

**T.C.
SÜLEYMAN DEMİREL ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ**

**BLOK ZİNCİRİ MİMARİSİNİN NESNELERİN İNTERNETİNE
UYARLANMASI VE PERFORMANS ANALİZLERİ**

Cihan YALÇIN

**Danışman
Dr. Öğr. Üyesi Mevlüt ERSOY**

**YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
ISPARTA - 2019**



© 2019 [Cihan YALÇIN]

TEZ ONAYI

Cihan YALÇIN tarafından hazırlanan "**Blok Zinciri Mimarisinin Nesnelerin İnternetine Uyarlanması ve Performans Analizleri**" adlı tez çalışması aşağıdaki jüri üyeleri önünde Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü **Bilgisayar Mühendisliği Anabilim Dalı**'nda **YÜKSEK LİSANS TEZİ** olarak başarı ile savunulmuştur.

Danışman

Dr. Öğr. Üyesi Mevlüt ERSOY
Süleyman Demirel Üniversitesi



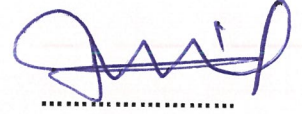
Jüri Üyesi

Dr. Öğr. Üyesi Asım Sinan YÜKSEL
Süleyman Demirel Üniversitesi



Jüri Üyesi

Dr. Öğr. Üyesi Bekir AKSOY
Isparta Uygulamalı Bilimler Üniversitesi



Enstitü Müdürü

Doç. Dr. Şule Sultan UĞUR

.....

TAAHHÜTNAME

Bu tezin akademik ve etik kurallara uygun olarak yazıldığını ve kullanılan tüm literatür bilgilerinin referans gösterilerek tezde yer aldığını beyan ederim.

Cihan YALÇIN



İÇİNDEKİLER

	Sayfa
İÇİNDEKİLER.....	i
ÖZET	iii
ABSTRACT	iv
TEŞEKKÜR.....	v
ŞEKİLLER DİZİNİ	vi
ÇİZELGELER DİZİNİ	vii
SİMGELER VE KISALTMALAR DİZİNİ	viii
1. GİRİŞ.....	1
2. KAYNAK ÖZETLERİ	3
3. MATERYAL VE YÖNTEM	11
3.1. Nesnelerin İnterneti	11
3.1.1. Cihazlar	13
3.1.2. Algılayıcılar	13
3.1.3. İletişim.....	14
3.1.4. Aktüatörler	14
3.1.5. Depolama.....	15
3.1.6. İşleme.....	15
3.1.7. Takip	15
3.1.8. Kimliklendirme	16
3.2. Nesnelerin İnterneti Yönetim Aracı	17
3.2.1. Node-Red Veri Akışı.....	18
3.2.2. Node-Red Ön Yüz	19
3.3. Blok Zinciri.....	20
3.3.1. Blok Yapısı	22
3.3.2. Merkle Ağacı (Merkle Tree)	24
3.3.3. Kriptografi.....	25
3.3.3.1. Simetrik Şifreleme	25
3.3.3.2. Asimetrik Şifreleme	26
3.3.4. Blok Zinciri Güvenlik Servisleri	27
3.3.5. Fikir Birliği (Konsensus) Protokolleri.....	28
3.3.5.1. Akıllı Anlaşmalar	29
3.3.5.2. Madencilik	29
3.3.6. Dijital İmzalama – Doğrulama.....	29
3.4. Blok Zinciri Alt Yapısı.....	31
3.4.1. Blok Zinciri Alt Yapı Çatısı	32
3.4.1.1. Hyperledger Fabric Protokolü:.....	32
3.4.1.2. Düzenleme ve iyileştirme	35
3.4.2. Çatı Mimarisi.....	36
3.4.2.1. Dağıtma işlemi	37
3.4.2.2. İşlemin başlaması	38
3.4.2.3. Sorgu İşlemi	38
3.4.3. Blok Zinciri Yürütme Mimarisi	41
3.4.3.1. Sıralı Yürütme	41

3.4.3.2. Deterministik Olmayan Kod	42
3.4.3.3. Uygulama Gizliliği	42
3.4.4. Yapıya Ek Akıllı Sözleşme	43
3.4.4.1. Sabit Güven Modeli.....	43
3.4.4.2. Kodlanmış Fikir Birliği.....	43
3.4.4.3. Modüler Fikir Birliği:	44
3.4.4.4. Blok Zinciri ile Hizmet Yönetimi	44
3.4.5. Yüksek İşlem Akışı	44
3.4.5.1. Tasarım Tartışması	46
3.4.6. Doğrulama Aşaması.....	47
4. ARAŞTIRMA BULGULARI VE TARTIŞMA.....	49
4.1. Nesnelerin İnterneti Veri İletişim Sistemi	50
4.1.1. Node-Red Hyperledger İşlem Akış	51
4.1.2. Düğümlerin Oluşturulması	52
4.1.3. IoT Yapısının Oluşturulması	55
4.1.4. Blok Zinciri Yapısının Oluşturulması.....	58
4.1.5. Hyperledger Fabric Kurulumu	59
4.1.6. Hyperledger Composer Kurulumu	61
4.1.7. Blok Zincirini Nesnelerin İnternetine Dahil Etmek	62
4.2. Nesnelerin İnterneti Blok Zinciri Performans Analizleri	67
5. SONUÇ VE ÖNERİLER.....	71
KAYNAKLAR	73
ÖZGEÇMİŞ.....	77

ÖZET

Yüksek Lisans Tezi

BLOK ZİNCİRİ MİMARİSİNİN NESNELERİN İNTERNETİNE UYARLANMASI VE PERFORMANS ANALİZLERİ

Cihan YALÇIN

Süleyman Demirel Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Mevlüt ERSOY

Günümüz koşullarında Nesnelerin İnterneti (IoT) teknolojileri insanlığın fayda ve refahı için kullanılmaktadırlar. İnsan faktörünün söz konusu olması nedeniyle veri güvenliği ve gizliliği de bir o kadar önem kazanmıştır. Bunun yanı sıra bu tür teknolojik işlemlerde, verinin elde edilmesindeki hız ve enerji tasarrufu gibi konular da öncelikli hale gelmişlerdir. IoT teknolojisinin mimari yapısı gereği, güvenlik mekanizmaları, istenilen veri güvenliği kıstaslarını yakalayamamışlardır. Bu sebeple, bu alanda birçok çalışma halen sürdürülmekte ve belirli bir güvenlik standardı oluşturulmaya çalışılmaktadır.

Bu sebepler nedeniyle; Nesnelerin İnternetinden elde edilen verilerin kimlik doğrulaması yapılmış olup; IoT yapısında enerji tasarrufu ve tek bir kriptografik yapıda toplanması adına çalışma gerçekleştirilmiştir. Gömülü bir sistem üzerine entegrasyonu sağlanmış cihazlar ve çeşitli algılayıcılar vasıtasıyla bir IoT sistemi oluşturulmuştur. Nesnelerin İnterneti (IoT) yapısını kurarken oluşturulan düğümlerde, veriler Blok Zinciri veri yapısı ile iletilmiştir. Blok Zinciri teknolojisinin nasıl kullanılabileceğine karşı fikirler geliştirilmiş ve literatüre bu yönlü bir çalışma ile destek verilmiştir.

Anahtar Kelimeler: Nesnelerin İnterneti, Veri İletişimi, Blok Zinciri, Kriptografi, Veri Gizliliği, Enerji Sarfıyatı, Performans Analizi

2019, 89 sayfa

ABSTRACT

M.Sc. Thesis

ADAPTING BLOCKCHAIN ARCHITECTURE TO THE INTERNET OF THINGS AND PERFORMANCE ANALYSIS

Cihan YALÇIN

**Süleyman Demirel University
Graduate School of Natural and Applied Sciences
Department of Computer Engineering**

Supervisor: Asst. Prof. Dr. Mevlut ERSOY

In today's conditions, Internet of Things (IoT) technologies are used for the benefit and prosperity of humanity. Because of the human factor, data security and confidentiality are also important. In addition, in such technological processes, issues such as speed and energy savings in obtaining data are prioritized. Due to the architectural structure of IoT technology, security mechanisms have not been able to achieve the required data security criteria. For this reason, many studies are being carried out in this area and a certain safety standard is tried to be established.

Due to these reasons, data obtained from the Internet of objects, to authenticate, to save energy in the IoT structure, and to collect all of these situations in a single cryptographic structure. First, an IoT system was established by means of devices integrated into an embedded system and various sensors. The data obtained from the IoT devices, transferred by Blockchain data structure. Ideas, developed against the use of Blockchain technology for emerging problems and solutions. Ultimately, this study will make a versatile and a sophisticated contribution to the literature.

Keywords: Internet of Things, Data Communication, Blockchain, Cryptography, Data Privacy, Energy Consumption, Performance Analysis

2019, 89 pages

TEŞEKKÜR

Bu araştırma için beni yönlendiren, karşılaştığım zorlukları bilgi ve tecrübesi ile aşmamda yardımcı olan değerli Danışman Hocam Dr. Öğr. Üyesi Mevlüt ERSOY'a teşekkürlerimi sunarım. Değerli fikirleri ile yön gösteren Dr. Öğr. Üyesi Asım Sinan YÜKSEL Hocam'a teşekkürlerimi sunarım. Literatür araştırmalarımnda yardımcı olan ve manevi desteğini esirgemeyen Ezgi TUFAN'a teşekkür ederim.

Tezimin her aşaması ve eğitim hayatımda beni yalnız bırakmayan AİLEME sonsuz sevgi ve saygılarımı sunarım.

Cihan YALÇIN
ISPARTA, 2019



ŞEKİLLER DİZİNİ

	Sayfa
Şekil 3.1. Kurulu Küresel İnternet Cihazları Baz Tahmini	12
Şekil 3.2. IoT Bileşenleri.....	13
Şekil 3.3. Node-Red Çizelge Görünümü.....	17
Şekil 3.4. Veri Akış Modeli	19
Şekil 3.5. Dağıtık Defter Teknolojisi	21
Şekil 3.6. Blok Yapısı.....	22
Şekil 3.7. Blok Zinciri	22
Şekil 3.8. Blok İçeriği	23
Şekil 3.9. Merkle Ağacı	24
Şekil 3.10. Sha-256 Çıktı Sonuçları	25
Şekil 3.11. Simetrik Şifreleme	26
Şekil 3.12. Asimetrik Şifreleme	26
Şekil 3.13. Dijital İmza – Doğrulama Adımları.....	30
Şekil 3.14. Alt Yapı Araçları.....	31
Şekil 3.15. Hyperledger Hareket İşleme Protokolü	34
Şekil 3.16. Emir – İşlem Yürütme Mimarisi	41
Şekil 3.17. Yüksek Seviye Veri Akış Diyagramı	45
Şekil 3.18. Tamamen Doğrulanmış Bir Blok Zinciri Düğümü.....	48
Şekil 4.1. Watson- Nesnelerin İnterneti Platform Gösterimi.	51
Şekil 4.2. Yöntem.....	52
Şekil 4.3. Meteoroloji Düğümü	53
Şekil 4.4. Hırsız Sezinleme Düğümü	53
Şekil 4.5. Hareket Algılayıcı Kodları	54
Şekil 4.6. Ultrasonik Mesafe Ölçer Kodları.....	55
Şekil 4.7. Örnek Düğüm	56
Şekil 4.8. Node-Red Başlangıç.....	57
Şekil 4.9. Tetikleme	57
Şekil 4.10. Deprem Düğümü	58
Şekil 4.11. Meteoroloji Düğümü	58
Şekil 4.12. Başlangıç Bloğu.....	60
Şekil 4.13. Eşler	61
Şekil 4.14. Sistem Testi.....	61
Şekil 4.15. Playground	63
Şekil 4.16. Model Tanımı.....	64
Şekil 4.17. Perishable Network	64
Şekil 4.18. IoT-Chain Network.....	65
Şekil 4.19. ChainCode	65
Şekil 4.20. IoT Blok Zinciri Hizmet Servisi.....	66
Şekil 4.21. Node - Red Hareket Algılama Çıktısı	66
Şekil 4.22. Watson Hareket Algılama Çıktısı	67
Şekil 4.23. Blok Zinciri Olmaksızın Enerji Sarfıyatı (Ma)	67
Şekil 4.24. Blok Zincirine Entegrasyon Sonrası Enerji Sarfıyatları (Ma)	68
Şekil 4.25. Blok Zinciri Olmaksızın/Blok Zincirine Entegre Enerji Sarfıyatları... ..	69
Şekil 4.26. Blok Zinciri/TCP/IP Veri Transfer Hızları	70

ÇİZELGELER DİZİNİ

	Sayfa
Çizelge 3.1. Blok Zinciri ile Merkezi/Dağıtık Veritabanlarının Güvenlik Servislerinin Karşılaştırılması.....	28



SİMGELER VE KISALTMALAR DİZİNİ

API	Uygulama Programlama Ara yüzü - (Application Programming Interface)
BFT	Bizans Hata Toleransı - (Byzantine Fault Tolerance)
CPU	Merkezi İşlem Birimi - (Central Processing Unit)
DDoS	Dağıtık Hizmet Reddi Saldırısı - (Distributed Denial of Service Attack)
DLT	Dağıtılmış Defter Teknolojisi - (Distributed Ledger Technology)
DoS	Hizmet Reddi saldırısı - (Denial of Service Attack)
GPIO	Genel Amaçlı Giriş Çıkış - (General Purpose Input Output)
GPU	Graphics Processing Unit - (Grafik İşlem Birimi)
IoT	Nesnelerin İnterneti - (Internet of Things)
MSP	Yönetilen Hizmet Sağlayıcısı - (Managed Service Provider)
OSN	Açık Sistem Ağı - (Open System Network)
P2P	Eşler arası ağ - (Peer to Peer Network)
PBFT	Pratik Bizans Hata Toleransı - (Practical Byzantine Fault Tolerance)
PoS	Çalışma İspatı - (Proof of Stake)
PoW	İşin İspatı - (Proof of Work)
RFID	Radyo Frekanslı Tanımlama - (Radio Frequency Identification)
SMR	Mobil Telsiz - (Specialized Mobile Radio)
VLAN	Sanal Yerel Alan Ağı - (Virtual Local Area Network)

1. GİRİŞ

Bilgisayar ağlarının ve bilgisayarlar arası iletişime olanak tanıyan cihazların uygulamalarda yoğun bir şekilde kullanılması, farklı ihtiyaç ve uygulamalara cevap verebilecek haberleşme sistemlerinin ve protokollerinin geliştirilmesine neden olmuştur. Nesnelerin İnterneti (IoT- Internet of Things), günümüzde milyarlarca cihazdan veri toplayan ve bu verilerin İnternet üzerinden yayınlanmasını sağlayan haberleşme sistemlerinin bütününe verilen isimdir. Bu sistemlerin amacı, belirli bir bölge için toplanan veriler üzerinden insanlara bilgi sağlamak veya bölge hakkında analizler yapılmasını kolaylaştırmaktır. Bu sayede hayatımızı, çalışma şartlarını iyileştirirken; evlerimiz, araçlarımız, çalışma alanlarımız ve yaşamımızı sürdürdüğümüz şehirleri akıllı interaktif alanlara dönüştürebilmektedir. Açık ve kapalı alanlarda, içinde insan ve teknoloji faktörünün bulunduğu her sektör Nesnelerin İnterneti konusu kapsamında yer almaktadır.

Ülkemizde Nesnelerin İnterneti; şehir takip sistemlerinde, sağlık hizmetlerinin takibinde, eğitim faaliyetlerinde, turizm sektöründe yoğun olarak kullanılmaktadır. Bu uygulamalarda akıllı algılayıcı sistemleri, anlık büyük veri analiz teknolojileri, bulut sistemleri, ölçeklenebilir tümleşik veya dağıtık altyapı mimarileriyle birlikte kullanılmaktadır. Tüm bu nesnelerin İnternet'e bağlanması ile bugüne kadar gerçekleşmeyen büyüklükte veriler toplanarak analiz edilmektedir. Bu analizler için siber ve fiziksel dünya arasında entegrasyon sağlanmaktadır. Cihazların iletişim özellikleri sayesinde günlük işlerin işlevselliği sezilmekte; böylelikle pek çok yaşamsal işlem otomatik hale gelmektedir.

Tüm bu işlevselliği sağlamak ve kullanıcılar için özel olan bilgileri saklamak için veri gizliliği de bu alanda önemli bir hale gelmiştir. Veri gizliliğinin sağlanması amacıyla birçok şifreleme algoritması bulunmaktadır. Ancak bu algoritmaların işlevsel oluşu, şifreleme ve şifre çözme yöntemleri ile ilgili çalışmalar devam etmektedir. Son dönemlerde bu amaca yönelik olarak "Blok Zinciri" yapıları geliştirilmiştir. Bir Blok Zincirinin temel ilkesi, birden fazla bilgisayar arasında dağılmış ve merkezi olmayan birden çok alanda kayıt tutmaktır; böylece sonraki

bloklar deęiřtirilemeden veya zincirin tm dęmleri akıřmadan bir kayıt deęiřtirilemez (Armstrong,2016;Gan,2017).

Bu tezin konu ve kapsamında, IoT yapısının temel yapısına deęinilmiř; Blok Zinciri gereksinimleri ile veri akıř diyagramları mekanizması, Blok Zincirindeki gvenlik operasyonlarının netleřtirilmesi ve daha iyi zmlerin keřfedilmesi adına alıřmalar yapılmıř, ıkan sonular ıřıęında Blok Zinciri teknolojisinde enerji sarfiyatları ve veri transfer hızı karřılařtırmalı analizleri yapılmıřtır.

Bu tez alıřmasında IoT ve Blok Zinciri'nin kriptografik yapısı incelenip, performans analizleri yapılmıřtır. Aynı zamanda gvenilir, verimli ve leklenebilir IoT gvenlik zmleri sunmak iin ele alınması gereken arařtırma konuları ile zorlukları zetlenmiřtir. Bu alıřma ile gndelik hayatta kullanılan IoT ve Blok Zinciri teknolojisi bir birleřim olarak dřnlmřtr. IoT ve Blok Zinciri gibi iki teknolojinin temel bileřenlerinin ortaya ıkarılacak olması ve enerji tasarrufunun saęlanma amacı gdlmesi, bu alıřma iin temel motivasyon kaynaęı olmuřtur.

2. KAYNAK ÖZETLERİ

Akyıldız vd. (2002) ve Kumar vd. (2006) yapmış oldukları çalışmalarda, askeri, akıllı ev sistemleri, biyolojik ve kimyasal saldırılara karşı güvenliğin öneminden bahsetmişlerdir. Ayrıca bu ağların mimari yapılarından dolayı DoS saldırılarına karşı herhangi bir savunma sisteminin geliştirilmediği ve savunma mekanizmalarının yetersiz olduğunu açıklamışlardır. Araştırmalarında Kablosuz Algılayıcı Ağlarda ortam erişim protokolleri üzerinde gerçekleştirilen güvenlik önlemlerinden bahsetmişler ve bu katmanda yapılması gereken güvenlik önlemlerini açıklamışlardır. Gerçekleştirilen çalışmada ortam erişim katmanlarının IoT güvenliğinde yetersiz kaldığını göstermişlerdir.

Zhou ve Fang (2008), yapmış oldukları çalışmada çevresel izleme, tıbbi bakım ve ev aletlerinin yönetiminde birçok kablosuz algılayıcı ağ yapısının geliştirildiğinden bahsetmişlerdir. Bu çalışmada Kablosuz Algılayıcı Ağlar ile gerçekleştirilen uygulamaların hızlı bir şekilde dağıtık hale dönüştürülmesi nedeniyle önemli ticari olayların olduğu ve bu nedenle güvenlik tehditleri ile karşı karşıya olduklarını belirtmişlerdir. Bu ağlarda güvenlik mekanizmalarının önemli olduğu ve dağıtık bir mimari de çalıştığı için zorlu bir tasarım görevi gerekmektedir. Bu kapsamda yeni bir simetrik şifreleme algoritması geliştirmişler ayrıca farklı saldırılara karşı alınabilecek önlemler hakkında bilgiler vermişlerdir.

Meghdadi vd. (2008)' ne göre, kablosuz algılayıcı ağ saldırıları, iç saldırılar ve dış saldırılar olmak üzere ikiye ayrılmaktadır. İç saldırılarda gerçekleştirilen işlerde gizli anahtarların ele geçirilmesi ile yapılan saldırılar betimlenirken; dış saldırılar ise gizli anahtara gerek kalmadan algılayıcıların işlevselliği bozan saldırılar olarak adlandırılmaktadır. Bu nedenle hem dış saldırılar hem de iç saldırılarda dikkat edilmesi gereken durumlar bu çalışmada yer almaktadır.

Ercan ve Nacak (2009), günümüzde, bilişim alanında kötü amaçlı yazılım kullanıcılarının artması nedeniyle, bilgisayarlara karşı yapılan siber saldırıların da bir o kadar artış gösterdiğini söylemişlerdir. Bu duruma karşılık olarak bilgi

güvenliği veya siber güvenlik konularının araştırılması gerektiğini, bilgi güvenliği konularının önem kazandığını belirtmişlerdir. Bu konuda gerek devlet kurumlarında gerekse ticari kuruluşlarda bilgi güvenliği farkındalığının oluşması gerektiğini söylemişlerdir. Bu hususta, bilgi güvenliği açısından birçok savunma sistemlerinin oluşturulduğunu ve gerek yazılımsal gerek donanımsal testlerinin halen yürütüldüğünü savunmuşlardır. Bu kapsamda, ülkemizde de birçok yerli şirket tarafından güvenlik duvarları ve kayıt (log) cihazları geliştirilmektedir. Bunların yanında, zararlı yazılımları engellemek amacıyla "Antivirüs" yazılımları da kullanılmaktadır. Ayrıca ülkemizde birçok firma tarafından beyaz şapkalı bilgisayar korsanları için, devlet kurumlarında nüfuz tespit sistemleri ile bilişim güvenliği açıklarını tespit edici testler yapılmakta olduğunu belirtmişlerdir. Bunların yanı sıra, açıklık tarayıcıları ve şifreleme araçlarıyla bilgi güvenliğini sağlamak için de yeni teknolojiler kullanılması gerektiğini savunmuşlardır.

Xu (2011) yapmış olduğu çalışmada; Kablosuz Algılayıcı Ağlarda yeni bir güvenlik mimarisi geliştirmiştir. Bu uygulamada kablosuz algılayıcı ağlara yapılan saldırıları veri madenciliği yöntemi ile analiz etmiş ve analiz sonucunda elde edilen durumlara göre güvenlik sistemi geliştirilmiştir. Geliştirilen güvenlik altyapısını deneysel olarak gerçekleştirmişler ve performans ölçümlerinde tatmin edici sonuçlar elde etmişlerdir. Çalışma DDoS saldırılarına yönelik olarak gerçekleştirilen bir güvenlik alt yapısıdır.

Abramov, Raz ve Herzberg, (2012)' e göre, kablosuz ağları en çok etkileyen saldırı türü DoS ve DDoS saldırılarıdır. Bu ağlarda, kullanılan protokoller üzerinden gerçekleştirilen bu yazılımlara karşı herhangi bir savunma sistemi geliştirilmediğinden ve ağ cihazlarını önemli ölçüde etkilediğinden bahsetmektedirler.

Flauzac vd. (2015)'nin yaptığı çalışmada Yazılım Tanımlı Ağ Mimarisi'nden (Software Defined Network-SDN) bahsetmiştir ve SDN tabanlı IoT mimarisinin üzerinde durmuştur. Bu çalışmaya göre Nesnelerin İnterneti için güvenlik mimarisi kavramı, yazılım tanımlı ağlara (Software Defined Networking-SDN) dayalı olarak da açıklanabilir. Bu bağlamda SDN tabanlı mimari, SDN-Domain

olarak adlandırdığımız, altyapıya sahip olsun olmasın altyapı olmadan da çalışabilir. IoT çözümleri günümüz koşullarında diğer kullanıcılara karşı bir adım önde olabilmek adına önemli kullanıcı gereksinimlerini karşılayan hızlı bir geliştirme sürecine odaklanarak sunulmaktadır. Bu çözümlerin güvenlik yönleri dikkate alınarak tasarlanması konusunda maalesef çok az yatırım yapılmaktadır. Günümüzdeki çoğu IoT çözümünün buluta veri gönderen, kimlik denetimi olmayan cihazlar vasıtasıyla yapılması; dolayısıyla saldırganların yazılım kopyalarının oluşturmaya ve buluttaki IoT arka planına aynı biçimde kötü veriler sızdırmasına olanak sağlamaktadır. Cihazların buluta açık metinler halinde veri göndermesi çözümleri doğrudan saldırılara açık hale getirmekte ve kötü amaçlı kopyaların sisteme kaydedilmesini, önlemek için orijinal cihazların tanımlanması gerekmektedir.

Bir kullanıcının ağ üzerinden veri trafiğini analiz edebilmesini ve IoT cihazları tarafından iletilen hassas veriyi anlamasını sağlamak gereklidir ve bu durum iyi niyetli olmayan verinin işlenebilmesini önlemelidir. Söz konusu sistem, algılayıcı verilerinin toplandığı ve GSM, Hücresele veri gibi teknolojiler ile buluta yüklendiği cihazlara bağlı olarak, IoT algılayıcılarından oluşmaktadır. Cihazlar orijinallik sağlamak için bulut içinde barındırılan cihazlar ve kimlik doğrulama sunucuları arasında anahtar değişimi yaparak kopya (klon) saldırılarını önleyecek şekilde ilk kimliği doğrular. Kimlik doğrulama başarılıysa, cihaz hassas verinin kötü niyetli kopyalara maruz kalmasını önlemek için verileri şifreler. Teknolojik yeniliklerin aldığı ivmeyle Blok Zinciri son yıllarda hızla teknolojilerinden biri haline gelmiş olması bir diğer bulgusudur. Merkezi olmayan ve dağıtık bir veri yönetimi çözümü olan Blok Zinciri'nin, gömülü kriptografi ve fikir birliği mekanizması ile güven tanımını teknoloji dünyasına geri getirerek herhangi bir üçüncü partiye ihtiyaç duymadan güvenlik, anonimlik ve veri bütünlüğünü sağlayacağını savunmuşlardır. (Flauzac vd.,2015).

Popov (2016)'un yaptığı çalışmada, Blok Zinciri tabanlı yapılarda güvenlik konuları ile IoT konusu bileşik biçimde işlenmiştir ve yönetimlerine dair anektodlar verilmiştir. Ayrıca veri akışı, Nesnelerin İnternetinin mimari yapısı ve anatomisi ele alınmış, klasik protokolleriyle karşılaştırma yapılmıştır. Bu

çalışmaya göre IoT, küçük ancak çok sık oluşturulan bilgi hacimleri ile karakterizedir. Çok küçük bir veri paylaşımını büyük yüklerle aktarmak mantıklı değildir, çünkü bu durumda kullanıcı bir kaç kat daha fazla enerji harcamak zorunda kalacaktır. İşin kanıtı - talep edilen tarafın oldukça karmaşık, uzun vadeli çalışmaları yerine getirme ihtiyacına dayanarak, dağıtık sistemleri, hizmetlerin kötüye kullanılmasına karşı koruma ilkesi (DDoS saldırıları veya spam postalanması gibi) maksimum performans sunma ihtiyacı göz önüne alındığında gerçekleştirilemez. Bazı öğelerinin kaçınılmaz ayrımcılık yaratması da çatışmalar yaratır ve tüm unsurlar çatışmanın çözümü için kaynak harcar hale gelir. Ayrıca, mevcut standart blok zincirinin, kullanıcılar ve doğrulayıcılar gibi rollerinin de net bir şekilde ayrıldığı heterojen bir sisteme dönüşmesi gerektiği savunmuştur.

Karel (2017), bilgi güvenliği farkındalığının oluşması adına ülkemizde ve dünyada birçok önemli olay yaşandığını söylemiştir. Bunların en önemlilerinden biri olan Dağıtık Hizmet Reddi Saldırıları (DDoS) 'nın, dünya çapındaki birçok şirket ve sosyal medya sitelerini bizzat etkilediğini belirtmiştir. Bu durumun, insanların gündelik yaşamlarını etkilediğini; gerçekleştirilen saldırıların genellikle basit yazılımlar sayesinde gerçekleştirilmesine rağmen, geniş çapta sıkıntıları da beraberinde getirdiğini savunmuştur. Bu saldırılardan biri olan "Mirai" virüsü, 2016 yılında birçok akıllı ev cihazının ve kapalı devre güvenlik kamera sistemlerini etkilemiştir. Birçok insanın evlerinde kullandıkları özel algılayıcılar istemsiz bir şekilde çalışmış ve konutlarda kullanılan birçok cihaz zarar görmüştür.

