ark., 2019, "HomeChain: A blockchain-based secure mutual authentication system for smart homes." IEEE Internet of Things Journal **7**(2): 818-829.

Lin, I.-C. ve T.-C. Liao, 2017, "A survey of blockchain security issues and challenges." Int. J. Netw. Secur. **19**(5): 653-659.

Liu, G., ve ark., 2023, "Communitychain: Towards a scalable blockchain in smart home." IEEE Transactions on Network and Service Management.

Lu, Y., 2019, "The blockchain: State-of-the-art and research challenges." Journal of Industrial Information Integration **15**: 80-90.

Lu, Y. ve L. Da Xu, 2018, "Internet of Things (IoT) cybersecurity research: A review of current research topics." IEEE Internet of Things Journal **6**(2): 2103-2115.

Luu, L., ve ark., 2016, Making smart contracts smarter. Proceedings of the 2016 ACM SIGSAC conference on computer and communications security.

Ma, G., ve ark., 2020, "Achieving reliable timestamp in the bitcoin platform." Peer-to-Peer Networking and Applications **13**: 2251-2259.

Mamdouh, M., ve ark., 2021, "Authentication and identity management of IoHT devices: Achievements, challenges, and future directions." Computers & Security **111**: 102491.

Maull, R., ve ark., 2017, "Distributed ledger technology: Applications and implications." Strategic Change **26**(5): 481-489.

Mehta, D., ve ark., 2021, "Blockchain-based royalty contract transactions scheme for Industry 4.0 supply-chain management." Information Processing & Management **58**(4): 102586.

Merkle, R. C., 1988, A digital signature based on a conventional encryption function. Advances in Cryptology—CRYPTO'87: Proceedings 7, Springer.

Moniruzzaman, M., ve ark., 2020, "Blockchain for smart homes: Review of current trends and research challenges." Computers & Electrical Engineering **83**: 106585.

MultiChain (2022). "MultiChain: Open Platform for Building Blockchains." Ziyaret Tarihi February, 2023, URL https://www.multichain.com/.

Murthy, C. V. B., ve ark., 2020, "Blockchain based cloud computing: Architecture and research challenges." Ieee Access **8**: 205190-205205.

Nakamoto, S., 2008, "Bitcoin: A peer-to-peer electronic cash system." Decentralized Business Review: 21260.

Nofer, M., ve ark., 2017, "Blockchain." Business & Information Systems Engineering **59**: 183-187.

O'Dwyer, K. J. ve D. Malone, 2014, "Bitcoin mining and its energy footprint."

Pal, D., ve ark., 2018, "Analyzing the elderly users' adoption of smart-home services." Ieee Access **6**: 51238-51252.

Pittalia, P. P., 2019, "A comparative study of hash algorithms in cryptography." International Journal of Computer Science and Mobile Computing **8**(6): 147-152.

Popov, S., ve ark., 2019, "Equilibria in the tangle." Computers & Industrial Engineering **136**: 160-172.

Qashlan, A., ve ark., 2021, "Privacy-preserving mechanism in smart home using blockchain." Ieee Access **9**: 103651-103669.

Ravishankar, B., ve ark., 2020, Blockchain-based database to ensure data integrity in cloud computing environments. 2020 International Conference on Mainstreaming Block Chain Implementation (ICOMBI), IEEE.

Ren, Y., ve ark., 2021, "Multiple cloud storage mechanism based on blockchain in smart homes." Future generation computer systems **115**: 304-313.

Reyna, A., ve ark., 2018, "On blockchain and its integration with IoT. Challenges and opportunities." Future generation computer systems **88**: 173-190.

Rivest, R., 1992, The MD5 message-digest algorithm.

Sabry, S. S., ve ark., 2019, "The road to the blockchain technology: Concept and types." Periodicals of Engineering and Natural Sciences **7**(4): 1821-1832.

Samuel, O., ve ark., 2022, "Towards sustainable smart cities: A secure and scalable trading system for residential homes using blockchain and artificial intelligence." Sustainable Cities and Society **76**: 103371.

Sanguinetti, A., ve ark., 2018, "What's energy management got to do with it? Exploring the role of energy management in the smart home adoption process." Energy efficiency **11**(7): 1897-1911.

Sharma, P. K., ve ark., 2018, "DistArch-SCNet: blockchain-based distributed architecture with li-fi communication for a scalable smart city network." IEEE Consumer Electronics Magazine **7**(4): 55-64.

Shin, J., ve ark., 2018, "Who will be smart home users? An analysis of adoption and diffusion of smart homes." Technological Forecasting and Social Change **134**: 246-253.

Singh, A., ve ark., 2020, "Blockchain smart contracts formalization: Approaches and challenges to address vulnerabilities." Computers & Security **88**: 101654.

Solat, S., 2018, Rdv: An alternative to proof-of-work and a real decentralized consensus for blockchain. Proceedings of the 1st Workshop on Blockchain-enabled Networked Sensor Systems.

Sompolinsky, Y., ve ark., 2016, "Spectre: A fast and scalable cryptocurrency protocol." Cryptology ePrint Archive.

Statista (2022). "Number of Internet of Things (IoT) connected devices worldwide from 2019 to 2021, with forecasts from 2022 to 2030." Ziyaret Tarihi February, 2023, URL https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/.