Kuzmin (2017), Nesnelerin İnterneti'nin pek çok sektörü ve şirketlerin iş yapma biçimlerini değiştirdiğini belirtmiştir. Her geçen gün daha fazla gün yüzüne çıkan bu değişimin; akıllı sistemlerin, akıllı planlamalar ve stratejiler doğrultusunda iş yapma biçimlerine de önemli katkılar sağlamakta olduğunu söylemiştir. İnsanlar, makineler ve büyümekte olan bağlantısı, her sektörün kendi iş sürecini ve iş modellerini dijitalleştirerek, bu taleplere ayak uydurma zorunluluğunu öne sermektedir. Bu dijitalleşmeye de büyük bir değişim süreci eşlik etmektedir. Dijital dönüşüm olarak adlandırdığımız bu sürecin hayatımızda yer alıp,

dokunmamış olduđu bir alan da bulunmamaktadır. Nesnelerin İnternetinin; araçların, aygıtların, sokakların, binaların, makinelerin, kısaca bağı olan "Akıllı Nesneler" in, bilgi toplayıp iletişim halinde kalarak; yazılım, algılayıcı ve gerekli ağ bağlantıları ile birleştirilmiş bir ögeler ağı olduğunu savunmuştur. Nesnelerin İnterneti, birbirine entegre olmuş cihazların, entegre oldukları ağdan özgür veri trafiğini sağlamaktadır fakat, IoT sadece birbirine bağı cihazlar olarak düşünülmemelidir. Teorik olarak, dijital kimliğe sahip, takip edilebilen ve doğrulanabilen herhangi bir ürünün de Nesnelerin İnternetine dahil olabileceğini belirtmiştir. Son yıllarda, üretim, otomasyon ve akıllı hesaplama sistemlerinin de Nesnelerin İnternetine entegre olduğunu belirtmiştir.

Khan ve Salah (2017)'ın yaptığı çalışmada; IoT mimarisi ile ilgili güvenlik zorlukları ve savunma yolları anlatılmıştır. Ayrıca protokol ve standartlar, güvenlik ihtiyaçları, veri güvenliği ve enerji verimliliği gibi konuların önemi açıklanmıştır. Aynı zamanda IoT güvenliği için blok zinciri ve potansiyel çözümler ile kullanıcı girişleri konuya entegre edilmiştir. Bu çalışmada akıllı cihazların ve yüksek hızlı ağların büyümesi ile birlikte, Nesnelerin İnterneti (IoT), kaynakları kısıtlı olan Düşük Güç Kayıplı (LPN - Low-Power Lossy) ağların ana standardı olarak görülmüştür. Bu, "nesneler" veya algılayıcılara sahip gömülü cihazların özel veya genel bir ağ aracılığıyla birbirine bağlandığı bir ağı temsil etmektedir. IoT'deki cihazlar istenen işlevselliği gerçekleştirmek için uzaktan kontrol edilebilmekte ve cihazlar arasındaki bilgi paylaşımı standart iletişim protokollerini kullanan ağ veya şebeke vasıtasıyla gerçekleştirilebilmektedir. Akıllı bağlantılı cihazlar veya "nesneler", basit giyilebilir aksesuarlardan, her biri algılayıcı içeren büyük makinelere kadar değişmektedir. IoT'nin gelecekteki önemi, günlük yaşamdaki uygulamaları nedeniyle de belirgindir. Frekans spektrumu ve kanal atamalarının yaygın kullanılmamasına, radyo tabanlı ağlarda eklenerek bant genişliğinin geliştirilmesi ve genişletilmesi gibi tekniklerin evriminden ötürü Blok Zinciri tabanlı IoT hızla büyümeye devam etmektedir. Heterojen nitelikte ve yüksek hesaplama gücüne sahip çok sayıdaki güvenliksiz akıllı IoT cihazları, saldırganlar için kolay ve çekici bir hedef durumundadır ve güvenlik, IoT için en önemli konudur.

Kuzmin (2017)'in yaptığı çalışmaya göre, IoT ekosistemi kavramsal sistem katmanlarından oluşmaktadır. Her katmanın kendi risk unsuru vardır ve kendine has güvenlik yaklaşımları gerektirmektedir. Bazı açılardan, IoT'deki uç nokta ilişkileri, birbiriyle ilişkili uç noktaların yapılandırılmasındaki çok küçük değişikliklerin, sistemin son noktasındaki işlevselliğini önemli ölçüde değiştirebileceği şekilde kolayca kaotik ilişkiler üstlenebilmektedir. IoT'deki son nokta aygıtları, ağ geçitleri ve veri merkezleri, günümüzde akla gelebilecek her tür ağ teknolojisini kullanmaktadır.

Aynı çalışma (Kuzmin, 2017)'da, Kablosuz Algılayıcı Ağları (Wireless Sensor Networks-WSN) ile Makinadan-Makineye (Machine to Machine - M2M) veya Siber Fiziksel Sistemlerin (Cyber Physical System-CPS) artık geniş kapsamlı IoT terimi için ayrılmaz bir bileşen haline geldiğini savunmaktadır. Sonuç olarak WSN, M2M veya CPS ile ilgili güvenlik sorunları IoT bağlamında ortaya çıkmaya devam etmektedir. Ancak TCP/IP protokolü ana standart olmasından dolayı; tüm dağıtım mimarisinin, IoT tarafından sağlanan hizmetlerin engellenmesinde, verilerin mahremiyeti, bütünlüğü veya gizliliğine tehdit oluşturabilecek tüm saldırılardan korunması gerektiğini savunmuştur. IoT birbirine bağlı ağlardan ve heterojen aygıtlardan oluşan bir koleksiyonu temsil ettiğinden, bilgisayar ağlarıyla ilgili geleneksel güvenlik sorunlarını miras almaktadır. Sınırlı kaynaklar, IoT güvenliğiyle ilgili fazladan zorluk teşkil etmektedir. Çünkü küçük cihazlar veya algılayıcılar içeren "nesneler" sınırlı güç ve hafızaya sahiptir. Sonuç olarak, güvenlik çözümlerinin sınırlandırılmış mimarilere uyarlanması gerekir. Son yıllarda IoT modellerinde güvenlik sorunları ile baş etmek için çok büyük çaba gösterilmiştir. Bu yaklaşımlardan bazıları belirli bir katmandaki güvenlik konularını hedeflerken diğer yaklaşımlar IoT için uçtan uca güvenlik sağlamayı amaçlamaktadır.

Fangfang vd. (2017)'nin yaptığı çalışmada Blok Zinciri uygulaması ve güvenlik sorunlarını karşılaştırılmıştır. Blok Zinciri güvenlik avantajları ve veri mahremiyeti hususuna değinilmiş; Blok Zincirinde siber güvenlik ve güvenlik zafiyetleri konuları ayrı ayrı ele alınmıştır. Bu çalışmaya göre Blok Zinciri teknolojisinin gizliliği, güvenliği ve güvenilirliği kullanıcılar tarafından

gerçekleştirilmektedir. Blok Zinciri, herhangi bir güvenilir üçüncü parti organizasyonuna ihtiyaç duymadan, verileri merkezi bulunmayan fakat güvenli bir şekilde aktarımını sağlayan, yeni bir veri depolama, iletim ve yönetim mekanizması olarak düşünülebilmektedir. Kullanıcıların bilgi işlem, depolama, özgünlük doğrulaması ile güvenilir veri tabanının korunması konularında ortaklaşa yararlanmalarını sağlayan teknik bir çözüm olan Blok Zinciri, kripto bilimde varoluşuna başlamış, varlık ve kredileme alanlarında büyümüş ve uygulamasını yavaş yavaş bilgi ve iletişim alanında bulmuştur. Blok Zinciri teknolojisinin hızla gelişmesi ile birlikte, çeşitli teknoloji dalları da giderek Blok Zinciri kavramının teknolojik üstünlüğünü ve uygulama değerini fark etmeye başlamışlardır.

Kuzmin (2017)'e göre; Ağ, Blok Zinciri'nin boyutuna bağlı olarak, cihazların sayısı ile sınırlandırılabilir. Oluşturulan trafiğin kapasitesiyle bilgi işlem gücü kritik seviyelere ulaşmayacaktır lakin kanal kapasitesi VLAN (Virtual Local Area Network) mantığıyla büyük bir rol oynamaktadır. Blok Zinciri donanımı (Device of Blok Zinciri-DOB) olarak adlandırılan protokol uygulaması, konfigürasyonların bütünlüğünü geometrik olarak arttırmayı ve IoT verisini değiştirmeyi sağlayarak Blok Zinciri tabanlı yapıların oluşturulmasında tümleşik 5G uyumlu aygıtlarda dahil olmak üzere ana kart belleğine katılımı gerçekleştirir. Düğümlerin sürekli kayıt edilmesi için standart bir gereksinim yoktur.

Fangfang vd. (2017)'ne göre; Esasen Blok Zinciri, kullanıcıların güvenilir veri tabanını, merkezi olmayan bir şekilde topluca korumasını sağlayan temel bir teknik çerçevedir. Tipik bir blok zincir sisteminde veri, düğümlerin birimlerinde üretilir ve depolanır. Ardışık düğümler, zincirlenmiş bir veri yapısı oluşturmak için kronolojik sırayla birleştirilir. Tüm kullanıcı düğümleri verilerin geçerliliği, depolanması ve bakımına katılırlar. Genellikle yeni bir blok oluşturulması, kullanıcıların yarısından fazlası tarafından onaylanmalı ve ağ çapında bir senkronizasyon gerçekleştirmek için tüm kullanıcı düğümlerine yayınlanmalıdır. Objektif olarak, uygulama katmanındaki Blok Zinciri hâlâ keşif aşamasındadır. Teknolojinin mükemmelliğine ulaşma yolunda ve uygulamanın derinleşmesine kadar devam etmek için hala uzun bir entegrasyon ve geliştirme süreci vardır.

Blok Zincirinin yenilikçi deęişimlerine raęmen, teknolojinin kendisinde hala bazı doęal gvenlik riskleri de bulunmaktadır.

Khan ve Salah (2017)'ın yaptığı alıřmaya gre; IoT'deki gvenlik tehditleri, farklı dzeylerde mevcut olan uygulamalar, arabirimler, aę bileřenleri, yazılım ve fiziksel aygıtlar gibi eřitli bileřenlerin gvenlik aklarını kullanmaktadır. Bir IoT paradigması iindeki kullanıcılar, gvenlik nlemlerinden de ıkarılabilen/uygulanmayabilen protokoller aracılığıyla bu bileřenlerle etkileřim kurarlar. Gvenlik tehditlerine karřı olan nlemler, belirli bir gvenlik seviyesine ulařmak iin bu etkileřimin farklı katmanlardaki gvenlik aklarını ele almaktadır. Bileřenlerin daęıtımını destekleyen eřitli protokoller, bu nlemlerin karmařıklığına katkıda bulunurlar. Blok Zinciri algoritmasının alt katmanında ve orta katmanında; nakliye ve yine sırasıyla st katmanında dahi yksek dzey gvenlik tehditlerinin varlıkları bilinmektedir. Blok Zinciri teknolojisi, endstri ve arařtırma toplulukları tarafından, IoT cihazlarının ynetiminde, kontrol edilmesinde ve en nemlisi emniyet altına alınmasında nemli bir rol oynayacak yıkıcı bir teknoloji olarak ngrlmektedir.

Yılmaz ve Uludaę (2017)'ın yaptığı alıřmada, IoT tabanlı akıllı sistemlerde siber saldırıları azaltma yntemlerine ve olası saldırı tiplerine aęırlık verilmiřtir. Sistem modeli ıkartılmıř ve saldırıları azaltma nerileri ile birlikte algoritmalarına aęırlık verilmiřtir. Bu alıřmaya gre genellikle Hizmet Reddetme (DoS) saldırıları denilen kullanılabilirlik boyutunu hedefleyen siber saldırılar yeni deęildir. Bununla birlikte İnternet, DoS saldırıları iin etkin ve etkili zmler ile hafifletme tekniklerini saęlamak adına hala zorlayıcıdır. Akıllı řebeke sz konusu olduęunda, bu tr saldırıların potansiyel zararı daha derin ve altyapının kendine has zellikleri nedeniyle DoS saldırılarını zor kılmaktadır.

3. MATERYAL VE YÖNTEM

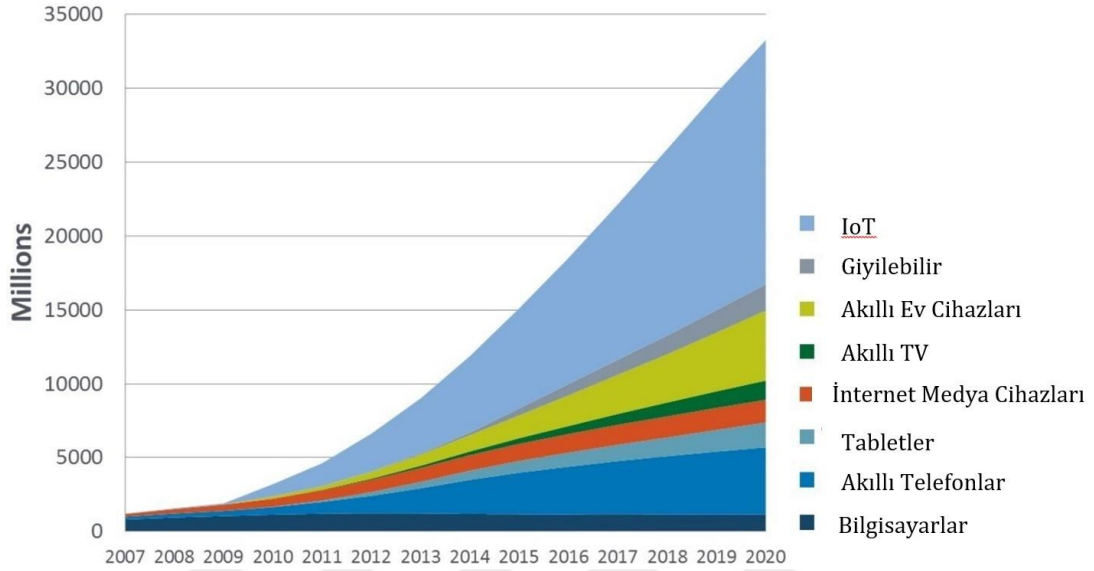
Günümüzün “Nesnelerin İnterneti” cihazları farklı algılayıcı ve altyapılardan oluşmaktadırlar. Bu cihazlar kötü niyetli yaklaşımlara karşı otonom savunma becerisine sahip değildirler. Bunun nedeni, IoT aygıtlarında kullanılan donanım ve yazılımların güvenli olmaması, IoT standartlarının olgunlaşmamış olması, tasarım, geliştirme ve dağıtım süreçlerinin hiyerarşik yapı içinde olmamasından kaynaklanmaktadır. Ayrıca ağa bağlı cihazların yoğun miktarda artışı, veri aktarım esnasında gerçekleşen siber saldırıların artması; bu alana yönelik çözüm geliştirme ihtiyacını ortaya çıkarmıştır. IoT cihazlarını ve veri aktarımlarını güvence altına almak için küresel bir güvenlik mekanizması tanımlama çabaları, Nesnelerin İnternetindeki kaynakların çeşitliliği nedeniyle de çözümü zor bir problem haline gelmiştir. Bu kapsamda ki problemin çözümü için, Blok Zinciri Tabanlı Nesnelerin İnterneti geliştirilmiştir.

3.1. Nesnelerin İnterneti

Nesnelerin İnterneti (Internet of Things - IoT), her biri birbirine benzerlik göstermeyen iletişim kanallarına sahip gömülü sistem cihazların, İnternet'e bağlanmaları yoluyla, birbirleri ile iletişim kurup veri alışverişine olanak sağlayan bir teknolojidir. Nesnelerin İnterneti kavramı; insan etkisine ihtiyaç duymadan, insanların beklentilerini ve ihtiyaçlarını karşılamak amacıyla ve interneti kullanan teknolojik aletler vasıtasıyla, aralarında veri alışverişi yapan sistemleri kapsamaktadır. Bu sistemler belirli bir ortamdaki veriyi toplayıp birbirleriyle iletişime geçerek kullanıcılara bilgi servisi oluşturmaktadır. Bu kavramın ana fikri, adresleme sistemlerini kendi bünyesinde kullanarak cihazlar ile işbirliği yapmak ve elde edilen kullanılabilir verileri işleyerek, sanal platformlar üzerinde anlamlandırılmış ve analize hazır halde kullanabilmektir.

Günümüzde IoT kapsamında tahmini olarak 10-11 milyar cihazın e bağlı olduğu varsayılmaktadır. Bu denli yüksek sayıda cihazın ağa bağlı olması, oldukça yüksek veri trafiği sorununu da beraberinde getirmektedir. Bu tip büyük verilerin analiz edilip anlamlandırma aşaması ise çeşitli zorlukları ortaya çıkarmaktadır. Tüm bu

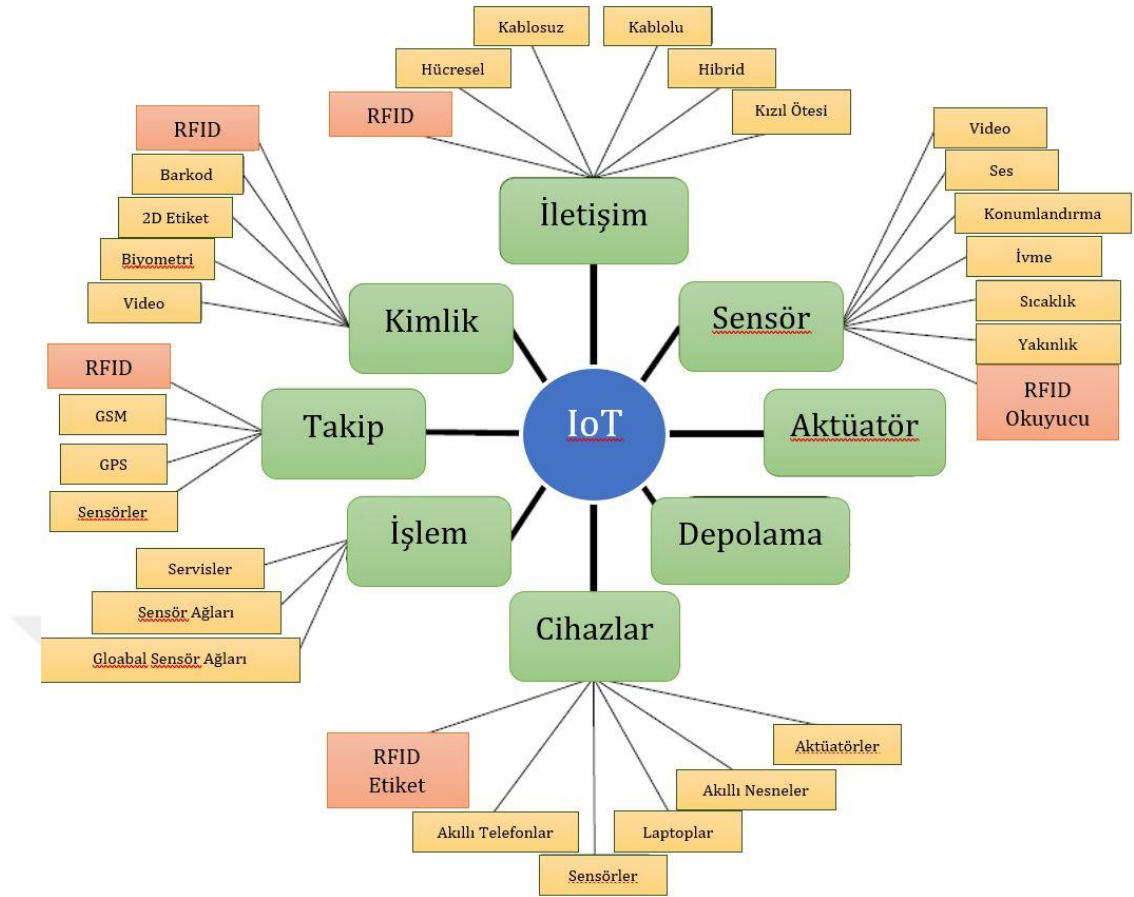
veri trafiğinin sanal ortamdaki güvenliği ve mahremiyeti de önemli bir çalışma alanı olarak karşımıza çıkmaktadır. Ağ üzerindeki cihaz sayısının artması ile (Şekil 3.1.) veri trafiğinde ciddi artış ve güvenlik risklerini oluşturmaktadır. 2025'e kadar İnternete bağlı cihaz sayısının 70 milyarı geçmesi beklenmektedir.



Şekil 3.1. Kurulu Küresel İnternet Cihazları Baz Tahmini (Prnewswire, 2014)

Dünyada cihaz artışına bağlı güvenlik riskleri, 2016 yılında yaşanan San Francisco ulaşım sisteminin ele geçirilmesi ile örneklenebilir. 2000 biletleme cihazı işlem yapmaya kapanmış ve yolcular ücretsiz seyahat etmişlerdir. Saldırı sonrası yetkililerden 73.000 USD (United States Dollar – Amerikan Doları) değerinde 100 BTC (Bitcoin) fidye isteyen saldırganlar ile bir anlaşma sağlanamayınca istasyonlardaki biletleme sistemleri devre dışı bırakılmıştır. Bu süre boyunca ziyaretçi ve bölge sakinlerinin bir hafta sonu boyunca ücretsiz yolculuk yapmaları, Amerika Birleşik Devletleri'nin 1 milyon USD'den fazla gelir kaybına uğramasına sebep olmuştur. Sonuç olarak bu siber saldırının kurbanı, büyük miktarda gelir kaybına uğrayan San Francisco şehri olmuştur.

Nesnelerin İnterneti sekiz ana bileşenden (Şekil 3.2) oluşmaktadır. Bu ana bileşenlerin bütünlüğü, gizliliği, orijinallliği; sistemlerin değerlendirilmesi adına incelenmesi gereken konulardır.



Şekil 3.2. IoT Bileşenleri (Christoph, 2009).

3.1.1. Cihazlar

Nesnelerin İnterneti ortamında, fiziksel dünyada bulunan insanlar ile etkileşime girebilmek için cihazlara ihtiyaç duyulur. Cihazların bütünlüğü, bir cihazın kötü amaçlı yazılım içermediği anlamına gelmektedir. Bir cihazın orijinalliği, çoğunlukla bağlantı uç noktası olarak görülmektedir. Bir cihazdaki gizlilik, üçüncü tarafın, cihazların dahili verilerine erişememesini sağlama konusundaki kararlılığıdır. Bu normalde cihazın bütünlüğü durumunda sağlanır. Bir cihazın kullanılabilirliği, cihazların bütünlüğüne ve güvenilirliğine, ayrıca cihazı bağlayan iletişim bölümünün mevcudiyetine bağlıdır.

3.1.2. Algılayıcılar

Fiziksel dünyayı sezinleme ve algılamada kullanılan algılayıcılar, yakaladıkları verileri dijital dünyaya aktarırlar. Algılayıcı verilerinin bütünlüğü ve gerçekçiliği,

ele alınabilecek güncel bir araştırma hedefidir. Algılayıcı verilerinin gizliliği zayıf bir ihtiyaçtır; çünkü saldırgan, fiziksel olarak kendi algılayıcısını yerleştirebilir ve aynı değerleri algılayabilir. Bu nedenle, algılayıcının kendisinde, algılayıcı gizliliğine olan ihtiyaç düşüktür ve bu nedenle gizlilik ihtiyacı iletişim gizliliğine dayanır. Algılayıcılardaki gizlilik esas olarak, algılanan fiziksel dünyayı hedeflemektedir. Görüntü verilerinde yüz bulanıklığı gibi mekanizmaların, fiziksel dünyadaki insan ve nesnelerin mahremiyetini korumak için kullanılması gerekmektedir. Algılayıcıların mevcudiyeti çoğunlukla iletişim altyapısına bağlıdır. Şu anda en sık algılayıcılardan habersiz olan insanların (Video Kameralar gibi) ortamlarındaki mahremiyetinin korumak için düzenlemeler yapılması gerekmektedir.

3.1.3. İletişim

İletişim protokolünde; cihazlar arasında bilgi alışverişini sağlamak için araştırma, bütünlük, özgünlük ve gizlilik sağlayan çözümler üretilmesi esastır (Örneğin; TLS veya IPSec). Gizlilik ihtiyacının Freenet, Onion, Thor gibi farklı yönlendirmelerle sağlanması mümkündür. Fakat geniş kitlelerce kullanılmamaktadır.

3.1.4. Aktüatörler

Dijital dünyada tetiklenen olayları, fiziksel dünyada eyleme dönüştürmek için aktüatörler kullanılır. Servo motorlar, uzaktan yönetilebilir su giriş çıkış vanaları gibi fiziksel dünyaya müdahale edebilen cihazlardır. Bir aktüatöre gönderilen verilerin bütünlüğü, orijinallliği ve gizliliği çoğunlukla iletişim güvenliğine bağlıdır. Bu nedenle aktüatörün kendisinin de düşük duyarlılığa sahip olması gerekmektedir. Bir saldırganın aktüatörü kontrol edemeyeceği durumu, güvence altına alınması gereken bir detaydır. Aktüatörlerdeki mahremiyet duruma özeldir, bu nedenle hassasiyet konusunda genel bir değerlendirme yapılamaz. Bir aktüatörün mevcudiyetinin kritik olup olmaması, aktüatörün türüne bağlıdır, ancak genellikle hassas olarak değerlendirilebilir. Düzenlemeler, algılayıcılara

benzer ve aktüatörlerin kullanımının, mahremiyeti bozmadığından emin olunmalıdır.

3.1.5. Depolama

Algılayıcılardan, tanımlama ve izleme sistemlerinden veri toplanabilmesi için depolama protokolü büyük önem teşkil etmektedir. Depolama cihazları için güvenlik mekanizmaları sağlamdır, ancak çalışma performansları halen zayıf olma durumunu korumaktadır. Verilerin depolanması, gizliliğe karşı oldukça duyarlı olduğundan ve veri ihlalleriyle ilgili raporların yaygın olması nedeniyle, kullanıcının gizliliğinin yeterli şekilde korunmasını sağlamak amacıyla düzenlemelerin genişletilmesi gereklidir. Depolamanın kullanılabilirliği, çoğunlukla iletişim altyapısının mevcudiyetine ve depolama yedeklemesi için iyi kurulmuş mekanizmalara bağlıdır.

3.1.6. İşleme

Verilerin işlenmesinde bütünlük; cihazın bütünlüğüne ve iletişimin bütünlüğüne dayanmaktadır. Ayrıca, veri işlemek için algoritmaların doğru tasarımı ve doğru uygulanması da büyük önem teşkil etmektedir. İşlemeyi çoğu zaman aktüatör eylemleri izleyebildiği için, aktüatörün yanlış komutlar alabilmesi bakımından da hassas bir durum söz konusudur. İşleme, yalnızca cihazın orijinalliğine; dolayısıyla iletişimin orijinalliğine bağlıdır. İşlemdeki gizliliğin niteliği, yalnızca cihazın bütünlüğüne bağlı olmayıp; "Dağıtılmış İşlem Durumunda" da iletişimin bütünlüğüne de bağlıdır. İşlenme aşaması, gizlilik ve depolama için kritik öneme sahiptir.

3.1.7. Takip

Fiziksel dünyadaki yer tespiti ve takibi için yerelleştirme ve izleme adımları uygulanır. Yerelleştirme ve izlemenin bütünlüğü, temelde iletişim bütünlüğüne dayanır. Ayrıca, yerelleştirmede kullanılan referans sinyallerinin bütünlüğü de çeşitli dayanak noktaları oluşturmaktadır. Aynı şekilde, orijinallik; iletişim ve

cihaz bütünlüğüne de bağlıdır. Gizlilik ve yerleştirme ile izleme verilerinin mahremiyeti, kullanıcının gizliliğini sağlamak nedeniyle hassas öneme sahiptir. Bu bağlamda gizlilik, bir saldırganın yerleştirme verilerini açığa çıkaramayacağı ve bu nedenle temel olarak iletişim gizliliğine dayanılacağı anlamına gelmektedir. Yerleştirme verilerinde gizlilik;

- Saldırganın, kişinin kimliğini göstermesine veya yerleştirme verilerinin eklendiği nesneyi açığa çıkarmasına imkan olmadığı anlamına gelir.
- Yerleştirme ve izlemenin açık bir anlaşma veya bilgi olmadan mümkün olmadığı anlamına gelir.

Yerleştirmenin kullanılabilirliği, yerleştirme için referans sinyallerinin sağlam olmasını ve saldırgan tarafından manipüle edilememesini sağlamak bakımından önemlidir.

3.1.8. Kimliklendirme

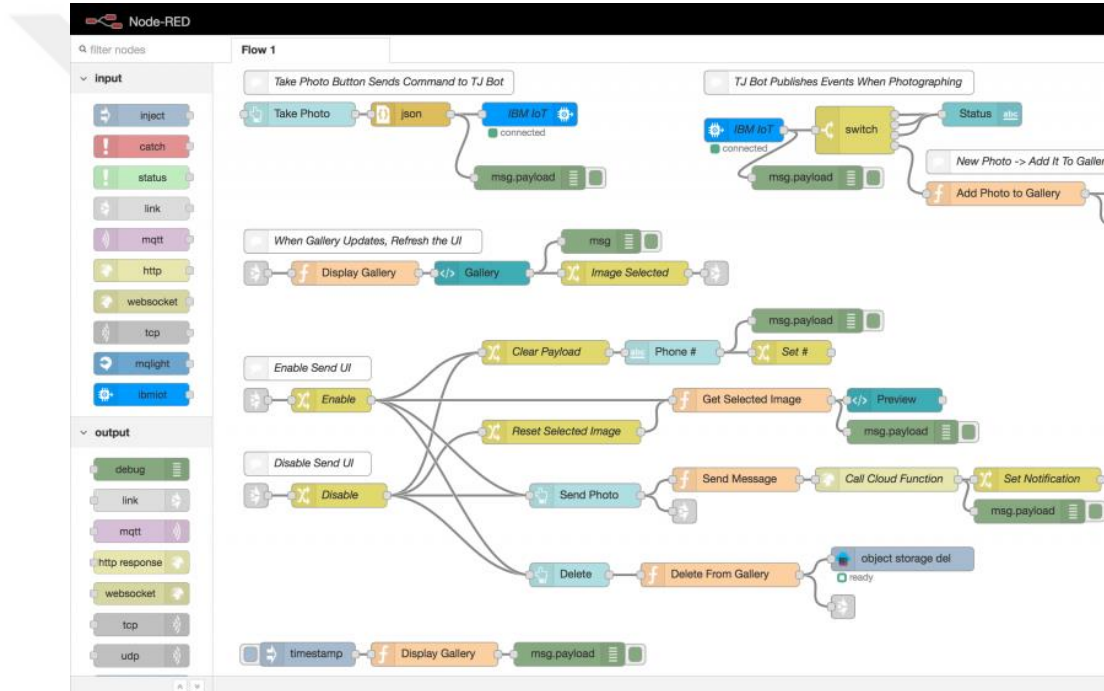
Dış dünyadan algılayıcılar aracılığıyla toplanan verilerin işleme, izleme, depolama ve diğer cihazlar arasındaki iletişimle birbirinden ayrışmasını sağlayan belirleyici özellikleri “Kimlik”i oluşturur. Kimlikleme, yerleştirme ve izleme maddesi ile aynı hassasiyetleri paylaşır. Tek fark, bütünlükte de yüksek hassasiyete sahip olmasıdır. Saldırgan, yerleştirme sürecini manipüle ettiği gibi tanımlama sürecini de kolaylıkla manipüle edebilir. Bu, esas olarak, bir saldırganın yerleştirme teknolojilerinden (Örneğin; GSM) ve manipülasyonun daha da kolay olduğu teknolojik gelişmelerden (Örneğin; RFID veya Biyometri) kaynaklanmaktadır.

Güvenlik açısından bakıldığında, IoT uygulamalarının ve platformlarının ana dezavantajı, merkezi bir buluta güvenmeleridir. Merkezi olmayan (Desantralize), Blok Zinciri temelli bir yaklaşım, merkezi bulut yaklaşımıyla ilişkili sorunların çoğunun üstesinden gelmektedir.

Bazı uzmanlar Blok Zincirinin IoT cihazları için asgari düzeyde güvenlik sağlayabileceğine işaret etmektedirler. Zaman baskısı için gereken süre dışında, Blok Zincirinde tek bir başarısızlık noktası veya kırılgnlık bulunmamaktadır (Kshetri, 2017).

3.2. Nesnelerin İnterneti Yönetim Aracı

Node-red (Şekil 3.3.); Nesnelerin İnterneti uygulamalarında donanım ve API (Application Programming Interface) bağlamak için kullanılan web tabanlı bir görsel araçtır.



Şekil 3.3. Node-Red Çizelge Görünümü (Blackstock ve Lea, 2014)

Node Red'deki veri akışı, kablolarla (modüler düğüm bağlantısı) bağlanan düğümlerden oluşan akışlar için kullanılır. Kullanıcı ara yüzü, akış diyagramına bırakılabilen düğüm şablonlarına sahip bir akış düzenleyiciden oluşur. Node Red'e erişen tüm kullanıcılar, birden fazla sayfada gösterilebilecek tek bir akışı yönetir. Bir akış oluşturulduktan veya değiştirildikten sonra, kullanıcı akışı sunucuya kaydederek yeniden Node Red sunucusunda dağıtmaya başlar (Giang vd., 2015).

Akıştaki düğümler, akış mantığını temel sınıflarından miras olarak kendilerine alırlar (Johnston vd., 2004). Bir düğüm ayrıca, kablolar tarafından abone listelerini sürdürmek için gözlemci tasarım modelini uygulayan “node.js” olay API’sindeki bir olay düzenleyicisinin alt sınıfıdır ve düğümler akışı aşağı düğümlere doğru yayarlar (Zhang vd., 2013). Başlatırken, giriş düğümleri harici servislere abone olabilir, çalıştırabilir, bir porttaki verileri dinlemeye başlayabilir veya “http” isteklerini işleme koymaya başlayabilir. Veriler, bir harici hizmetten, bir düğüm aracılığıyla işlendiğinde ya da bir “işleyici” vasıtasıyla bir yukarı akış düğümünden alındığında, düğüm bir “javascript” nesnesi ile temel sınıf düğümü arasında “send()” yöntemini çağırır. Kullanıcı, yönetici (Admin) veri gönderme yöntemi için “EventEmitter.emit()” biçiminde, verileri işleyen, ek olaylar oluşturan ya da dış hizmetler veya OS işletim sistemi (Operating System) ile iletişim kuran alt düğüm örneklerine adlandırılmış olayları gönderme yöntemini kullanırlar (Blackstock ve Lea, 2014).

Node Red, yeni düğüm ve akışlara katkıda bulunan geniş bir kullanıcı topluluğu olan IBM tarafından da desteklenir. Cihaz verileri kurulan sisteme eş biçimde IBM Watson’a (IBM tarafından geliştirilen Nesnelerin İnternet Kontrol Merkezi) gönderilebilir, alınabilir, okunabilir veya yönetilebilir. Tarayıcıya kullanıcı ara yüzü uygulamak ve veri işlemek için entegre bir HTML dosyası ile birlikte sisteme ekleyebilir.

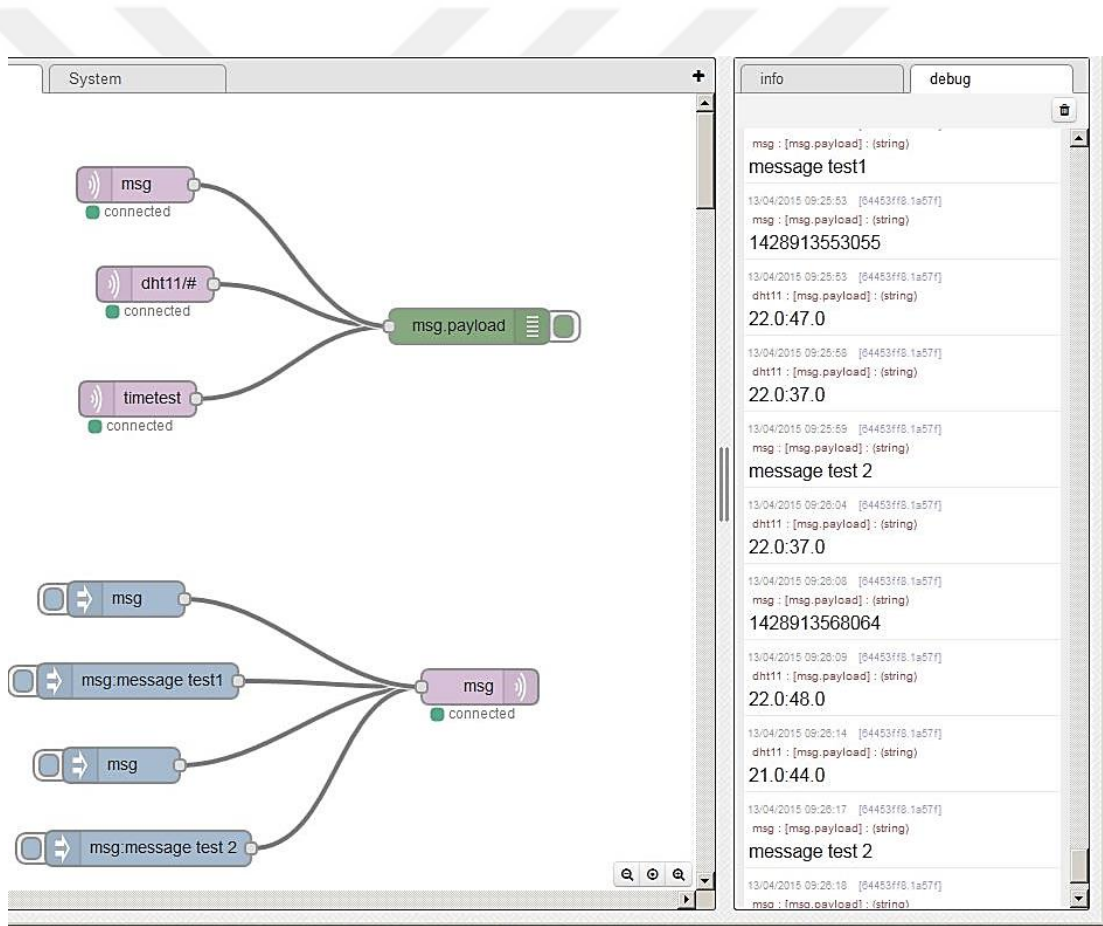
3.2.1. Node-Red Veri Akışı

Dağıtılmış akışları desteklemek için, kullanılan veri akış programı modeli (Şekil 3.4) olan Node Red’in çeşitli şekillerde genişletilmesi gerekmektedir. Sistemin herhangi bir, çok kullanıcı sistem için gerekliliği olan, belirli kullanıcıya ve cihazlara akış erişimi ve kontrol mekanizması gerekmektedir.

Düğümleri “cihaz düğümleri”, “sunucu düğümleri” veya “mobil düğümler” olarak işaretlemek mümkündür. Cihaz düğümleri, yerel cihaz bağlantısına veya cihazın özelliklerine bağlı olanlardır. Yerel dosya sistemine doğrudan erişim, algılayıcı pinlerine bağlı algılayıcı veya cihazdaki bir USB portuna bağlanması, bu özellikleri içerebilir. Bir sunucu düğümü, platformdaki belirli programlama dili,

bulutta barındırılan bir sunucunun işlem gücü, veri depolaması veya güvenlik duvarları dışındaki sunucu bağlantısı vb. için yardıma gereksinim duyabilir. Belirtilmemiş cihaz kimlikleri olan düğümler, mobil düğümler olarak kabul edilebilir ve bir aygıtta veya sunucuda barındırılabilir, kullanıcı tercihlerine bağlı olarak veya bir düğümün en iyi yerleşimini belirleyen bir sezgisel tarama yapabilir.

Benzer şekilde, farklı hat türlerinin de içerecek şekilde basit bir akış modeli genişletilebilir. Dağıtılmış bir akışta, düğümler arasındaki hatlar aynı yürütme motorundaki tüm yerel bağlantılar demek değildir, ancak yerel veya geniş alan ağı üzerinden sunucular ve cihazlar arasında veri transferini içerebilmektedir.



Şekil 3.4. Veri Akış Modeli

3.2.2. Node-Red Ön Yüz

IoT uygulamaları oluşturmak oldukça karmaşık bir işlemdir. Çeşitli protokoller ve hizmetlerin entegrasyonunun yanı sıra, “nesnelerden” ve hizmetlerden zaman

serisi veri akışlarını yönetmek ve toplamak için güvenli ve esnek bir bağlantı olanağı gerektirmektedir. Geliştiricilerin, hem fiziksel hem de çevrimiçi dünyadaki değişiklikleri tetiklemek ve bunlara tepki vermek için verileri, işlem öncesi ve sonrası yöntemleri kullanmak için bir araca ihtiyaçları vardır. Geleneksel programlama araçlarını kullanarak gerçek zamanlı etkileşimli karma makineleri oluşturmak mümkün olsa da, zor olabilmektedir. Geliştiricilerin yeni protokolleri ve API (Application Programming Interface-Uygulama Programlama Arayüzü) öğrenmeleri, veri işleme bileşenleri oluşturmalarını ve bunların birbirine bağlantısını gerektirmektedir. Kullanım kolaylığını korurken daha fazla esneklik sağlamak için, bilgisayar programlarının bağlı hatlar boyunca veri alışverişinde bulunan “siyah kutu” düğüm ağlarını bağlaması, yönlendirilmiş grafikler olarak modellenen bir veri akış paradigması oluşturmaya sebep olur. Görsel veri akışı programlama dilleri, diğer birçok alanda (Çok çekirdekli işlemciler, mühendisliklerden yararlanılarak yapılan yüksek performanslı paralel hesaplama gibi) kullanılmıştır (Blackstock ve Lea, 2016).

3.3. Blok Zinciri

Blok Zinciri, bileşenlerin sistemdeki her bir bileşen tarafından güvenilmesi gereken merkezi bir entegrasyon noktasına güvenmeden, güvenilmeyen büyük bir katılımcı ağında merkezi olmayan ve işlemsel veri paylaşımı için ortak eşler üzerinde anlaşmalar bulabilecekleri yeni yazılım mimarileri formları sağlayan bir teknolojidir (Xu vd., 2016).

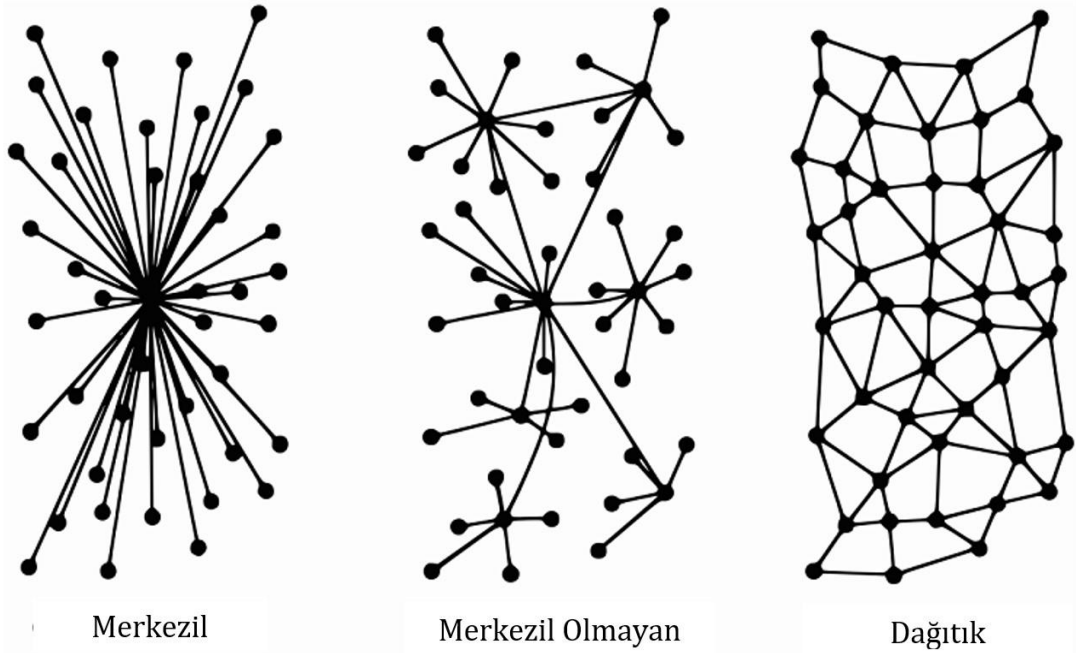
Blok Zinciri, zamana göre sıralanıp yayılan ve sürekli büyüme eğilimi gösteren bir veri alt yapısıdır. Bloklar, yapılan işlem(ler)i ve kendinden önceki bloğun adresini kendi bünyelerinde kayıt altında tutarlar. Blok Zinciri, işlemlerin değiştirilemez listesinin tutulduğu bir kayıt defteridir.

Blok Zinciri Sisteminin Temel Özellikleri şunlardır;

- İşlemler merkezi değildir.
- İşlemler Uçtan Uca (Peer to Peer - P2P) ağda tüm düğümlere yayınlanır (Broadcast).

- İşlemler birden fazla düğüm tarafından onaylanır ve sonunda Blok Zincirine eklenir.
- Sistemdeki bütün hesaplar halka açıktır (Public), ama anonimdir. Hesap ID'si aynı zamanda açık anahtar (Public Key) olarak kullanılır.
- Madenci düğümler, işlemleri bloklar halinde toparlarlar (Karaarslan ve Akbaş, 2017).

Blok Zinciri, birden çok düğümün veya veri işlemlerini takip eden bilgisayarların, dağıtık ağında çalışan bir kronolojik veri tabanıdır. İşlemlerin kaydedildiği ve doğrulandığı yollardan dolayı "Blok Zinciri" denir. Belirli sayıda işlem hakkında bilgi düzenlenir ve "bloklara" şifrelenir. Her yeni blok, düğümler veya bilgisayarlar ağ üzerinden bir Fikir Birliği (Konsensus) ulaştığı zaman onaylanır. Blok Zinciri ve çoğu DLT (Dağıtık Defter Teknolojisi) sistemleri; süreklilik, ölçeklenebilirlik, enerji tüketimi, güvenlik, mahremiyet, kişisel ve hassas verilerin korunması ile ilgili ihtilaflarla ilgilidir. Blok Zinciri sistemleri birçok durumda, eski BT sistemleri ile tamamlayıcı veya entegre olarak faaliyet göstermişlerdir. Başka bir yaklaşım, Blok Zincirinin IoT, Yapay Zeka, Robotik veya katkılı üretim gibi endüstriyel bağlamdaki diğer dijital teknolojilerle olası kesişimidir (Medium Corp, 2018). DLT yapısı (Şekil 3.5.) aşağıdaki resimde gösterilmiştir.



Şekil 3.5. Dağıtık Defter Teknolojisi (Garderen, 2016).

3.3.1. Blok Yapısı

Blok Zinciri içerisinde “Transactions” adı verilen ve sürekli büyüyen işlemlerin - kayıtların listesini tutan yapılar “Blok” (Şekil 3.6.) adını alır. İlk blok yapısına Başlangıç Bloğu (Genesis Block) adı verilir (Taş ve Kiani, 2018).

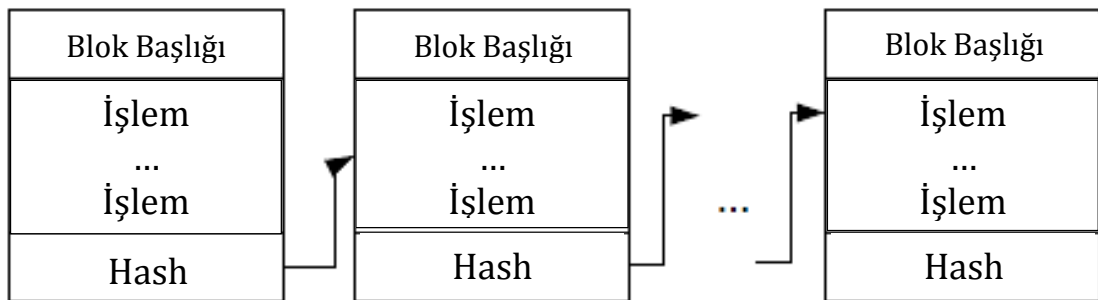
Başlangıç Bloğu (Genesis Blok)



Şekil 3.6. Blok Yapısı

Blokların güvenliği ve doğruluğu kriptografik algoritma ve/veya şifreler ile sağlanmaktadır. Her bir blok, kendinden önceki bloğun kriptografik şifresini kendi içeriğinde barındırır ve dosyanın doğruluğunu kontrol eder. Kriptografik şifreleme işlemi “Kriptografik Özetleme” yani “hash” olarak adlandırılır. Hash işlevi herhangi uzunluktaki bir girişi, belirlenen algoritmaya göre tek tip (Uniform String) metin dizisine kodlar. Girilen ve değişen her bit için dahi üretilen hash kodu değişir, böylelikle dağıtık kayıt defterine kaydedilen, zincire eklenen bir blok müdahale edilemez hale gelir.

Blok yapısı (Şekil 3.7.) aşağıdaki resimde gösterilmiştir (Sousa vd., 2018).



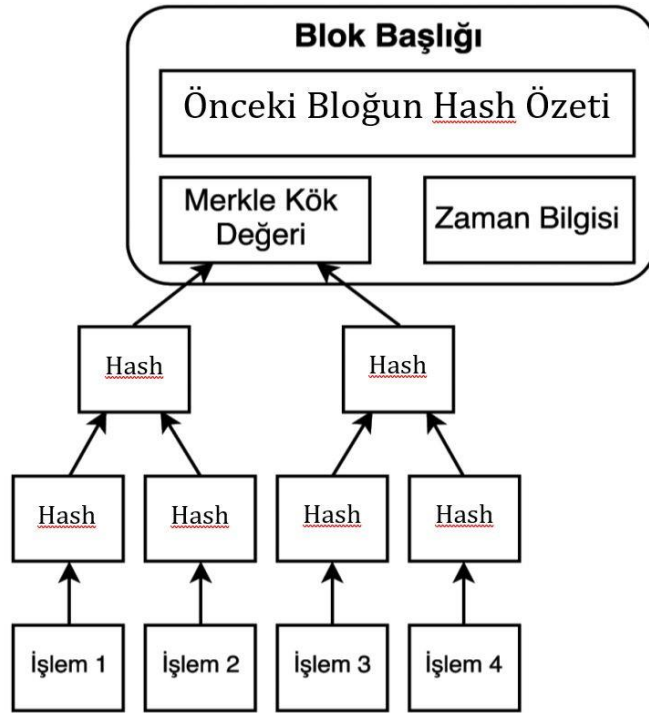
Şekil 3.7. Blok Zinciri (Sousa vd., 2018).

Blok Zinciri, işlemleri blok birimlerine kaydeder. Her blokta benzersiz bir kimlik ve bir önceki bloğun kimliği bulunur. Başlangıç (Genesis) adı verilen bu ilk blok, protokolün bir parçası olarak tanımlanır. Geçerli bir blok, mevcut bloktaki işlemlerde şifreleme işlemi için adreslenebilen bir eşsiz kimlik içerir. Bu işleme

madencilik adı verilir ve terminolojinin kötüye kullanılması ile blok madenciliği olarak da adlandırılır. Spesifik olarak şifreleme, sonucu belirlenmiş bir değerden daha küçük olması gereken çifte bir karmadır. Bu değere göre ayarlanan işlem zorluğu, dinamik olarak bloklar üretilecek şekilde ayarlanır, hash'in karması her 10 dakikada bir ortalama hızda bulunur (Eyal ve Sirer, 2018).

Birden fazla bloğun birleşmesiyle oluşan zincirin, blok içeriği (Şekil 3.8) temel olarak 2 parçadan oluşur;

- **İşlemler (Transactions);** Blok verileridir.
- **Blok Başlığı;** Veri bütünlüğü ve kontrol amaçlı üst bilgidir.
 - ✓ Bir önceki bloğun kriptografik özeti,
 - ✓ Transactions Merkle Kök değeri,
 - ✓ Zaman bilgisi verilerini barındırır.

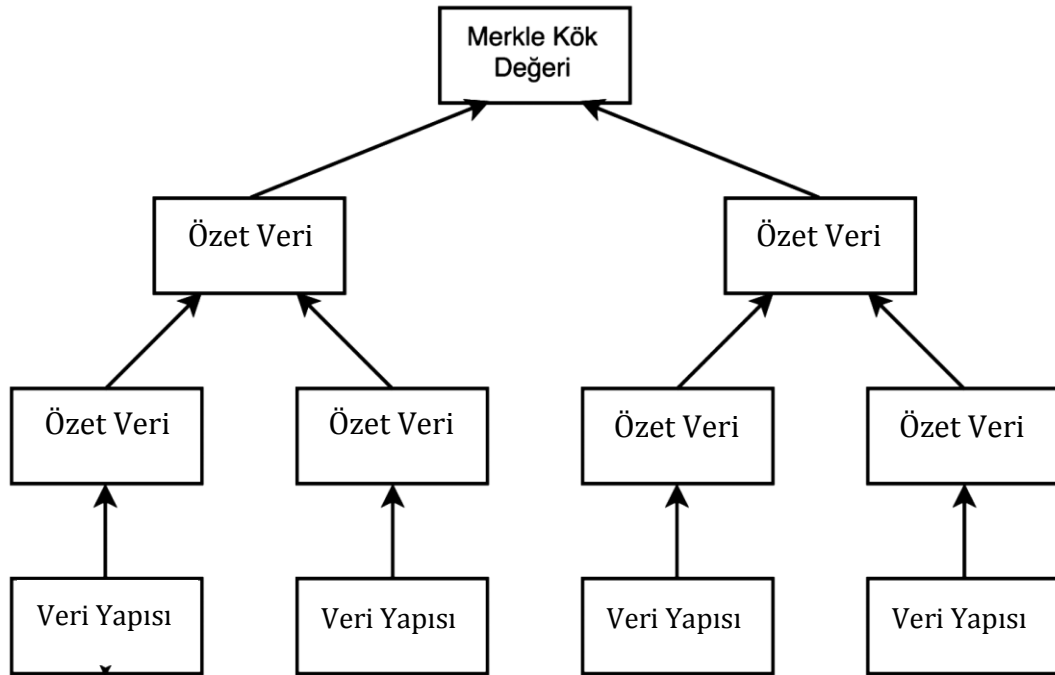


Şekil 3.8. Blok İçeriği (Usta ve Doğantekin, 2017)

Hangi blokların zincire ekleneceği, eşlenik durumu sağlamak için kullanılan ağ ve platformlara bağlı olarak çeşitli Fikir Birliği – Uzlaşma (Konsensus) protokolleri, Madencilik (Mining) veya açık kaynak Akıllı Sözleşmelere göre değişiklik göstermektedir.

3.3.2. Merkle Ağacı (Merkle Tree)

Merkle Ağacı yapısı, veri aktarım boyutunun aşılması durumunda güvenli ve hızlı bir şekilde doğrulama için kullanılan Eşler Arası “Peer-2-Peer – P2P” ve Blok Zinciri kullanan büyük platform ve sağlayıcılarının standardına oturmuş bir modeldir. İkili (binary) yapısında ağaç modeline benzer özetleme değerlerine ayırır ve ufak parçalar haline getirilmiş özetleme değerlerinin sıralı olarak hashlerini tutar. Bölünen her bir veri değeri hash fonksiyonundan geçerek kendi hashlerini oluşturur. Bu işlem gönderilen verinin büyüklüğüyle doğru orantılıdır ve taşınabilecek maksimum veri boyutuna indirilinceye dek devam eder. Bu safhada yapraklar (Merkle Leaf) bölünen her eşlenik parçanın hashlerini birleştirerek yeni bir hash elde eder. Bu işlem köke (Merkle Root) erişinceye kadar devam eder. Merkle Ağacı gösterimi (Şekil 3.9.)’da gösterilmiştir.



Şekil 3.9. Merkle Ağacı

3.3.3. Kriptografi

Kriptografi; bir bilgi ya da verinin istenmeyen kişilerce ele geçirilmesi ve dahi istenmeyen taraflarca okunamayacak hale dönüştürülmesinde kullanılan matematiksel yöntemler bütünüdür. Rastgele gibi görünen karmaşık sayı ve rakam karmaşasında, kullanılan kriptolama algoritmasına bağlı olarak, istenmeyen kişilerce veri ele geçirilmiş olsa dahi ilgili anahtara sahip olmayan kişilerce verinin orijinal yapısına geri çevrilemezler. Sadece gönderen ve gönderilen kişinin sahip olduğu eşsiz anahtarların doğrulanması ile veri orijinal haline dönüştürülebilir.

Her bir bloğun kendine has eşsiz bir kimliği (Unique Identity) vardır. Blok içerisinde yapılmaya çalışılan geriye dönük herhangi bir değişiklik teşebbüsünde, üretilen hash kodu değişeceği için, sonrasında yer alan blokta ki var olan hash kodu ile uyuşmayacaktır. Aşağıda “SHA-256” hash algoritmasının string veri girdisi değişiminde, çıktı sonuçlarının değişimi (Şekil 3.10.) görülmektedir.