Statista (2022). "Smart Home penetration rate forecast in the world from 2017 to 2025." 2022, URL https://www.statista.com/forecasts/887636/penetration-rate-of-smart-homes-in-the-world

Stojkoska, B. L. R. ve K. V. Trivodaliev, 2017, "A review of Internet of Things for smart home: Challenges and solutions." Journal of cleaner production **140**: 1454-1464.

Swan, M., 2015, Blockchain: Blueprint for a new economy, " O'Reilly Media, Inc.".

Szabo, N., 1997, "Formalizing and securing relationships on public networks." First monday.

Szydlo, M., 2004, Merkle tree traversal in log space and time. Eurocrypt, Springer.

Tchagna Kouanou, A., ve ark., 2022, "Securing data in an internet of things network using blockchain technology: smart home case." SN Computer Science **3**(2): 167.

Tewari, A. ve B. B. Gupta, 2020, "Security, privacy and trust of different layers in Internet-of-Things (IoTs) framework." Future generation computer systems **108**: 909-920.

Torre, I., ve ark., 2016, A framework for personal data protection in the IoT. 2016 11th international conference for internet technology and secured transactions (ICITST), IEEE.

Tschorsch, F. ve B. Scheuermann, 2016, "Bitcoin and beyond: A technical survey on decentralized digital currencies." IEEE Communications Surveys & Tutorials **18**(3): 2084-2123.

Vukolić, M., 2016, The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication. Open Problems in Network Security: IFIP WG 11.4 International Workshop, iNetSec 2015, Zurich, Switzerland, October 29, 2015, Revised Selected Papers, Springer.

Wang, J., ve ark., 2019, "Energy efficient routing algorithm with mobile sink support for wireless sensor networks." Sensors **19**(7): 1494.

Wilson, C., ve ark., 2017, "Benefits and risks of smart home technologies." Energy Policy **103**: 72-83.

Wood, G., 2014, "Ethereum: A secure decentralised generalised transaction ledger." Ethereum project yellow paper **151**(2014): 1-32.

Xiang, G., ve ark., 2012, "Research on trust model of sensor nodes in WSNs." Procedia Engineering **29**: 909-913.

Xiaoping, D., ve ark., 2021, Research on the intelligent settlement cloud platform of electric power materials based on the electronization of blockchain VAT special invoice. 2021 China International Conference on Electricity Distribution (CICED), IEEE.

Xu, X., ve ark., 2017, A taxonomy of blockchain-based systems for architecture design. 2017 IEEE international conference on software architecture (ICSA), IEEE.

Yakubu, B. M., ve ark., 2023, "Blockchain-based DDoS attack mitigation protocol for device-to-device interaction in smart home." Digital Communications and Networks.

Yan, Z., ve ark., 2014, "A survey on trust management for Internet of Things." Journal of network and computer applications **42**: 120-134.

Yang, Q. ve H. Wang, 2021, "Privacy-preserving transactive energy management for IoT-aided smart homes via blockchain." IEEE Internet of Things Journal **8**(14): 11463-11475.

Yli-Huumo, J., ve ark., 2016, "Where is current research on blockchain technology?—a systematic review." PloS one **11**(10): e0163477.

Yuan, Y. ve F.-Y. Wang, 2018, "Blockchain and cryptocurrencies: Model, techniques, and applications." IEEE Transactions on Systems, Man, and Cybernetics: Systems **48**(9): 1421-1428.

Zhao, W., ve ark., 2018, "ETC-IoT: Edge-node-assisted transmitting for the cloud-centric internet of things." IEEE Network **32**(3): 101-107.

Zheng, Z., ve ark., 2018, "Blockchain challenges and opportunities: A survey." International journal of web and grid services **14**(4): 352-375.

Zheng, Z., ve ark., 2017, An overview of blockchain technology: Architecture, consensus, and future trends. 2017 IEEE international congress on big data (BigData congress), Ieee.

Zhu, Q., ve ark., 2019, "Applications of distributed ledger technologies to the internet of things: A survey." ACM computing surveys (CSUR) **52**(6): 1-34.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Kadriye Nur ERMAN
Uyruğu : TC

EĞİTİM

Derece		**Adı, İlçe, İl**	**Bitirme Yılı**
Lise	:	Konya Atatürk Mesleki ve Teknik Anadolu Lisesi	2015
Üniversite	:	Selçuk Üniversitesi	2020
Yüksek Lisans	:	Selçuk Üniversitesi	Devam
Doktora	:		

İŞ DENEYİMLERİ

Yıl	**Kurum**	**Görevi**
2020-2021	Milvasoft	Frontend Developer
2022-Devam	Kırıkkale Üniversitesi	Araştırma Görevlisi

UZMANLIK ALANI

YABANCI DİLLER

BELİRTMEK İSTEĞİNİZ DİĞER ÖZELLİKLER

YAYINLAR

Erman, K.N. ve Çınar, A.C., 2023, Ensuring Data Security in The Blockchain Based Smart Home Network, *The 7th International Conference on Computational Mathematics and Engineering Sciences,* Elazığ – Türkiye, 51.