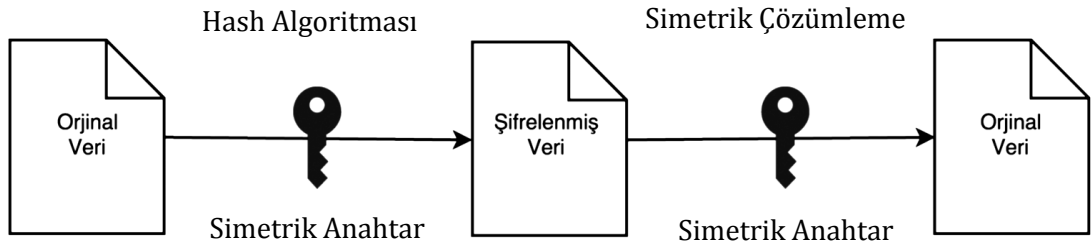
Örnek	SHA-256 Karşılığı (okunabilirlik açısından 16'lık sayı düzeninde gösterilmiştir)
erhan	A7C3962E7BD1F5C65FDD9D97CC993B231CFF60C8296ED9F9590EAD5B0813D1D0
serhan	37B081FA6506D4B937F5A9EB893B45823BDBA49D5DF840B24AF4122BA29E540D
Serhan	508B4498D3A57B759CC171A541CA4F2BBB2DC2B18442665E5EE1E50AF37F7F7A

Şekil 3.10. Sha-256 Çıktı Sonuçları

3.3.3.1. Simetrik Şifreleme

Simetrik şifreleme algoritmalarında, şifreleme ve şifre çözme için tek bir anahtar kullanılmaktadır. Güçlü yanı hızlı olmasına karşın şifreleme ve şifre çözme

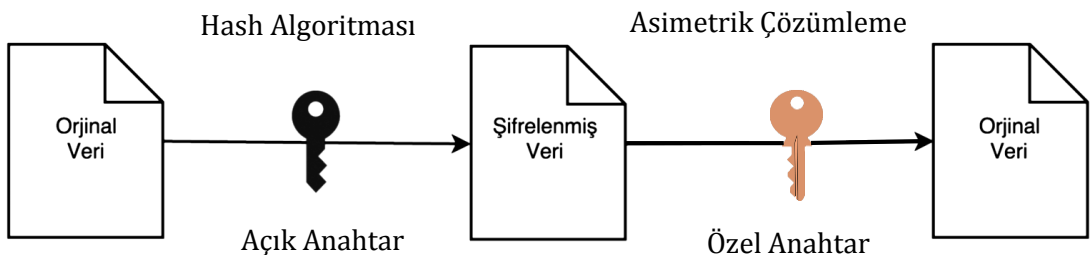
işleminde tek bir gizli anahtarla işlem yapıldığı için gizli – simetrik anahtarın üçüncü şahıslar tarafından ele geçirilmesi durumunda gelen, gönderen ve içerik bilgilerinin hepsi okunabilmektedir. Sık kullanılan bir yöntem olmasına karşın, gönderim sonrası değiştirilmesi ve tekrar paylaşılması ayrı bir yük ve açık oluşturmaktadır. Aşağıda simetrik şifreleme (Şekil 3.11.) yönteminin gösterimi yapılmaktadır.



Şekil 3.11. Simetrik Şifreleme

3.3.3.2. Asimetrik Şifreleme

Asimetrik şifreleme algoritmalarında şifreleme ve şifre çözme işleminde iki farklı anahtar kullanılmaktadır. Şifreleme yapan anahtar açık anahtar (Public Key), şifreyi çözen anahtar ise özel anahtar (Private Key) olarak adlandırılır. Her kullanıcıda gönderim ve alım işlemlerinin yapılacağı bir adet açık anahtar ve çözümleme işleminin yapılacağı, asla paylaşılmaması gereken özel anahtar bulunmaktadır (Şekil 3.12.). Açık anahtarlar herkese dağıtılabilir lakin hangi anahtarın kime ait olduğundan emin olunmalıdır. Açık anahtar ile şifrelenmiş bir veri yalnızca ilgili özel anahtar ile çözülebilmektedir. Aynı şekilde özel anahtarla şifrelenmiş bir içerikte yalnızca ilgili açık anahtar ile çözülebilmektedir.



Şekil 3.12. Asimetrik Şifreleme

Asimetrik şifreleme yöntemi, sadece ilgili alıcı tarafından okunması istenilen durumlarda kullanılmaktadır. Bu kapsamda gönderilen mesaj başka biri tarafından ele geçirilse dahi, ilgili özel anahtar bilgisine sahip değilse içeriğine ulaşması mümkün olmayacaktır.

3.3.4. Blok Zinciri Güvenlik Servisleri

Saldırganların sistemi ele geçirmesi için, ağdaki düğümlerin çoğunluğunu ele geçirmesi gerekmektedir. Düğümlerin dağıtık olması, bu olasılığı da oldukça düşürmektedir. Blok Zinciri yapısında hash fonksiyonları aktif olarak kullanılmaktadır. Sistemdeki bir işlemi değiştirmek, zincirdeki tüm blokları da hesaplamayı gerektirecektir. Bu faaliyet, çok yüksek bir işlem gücüne gereksinim duyacaktır. Zincirdeki her değiştirilen blok için diğer düğümlerin de ikna edilmesi ve buna dayanarak İşin İspatı (Proof of Work - PoW) hesaplamalarının da gerçekleştirilmesi gerekecektir. Bu da %51 saldırısı olarak tanımlanmaktadır. Çünkü ağdaki bütün düğümlerin madencilik işlemci gücünün en az %51'ine sahip olması gerekecektir. Saldırı teorik olarak mümkün olsa da pratikte bu tür bir saldırı olası değildir ve etkisinin kısa süreceği ifade edilmektedir (Sousa ve Bessani, 2015).

Destek Kanıtı (Proof of Stake - PoS) kullanıldığında ise, saldırganın bütün kripto paranın en az %51'ine sahip olması gerekecektir ki Ethereum'da sadece konsorsiyumun elinde bulunan bir güçtür (Sousa vd., 2018).

Güvenlik servisleri açısından Blok Zincirinin, merkezi ve dağıtık veri tabanlarından farkı Çizelge 3.1.'de verilmiştir. Blok Zinciri ile kayıt bütünlüğü, kullanılabilirlik (Availability) ve hata toleransı (Fault Tolerance) karşılaştırılabilir. Blok Zinciri tabanlı sistemler, gizlilik servisini (Çizelge 3.1.) hedeflememektedir (Karaarslan ve Akbaş, 2017).

Çizelge 3.1. Blok Zinciri ile Merkezi/Dağıtık Veritabanlarının Güvenlik Servislerinin Karşılaştırılması (Karaarslan ve Akbaş, 2017).

	Blok Zinciri	Merkezi Veritabanı	Dağıtık Veritabanı
Kayıt Bütünlüğü	Yüksek	Orta	Orta
Kullanılabilirliği	Yüksek	Düşük	Orta
Hata Toleransı	Yüksek	Düşük	Yüksek
Gizlilik	Düşük	Yüksek	Orta

İşlemi gerçekleştiren makinelerin bütün kayıtları ortada olsa da kime ait olduklarının belirli olmamasından dolayı mahremiyet (Privacy) tabanlı servisler de verilebilmektedir.

Başlangıçta finansal kayıtları tutmak için tasarlanan Blok Zinciri, bireysel hesaplama kaynaklarını da uygulayabilmek adına genelleştirilmiş bir çerçeve haline dönüştürülebilmektedir (Wood, 2014; Azaria vd, 2016).

Her bir hesaplama kaynağı, şifreleme ile güvence altına alınan işlemler ile durumlar arasında geçiş yapabilen tekil bir işlem yöneticisi olarak düşünülebilir. Yeni bir işlem yöneticisi oluştururken, düğümler geçerli işlem geçişlerini tanımlayan ve Blok Zincirine yükleyen mantığı kodlarlar. Buradan itibaren bloklar, artan bir işlem trafiği ile birlikte, bir önceki bloktan bir dizi geçerli geçiş yapıldığında, işlem yöneticisini mevcut duruma uyarlarlar. İş kanıtı (PoW), Fikir Birliği algoritması ve onun altında bulunan eşler arası protokoller, makinaların durumunu, geçiş mantığını kötü niyetli saldırılara karşı korur ve bu bilgiyi sisteme katılan tüm düğümlerle paylaşır. Bu nedenle, düğümler makinaların durumlarını sorgulayabilir ve tüm ağın kabul edeceği şekilde, kesinliği yüksek bir sonuç elde edebilirler (Azaria vd., 2016).

3.3.5. Fikir Birliği (Konsensus) Protokolleri

Blok zincirlerinin bütün düğümlerde aynı olabilmesi için kimin değişiklik yapacağını belirleyen kurallar bütünüdür. Çalışma Kanıtı (Proof of Work, PoW) ve İş Kanıtı PoS (Proof Of Stake) yaklaşımlarından burada söz etmek mümkündür. PoW her düğümün değişiklik önerisi yapabilme hakkı kazanmak

için öncelikle çözmesi gereken bir bulmaca gibidir. Başkalarının çözmesinin zor olduğu ama işleyen tarafından kolaylıkla doğrulanabilecek bir değerdir. PoS, PoW'deki hesaplama yerine, sisteminde sahip olduğu zenginliğe (Kripto Para) göre bloğu yaratacak olanın seçildiği bir yaklaşımdır (Karaarslan ve Akbaş, 2017).

3.3.5.1. Akıllı Anlaşmalar

Ethereum, Hyperledger, AWS (Amazon Web Service) gibi proje ve hizmetler ile Blok Zincirinde akıllı anlaşmalar yapmak mümkündür. Bu anlaşmalarla; değer tutan, veri kaydeden ve çeşitli hesaplama görevleri için bloklara çalıştırabilir kod ekleyen uygulamaların geliştirilmesi mümkün olmaktadır (Karaarslan ve Akbaş, 2017).

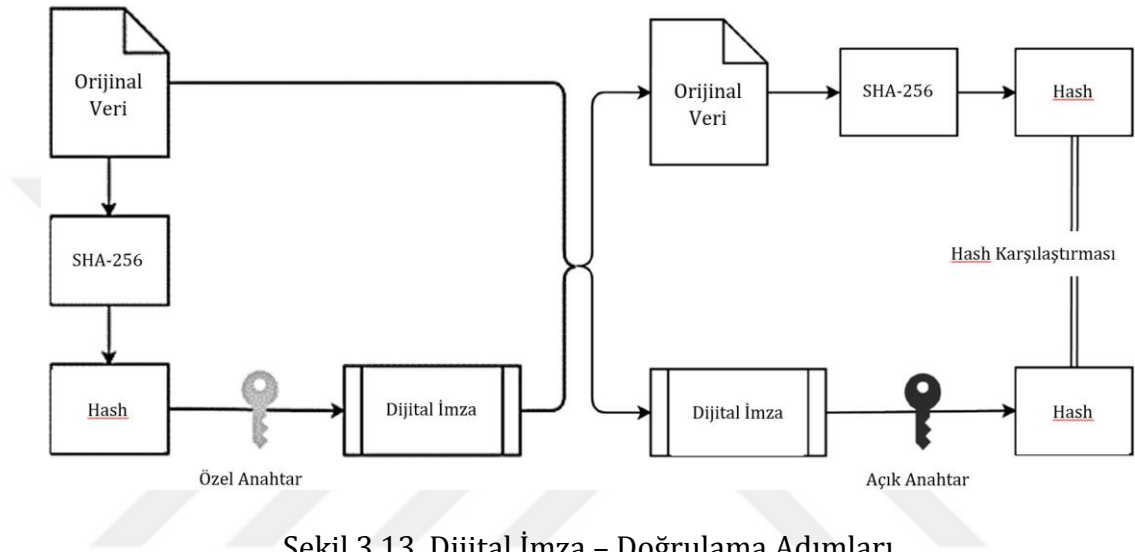
3.3.5.2. Madencilik

İşlemlerin gerçekleşmesini sağlayan bilgisayarlardır. Önceleri işlemci gücü kullanılırken, ekran kartlarındaki işlemcilerin veya bu iş için üretilmiş özel kartların kullanılması söz konusu hale gelmiştir ve ciddi maliyet hesaplaması gerektirmektedirler. Hash işlemleri çoğunlukla ekran kartlarının işlemcileri üzerinde GPU hesaplama gerçekleştirilmekte ve H/s (saniyede hash hesaplama) birimi ile Kilo (Bin)-Mega (Milyon)-Giga (Milyar) biriminden güçleri tanımlanmaktadır. Bir ekran kartı Mh/s güçlerinde çalışmakta, makinelere takılan çoklu kartlarla yüksek madencilik güçlerine ulaşabilmektedir (Karaarslan ve Akbaş, 2017).

3.3.6. Dijital İmzalama – Doğrulama

Bu safhada, gönderilen bir verinin gerçekten gönderici tarafından gönderildiği ya da transfer esnasında bozulmaya uğradığı, eksik gönderildiği, paket kaybına uğradığı veya dış bir kaynak tarafından müdahale edilmediğini doğrulamak amacıyla dijital imza (Şekil 3.13) kullanılmaktadır.

Gönderici tarafından oluşturulan özetleme algoritması (SHA-256) kendi özel anahtarıyla şifrlenerek gönderilir ve alıcı, kendi özet değerini oluşturarak, göndericinin açık anahtarıyla şifreyi çözer. Alıcı kendi oluşturduğu özet değer ile göndericinin özet değerini karşılaştırarak doğru yerden gelip gelmediği bilgisine ulaşabilir. Şayet iki özet bilgi karşılaştırıldığında içerik uyuşması yaşanmıyorsa mesajı içeriğini oluşturan ve özet değeri imzalayan kişi sizin alıcınız/göndericiniz değildir ya da mesaj içeriğine transfer sırasında müdahale edilmiş demektir.



Şekil 3.13. Dijital İmza – Doğrulama Adımları

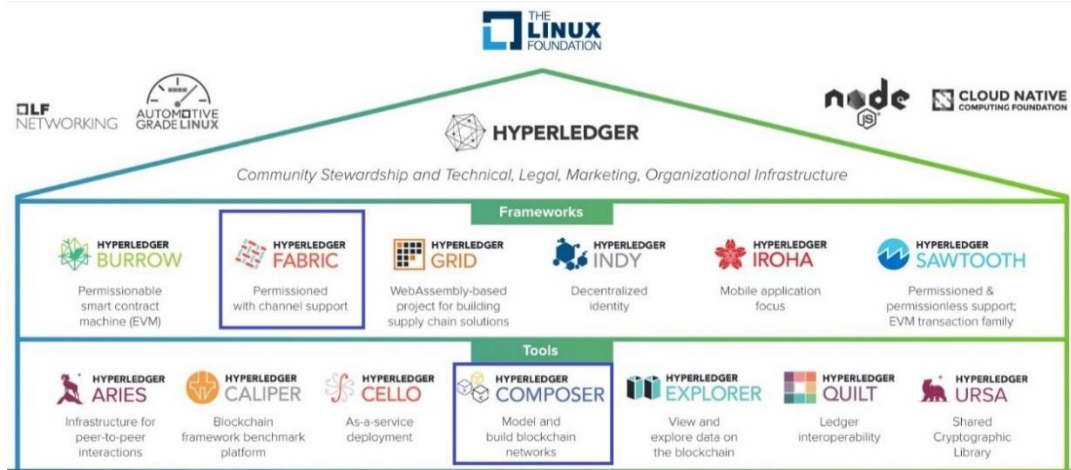
2015 yılında ortaya çıkan “Mirai” saldırısı, Nesnelerin İnterneti alanında yapılan çalışmalarda veri gizliliği, veri güvenliği, kimliklendirme gibi konuları gündeme getirmiştir. Veri mahremiyetine had safhada ihtiyaç duyulan bu platformda, mimari olarak da güvenli ve büyük veri kümelerini kendi bünyesinde toplayabilecek bir yapı arayışı, karşımıza “Blok Zinciri” kavramını çıkarmaktadır.

Nesnelerin İnterneti, mimari savunmasızlığı ile veri aktarımının düğümler üzerinden merkezi noktalara aktarılmasının, güvenliğin sadece kimliklendirme mekanizmasıyla yetersiz kalacağını göstermektedir. Verinin belli bir kanal üzerinden aktarılması ve herkese açık olması, düğümler arasında da verinin şifreli bir şekilde aktarılmasını gerektirmektedir. Bu alanda Makineden Makineye İletişim, Makineden İnsana İletişim, Araçtan Araca İletişim, Araçtan İnsana İletişim ve Kablosuz Vücut Ağı gibi kişisel verilerin merkezi noktalara aktarılmasında verinin gizliliği ve güvenliği önemli hale gelmektedir. Bu nedenle düğümler üzerinde hızlı, güvenli ve güvenilir bir şifreleme mekanizmasının

olması gerekmektedir. Aynı zamanda düğümler arasındaki veri iletişiminin de gizlilik ihlallerinin yapılmaması gerekmektedir. Bu amaçla düğümlerdeki algılayıcılardan elde edilen verilerin merkezi düğüme aktarılırken verinin güvenliğini ve gizliliğini sağlayacak bir sistemin geliştirilmesi gerekmektedir. Simetrik veya asimetrik şifreleme algoritmalarında anahtarlar kullanıldığı için kişisel anahtarların ele geçirilmiş olması kimliklendirme konusunda gizlilik ihlallerini oluşturabilmektedir. Asimetrik şifreleme algoritmalarında iki anahtarlı yapının kullanılması bile, kişilerin özel anahtarlarının tespit edilmesi sayesinde veri dinlemesi ile veri gizliliğini ortadan kaldırabilmektedir. Bunun yanında asimetrik şifreleme algoritmalarında her ne kadar kimlik doğrulama problemi giderilmiş olsa da kullanılan sistemde çok fazla işlemci harcanmasında ve bu nedenle enerji sarfiyatının artmasına neden olmaktadır.

3.4. Blok Zinciri Alt Yapısı

Hyperledger, Linux tarafından fonlanan, iş uygulamalarını hedefleyen ve IBM'in ev sahibi olduğu Blok Zinciri tabanlı modüler yaklaşım bir Blok Zinciri çatı platformudur. Genellik ve esneklik kavramları üzerine inşa edilmiş olup temel tasarım olarak geniş bir yelpazede akıllı sözleşmelere destek verir. Bu sistemin temel bir özelliği de, genişleyebilir oluşu ve özellikle Blok Zinciri oluşturmak için çoklu hizmet alt yapı araçları (Şekil 3.14) sağlayabiliyor olmasıdır. Blok Zinciri yönetimi için birden fazla akıllı sözleşme kullanım olanağı sağlar (Sousa vd., 2018).



Şekil 3.14. Alt Yapı Araçları (Medium.corp, 2018).

Tipik olarak eşler arası, bir ağ içinde konuşlandırılmış ve dağıtılmış bir verinin kaydını tutan açık bir veri tabanıdır. Blok Zincirin temel mantığıyla eş misyon, işlemleri (Transaction) içeren, blok ve sürekli büyüyen bir kayıt listesinden oluşur (Vukolic, 2015).

Bloklar, kriptografik karmalar (hash) ve uzlaşma mekanizması tarafından müdahalelere karşı korunmaktadır (Androulaki vd., 2018).

Blok Zinciri, izin verilen ya da izne gerek olmaksızın modele bağlı kalabilir. İzne gerek olmaksızın kayıtlar, merkezi olmayan ve anonim bir şekilde DLT (Distributed Ledger Technology) tarafından muhafaza edilebilir (Cachin, 2016).

3.4.1. Blok Zinciri Alt Yapı Çatısı

Hyperledger, içerisinde bir açık kaynak projesinin çatısını oluşturmaktadır. Çeşitli bileşenlerin eklenebildiği uygulamalar ve bunlar için var olan akıllı sözleşmeleri desteklemek amacıyla izne gerek olmayan bir Blok Zinciri sistemidir.

Fabric kullanıcıları, "Chaincode" kullanarak işlemlerini yönetmek için, akıllı sözleşmeleri kullanırlar. Chaincode'lar; Fabric içerisinde akıllı sözleşmelerin tam eşleniğidir. Fabric üzerinde işlenmiş olan kodlar, veri tabanında (sürümlü bir anahtar olarak modellenmiştir) ve onay ilkelerine uygun davranan, onaylayan eşlenikler tarafından yürütülür ve tamamlanır. Servis, sorumlu olduğu işlemi eklemek için blok dağıtımın yanı sıra her komut ile de Kayıt Defterine (ledger) eklenir. Servis sağlayıcısının sorumluluğu, her eklenen blok için dağıtık kayıt defteri atamak, hem de hangi blokların deftere ekleneceğine karar vermektir.

3.4.1.1. Hyperledger Fabric Protokolü:

Onaylanan eşler tarafından yürütülüp geçerli kılınan ve veri tabanının durumunu onay ilkelerine bağlı olan bir koddan oluşur. Fabric Protokolü, dağıtık defter için

bloklar oluşturmaktan ve her bloğun deftere eklendiği hizmetlerden sorumludur. Tüm bu işlemler, aşağıda (Şekil 3.15.) adımlar halinde açıklanmıştır:

Adım 1.

İstemciler bir eş misyon faaliyeti yaratıp eşlenikler için mesaj gönderirler. Bu mesaj bir zincir kodu fonksiyonunu işleme çağırmak için kullanılır. Bu durum bir zincir kodunu ID (Kimlik)'e dahil etmek amacıyla ve zaman süresi ile işlem yükünü belirlemek amacıyla yapılır.

Adım 2.

Onaylanan eşlenikler eş misyon eylemlerini benzetirler ve bir uzlaşma anlaşması imzalarlar. İstemci uygun biçimde yetkili kılındıysa, bir zincirin erişim kontrol politikalarını değerlendirerek işlemi gerçekleştirebilir. İşlemler daha sonra mevcut duruma göre yürütülür. Eşlenikler istemciye bu oluşumun sonucunu iletir (mevcut durumlarıyla ilişkili “node”ları okuyabilir ve yazabilir). Bu noktada ek bir güncelleme yapılmaz.

Adım 3.

İstemciler eş misyon fonksiyonunu toplayabilirler ve birleştirebilirler. İstemciler karara varmak amacıyla eşlenikleri onaylar, eğer kümelerin okuma ve yazma sonuçları eşleniklerle uyuyorsa onay politikası yerine getirilmiş demektir. Eğer bu şartlar sağlanırsa, istemciler onaylı bir okuma yazma kümesini barındıran “zarf” yaratır, bu zarfın içinde onaylı kanal ID’si de bulunur. Kanal, Fabric ağı içerisinde özel bir Blok Zinciri ve veri bölümünü sağlayan noktadır. Kanalin her bir eşleniği, kanalın spesifik kayıt defterini paylaşır. Bahsi geçen söz konusu zarf, bir işlemi temsil eder.

Adım 4.

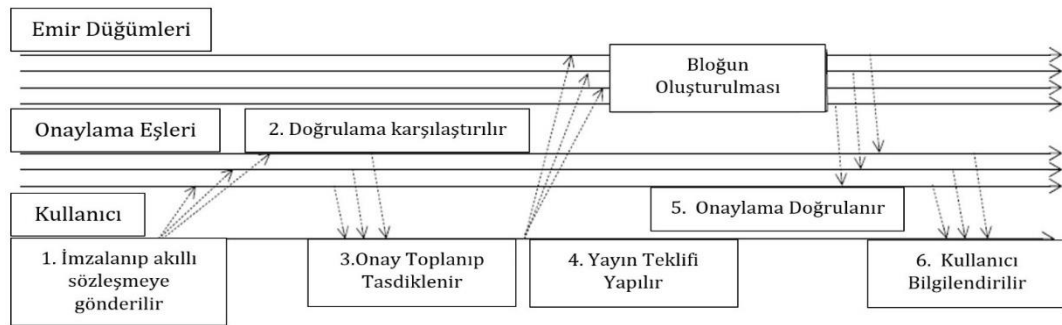
İstemciler eş misyon eylemi yayını için öneri sunarlar. Hizmet sağlayıcı, zarfın içeriği okumaz. Yalnızca ağdaki tüm kanallardan zarf toplar, zarfları yönlendirir ve bu zarfları içeren işaretli blok zincirlerini oluşturur.

Adım 5.

Zarf; blokları kanaldaki eşleniklere iletir. Blok içindeki zarflar (1) onay politikalarını sağlamak ve yerine getirmek amacıyla, (2) okuma seti değişkenleri için eş durumlarında değişiklik olup olmadığını kontrol etmek için (okuma seti işlem yürütücüsü tarafından üretildikten itibaren) kontrol edilip doğrulanır. Bu maksatla, okuma seti bir işlemi benzetimler iken, eşleniklerin okunmasını sağlayan bir dizi sürümlü anahtar içerir (Adım 2). Bu doğrulamaların başarısına bağlı olarak, zarflar içerisinde bulunan hareket önerisi geçerli veya geçersiz olarak işaretlenir.

Adım 6.

Eşler, alınan bloğu kanalın blok zincirine bağlar. Her geçerli işlem için, yazma kümeleri eşlerin mevcut durumları için kararlılık gösterir. İstemciye, işlemin kanalın blok zincirine kesin olarak eklendiğini bildirmesi için bir olay tetiklenir, işlemin geçerli veya geçersiz kabul edilmesine karar verilir. Geçersiz işlemlerin de deftere eklendiğine dikkat edilir, ama eşleniklerde bu durum geçerli olmaz. Bu aynı zamanda, kötü niyetli istemcileri tespit etmeyi mümkün kılma avantajını da beraberinde getirir, çünkü eylemleri de kaydedilir (Sousa vd., 2018).



Şekil 3.15. Hyperledger Hareket İşleme Protokolü (Androulaki vd., 2018; Sousa vd., 2018).

Fabric protokolünün önemli yönü, onaylamanın (2.adım) ve onaylamanın (5.adım) farklı eşleniklerde yapılabilmesidir. Ayrıca, onay sırasında zincir kod uygulamasının aksine, onaylama kodunun deterministik olması gerekmektedir. Yani aynı durumdaki farklı eşler tarafından onaylanan aynı işlem aynı çıktıyı üretecektir.

3.4.1.2. Düzenleme ve iyileştirme

Hizmet servisinin uygulanması, bütünlüğünü korumayı amaçlayan blok başına tek bir imza ile sağlanır. Fabric her bloğun kendisiyle ilişkilendirilmiş iki imza içermesini bekler. Biri imzanın ilişkili bloğun bütünlüğünü koruması amaçlanırken, diğeri blok işlemlerini bir yürütme bağlamında adapte edebilmek için tasarlanmıştır. Bununla birlikte, sınırlayıcı faktör imza oluşturma oranı, yani veri boyutu küçük ön sınırı olan küçük işlemler olduğunda, bu önemli bir performans düşüşüne yol açabilir. Temel olarak normal yükü ve son yapılandırma zarfını depolayan bir bloğun kimliğini kapsadığı için; ikinci imzayı almak için yeterli neden bulunmamaktadır. Sadece ilk imzanın üretilmesinin Blok Zincirinin bütünlüğünü korumak için yeterli olduğu anlaşılmaktadır. İstem düğümlerinden kötü niyetli davranışlara karşı her biri yerel olan bloklar toplanır ve imzalarını üretirler. Bu da, defterin ön uçlarında tutulan yerel kopyalara eklenen bir blok akışı ile sonuçlanır.

Ancak Fabric kod temeli, bir blok akışı yerine zarf akışını üreten ve yalnızca kilitlenme emri veren servisler için daha uygundur. Örneğin, bir zarf akışı aldıktan sonra fabric, hem blok oluşturmak hem de bunları zincire eklemek için yöntemler kullanır. Ayrıca, blokları zincire ekleyen yöntemler de daha önce tartışılan imzaları oluşturur (Sousa vd., 2018).

Blok Zinciri, üzerinden bağlı düğümler tarafından işletilen, dağıtılmış bir protokol aracılığıyla "Güvenilir" bir bilgisayar servisine benzer. Hizmet, tüm düğümlerin bir miktar hissesine sahip olduğu bir varlığı temsil eder veya yaratır. Düğümler hizmeti yürütme hedefini paylaşır, ancak daha fazlası için mutlak biçimde birbirlerine güvenmezler. "İzinsiz" bir Blok Zincirinde, herkes bir düğümü çalıştırabilir ve CPU döngülerini harcama yoluyla işleme katılabilir ve bir "iş güvenliği" örneği sergileyebilir. Öte yandan, onaylama ve protokolde yer alan "izinli" model kontrolündeki bloklar, bu düğümler ile tipik olarak kimlikleri oluşturabilir ve bir uzlaşma protokolü ortaya koyabilirler.

Hyperledger Fabric, akıllı sözleşmeler yapmak için dağıtık bir defter platformu kullanan, çeşitli fonksiyonların takılabilir uygulamalarına izin veren modüler bir yapıya sahip olan, bilinen ve kanıtlanmış bir teknolojidir.

3.4.2. Çatı Mimarisi

Fabric Blok Zinciri, bir ağı oluşturan bir dizi düğümden oluşur. Yapıya izin verildiği için, ağa katılan tüm düğümlerin modüler bir üyelik servis sağlayıcısı (MSP) tarafından sağlanan bir kimliği vardır. Bir Fabric ağındaki düğümler aşağıdaki üç rolden birini üstlenirler:

- İstemciler, işlem tekliflerinin yürütmek için bilgi sunumu yaparlar, yürütme aşamasını düzenlemeye yardım eder ve son olarak sipariş vermek için işlemleri yayınlarlar.
- Eşler, işlem tekliflerinin yürütür ve işlemleri doğrularlar. Tüm eşler, bir karma zincir şeklinde, tüm işlemler deftere kaydedilene dek, ekleme yaparak veri yapısını korurlar. Tüm eşler tüm işlemleri yürütmezler. Teklifler, yalnızca onaylayan eşler (veya basitçe onaylayıcılar) olarak adlandırılan alt küme tarafından, işlemin ilgili olduğu zincir kod politikasında belirtildiği şekilde yürütülür. Geçerlilik aşamasında değerlendirilen bir onay politikası, güvenilmeyen uygulama ve geliştiricileri tarafından seçilemez veya değiştirilemez. Bir onay politikası, yalnızca zincir kodu ile parametrelendirilebilen, Fabricteki işlem doğrulaması için statik bir kütüphane görevi görür. Yalnızca belirlenmiş yöneticiler, onay politikalarını sistem yönetimi fonksiyonları ile değiştirme iznine sahip olabilir. Tipik bir onay ilkesi, zincir kodunun, bir işlem için onaylayanları, onay için gerekli olan bir grup eşlenikler biçiminde belirtmesini sağlar. Setlerde “beşte üç” gibi matematiksel monoton bir ifade kullanılır.
- Hizmet Servis Düğümleri (OSN) veya sadece hizmet verenler, hizmet servisinin toplu olarak oluşturan düğümlerdir. Kısacası, sipariş hizmeti, her işlemin yürütme aşamasında hesaplanan durum güncellemelerini ve bağımlılıkları içerdiği için tüm işlemlerin toplam sırasını belirler.

Onaylayan eşler, kriptografik imzaları ile birlikte onaylayarak bir eş şifreleme imzası oluşturur. Bu tasarım seçimi, Fabric'te fikir birliğini mümkün olduğunca modüler kılar ve fikir birliği protokollerinin değiştirilmesini basitleştirir.

Bir istemci, onay ilkesi gereği belirtilen protokoller vasıtasıyla işlem talebi gönderir. Her işlem daha sonra belirli eşler tarafından gerçekleştirilir ve çıktısı kaydedilir. Yürütmeden sonra, işlemler, bloklar halinde gruplandırılmış, hizmet sağlanmış ve bir onaylanmış işlem sırası üretmek için, takılabilir bir konsensüs protokolü kullanan servis aşamasına girer. İşlem girdilerini tamamen sipariş eden standart aktif çoğaltmadan farklı olarak, Fabric, işlem çıktılarını yürütme aşamasında hesaplandığı gibi durum bağımlılıklarıyla birleştirir. Her bir eşlenikten sonra, onaylama işlemlerinde onaylama işlemlerinden denetleyici değişikliklerini ve onaylama aşamasında yürütmenin tutarlılığı inceler. Tüm eşler işlemleri aynı sırada onaylarlar ve bu onaylama eylemi deterministiktir.

Bir Fabric ağı aslında aynı hizmet servisine bağlı birden fazla Blok Zinciri'ni destekler. Bu tür ağlarda, her bir Blok Zinciri bir kanal olarak adlandırılır ve üyeleri olarak farklı eşleniklere sahip olabilir. Kanal, Blok Zinciri ağının bir bölümü gibi düşünülebilir ancak, kanallar arasında fikir birliği koordine edilmez ve her bir kanaldaki toplam işlem sırası diğerlerinden ayrıdır. Tüm istemcileri güvenilir olarak kabul eden belirli dağıtımlar, eşler için kanallar arası erişim kontrolü de yapabilir.

3.4.2.1. Dağıtma işlemi

Parametre olarak bir zincir kodunu (akıllı sözleşmeyi), yine zincir kodu olan eşlere kurdurur ve eşleri üzerine yükler.

3.4.2.2. İşlemin başlaması

Daha önce bir konuşlandırma işlemi ile kurulmuş olan belirli bir zincir kodunun işlemi başlatılır, (argümanlar işlemin türüne özgüdür) duruma göre girişleri okuyabilir, yazabilir ve başarılı veya başarısız olduğunu gösterebilir.

3.4.2.3. Sorgu işlemi

Her bir zincir kodu, işlem sırasında kendi kalıcı girişlerini tanımlayabilir. Okuma izni verilen bir defterde, işlem uygulandığı için kimlik doğrulama ve yetkilendirme için bir güvenlik altyapısı da bulunur. Açık anahtar sertifikaları aracılığıyla kayıt ve işlem yetkilendirmesini, bant içi şifreleme yoluyla gerçekleşen zincir kodunun gizliliğini destekler. Daha basit bir tabirle, ağa bağlanmak için her eşleniğin üyelik hizmetlerinin bir parçası olan kayıt sisteminden bir kayıt sertifikası alması gerekmektedir.

Ağa bağlanmak ve işlem yapabilmek için, gerekli olan işlem sertifikalarını almak ve bir eşleniğe (Peer) yetki verilmesi elzemdir. Akıllı sözleşmeler için var olan zincir kodu, simetrik anahtar şifrelemesiyle sağlanır ve Blok Zinciri için kayıt sertifikasına sahip tüm eşler için kullanılabilen blok zincire özgü bir anahtarla işlemler yürütülür (Cachin, 2016).

Hyperledger ayrıca; standart ve genel amaçlı programlama dillerinde yazılmış uygulamaları, yerel bir şifreleme ve para birimlerine bağlılık olmadan çalıştıran ilk Blok Zinciri sistemidir. Bu alana özgü dillerde veya kripto para birimlerine güvenilmesi için “Akıllı Sözleşmeler” gerektiren mevcut Blok Zinciri platformlarına böylelikle keskin bir tezat oluşturmaktadır. Her eş, defterin bir kopyasını saklar, eşler hareketleri doğrulamak için, bloklar halinde karma bir zincir oluşturmak amacıyla uzlaşma protokolü yürütürler. Bu süreç tutarlılık için gerekli olduğu gibi işlemleri de istem kabul ederek defteri oluştururlar (Nakamoto, 2008).

Açık (Public) veya izinsiz bir zincirde, herkes belirli bir kimlik sahibi olmadan katılım sağlayabilir. Kapalı veya izinli (Private) zincirler ise tipik olarak yerel bir şifreleme para birimi içerir ve genellikle Çalışma İspatı (PoW)'na ve ekonomik teşviklere dayanan fikir birliği kullanırlar (Chain.co, 2014).

Öte yandan açık blok zincirleri, bilinen ve tanımlanmış bir dizi katılımcı arasında bir Blok Zinciri arasında çalıştırmaktadır. İzinli bir Blok Zinciri, ortak bir amacı olan ancak birbirine güvenmeyen bir grup varlık (Fon, Mal veya Bilgi Alışverişinde Bulunan İşlemler gibi) arasındaki etkileşimi güvence altına almanın bir yolunu sağlar (Androulaki vd., 2018).

Fabric'in mimarisi, güvenilmeyen bir ortamda güvenilmeyen bir kodun dağıtılması için yeni bir yürütme sırası doğrular. İşlem akışını, sistemdeki farklı varlıklarda da çalıştırabilmek için üç adıma ayırır:

- (1) İşlem yapmak, doğruluğunu kontrol etmek ve onaylamak,
- (2) İşlem semantiğinden bağımsız olarak bir uzlaşma protokolü yoluyla hizmet vermek,
- (3) Eş-zaman nedeniyle zaman ve sonuç koşullarını etkileyebilecek uygulamalara özgü, güven varsayımlarına göre işlem onayı vermek.

Fabric merkezi olmayan veri tabanlarında sıklıkla bulunan pasif ya da birincil yedek kopyalamayı kullanır, ancak ara katman tabanlı asimetrik güncelleme işlemleri ve hata toleransı (Byzantine Fault Tolerance) güvenilen ortamlara taşınır. Fabric'te, her işlem yalnızca bir eş alt kümesi tarafından gerçekleştirilir (onaylanır), paralel yürütmeye izin verilir ve potansiyel belirsizlik giderilebilir. Esnek bir onay politikası, belirli bir akıllı sözleşmenin doğru bir şekilde yürütülebilmesi için hangi emsallerin veya kaç tanesinin onaylanması gerektiğini belirler (Miller vd., 2016).

İkinci olarak, işlemlerin kayıt durumu üzerindeki etkileri, her eşlenik tarafından bireysel olarak yürütülen deterministik doğrulama adımında, ancak aralarındaki

toplam istem üzerinde uzlaşmaya varıldıktan sonra yazılır. Bu, Fabric'in işlem onayına göre uygulamaya özgü güven varsayımlarına saygı duymasına olanak verir. Dahası, durum güncellemelerinin sıralaması, fikir birliği için modüler bir bileşen olan atom yayını (Atomic Broadcast)'na devredilir, böylece merkezi olmayan ve mantıksal olarak işlem yürüten ve defteri tutan eşleniklerden ayrıştırılır. Fikir birliği modüler olduğundan, uygulaması belirli bir dağıtımın güven varsayımına göre uygulanabilir. Blok Zinciri eşlerini aynı fikir birliğini uygulamak için de kullanmak mümkün olsa da, iki rolün ayrılması esneklik sağlar ve birinin CFT (Çökmeye hata toleransı) veya BFT (Bizans Hata Toleransı) istemi için iyi kurulmuş araç setlerine güvenmesine izin verir.

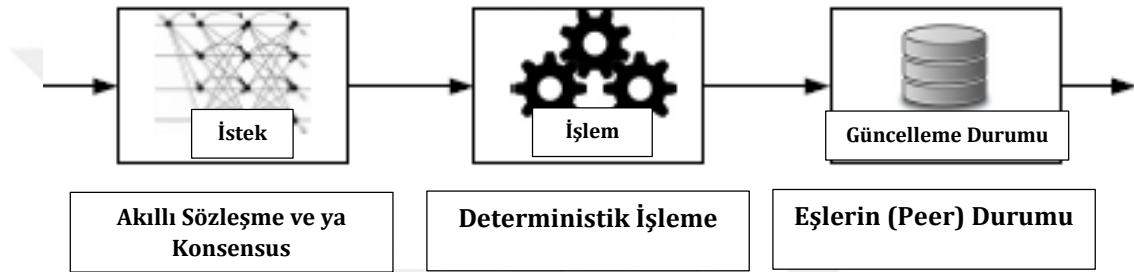
Genel olarak, hata toleransı modelinde pasif ve aktif replikasyonu birleştiren bu hibrid replikasyon tasarımı ve yürütme-yönetme paradigması, Fabric mimarisindeki temel yeniliği temsil eder ve Fabric'e esnek güven varsayımlarını destekleyen izin verilen bloklar için ölçeklenebilir bir sistem oluşmasında destek olur.

Bu mimariyi uygulamak için, Fabric aşağıdaki bileşenlerin her biri için modüler yapı taşlarını içerir:

- Bir istem servisi atomik olarak denetleyici güncellemelerini yayınlar ve işlemlerin sırası üzerinde fikir birliği kurulmasını sağlar.
- Bir üyelik hizmeti sağlayıcısı, eşleri kriptografik kimliklerle ilişkilendirmekten sorumludur.
- İsteğe bağlı eşler arası bilgi akışı servisi, tüm eşlere servis hizmeti vererek blok üretimini yayar.
- Fabric'te yapılan akıllı sözleşmeler, izolasyon için bir "Konteyner" ortamında gerçekleştirilir.
- Her bir eş, yerel olarak yerel bilgiyi, sadece ekleme bloğu şeklinde ve anahtar değer deposundaki en son durumun anlık görüntüsü olarak saklar (Androulaki vd., 2018).

3.4.3. Blok Zinciri Yürütme Mimarisi

Tüm önceki blok zincir sistemleri (izinli veya izinsiz) emir-işlem yürütme mimarisini (Şekil 3.16.) izler. Bu, Blok Zinciri ağının önce bir uzlaşma protokolü kullanarak işlemleri istem olarak kabul ettiği ve ardından, bunları tüm sıralarda aynı anda yürütmesi anlamına gelir. Mevcut izin verilen blok zincirleri, tipik olarak PBFT (Practical Byzantine Fault Tolerance) için diğer protokoller tarafından sağlanan BFT fikir birliği mekanizmasını kullanır. Bu esnada, aynı sıra yürütme yaklaşımı ve klasik aktif SMR uygulamasını takip eder.



Şekil 3.16. Emir – İşlem Yürütme Mimarisi (Androulaki vd., 2018)

3.4.3.1. Sıralı Yürütme

İşlemlerin tüm eşler üzerinde art arda yapılması, Blok Zinciri'nin gerçekleştirebileceği etkin verimi sınırlandırır. Bilhassa, iş hacmi uygulama gecikmesi ile ters orantılı olduğu için, bu en basit akıllı sözleşmeler hariç herkes için bir performans darboğazına dönüşebilir. Böyle bir Blok Zincirinin performansını ciddi şekilde azaltan bir hizmet reddi (DoS) saldırısı, uygulanması çok uzun zaman alan akıllı sözleşmelerde tahribata yol açabilir. Örneğin, sonsuz bir döngüyü yürüten akıllı bir sözleşmenin ölümcül bir etkisi vardır, ancak durma sorunu çözülemez olduğu için otomatik olarak algılanamaz. Bu sorunla başa çıkmak için, yürütme, kriptografi temelli halka açık programlanabilir bloklar kullanır.

3.4.3.2. Deterministik Olmayan Kod

Hizmet yürütme mimarisi için bir diğer önemli sorun deterministik olmayan işlemlerdir. Akıllı sözleşmelerin genel amaçlı bir dilde (Örneğin; Go, Java, C/C++) yazılması daha anlaşılır görünmekte ve Blok Zinciri çözümlerinin benimsenmesini hızlandırmaktadır.

Maalesef, genel diller uygulamanın sağlanması için pek çok sorun teşkil etmektedir. Uygulama geliştiriciler açıkça deterministik olmayan işlemler sunmasa bile, gizli uygulama detayları aynı yıkıcı etkiye sahip olabilir. İşleri daha da zorlaştırmak için, bir Blok Zincirinde, deterministik uygulama yaratma yükü, potansiyel olarak güvenilmeyen programcıya aittir. Kötü niyetle oluşturulan, sadece deterministik olmayan bir sözleşme tüm Blok Zincirini durma noktasına getirmek için yeterlidir.

3.4.3.3. Uygulama Gizliliği

Denetleyici düğümlerin planına göre, birçok izin verilen sistem tüm eşler üzerindeki akıllı sözleşmeleri yönetmektedir. Bununla birlikte, izin verilen bloklar için amaçlanan birçok kullanım durumu gizlilik gerektirir, yani akıllı sözleşmeler mantığına, işlem verilerine veya defter durumuna erişimin kısıtlanabileceği durumlar için gizlilik protokolü şarttır. Aynı kodu her yerde çalıştırmak yerine, aynı durumu tüm eşlere yaymak yeterlidir. Bu nedenle, akıllı bir sözleşmenin uygulanması, bu göreve güvenilen meslektaşların bir alt kümesi ile sınırlandırılabilir, bu da yürütmenin sonuçları için de geçerlidir. Bu tasarım, aktif taklitten (replikasyon), Blok Zinciri güven modeline adapte edilmiş bir pasif taklit varsayımına doğru yönelmektedir.

3.4.4. Yapıya Ek Akıllı Sözleşme

3.4.4.1. Sabit Güven Modeli

İzin verilen blokların çoğu, fikir birliği oluşturmak için asenkron BFT taklit protokollerine güvenir. Bu tür protokoller tipik olarak $n > 3f$ eşlenikleri arasında, f 'ye kadar uyumsuz eylem gösteren güvenlik varsayımına dayanırlar. Aynı eşler, genellikle aynı güvenlik varsayımı altında da uygulamalarını yürütürler. Bununla birlikte, bu tür niceliksel bir güven varsayımı, sistemdeki meslektaşların rollerinden bağımsız olarak akıllı sözleşme uygulaması için gereken güvenceyle eşleşmeyebilir (Androulaki vd., 2018)

Esnek bir sistemde, uygulama düzeyinde güven, protokol seviyesinde güvenmeye sabitlenmemelidir. Genel amaçlı bir Blok Zinciri, bu iki varsayımı ayrıştırmalı ve uygulamalar için esnek güven modellerine izin vermelidir.

3.4.4.2. Kodlanmış Fikir Birliği

Takılabilir uzlaşma sistemini tanıtan ilk Blok Zinciri örneği Fabric'tir. Fabric'ten önce, neredeyse tüm blok zincir sistemleri, izin olsun veya olmasın, kodlanmış bir fikir birliği protokolü ile süregelmıştır. Örneğin, BFT protokolleri, konuşlandırıldıklarında performansları bakımından büyük ölçüde farklılık gösterirler. Zincir iletişim modeline sahip bir protokol, simetrik ve homojen bağlantılara sahip bir LAN kümesinde ispatlanmış bir şekilde optimum verim sergilemektedir ancak; geniş alanlı, heterojen bir ağda bozulmaya uğramaktadır. Ayrıca; yük, ağ parametreleri ve gerçek hatalar veya saldırılar gibi dış koşullar, belirli bir ağ içerisinde zaman içinde değişikliğe uğrayabilir. Bu nedenlerden dolayı, BFT uzlaşma mekanizması doğal olarak yeniden yapılandırılabilir olmalı ve ideal olarak değişen bir ortama dinamik olarak adapte edilmelidir. Bir diğer önemli husus, protokolün güven varsayımını, bir Blok Zinciri dağıtım senaryosuyla eşleştirmektir.

3.4.4.3. Modüler Fikir Birliğı:

Daha önce de belirtildiğı üzere, Fabric modüler bir Blok Zincir sistemidir. Özellikle tak-çalıştır özelliğini destekleyen bileşenlerden biri istem hizmetidir. Çoğunlukla sistemi test etmek için kullanılan herhangi bir dağıtık protokol yürütölmeyen merkezi ve çoğaltılmamış bir istem hizmetidir (Sousa vd., 2018).

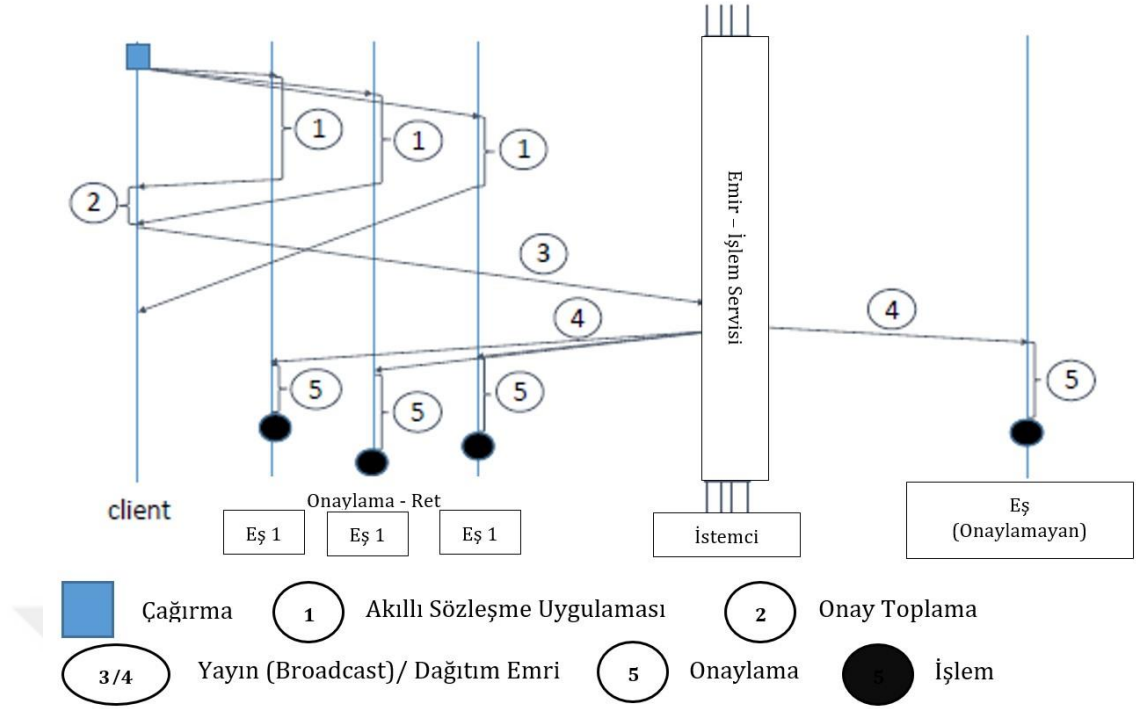
3.4.4.4. Blok Zinciri ile Hizmet Yönetimi

Fabric'in yürütme ve hizmet doğrulama mimarisini gerçekleştirmeden önce, servis-yürütme modelinde PBFT ile izin verilen bir Blok Zinciri platformu oluşturmak gerekmektedir.

Birçok kavram kanıtının uygulama esnasında elde edilen geri dönüşlerinden, bu yaklaşımın sınırları belirginleştirilmiştir. Kullanım esnasında, eşlerde farklı durumları gözlemlenebilmektedir ve fikir birliğı protokolünde bir hata ortaya çıkma ihtimali bulunmaktadır. Her durumda, sorumlunun belirleyici olmayan işlem kodu olduğı ortaya konulmaktadır. İşlemlerin ortalama gerçekleştirilmesi için süre bazında 200 ms sürmesine rağmen "Saniyede Yalnızca Beş İşlem" gibi sınırlı bir performans gösterme problemi ortaya çıkmaktadır. Bir Blok Zinciri sisteminin temel özellikleri tutarlılık, güvenlik ve performanstır. Özellikle Blok Zincirinin güvenilmeyen bir ortamda çalışması gerektiğinden, kullanıcılarının bilgisine ve iyi niyetine bağılı olmaması gerekmektedir.

3.4.5. Yüksek İşlem Akışı

Yüksek seviye veri akış yürütme aşamasında (Şekil 3.17), kullanıcılar, işlem teklifini (veya basit bir şekilde teklifi) yürütmek için bir veya daha fazla onaylayıcıya imzalar ve gönderirler. Bir teklif, gönderen kullanıcının kimliğini, yapılacak işlem şeklinde işlem yükünü, parametreleri ve zincir kodunun tanımlayıcısını, her kullanıcı tarafından yalnızca bir kez kullanılacak olan bir not ve kullanıcı tanımlayıcısından ve olmayan işlemden türetilmiş bir işlem tanımlayıcısı içerir.



Şekil 3.17. Yüksek Seviye Veri Akış Diyagramı (Androulaki vd., 2018).

Onaylayıcılar teklifi, Blok Zincirine kurulmuş olan, belirtilen zincir kodunda yürüterek işlemin benzetimi gerçekleştirilir. Zincir kodu ana onay işleminden izole edilmiş bir konteynerde çalışır.

Diğer eşlerle senkronizasyon yapılmaksızın, cihaza ait yerel Blok Zinciri durumuna karşı bir teklif benzetimi yapılır. Onaylayıcılar simülasyonun sonuçlarını defter durumuna alamazlar. Blok Zincirinin durumu, işlem yöneticisi tarafından versiyonlanmış bir anahtar-değer deposu biçiminde tutulur. Bir anahtara yapılan art arda güncellemeler monoton olarak artan sürüm numaralarına sahiptir. Bir zincir kod tarafından oluşturulan durum, yalnızca bu zincir koduna dahil edilir ve doğrudan başka bir zincir kodu tarafından erişilemezdir. Uygun görüldüğünde bir zincir kodu, aynı kanal içindeki durumuna erişmek için başka bir zincir kodu çağırabilir.

Benzetim sonucu olarak, her bir onaylayıcı, benzetim tarafından üretilen durum güncellemelerini kapsayan, yeni değerleriyle birlikte oluşan bir değer yazma sürecine girmektedir ve aynı zamanda, teklif benzetiminin versiyon bağımlılıklarını temsil eden bir okuma setine de sahip olur. Benzetimden sonra

onaylayıcılar, cihazda okunan bir mesajı imzalarlar. İşlem kimliği, onaylayıcı kimliği ve onaylayıcı imzası gibi veriler ile birlikte teklif, kullanıcıya geri gönderilir. Kullanıcı, işlemin çağırdığı zincir kodunun onay politikasını yerine getirinceye kadar onayları toplar. Özellikle, aynı yürütme sonucunu üretmek için politika tarafından belirlenen tüm onaylayıcılar gerekmektedir. Daha sonra, istemci işlemi yaratır ve hizmet servisine iletir.

3.4.5.1. Tasarım Tartışması

Onaylayıcılar teklifi diğer onaylayıcılarla senkronize etmeden benzetim yaptığı için, iki onaylayıcı, defterin farklı durumlarında da işlemi yürütebilir ve farklı çıktılar üretebilirler. Aynı sonucu elde etmek için birden fazla onaylayıcı gerektiren standart onay politikası için, bu aynı zamanda anahtara erişen işlemlerin yüksek çekişme altında olduğu anlamına da gelmektedir ve bu durumda müşteri onay politikasını yerine getiremeyebilir. Bu çoğaltılmış veri tabanlarında birincil yedekleme çoğaltması ile karşılaştırıldığında yeni katman senkronizasyonu ile yeni bir değerlendirmeye ihtiyaç duyulduğunu göstermektedir. Bu durum, bir Blok Zincirinde doğru uygulama için tek bir eşe güvenilmediği varsayımının sonucudur (Sousa vd., 2018).

Mimariyi oldukça basitleştirdiği ve tipik Blok Zinciri uygulamaları için yeterli olduğundan bu tasarım bilinçli bir şekilde benimsenmektedir. Dağıtılmış uygulamalar, aynı duruma erişen işlemlerle çekişmelerin azaltılabileceği biçimde formüle edilebilir veya normal durumda tamamen ortadan kaldırılabilir.

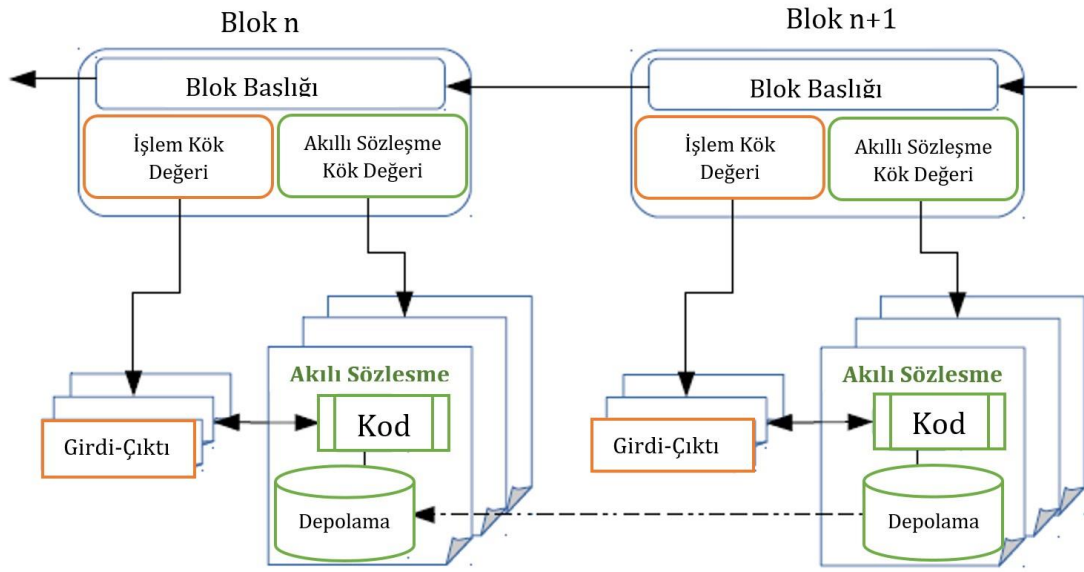
Sipariş aşamasından önce bir işlem yapmak deterministik olmayan zincir kodlarını tolere edebilmek için çok önemlidir. Deterministik olmayan işlemlere sahip Fabric üzerinde bir zincir kodu yalnızca kendi işlemlerinin canlılığını tehlikeye atabilir çünkü bir müşteri, örneğin yeterli sayıda onay toplayamayabilir. Bu deterministik olmayan işlemlerin meslektaşların durumunda tutarsızlıklara yol açması, sipariş yapılı mimariye göre temel bir avantajdır.

Son olarak, deterministik olmayan uygulamaların tolere edilmesi, güvenilmeyen zincir kodundan gelen DoS ataklarına da yöneliktir. Onaylayıcı, bir DoS saldırısından şüpheleniyorsa, yerel bir politikaya göre bir işlemi basitçe iptal edebilir. Bu, sistemin tutarlılığını tehlikeye sokmamaktadır ve yine, bu tür tek taraflı yürütmenin göz önüne alınması, mazeretli mimarilerde mümkün değildir.

3.4.6. Doğrulama Aşaması

Bloklar, eşlere doğrudan istem yoluyla iletilir. Yeni bir blok, üç ardışık adımdan oluşan doğrulama aşamasına girer (Şekil 3.18):

- (1) Onay politikası değerlendirmesi, blok içindeki tüm işlemlere paralel olarak gerçekleşir. Değerlendirme, doğrulama sistemi olan zincir kodunun görevidir. Blok Zinciri; konfigürasyonunun bir parçası olan ve zincir kodu için yapılandırılmış onay politikasına göre, onayın yapılmasından sorumlu olan statik bir kütüphanedir. Onayın yerine getirilmemesi durumunda, işlem geçersiz olarak işaretlenir ve etkileri dikkate alınmaz.
- (2) Bloktaki tüm işlemler için sırayla okuma-yazma çakışması kontrolü yapılır. Her işlem için, yeniden okuma alanındaki anahtarların sürümleri, defterin şu andaki durumuyla karşılaştırılır ve eşler tarafından yerel olarak depolandıkları ve hala aynı olmalarını sağlar. Sürümler eşleşmezse, işlem geçersiz olarak işaretlenir ve etkileri dikkate alınmaz.
- (3) Defter güncelleme aşaması, bloğun yerel olarak depolanmış deftere eklendiği ve blok zincir durumunun güncellediği son şekilde çalışır. Bilhassa, bloğu deftere eklerken, ilk iki adımda geçerlilik kontrollerinin sonuçları da devam eder. Ayrıca tüm durum güncellemeleri, yazma anahtarındaki tüm anahtar/değer çiftlerini yerel duruma yazarak uygulanır.



Şekil 3.18. Tamamen Doğrulanmış Bir Blok Zinciri Düğümü (Blackstock ve Lea, 2014).

4. ARAŞTIRMA BULGULARI VE TARTIŞMA

IoT teknolojilerinin dünyada yoğun bir şekilde ilgi görmesi, bu teknolojiye yeni bir iletişim altyapısının oluşturulmasını gerekli kılmıştır. Teknolojide bu gelişme ile farklı yönlendirme algoritmaları veya veri aktarım yöntemleri de geliştirilmeye başlanmıştır. Ancak IoT yapılarının neredeyse her alanda kullanılması dikkatleri üzerine çekmiş ve IoT güvenliği önemli bir konu haline gelmiştir. Bu tez çalışmasında incelenen çalışmalar, IoT teknolojilerinin araştırılması adına önemli bilgiler olarak görülmektedir.

Yapılmış olan bu çalışmada, IoT güvenliğinde kullanılması gereken güvenlik altyapıları ortaya çıkarılarak, özellikle akıllı ev ve akıllı şehir uygulamaları için bir öneri sistemi gerçekleştirilmesi düşünülmektedir. Bu sayede uzaktan kontrollü sistemlerde veya genel bir sistemde veri aktarımlarının güvenliği için literatüre katkı sağlayacak bir çalışma ortaya konmuştur. Bu kapsamda IoT üzerinde veri aktarımlarında Blok Zincirinin kullanılmasındaki avantajlar, dezavantajlar veya problemin çözümü için gerekli donanım veya yazılım gereksinimleri ortaya çıkarılması sağlanmış; aynı zamanda IoT teknolojilerinde olması gereken güvenlik standartlarının oluşturulması için örnek bir çalışma olmuştur. Dünya üzerinde milyonlarca kişinin kullandığı Blok Zinciri teknolojisi, bu çalışmanın yapılması vasıtasıyla güvenlik önlemleri gözden geçirilmiş olacak ve sistemler üzerinde uygulanabilirliği sınanmıştır.

Bu çalışmanın literatüre bir başka katkısı ise, simetrik veya asimetrik şifreleme tekniklerindeki şifreleme sürelerinin yavaş olmasından dolayı ortaya çıkan gecikmelerin en aza indirilmesidir. Çünkü “Hash fonksiyonları bahsi geçen şifreleme tekniklerinden daha hızlı şifreleme yapmaktadır” (Szabo, 1996).

Bu nedenle verinin güvenliliğinin alınması için harcanan zaman en aza indirilmesi denenmiştir. Harcanan zamanın azalması CPU kullanımlarını azaltacağı için; enerji sarfiyatından tasarruf sağlanması planlanmıştır. Bu çalışma, güvenlik ilkelerinin hepsini bir araya toplayacağı ve işlemci kullanımlarını en aza indirerek enerji tasarrufu sağlayacağı için IoT ile ilgili

birçok problemin ortadan kaldırılması ve literatüre kazandırılması planlamaktadır. Ülkemizde ise sadece finans sektörünü kapsamında görülen Blok Zinciri uygulamalarının yanında bilişim alanında da kullanılabilirliğini ispatlayan bir çalışma olması temel motivasyon kaynağıdır.

4.1. Nesnelerin İnterneti Veri İletişim Sistemi

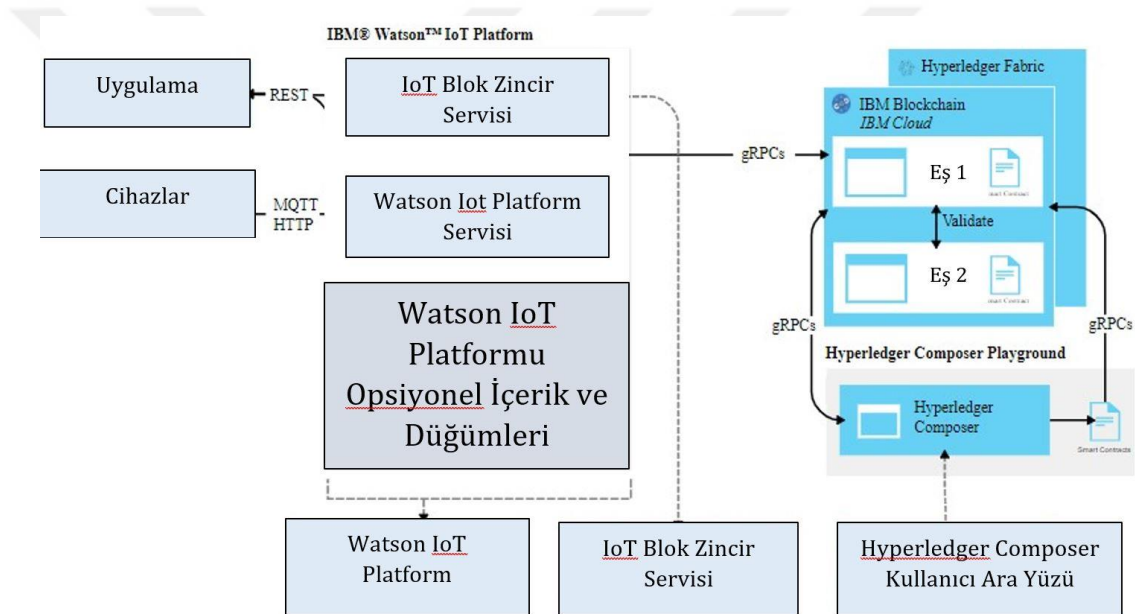
Günümüz koşullarında farklı altyapı ve algılayıcılar gerektiren Nesnelerin İnternetinin (IoT) her bir düğümü için kendini savunma becerisini sağlayacak “Blok Zinciri Tabanlı Nesnelerin İnterneti Veri İletişim Sistemi” adı verilen bir güvenlik sistemi tasarlanmıştır. Mevcut IoT yapılarının güvenlik mekanizmalarından yoksun olması sebebiyle, en küçük veri iletişimde bile herhangi bir siber saldırıya karşı güvenli bir sistemin ortaya çıkarılması gerekmektedir. Geliştirilen sistemde birden fazla düğüm ve bu düğümler arasında dağıtık halde bulunan verinin aktarımlarını, her bir düğüm içerisinde kayıt altına alarak değiştirilmez ve müdahale edilemez veri elde edilmesini sağlamak amacıyla Blok Zinciri kullanılmıştır. Blok zinciri yapısından var olan bloklar ve zincirin tüm düğümleri çakışmadan kayıt altına alınan verilerin değiştirilmesi ve elde edilmesi mümkün olmayacaktır.

Nesnelerin İnterneti Veri İletişim Sistemi projesinin temel aşamaları aşağıda gösterildiği gibidir:

- Gömülü bir sistem üzerinde kurulu düğümler ve çeşitli algılayıcılardan oluşan bir IoT yapısı oluşturulmuştur.
- Düğümlerden elde edilen verilerin Blok Zinciri yapısı ile güvenliği sağlanmış, aynı zamanda başka bir düğümün algılayıcılarında veri oluşmuş ise bu verilerinde mevcut Blok Zincirine eklenmesi ve bu sayede tek bir veri bloğu ile verinin aktarımı sağlanmıştır.
- Düğümlerde oluşan veriler, en kısa yoldan merkezi düğüme iletilmiştir.
- Geliştirilen sistemin işlemci kullanımları ve enerji sarfiyatlarının diğer simetrik ve asimetrik sistemler ile karşılaştırılmış ve analizleri yapılmıştır.

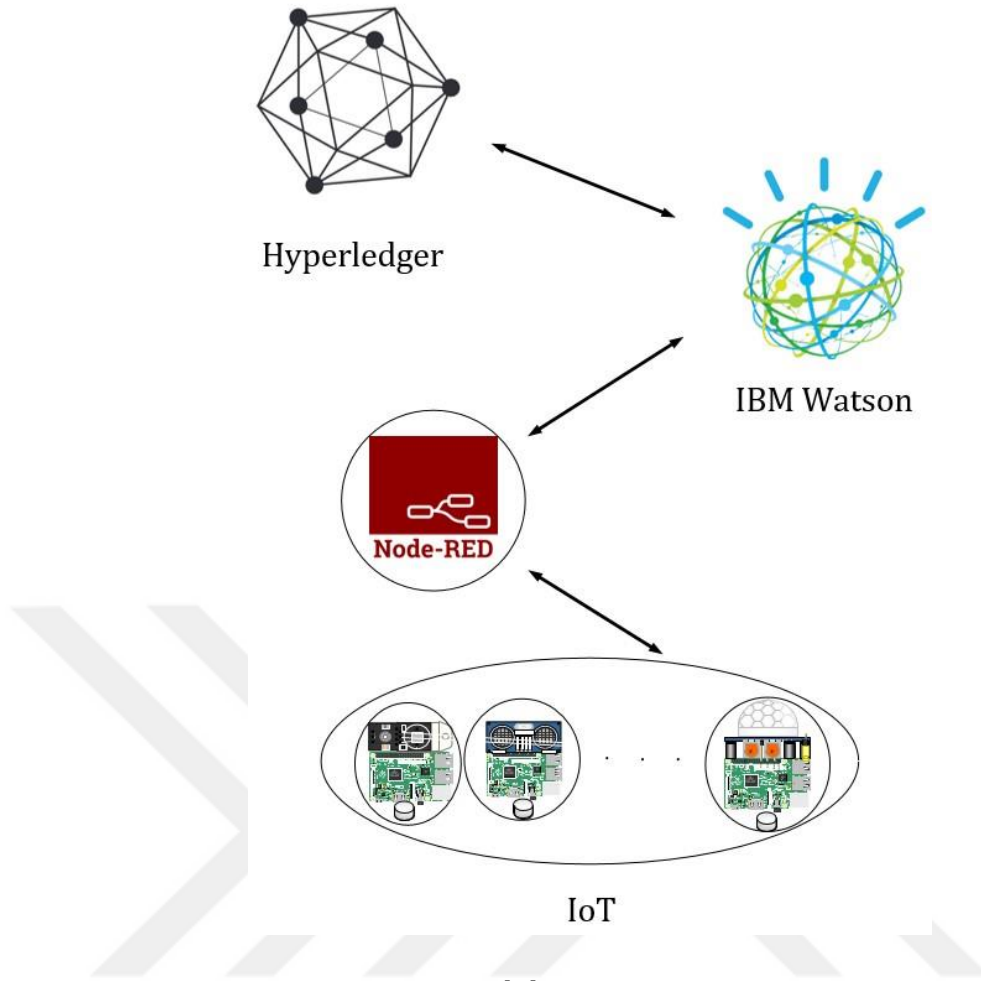
4.1.1. Node-Red Hyperledger İşlem Akış

IoT yapısını oluşturulurken Linux tabanlı çalışabilen Raspberry Pi 3B+ gömülü sistemi üzerinde çalışan ısı, nem, hareket, yakınlık, uzaklık algılama, yol-iz, gaz, yağmur ve ıslaklık ölçüm, eğim algılama ve titreşim algılayıcılarının kodları Python ve C++ ile yazılmıştır. Kiralanan bir sunucu üstüne Ubuntu – Hyperledger Fabric – Composer ve Node-Red kurulmuştur. IBM'in sunduğu Bluemix, Cloud Service, Watson gibi tüm servislerde hesap oluşturulmuş ve birbirine entegre edilmiştir. IBM SaaS (Software As A Service) hizmetleri çalışma prensibi (Şekil 4.1.) aşağıdaki resimde gösterilmiştir.



Şekil 4.1. Watson- Nesnelerin İnterneti Platform Gösterimi (IBM, 2019).

Kurulan tüm cihazlar (Raspberry Pi 3B+), Hyperledger Composer içinde bir ağ (Persihable-Network) oluşturularak eş olarak eklenmiştir. Cihazlara kurulan ve Node-Red üzerinden tetiklenen - elde edilen - gönderilen algılayıcı verileri Watson üzerinden Blok Zincirine dahil edilmiştir (Şekil 4.2). Hyperledger Composer dokümantasyonundan yararlanılan ve kurulan sisteme uygun şekilde düzenlenen kodlar, ara yüz (Playground) ile Watson'a entegre edilmiş ve veri okumaları Watson üzerinden sağlanmıştır.



Şekil 4.2. Yöntem

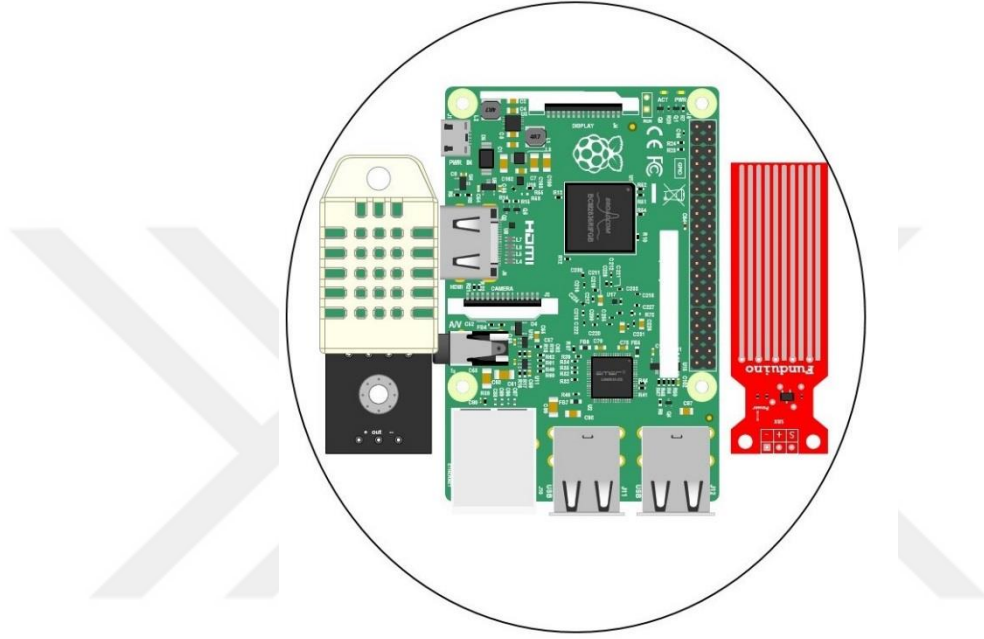
4.1.2. Düğümlerin Oluşturulması

Raspberry Pi gömülü sistem düğümlerin enerjileri 5V/2,5A DC özelliklere sahip adaptör ile sağlanmış olup, düğümler arasına enerji harcamalarını tespit etmek amacıyla piyasada bulunan Watt Metreler bağlanmıştır. Bu Watt Metreler iki farklı deney grubunda da kontrol edilmiştir.

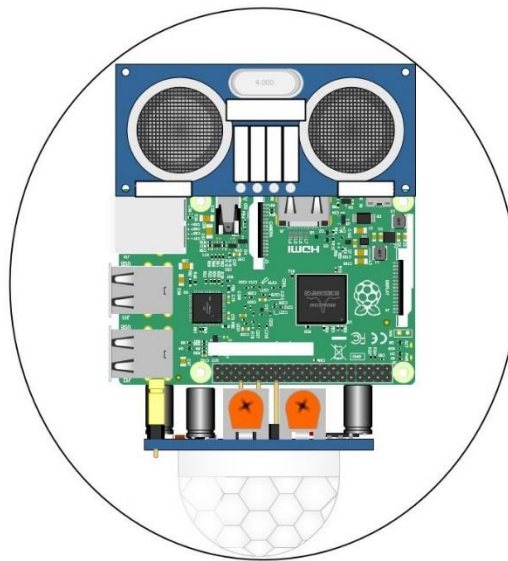
- Birinci deney grubunda Blok Zinciri kullanılmadan veri aktarımı sağlanmış ve Raspberry'lerin ne kadar enerji harcadığı, gönderilen verinin ne kadar sürede iletildiği tespit edilmiştir.
- İkinci deney grubu ise, Blok Zinciri kullanılarak aynı deneylere tabi tutulmuştur. Verinin doğru adrese gittiğinden emin olunması, verinin değişiklik veya manipülasyona uğrayıp uğramadığı ve verinin güvenli bir şekilde merkezi düğüme aktarılıp aktarılmadığı üzerine testler gerçekleştirilmiştir. Buradan elde edilen sonuçlara göre, kurulan projede

veri aktarım hızları, sıcaklık ve algılayıcı verileri analiz edilmiş, batarya verimlilikleri ve enerji tüketimleri karşılaştırılmıştır.

Düğüm (Şekil 4.3.), her cihaz başına en az bir, en fazla üç algılayıcıdan oluşmak üzere oluşturulmuş olup oluşturulan düğümlerden bazıları aşağıda tasnif edilmiştir.



Şekil 4.3. Meteoroloji Düğümü



Şekil 4.4. Hırsız Sezinleme Düğümü

Düğümleeri oluştururken algılayıcı kodları Python programlama dili ile yazılmış olup, yukarıdaki şekilde verilen ve oluşturulan düğüm içerisinde ki HC-SR01 Hareket Algılama Kodları (Şekil 4.5.) ve HC-SR04 Ultrasonik Mesafe Ölçüm (Şekil 4.6.) aşağıdaki gibidir.

```
GNU nano 2.7.4 File: hareket.py
# -*- coding: utf-8 -*-

#sag 1. 5v
#sag 3. pin ground (orta)
#sag 20. pin data

import RPi.GPIO as GPIO
import time
import datetime

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
pir=40
GPIO.setup(pir, GPIO.IN)

print "Sensör bekleniyor..."
time.sleep(1)

print "Sensör hazır. İşlem başladı."

while True:
    if GPIO.input(pir):
        print "-----"
        print "Hareket tespit edildi. " + str(datetime.datetime.now())
        time.sleep(2)

# -*- coding: utf-8 -*-
```

Şekil 4.5. Hareket Algılayıcı Kodları

```

GNU nano 2.7.4

# -*- coding: utf-8 -*-
#!/usr/bin/python
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

TRIG = 23
ECHO = 24

GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(ECHO,GPIO.IN)

while True:

    GPIO.output(TRIG, False)
    print "Mesafe Ölçülüyor..."
    time.sleep(2)

    GPIO.output(TRIG, True)
    time.sleep(0.00001)
    GPIO.output(TRIG, False)

    pulse_start = time.time()
    while GPIO.input(ECHO)==0:
        pulse_start = time.time()

    pulse_end = time.time()
    while GPIO.input(ECHO)==1:
        pulse_end = time.time()

    pulse_duration = pulse_end - pulse_start

    distance = pulse_duration * 17150
    distance = round(distance, 2)

    if distance > 2 and distance < 400:
        print "Mesafe:",distance - 0.5,"cm"
    else:
        print "Menzil aşıldı"

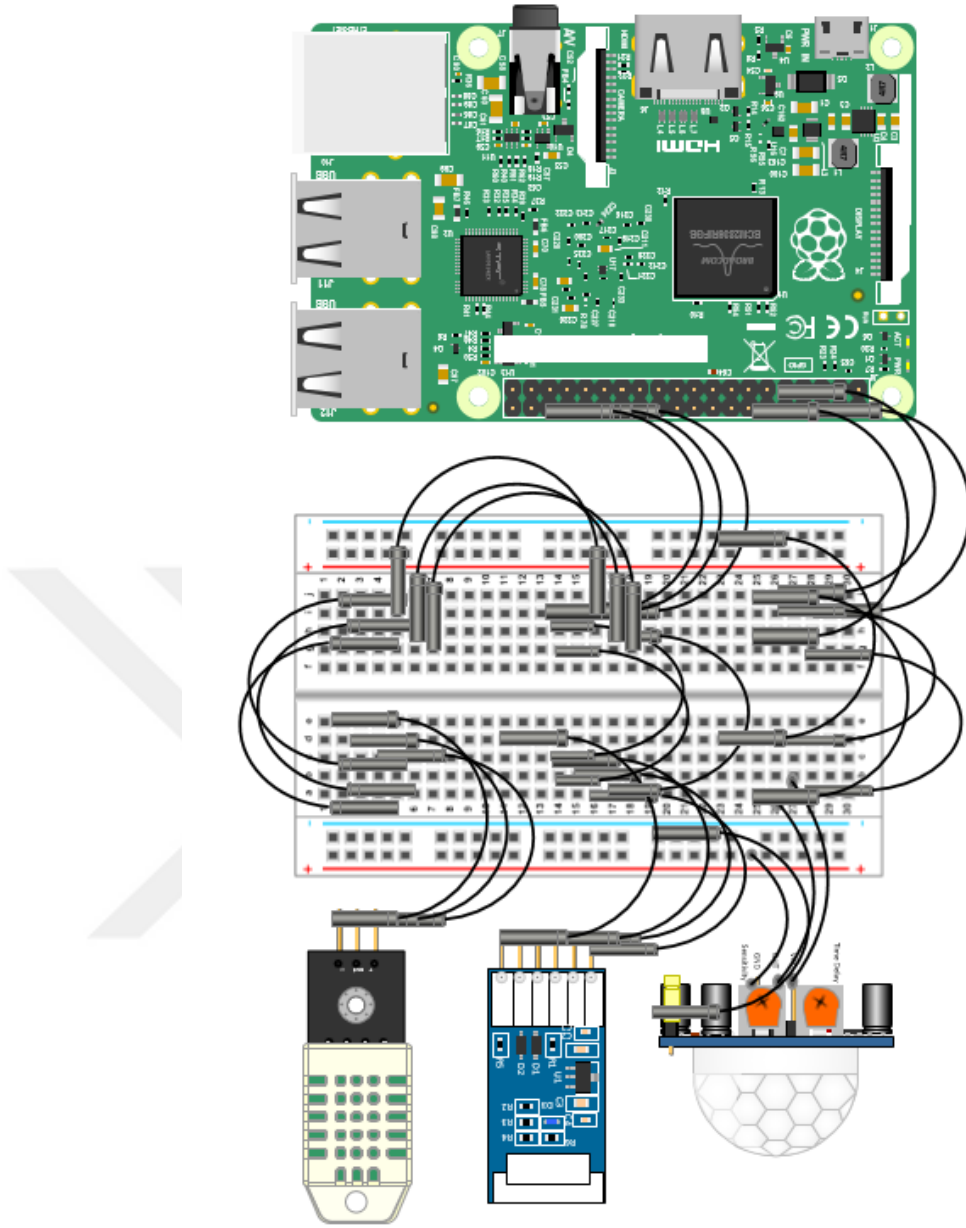
    GPIO.cleanup()
# -*- coding: utf-8 -*-

```

Şekil 4.6. Ultrasonik Mesafe Ölçer Kodları

4.1.3. IoT Yapısının Oluşturulması

Düğümlemler Node-red kullanılarak Raspberry Pi 3B+ gömülü sistemleri üzerinde oluşturulmuş ve statik IP'ler verilerek farklı VLAN'lar ile birbirinden ayrılmıştır. Aşağıda gösterilen çizim, oluşturulan düğümlerden birinin fiziki görüntüsünü (Şekil 4.7.) tasvir etmektedir.



Şekil 4.7. Örnek Düğüm

Tüm kurulum işlemlerinin ardından local olarak çalıştırılan Node-Red (Şekil 4.8) , *“sudo systemctl enable nodered.service”* komutuyla sistem düşmesi, yeniden açılma, ısınma nedeniyle kapanmalar gibi durumlarda tekrar müdahaleye gerek kalmaksızın, cihaz başlangıcında otomatik olarak aktif hale gelip kurulan sisteme binaen periyodik, zaman ayarlı, tespit halinde gönderim yada tetikleme yöntemiyle tekrar faal çalışabilir hale gelmektedir.

```
pi@raspberrypi:~/sensor_py $ node-red
14 May 11:50:08 - [info]
Welcome to Node-RED
=====
14 May 11:50:08 - [info] Node-RED version: v0.20.5
14 May 11:50:08 - [info] Node.js version: v10.15.3
14 May 11:50:08 - [info] Linux 4.14.98-v7+ arm LE
14 May 11:50:10 - [info] Loading palette nodes
14 May 11:50:16 - [info] Settings file : /home/pi/.node-red/settings.js
14 May 11:50:16 - [info] Context store : 'default' [module=memory]
14 May 11:50:17 - [info] User directory : /home/pi/.node-red
14 May 11:50:17 - [warn] Projects disabled : editorTheme.projects.enabled=false
14 May 11:50:17 - [info] Flows file : /home/pi/.node-red/flows_raspberrypi.json
14 May 11:50:17 - [warn]

-----
Your flow credentials file is encrypted using a system-generated key.

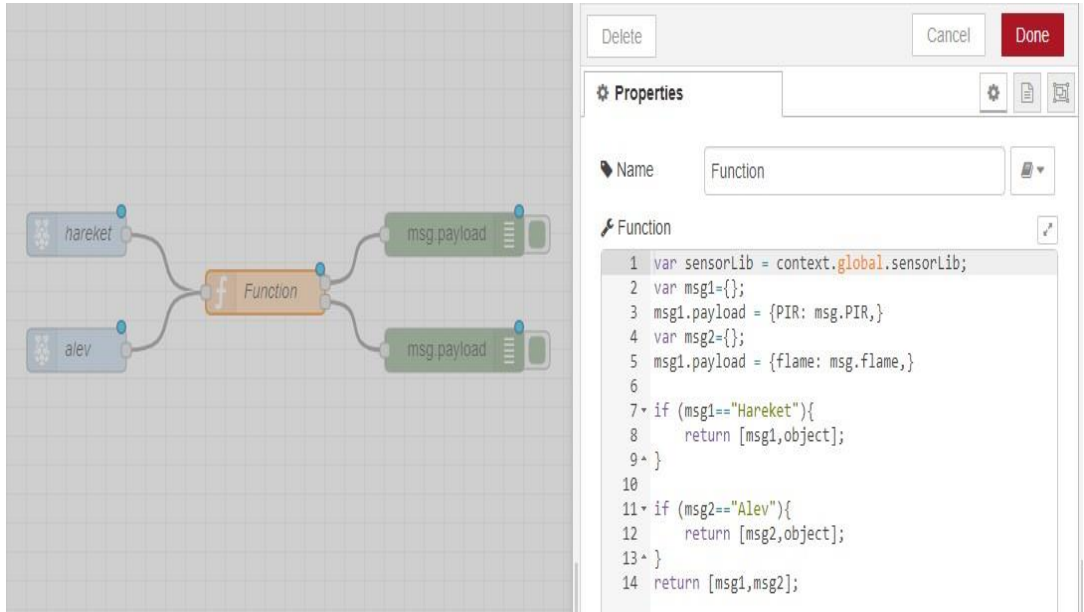
If the system-generated key is lost for any reason, your credentials
file will not be recoverable, you will have to delete it and re-enter
your credentials.

You should set your own key using the 'credentialSecret' option in
your settings file. Node-RED will then re-encrypt your credentials
file using your chosen key the next time you deploy a change.
-----

14 May 11:50:17 - [info] Starting flows
14 May 11:50:17 - [info] [wiotp out:447bb87f.c0c408] Connecting to Quickstart service as device node-red-wiotp/447bb87f.c0c408
[BaseClient:connect] Connecting to IoT with host : tcp://quickstart.messaging.internetofthings.ibmcloud.com:1883 and with client id
de-red-wiotp:447bb87f.c0c408
14 May 11:50:17 - [info] Started flows
14 May 11:50:17 - [info] Server now running at http://127.0.0.1:1880/
[DeviceClient:connect] DeviceClient Connected
```

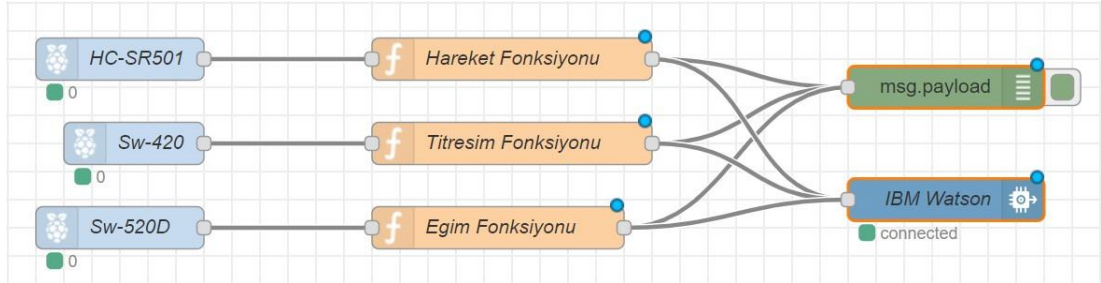
Şekil 4.8. Node-Red Başlangıç

İot düğümleri ve yapısını oluştururken, birbirine uyumlu ve yakın kategoride, denk sekmelere konabilecek gruplar oluşturulacak şekilde sınıflandırma yapılmıştır. Önceden “Uzaktan GPIO Kontrolüne” izin verilen cihazlarda, Node-Red içerisinde hazır bulunan ve “RPI GPIO” düğümü ile algılayıcılar tanıtılmıştır. “Function” düğümü ile parametreler ve fonksiyonlar girilerek hareket ya da alev tespiti halinde verinin gönderilmesi belirtilmiştir. Aşağıda tetikleme (Trigger) yöntemiyle çalışan bir düğüm (Şekil 4.9.) görülmektedir.



Şekil 4.9. Tetikleme

Aşağıda sismik çalışmaları hedefleyen; hareket, titreşim ya da deprem veya olası eğim kayması algılanması durumunda tetiklenen, algılayıcı verilerini IBM Watson IoT Kontrol Merkezine gönderen, Hyperledger'a (Blok Zinciri) entegre olan bir düğüm (Şekil 4.10.) tasvir edilmektedir.



Şekil 4.10. Deprem Düğümü

Aşağıda periyodik aralıklar ile saptanan sıcaklık, nem ve yağmur durumu verilerini periyodik aralıklar ile IBM Watson'a gönderen meteoroloji düğümü (Şekil 4.11.) gösterilmiştir.



Şekil 4.11. Meteoroloji Düğümü

4.1.4. Blok Zinciri Yapısının Oluşturulması

İzin verilen blok, eş (Peer), ve düğümleri (node) dağıtmak ve işletmek için esnek bir çatı sistemidir. Hyperledger Fabric ve Composer kurulumu tek tuşla ya da birkaç parça kod ile kurulmanın aksine, tam kurulum gerçekleşinceye kadar kullanıcıları yoğun ön koşullar ile kısıtlamaktadır.

Ön Koşullar;

- Ubuntu Linux 14.04 / 16.04 LTS (64-bit) İşletim Sistemi,
- Docker Engine: 17.03 veya üzeri versiyon,

- Docker-Compose:1.8 veya üzeri versiyon,
- Node: 8.9 veya üzeri versiyon (Versiyon 9 Desteklenmemektedir),
- npm: v5.x,
- git: 2.9.x veya üzeri versiyon,
- Python: 2.7.x versiyon.

4.1.5. Hyperledger Fabric Kurulumu

Hata düzeyini minimize etmek adına ön koşulları edinebileceğimiz “Docker Mirror”lar bulunmakla birlikte, kullanıcı geri bildirimleri doğrultusunda makineden makineye farklı hatalar alındığı bildirilmiştir.

- *sudo dpkg-reconfigure locales*
- *sudo apt-get update*
- *sudo apt-get upgrade*

Cihaz güncellemeleri ve paketlerin en güncel versiyona sahip olduğundan emin olunduktan sonra, aşağıda verilen komut ile ön koşul ve gereklilikleri indirecektir.

- *sudo apt-get install curl git docker.io docker-compose golang nodejs npm*

Geliştirme ortamı bileşenlerini yüklemek için komutlar aşağıdaki gibidir:

- *npm install -g composer-cli*
- *npm install -g composer-rest-server*
- *npm install -g generator-hyperledger-composer*
- *npm install -g yo*
- *npm install -g composer-playground*

Docker ve kullanıcıya gerekli izinler verilip, sistemi başlangıç olarak çalıştırmayı aktive hale getiririz:

- *sudo usermod -a -G docker \$USER*
- *sudo systemctl start docker*

- `sudo systemctl enable docker`

Ön Koşul ve gereklilik yüklemelerinin ardından Hyperledger kurulumu için aşağıdaki komutlar kullanılacaktır:

- `mkdir ~/fabric-dev-servers && cd ~/fabric-dev-servers`
- `curl -O https://raw.githubusercontent.com/hyperledger/composer-tools/master/packages/fabric-dev-servers/fabric-dev-servers.tar.gz`
- `tar -xvf fabric-dev-servers.tar.gz`
- `./downloadFabric.sh`

~/fabric-dev-servers klasörünün içerisinde `./startFabric.sh` komutuyla Hyperledger Fabric çalıştırılabilir.

`./byfn.sh generate` komutu ile Başlangıç Bloğu (Genesis Block) (Şekil 4.12.) oluşturulması gösterilmektedir:

```
File Edit View Search Terminal Help
cihan@cihan-virtual-machine: ~/fabric-samples/first-network

cihan@cihan-virtual-machine:~/fabric-samples$ cd first-network/
cihan@cihan-virtual-machine:~/fabric-samples/first-network$ ls
base                docker-compose-cli.yaml          docker-compose-org3.yaml
byfn.sh             docker-compose-couch-org3.yaml  byfn.sh
channel-artifacts   docker-compose-couch.yaml       org3-artifacts
configtx.yaml       docker-compose-eze-template.yaml README.md
crypto-config.yaml  docker-compose-kafka.yaml       scripts
cihan@cihan-virtual-machine:~/fabric-samples/first-network$ ./byfn.sh generate
Generating certs and genesis block for channel 'mychannel' with CLI timeout of '10' seconds and CLI delay of '3' seconds
Continue? [Y/n] y
proceeding ...
/home/cihan/fabric-samples/first-network/./bin/cryptogen
#####
#### Generate certificates using cryptogen tool ####
#####
+ cryptogen generate --config=./crypto-config.yaml
org1.example.com
org2.example.com
+ res=0
+ set +x

/home/cihan/fabric-samples/first-network/./bin/configtxgen
#####
##### Generating Orderer Genesis block #####
#####
CONSENSUS_TYPE=solo
+ '[' solo == solo ']'
+ configtxgen -profile TwoOrgsOrdererGenesis -channelID byfn-sys-channel -outputBlock ./channel-artifacts/genesis.block
2019-05-20 16:46:22.970 +03 [common.tools.configtxgen] main -> INFO 001 Loading configuration
2019-05-20 16:46:22.990 +03 [common.tools.configtxgen.localconfig] completeInitialization -> INFO 002 orderer type: solo
2019-05-20 16:46:22.998 +03 [common.tools.configtxgen.localconfig] Load -> INFO 003 Loaded configuration: /home/cihan/fabric-samples/first-network/configtx.yaml
2019-05-20 16:46:23.025 +03 [common.tools.configtxgen.localconfig] completeInitialization -> INFO 004 orderer type: solo
2019-05-20 16:46:23.025 +03 [common.tools.configtxgen.localconfig] LoadTopLevel -> INFO 005 Loaded configuration: /home/cihan/fabric-samples/first-network/configtx.yaml
2019-05-20 16:46:23.028 +03 [common.tools.configtxgen] doOutputBlock -> INFO 006 Generating genesis block
2019-05-20 16:46:23.028 +03 [common.tools.configtxgen] doOutputBlock -> INFO 007 Writing genesis block
+ res=0
+ set +x

#####
### Generating channel configuration transaction 'channel.tx' ###
#####
+ configtxgen -profile TwoOrgsChannel -outputCreateChannelTx ./channel-artifacts/channel.tx -channelID mychannel
ShowApplications 23.003 +03 [common.tools.configtxgen] main -> INFO 008 Loading configuration
23.003 +03 [common.tools.configtxgen.localconfig] Load -> INFO 009 Loaded configuration: /home/cihan/fabric-samples/first-network/configtx.yaml
```

Şekil 4.12. Başlangıç Bloğu

`./byfn.sh up` komutu ile eş (Peer) durumları (Şekil 4.13.) aşağıdaki resimde gösterilmiştir:

```

cihan@cihan-virtual-machine: ~/fabric-samples/first-netw
File Edit View Search Terminal Help

cihan@cihan-virtual-machine:~/fabric-samples/first-network$ ./byfn.sh up
Starting for channel 'mychannel' with CLI timeout of '10' seconds and CLI delay of '3' seconds
Continue? [Y/n] y
proceeding ...
LOCAL_VERSION=1.4.0
DOCKER_IMAGE_VERSION=1.4.0
Creating network "net_byfn" with the default driver
Creating volume "net_peer0.org2.example.com" with default driver
Creating volume "net_peer1.org2.example.com" with default driver
Creating volume "net_peer1.org1.example.com" with default driver
Creating volume "net_peer0.org1.example.com" with default driver
Creating volume "net_orderer.example.com" with default driver
Creating peer1.org2.example.com ...
Creating peer1.org1.example.com ...
Creating peer0.org1.example.com ...
Creating orderer.example.com ...
Creating peer0.org2.example.com ...
Creating peer1.org2.example.com
Creating orderer.example.com
Creating peer1.org1.example.com
Creating peer0.org1.example.com
Creating peer0.org2.example.com ... done
Creating cli ...
Creating cli ... done
Channel name : mychannel
Creating channel...
+ peer channel create -o orderer.example.com:7050 -c mychannel -f ./channel-artifacts/channel.tx --tls true --ca
rOrganizations/example.com/orderers/orderer.example.com/msp/tlsacerts/tlsca.example.com-cert.pem
+ res=0
+ set +x
2019-05-20 13:53:39.434 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2019-05-20 13:53:39.498 UTC [cli.common] readBlock -> INFO 002 Received block: 0
===== Channel 'mychannel' created =====

Having all peers join the channel...
+ peer channel join -b mychannel.block

```

Şekil 4.13. Eşler

docker ps -a komutu ile sistem doğrulaması, eş (peer) ve blok durumları kontrolü için gönderilen yayın (broadcast) sonucu (Şekil 4.14) aşağıda gösterilmektedir:

CONTAINER ID	IMAGE	PORTS	NAMES	COMMAND	CREATED	STATUS
2e2684c4d781	dev-peer1.org2.example.com-myc-1.0-26c2ef32830554aac4f7ad0f100aca865e87589c9a126e8d764c8d01f0346ab		dev-peer1.org2.example.com-myc-1.0	"chaincode -peer.add..."	3 minutes ago	Up 3 minute
cde89fffb66a	dev-peer0.org1.example.com-myc-1.0-384f11f484b9302df90b453200cfb25174305fce08f53f4e94d45ee3b6cab0ce9		dev-peer0.org1.example.com-myc-1.0	"chaincode -peer.add..."	3 minutes ago	Up 3 minute
0ea63c4614a6	dev-peer0.org2.example.com-myc-1.0-15b571b3ce049060b7ec74497da3b27e54e0df1345daff3951b94245ce09c42b		dev-peer0.org2.example.com-myc-1.0	"chaincode -peer.add..."	3 minutes ago	Up 3 minute
f2a818c4430e	hyperledger/fabric-tools:latest		dev-peer0.org2.example.com-myc-1.0	"/bin/bash"	4 minutes ago	Up 4 minute
0f6897c1467c	hyperledger/fabric-peer:latest		cli	"peer node start"	4 minutes ago	Up 4 minute
706f736aa669	0.0.0.0:8051->7051/tcp, 0.0.0.0:8053->7053/tcp		peer1.org1.example.com	"peer node start"	4 minutes ago	Up 4 minute
7406f775adff	hyperledger/fabric-peer:latest		peer0.org2.example.com	"peer node start"	4 minutes ago	Up 4 minute
a4327744a93f	0.0.0.0:8051->7051/tcp, 0.0.0.0:8053->7053/tcp		peer1.org2.example.com	"peer node start"	4 minutes ago	Up 4 minute
400af1ea1445	hyperledger/fabric-peer:latest		peer0.org1.example.com	"peer node start"	4 minutes ago	Up 4 minute
ae1b3494d303	0.0.0.0:7051->7051/tcp, 0.0.0.0:7053->7053/tcp		orderer.example.com	"orderer"	4 minutes ago	Up 4 minute
f6d2f55057ad	hyperledger/fabric-orderer:latest		dev-peer0.org1.example.com-fabcar-1.0	"chaincode -peer.add..."	6 weeks ago	Exited (2)
823e976999f2	dev-peer0.org1.example.com-fabcar-1.0-5c906e402ed29f20260ae42283216aa75549c571e2e380f3615826365d8269ba		dev-peer0.org1.example.com-fabcar-1.0	"tini -- /docker-ent..."	6 weeks ago	Exited (143)
1a804ab74f58	hyperledger/fabric-couchdb		couchdb	"sh -c 'fabric-ca-se..."	6 weeks ago	Exited (137)
ca.example.com	1a804ab74f58		ca.example.com			

Şekil 4.14. Sistem Testi

4.1.6. Hyperledger Composer Kurulumu

Hyperledger Composer arayüzü olan "*playground*"u yüklemek için aynı isimle bir kullanıcı oluşturulup, root yetkisi verilen kod aşağıdaki gibidir:

- *sudo adduser playground*

Gerekli root yetkisi için kullanılan kod aşağıdaki gibidir:

- `sudo usermod -aG sudo playground`

Oluşturulan yeni kullanıcı “*playground*”a geçiş yapıp, ön koşul yüklemeleri bu kullanıcı içine yapılır:

- `su - playground`

Composer ön koşul yüklemesini gerçekleştirecek olan kod aşağıdaki gibidir:

- `curl -O https://hyperledger.github.io/composer/latest/prereqs-ubuntu.sh`
- `chmod u+x prereqs-ubuntu.sh`
- `./prereqs-ubuntu.sh`

Ön koşul yüklemelerinin ardından playground hesabından çıkıp tekrar giriş yapılması gerekmektedir, aksi takdirde çalışma işleminin gerçekleşmediği vurgusuna dikkat ediniz.

- `exit`
- `su - playground`

Herhangi bir hata alınmadıysa, composer yüklenilen kod aşağıdaki gibidir:

1. `curl -sSL http://bit.ly/2ysbOFE | bash -s 1.4.0`

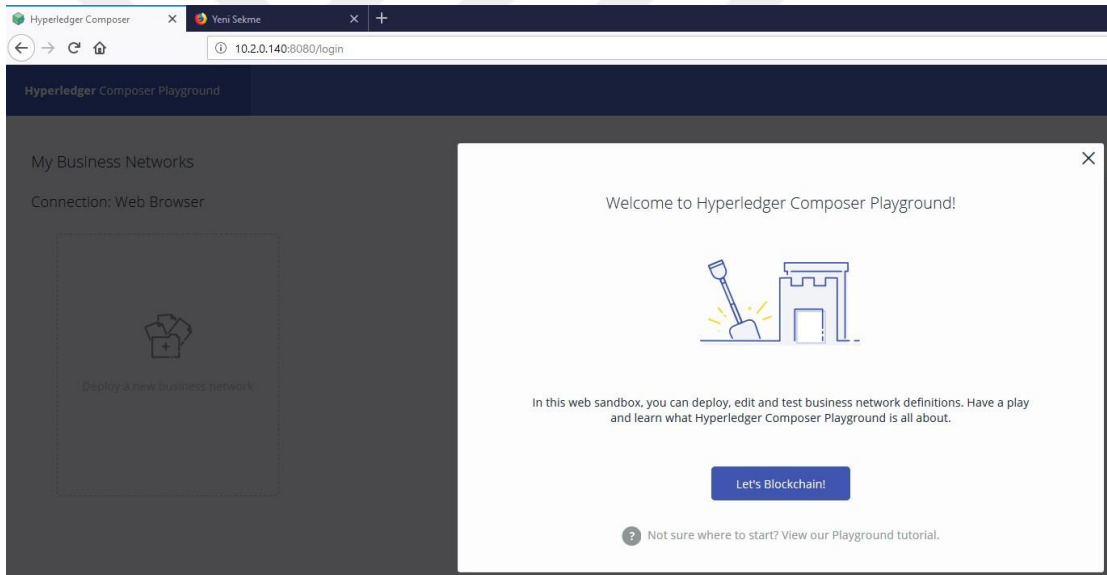
4.1.7. Blok Zincirini Nesnelerin İnternetine Dahil Etmek

Blok Zincirinin, Nesnelerin İnternetine dâhil edilmesi, güvenliği güçlendirmeyi amaçlayan çeşitli önlemlerle desteklenmektedir. Birkaç kuruluş, Blok Zincirini, üretim ve tedarik zincirlerine entegre etme konusunda öncülük etmektedirler. Örneğin; IBM, büyük zincir hizmetleri, tedarik zincirlerinde ve bilhassa yüksek değerli ürünlerin takip edilmesinde kullanmaktadır (Kshetri, 2017).

IBM, Nesnelerin İnterneti platformunun yerleşik özelliğini, kullanıcıların seçilen işlem verilerinin, paylaşılan işlemlere dâhil edilebilecek özel blok zincirleri

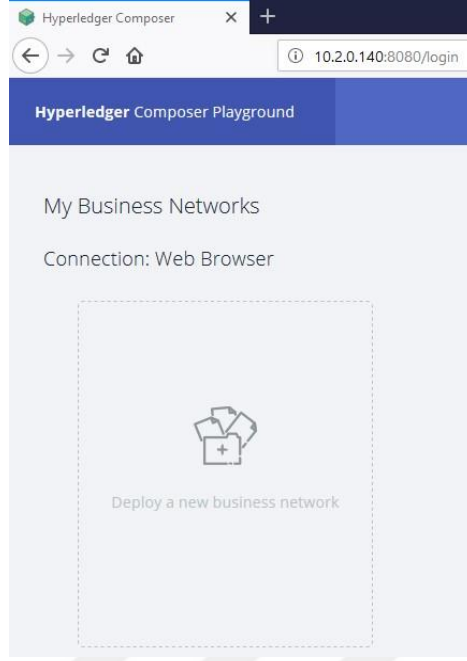
hesaplarına eklenmesine olanak tanımaktadır. Platform, bağılı cihazlardan gelen verileri Blok Zinciri sözleşmelerinin ihtiyaç duyduğu formata çevirmektedir. Blok Zinciri sözleşmesinin, cihaz verilerinin özelliklerini bilmesi gerekli değildir. Platform, aygıt işlemlerini filtreler ve yalnızca sözleşmeyi yerine getirmek için gereken verileri aktarır, veriler merkezi olarak depolanmaz, saklanmaz ve yönetilmez.

Cihaz üzerine Fabric'ten ayrı "playground" kullanıcısı oluşturularak kurulum bu kullanıcı içine gerçekleştirilmiştir. Composer kurulduktan sonra, "*composer-playground*" komutu ve "Local IP:8080" tarayıcı üzerinden composer ara yüzü (playground)'a ulaşabilmekteyiz. Açılış ekranında karşılayan resim (Şekil 4.15.) aşağıda gösterilmiştir.



Şekil 4.15. Playground

Model tanımı, kod ve akıllı sözleşme düzenlemeleri, ağ tanımı (Network Definiton) yapmak için "Deploy a New Business Network" seçerek sistem tanımlaması yapılması (Şekil 4.16.) aşağıda gösterilmiştir.

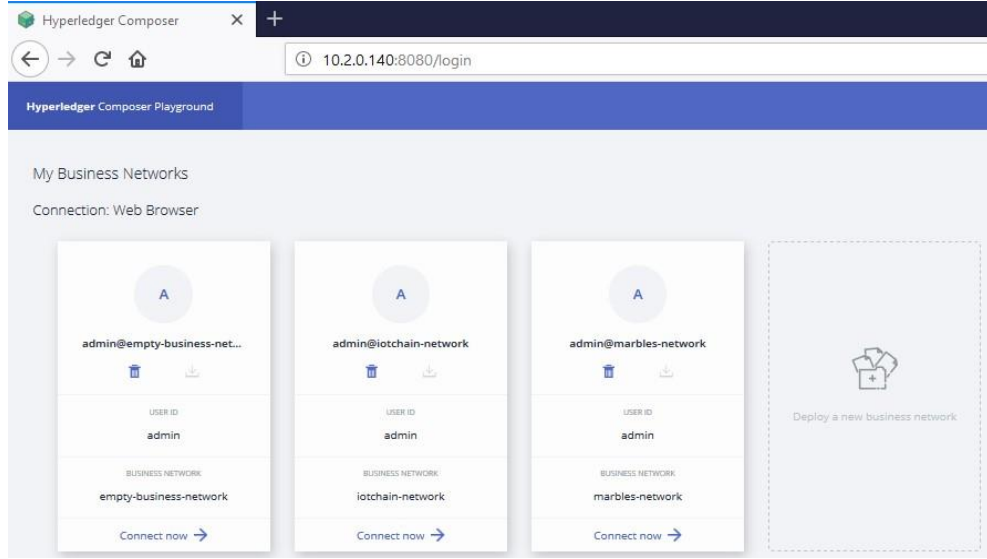


Şekil 4.16. Model Tanımı

Sistem isimlendirmesi, admin hesabı ve kullanım amaçları gibi ileride görmek istediğimiz tanımlar burada girilir ve hazır olan bir networke dâhil olmak ya da kendi sistemimizi tanımlayabildiğimiz durum (Şekil 4.17.) aşağıdaki gibidir.

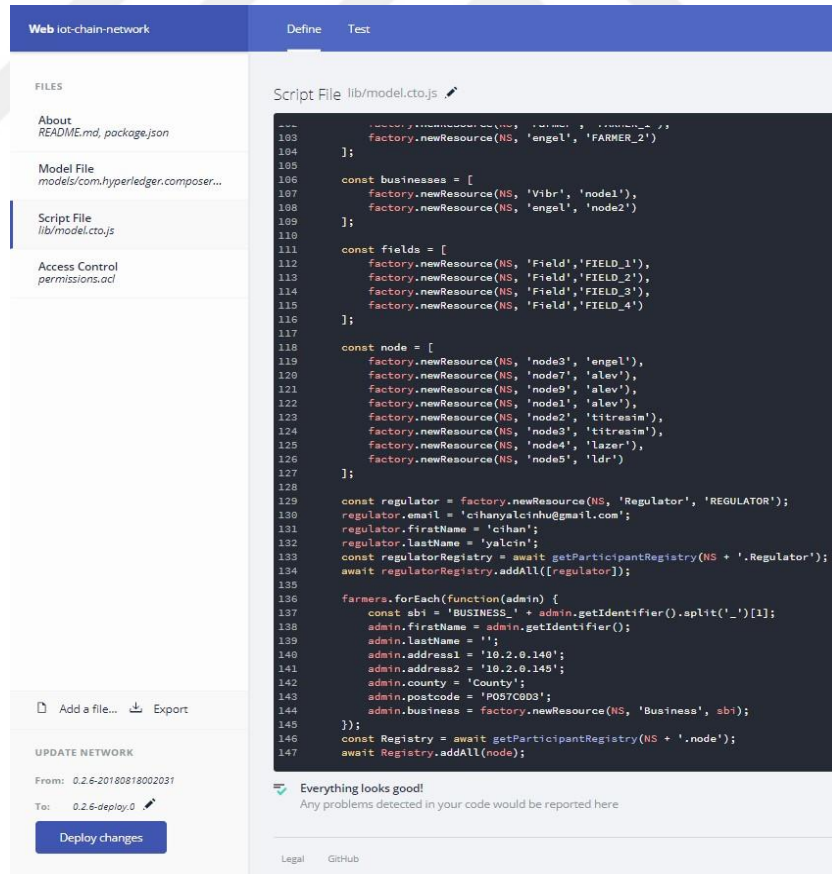
Şekil 4.17. Perishable Network

Sistem tanımlaması, kod güncellemeleri, servis ayarları, API key tanımlamaları yapıldıktan sonra oluşturulan ağ (Tablo 4.18.) aşağıdaki gibi gösterilmektedir.



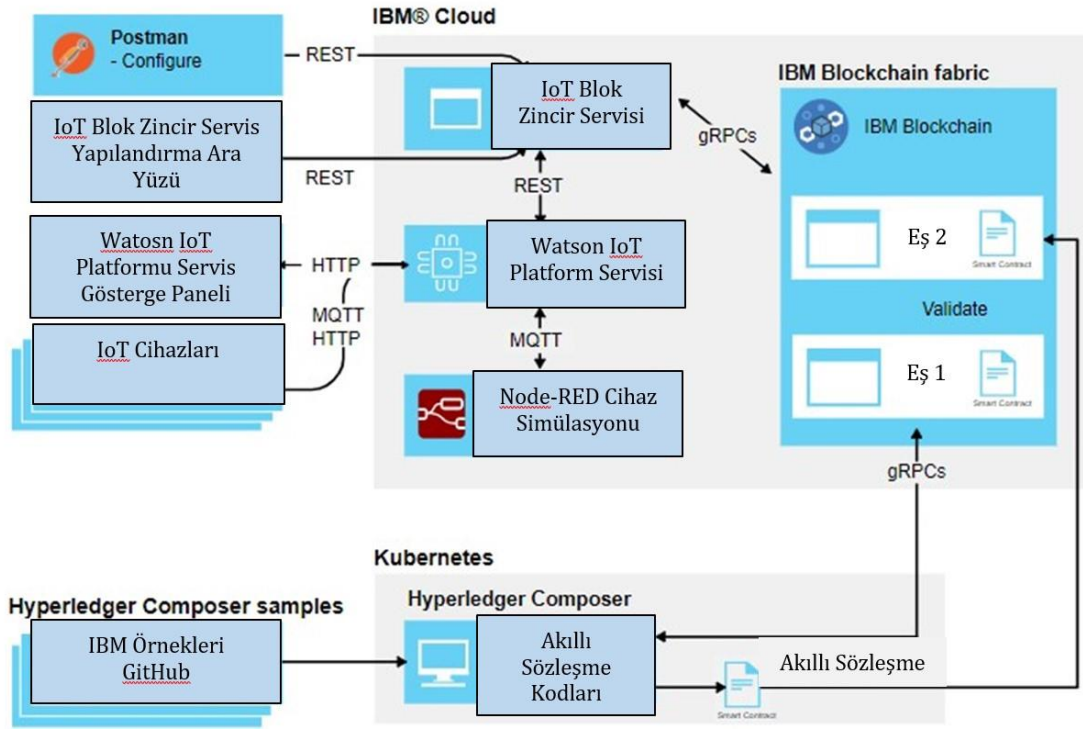
Şekil 4.18. IoT-Chain Network

Ağ modeli, özelleştirme ve akıllı sözleşme kodlarının (Tablo 4.19) aşağıdaki gibidir.



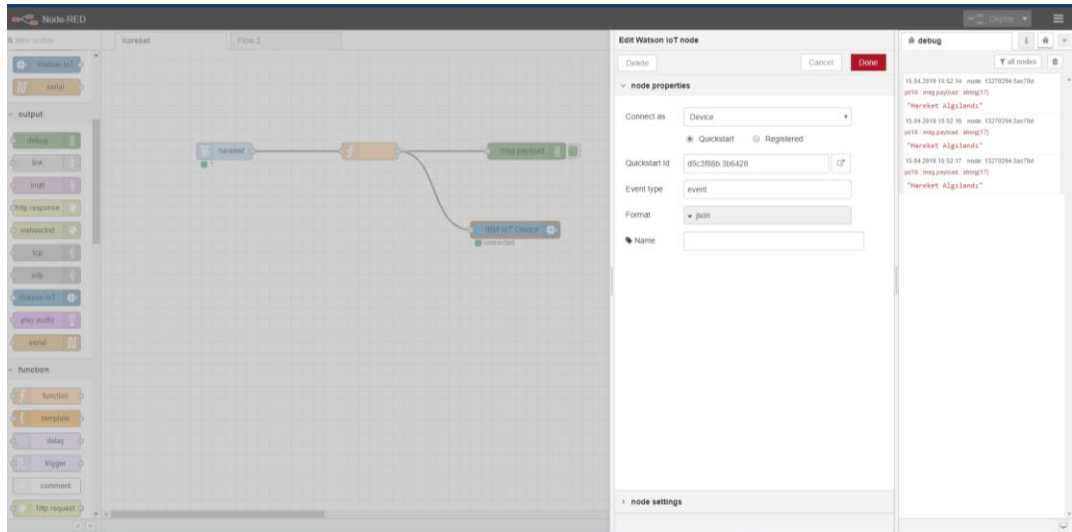
Şekil 4.19. ChainCode

IBM sistemleri çalışma prensibi (Şekil 4.20.) aşağıdaki gösterilmiştir.

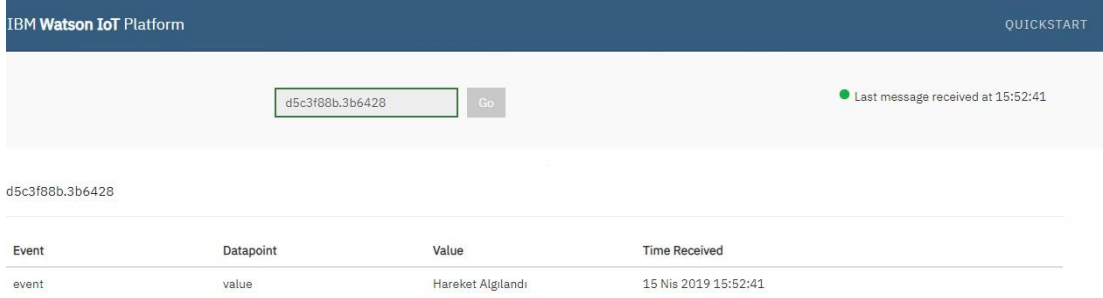


Şekil 4.20. IoT Blok Zinciri Hizmet Servisi (IBM, 2019).

Yapılan tüm kurulum, konfigürasyon, sistem özelleştirme ve tanıtların sağlıklı çalışmasının ardından, Node-Red üzerinden, Blok Zincirine (Hyperledger) dahil ettiğimiz verilerin Watson ve Node-Red üzerinde ki çıktıları (Şekil 4.21. ve 4.22.) aşağıdaki gibi gösterilmiştir.



Şekil 4.21. Node - Red Hareket Algılama Çıktısı

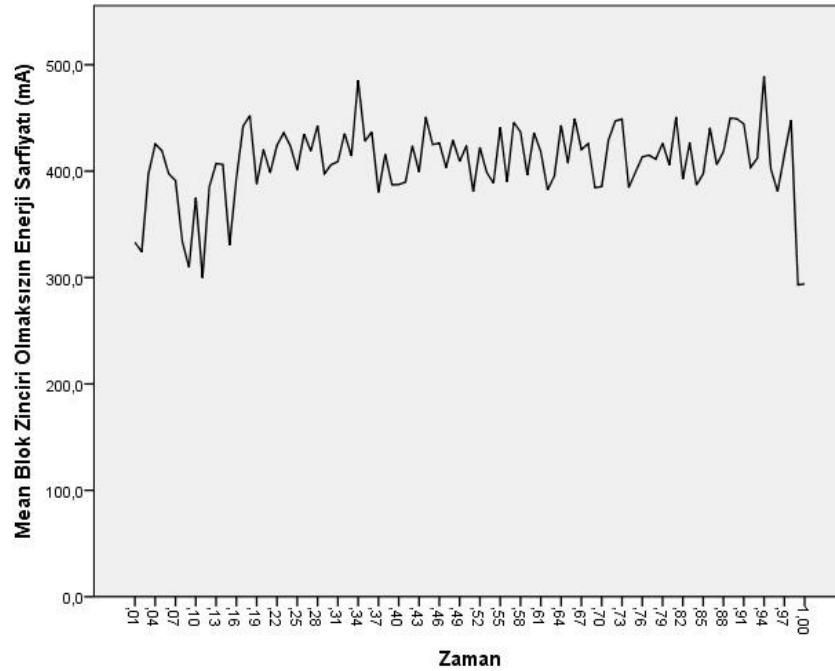


Şekil 4.22. Watson Hareket Algılama Çıktısı

4.2. Nesnelerin İnterneti Blok Zinciri Performans Analizleri

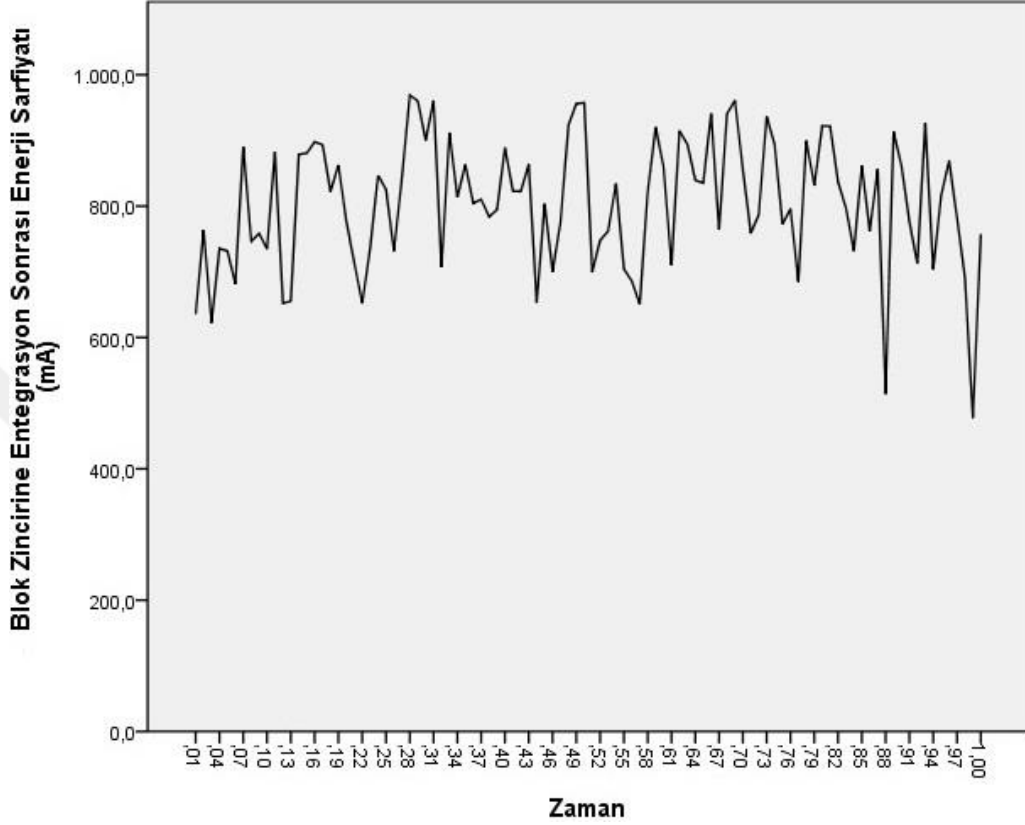
Blok Zinciri Mimarisinin, Nesnelerin İnterneti Sistemine entegresinde; Linux açık kaynak Blok Zinciri uygulaması, Hyperledger Fabric ve Composer kullanılmıştır. Bütünleşik hale getirilen Nesnelerin İnterneti ve Blok Zinciri sistemlerinin çalışma esasları üzerinden, enerji tüketimleri ve veri gönderim hızları üzerine karşılaştırmalı performans analizleri yapılmıştır.

Blok Zinciri Olmaksızın Enerji Sarfıyatı Analizi, cihazlarda tüm uzaktan erişim özellikleri açık olmak üzere, led, GPIO, iki adet algılayıcı bağlı ve ağ bağlantısı aktif iken 3 düğümün ortalaması (Şekil 4.23) hesaplanarak aşağıda gösterilmiştir.



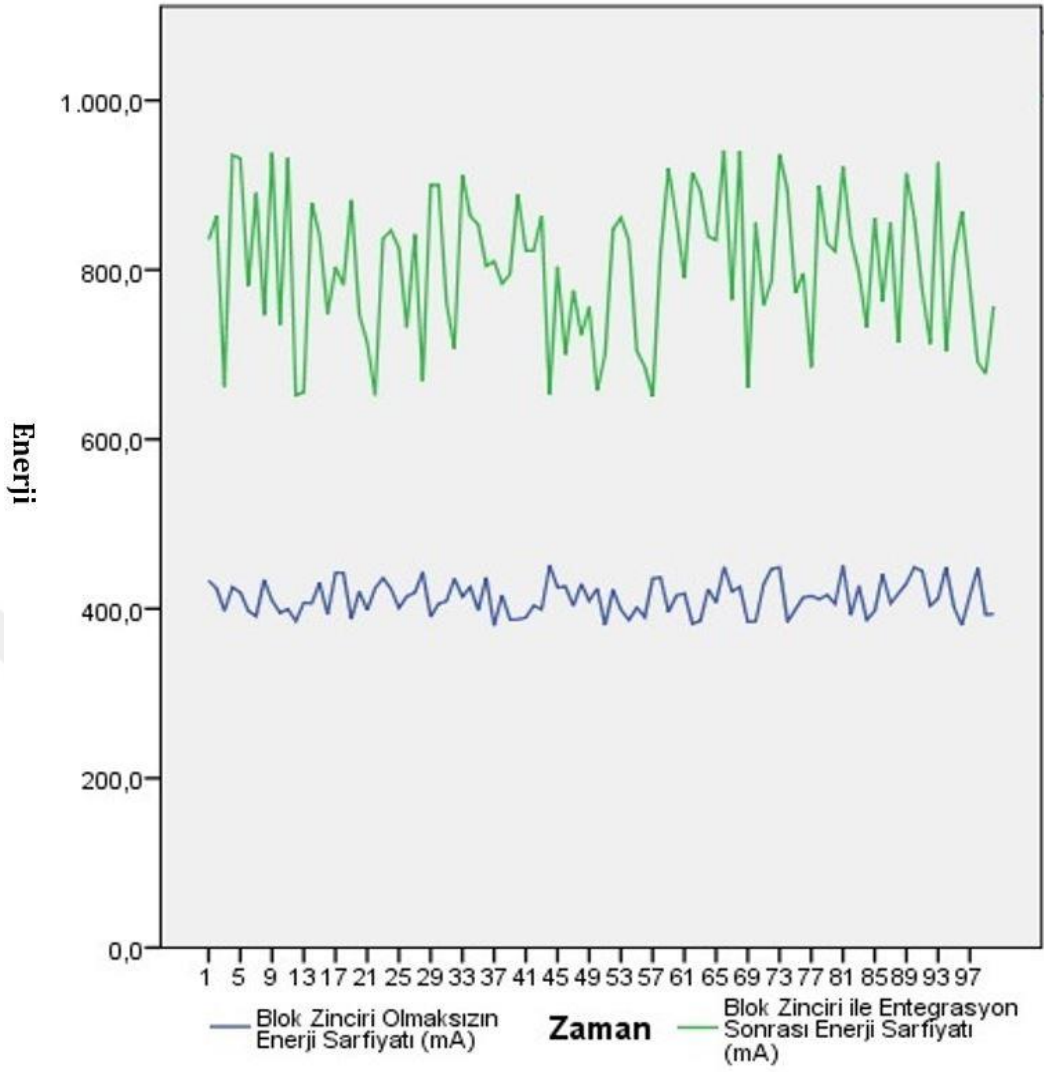
Şekil 4.23. Blok Zinciri Olmaksızın Enerji Sarfıyatı (Ma)

Blok Zincirine Entegre Enerji Sarfıyatı Analizi, tüm uzaktan erişim özellikleri aktif, her 5 saniyede algılayıcı verilerini gönderen ve cihaz muhafaza kutusundaki fanlar aktif halde iken 3 farklı düğüm ve cihazın ortalaması (Şekil 4.24) olarak hesaplanmıştır.



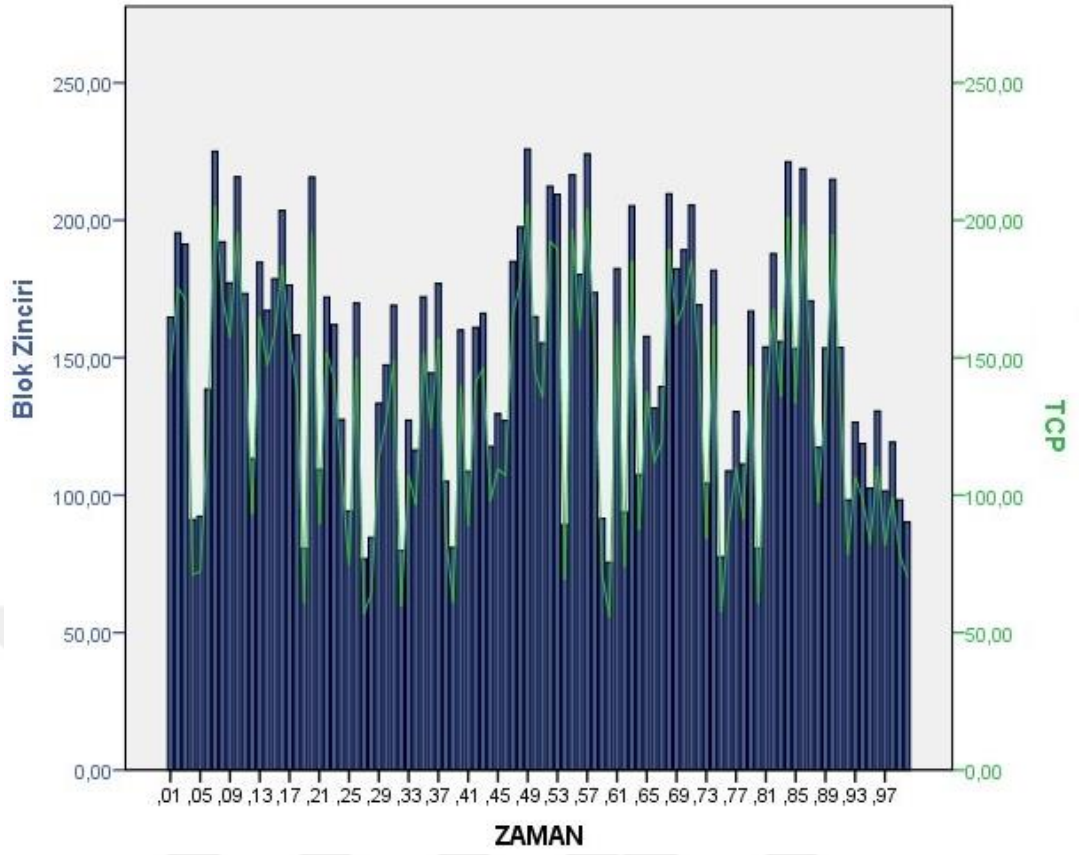
Şekil 4.24. Blok Zincirine Entegrasyon Sonrası Enerji Sarfıyatları (Ma)

Aşağıda (Şekil 4.25), mavi renk ile gösterilen Blok Zinciri Olmaksızın Enerji Sarfıyatı ve yeşil renk ile gösterilen Blok Zincirine Entegre Enerji Sarfıyatları karşılaştırmalı analiz grafiğini gösterilmektedir. Blok Zincirine entegrasyon sağlandığında, ısınma kaynaklı donma ve kapanmaların önüne geçmek adına, Raspberry Pi 3 B+ muhafaza kutusu üzerine gömülü entegre fanlar çalıştırılmıştır. Cihaz adaptörü ve GPIO pinlerinden beslenen fanların, enerji sarfıyatlarında yukarı yönlü hareketleri izlenebilmektedir. Cihaz üzerinde 2'den fazla algılayıcının Blok Zincirine veri sağlaması halinde, cihazlarda enerji kaynaklı donma ve yeniden başlatmalar yaşandığı gözlemlenmiştir.



Şekil 4.25. Blok Zinciri Olmaksızın/Blok Zincirine Entegre Enerji Sarfıyatları

Aşağıda (Şekil 4.26), TCP/Ip protokolü ile gönderilen ve Blok Zinciri uygulamasından sonra kullanıma hazır veri analizi gösterilmektedir. Mavi barlar ile gösterilen kısım, cihazlardaki algılayıcı verilerinin Node-Red üzerinden Blok Zincirine gönderilen; yeşil renk çizgi grafiği ise, algılayıcı verilerinin doğrudan Blok Zinciri entegrasi yapılmadan TCP yolu ile gönderimini göstermektedir.



Şekil 4.26. Blok Zinciri/TCP/IP Veri Transfer Hızları

Enerji sarfıyatı analizde, Blok Zinciri kullanıldığında gözle görülebilen artış, veri transferi analizinde minimal farklara inmiştir. Kullanılan akıllı sözleşme türlerine göre farklılık gösteren transfer hızları, PBFT (Practical Byzantine Fault Tolerance) kullanılmasıyla son kullanıcı açısından yavaşlık sebebiyle tercih edilmemesi sorununa çözüm getirmektedir.

5. SONUÇ VE ÖNERİLER

Sistemimizin donanımsal altyapısı için fiziksel düğümler oluşturularak, Node-Red üzerinde gruplandırılmıştır. Raspberry Pi 3 B+ kartları üzerine ısı, nem, hareket, yakınlık, uzaklık algılama, yol-iz, gaz, yağmur ve ıslaklık ölçüm, titreşim, eğim algılayıcıları eklenmiştir. Bu kapsamda kullanılan algılayıcıların piyasada bulunan genel amaçlı kullanılan algılayıcılar olduğu tekrar belirtilmelidir.

Gelişme döneminde bulunan iki büyük teknolojik yeniliğin, birleşimlerinden doğabilecek olası potansiyelini gördüğümüz, olgunlaşmamış güvenlik standartları, hızla artan cihaz sayısı ve bu cihazların kendini savunma yeteneğinin olmaması; Nesnelerin İnterneti dünyasında güvenlik standardı oluşturmayı giderek zorlaştırmaktadır. Diğer yandan, son yıllarda popülerleşen Blok Zinciri teknolojisi, gözlerin bu alanda yer alacak yeniliklere çevrilmesine sebep olmuştur. Bu iki büyük teknolojinin birleşimi, gerek akademik çalışmalarda, gerek devlet ve özel sektörde yer alan büyük bir eksiklik olan güvenlik ve verimlilik sorularının ortadan kalkacağı düşüncesini doğurmuştur.

Kullanıcılar tarafından özellikle finansal sektördeki patlamasıyla tanınan Blok Zinciri, aslen geleceğin güvenlik sektörünü de içerisinde barındırmaktadır. Bir hız çağında yaşadığımız düşünüldüğünde, Blok Zincirinin taşıdığı potansiyel güvenlik, gizlilik ve anonimlik ilkeleri; bu teknolojiyi kullanan bireyler adına hukuk, finans, sağlık gibi başlıca konularda, insanların birden fazla eyleme dönüşen durumlarını minimuma indirecektir. Blok Zincirinin taşıdığı potansiyel, kullanıcıları birden fazla iş yükünden kurtarabilecek niteliğe sahiptir.

Nesnelerin İnternetinde birbirine bağlı cihazların sayısı arttıkça, kullanıcılar üzerinde de beklentiler giderek arttırmaktadır. Fakat bu teknolojide, bu kadar fazla opsiyonu; hem verimli, hem hızlı ve hem de modüler bir şekilde ne yazık ki kullanamamaktayız. Sahip olduğumuz teknoloji ve aracı programlar, sadece büyük şirketler tarafından iş piyasalarına yönelik olduğu sürece, kullanıcıların ücretsiz ve kolay erişebileceği bir yapıya bu zaman zarfı içerisinde ulaşamayacağı sonucunu karşımıza çıkarmaktadır. Sonuç olarak Blok Zinciri

mimarisinin Nesnelerin İnterneti sisteminde kullanılması, ön koşulları yerine getirmenin zorluğu ve tek bir platform (Linux) üzerinde sunulması, son kullanıcı açısından sorun teşkil etmektedir. Ayrıca IoT sistemin kurulduğu cihazlar Raspberry Pi'ye ikiden fazla algılayıcıyı aktif çalıştırması Blok Zinciri kullanımında, işlemci tarafında stabilite ve sıcaklık sorunlarını beraberinde getirmektedir. Öte yandan Blok Zincirine olan ön yargının, kullanılan akıllı sözleşme protokolleri ile kullanım amacı doğrultusunda modüler hale getirilip veri transfer hızında bir aksama yaşatmadığı ve artan enerji sarfiyatı ise olası bir felaket senaryosunda çıkacak olan maliyetin çok altında olduğu ispatlanmıştır.

Blok Zinciri Tabanlı Nesnelerin İnternet Sistemi, Nesnelerin İnterneti için güvenlik çözümleri sunuyor olsa dahi, bloklar ve Blok Zinciri tek başına güvenlikle ilgili bir sorunu, otomatik olarak ve tek başına çözebilecek yetiye sahip değildir. Blok Zinciri, güvenlik ve mahremiyet açısından avantajlar sunmaktadır. Avantaj olarak ise, Blok Zinciri tabanlı siber güvenlik sistemlerinin ele geçirilmesinin diğer çözümlere göre çok daha güç olduğunu söylemek mümkündür.

Kripto tabanlı doğrulama hizmeti ve anahtar şifreleme protokolü, diğer algılayıcı ağlara göre daha az sarfiyatla, daha iyi enerji tüketimi sonuçları sunmaktadır. Bu protokol, ağ üzerinde bulunan düğümlerin miktarına bağlı olarak, ağın kendi boyutuna göre yayılım gösterebilmektedir.

Blok Zinciri tabanlı veri aktarım gücü arttıkça; güvensiz veri aktarımı da buna karşılık azalmaktadır. Bununla birlikte olası bir güvenlik sorununa karşılık, Blok Zinciri ile güvenli ve anonim veri aktarımı tolere edilebilir bir maliyete yol açmaktadır.

KAYNAKLAR

- "Chain protocol whitepaper," 2014. [Çevrimiçi]. Erişim Tarihi: 22.05.2019. Available: <https://chain.com/docs/1.2/protocol/papers/whitepaper>
- Abramov, Raz, ve Amir Herzberg. "TCP Ack storm DoS attacks." *Computers & Security* (2012), Volume 33, March 2013, Pages 12–27.
- Akyildiz, I.F., W. Su, Y. Sankarasubramaniam, E.Cayirci, 2002. "Wireless Sensor Networks: A Survey," *Computer Networks*, Cilt 38, Sayı 4, 393–422.
- Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., ... & Muralidharan, S. (2018, April). Hyperledger fabric: a distributed operating system for permissioned blockchains. In *Proceedings of the Thirteenth EuroSys Conference* (p. 30). ACM.
- Azaria, A., Ekblaw, A., Vieira, T., Lippman, A., 2016. MedRec: Using Blockchain for Medical Data Access and Permission Management. 2016 2nd International Conference on Open and Big Data, 25-30.
- Blackstock, M., & Lea, R. (2014, October). Toward a distributed data flow platform for the web of things (distributed node-red). In *Proceedings of the 5th International Workshop on Web of Things*(pp. 34-39). ACM.
- Blackstock, M., & Lea, R. (2016). Fred: a hosted data flow platform for the IOT built using node-red. *Proceedings of MoTA*.
- Cachin, C. (2016, July). Architecture of the hyperledger blockchain fabric. In *Workshop on distributed cryptocurrencies and consensus ledgers* (Vol. 310).
- Christoph, P. M. (2009). Security and privacy challenges in the internet of things. *Proc WowKiVS*, 2009-1.
- Ercan, T., Nacak, D., 2009. Kablosuz Ağlardaki Paket Trafiğine Adli Bilişim Yaklaşımı, *Journal of Yaşar University*, 4(13), 1909-1921.
- Eyal, I., Sirer, E.G., 2018. Majority is not Enough: Bitcoin Mining is Vulnerable. *Communication of the ACM*, July 2018, Vol. 61, No.7, 95-102. DOI: 10.1145/3212998.
- Fangfang, D., Yue, S., Nan, M., Liang, W., Zhiguo, Y., 2017. From Bitcoin to Cybersecurity: a Comparative Study of Blockchain Application and Security Issues. *The 2017 4th International Conference of Systems and Informatics (ICSAI 2017)*, 975-979.
- Flauzac, O., Gonzalez, C., Nolot, F., 2015. New Security Architecture for IoT Network. *International Workshop on Big Data Mining Challenges on IoT and Pervasive Systems (BigD2M 2015)*, 1028-1033.

- Gan, S., 2017. An IoT Simulator in NS3 and a Key Based Authentication Architecture for IoT Devices Using Blockchain. Department of Computer Science and Engineering Indian Institute of Technology KANPUR, 8-42.
- Garderen, P.V., 2016. Introduction to Blockchain and Recordkeeping. Recordkeeping Roundtable Synded, Australia. [Çevrimiçi]. Erişim Tarihi: 22.05.2019.
http://www.interpares.org/display_file.cfm?doc=ip1_dissemination_ss_van-garderen_rr_2016.pdf
- Giang, N.K. et al. 2015. Developing IoT applications in the Fog: A Distributed Dataflow approach. *Internet of Things (IOT), 2015 5th International Conference on the* (Oct. 2015), 155–162.
https://www.ibm.com/support/knowledgecenter/en/SSCG66/iot-blockchain/getting_started/getting_started.html
- IBM CO, 2019. The IBM Watson IoT Platform On Blockchain Add-On. Erişim Tarihi: 22.05.2019.
<https://www.ibm.com/support/knowledgecenter/SSQP8H/iot/blockchain/index.html>
- IBM CO, 2019. Tutorials to get Started With IoT Blockchain Service. Erişim Tarihi: 22.05.2019.
- Johnston, W.M. et al. 2004. Advances in Dataflow Programming Languages. *ACM Comput. Surv.* 36, 1 (Mar. 2004), 1–34.
- Karaarslan, E., & Akbaş, M. F. (2017). BLOKZİNCİRİ TABANLI SİBER GÜVENLİK SİSTEMLERİ. *ULUSLARARASI BİLGİ GÜVENLİĞİ MÜHENDİSLİĞİ DERGİSİ*, 3(2), 16-21.
- Karel, 2017. Erişim Tarihi: 22.05.2019. <https://www.karel.com.tr/blog/blok-zinciri-ile-nesnelerin-interneti-icin-guvenli-bir-model>
- Khan, M.A., Salah, K., 2017. IoT Security: Review, Blockchain Solution, and Open Challenges. *Future Generation Computer System*, <https://doi.org/10.1016/j.future.2017.11.022>.
- Kshetri, N. (2017). Can blockchain strengthen the internet of things?. *IT professional*, 19(4), 68-72.
- Kumar, S., Vineet S. Raghavan, Jing Deng, 2006. “Medium Access Control protocols for ad hoc wireless networks: a survey” *Ad Hoc Networks* (ELSEVIER), Cilt 4, Sayı 3, 326–358.
- Kuzmin, A., 2017. Blockchain-Based Structures For a Secure and Operate IoT, 978-1-5386-3197-3/17.

- Medium Corporation, 2018. Blockchain ve Dağıtık Defter Teknolojileri. Erişim Tarihi: 22.05.2019. <https://medium.com/patikan/blockchain-ve-da%C4%9F%C4%B1t%C4%B1k-defter-teknolojileri-4e76083d2c2f>
- Medium Corporation, 2018. Blok Zincirde Farklı Bir Perspektif; Hyperledger. Erişim Trihi: 22.05.2019. <https://medium.com/unichain-tr/hyperledger-fabric-ve-hyperledger-composer-4fe95ce91d18>
- Meghdadi, M., Özdemir, S., Güler, İ.,2008. “Kablosuz Algılayıcı Ağlarında Güvenlik: Sorunlar Ve Çözümler”, Bilişim Teknolojileri, 1.
- Miller, A., Y. Xia, K. Croman, E. Shi, and D. Song, “The honey badger of BFT protocols,” in Proceedings of the 2016 ACM Conference on Computer and Communications Security, Vienna, Austria, 2016.
- Nakamoto, S., Bitcoin: A peer-to-peer electronic cash system, 2008.
- Prnewswire, 2014. 33 Billion Internet Devices By 2020: Four Connected Devices for Every Person in the World, Says Strategy Analytics. Erişim Tarihi: 18.06.2019. <https://www.prnewswire.com/news-releases/33-billion-internet-devices-by-2020-four-connected-devices-for-every-person-in-the-world-says-strategy-analytics-279166141.html>
- Popov, S., 2016. The Tangle Whitepaper-IOTA.
- Sousa, J. and A. Bessani, “Separating the WHEAT from the chaff: An empirical design for geo-replicated state machines,” in Proceedings of the IEEE 34th Symposium on Reliable Distributed Systems, Montreal, Quebec, Canada, 2015.
- Sousa, J., Bessani, A., & Vukolic, M. (2018, June). A byzantine fault-tolerant ordering service for the hyperledger fabric blockchain platform. In *2018 48th annual IEEE/IFIP international conference on dependable systems and networks (DSN)* (pp. 51-58). IEEE.
- Szabo, N., “Smart contracts: Building blocks for digital markets,” *EXTROPY: The Journal of Transhumanist Thought*, no. 16, 1996.
- Taş, O., Kiani, F., 2018. Blok Zinciri Teknolojisine Yapılan Saldırıları Üzerine Bir İnceleme. Bilişim Teknolojileri Dergisi, Cilt:11, Sayı:4, Ekim 2018.
- Usta, A., Doğanekin, S., 2017. BLOCKCHAIN 101. Kapital Medya Hizmetleri A.Ş. MediaCat Kitapları. ISBN: 978-605-4584-97-0
- Vukolić, M., “The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication,” in Open Problems in Network Security - IFIP WG 11.4 International Workshop, Zurich, Switzerland, 2015.

- Xu, J.F., 2011. "A Defense System for Wireless Sensor Networks", *The Journal of China Universities of Posts and Telecommunications*, 18(2), 119-122.
- Xu, X., Pautasso, C., Zhu, L., Gramoli, V., Ponomarev, A., Chen, S., 2016. The Blockchain as a Software Connector. 2016 13th Working IEEE/IFIP Conference on Software Architecture, 182-191.
- Yilmaz, Y., Uludağ, S., 2017. Mitigating IoT-Based Cyberattacks on the Smart Grid. 2017 16th IEEE International Conference on Machine Learning and Application, 517-522.
- Zhang, J. et al. 2013. Supporting Personizable Virtual Internet of Things. *Ubiquitous Intelligence and Computing, 2013 IEEE 10th International Conference on Autonomic and Trusted Computing (UIC/ATC)* (Dec. 2013), 329-336.
- Zhou, Y., Fang, Y., 2008. "Securing Wireless Sensor Networks: A Survey", *IEEE Communications Surveys & Tutorials*, 10(3), 6-28.

ÖZGEÇMİŞ

Adı Soyadı : Cihan YALÇIN
Doğum Yeri ve Yılı : Isparta, 1990
Medeni Hali : Bekâr
Yabancı Dili : İngilizce, Macarca, Litvanca
E-posta : cihanyalcin@sdu.edu.tr
cihanyalcinhu@gmail.com



Eğitim Durumu

Lise : Isparta Anadolu Lisesi, 2009
Lisans : Kaunas Teknoloji Üniversitesi, Mühendislik Fakültesi,
Bilişim Mühendisliği

Mesleki Deneyim

SDÜ Bilgi İşlem Daire Başkanlığı 2016 - (halen)