T.C.

TRAKYA ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

NESNELERİN İNTERNETİ UYGULAMALARINDA AES BLOK ŞİFRESİNİN YAZILIMSAL VE DONANIMSAL PERFORMANSLARININ KARŞILAŞTIRILMASI

IŞIL ÇETİNTAV

YÜKSEK LİSANS TEZİ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

Tez Danışmanı: DR. ÖĞR. ÜYESİ Deniz TAŞKIN

EDİRNE-2018

Işıl ÇETİNTAV'ın hazırladığı "Nesnelerin İnterneti Uygulamalarında AES Blok Şifresinin Yazılımsal ve Donanımsal Performanslarının Karşılaştırılması" başlıklı bu tez, tarafımızca okunmuş, kapsam ve niteliği açısından Bilgisayar Mühendisliği Anabilim Dalında bir Yüksek lisans tezi olarak kabul edilmiştir.

Jüri Üyeleri (Ünvan, Ad, Soyad): İmza

Dr. Öğr. Üyesi Deniz TAŞKIN .....

Dr. Öğr. Üyesi Tarık YERLİKAYA .....

Dr. Öğr. Üyesi Murat Olcay ÖZCAN .....

Tez Savunma Tarihi: 03/08/2018

Bu tezin Yüksek Lisans tezi olarak gerekli şartları sağladığını onaylarım.

İmza

Dr. Öğr. Üyesi Deniz TAŞKIN
Tez Danışmanı .....

Trakya Üniversitesi Fen Bilimleri Enstitüsü onayı

.....
Prof. Dr. Murat YURTCAN
Fen Bilimleri Enstitüsü Müdürü

T.Ü.FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI YÜKSEK LİSANS PROGRAMI
DOĞRULUK BEYANI

Trakya Üniversitesi Fen Bilimleri Enstitüsü, tez yazım kurallarına uygun olarak hazırladığım bu tez çalışmasında, tüm verilerin bilimsel ve akademik kurallar çerçevesinde elde edildiğini, kullanılan verilerde tahrifat yapılmadığını, tezin akademik ve etik kurallara uygun olarak yazıldığını, kullanılan tüm literatür bilgilerinin bilimsel normlara uygun bir şekilde kaynak gösterilerek ilgili tezde yer aldığını ve bu tezin tamamı ya da herhangi bir bölümünün daha önceden Trakya Üniversitesi ya da farklı bir üniversitede tez çalışması olarak sunulmadığını beyan ederim.

03/08/2018

Işıl ÇETİNTAV

I. Cetintav

Yüksek Lisans Tezi

Nesnelerin İnterneti Uygulamalarında AES Şifreleme Metodunun Yazılımsal ve Donanımsal Performanslarının Karşılaştırılması

T.Ü. Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

ÖZET

Nesnelerin interneti uygulamalarının artışı ile ortaya çıkan güvenlik problemleri oldukça fazladır. Düşük enerji tüketimli ve güvenli sistemler üretmek, günümüzde nesnelerin interneti uygulamalarının en öncelikli amacı olmuştur.

Bu tez çalışmasında, Bluetooth düşük enerji tabanlı nesnelerin interneti sistemleri üzerinde güvenlik analizi yapılmıştır. Bu analiz, AES şifreleme metodunun nesnelerin interneti sistemleri üzerinde yazılımsal ve donanımsal uygulamalarının karşılaştırılması üzerinedir. Çalışmada CC254x Bluetooth düşük enerji yongası kullanılmış olup bu yonganın donanımsal olarak 128-bit AES şifreleme desteği bulunmaktadır. Ayrıca yazılım uygulamasında da aynı yonga üzerinde AES şifreleme metodu bu kez yazılımsal olarak gerçekleştirilmiştir. Her iki uygulama da nesneler ile ağ geçitleri arasındaki kablosuz haberleşme güvenliği sağlanmaktadır.

AES şifreleme yazılım ve donanım uygulaması karşılaştırıldığında donanım uygulamasının çok daha hızlı çalıştığı ve daha az güç tükettiği gözlenmiştir. Bu çalışmanın kablosuz haberleşme güvenliği ve analizi konusunda yararlı olacağı düşünülmektedir.

Yıl : 2018

Sayfa Sayısı : 69

Anahtar Kelimeler : aes şifreleme, nesnelerin interneti, bluetooth düşük enerji, güvenlik

Master Thesis

Comparing the Hardware and Software Performance of AES Encryption Method on Internet of Things Applications

Trakya University Institute of Natural Sciences

Computer Engineering Department

ABSTRACT

Security problems due to increasing of Internet of Things application are quite a lot. Creating secure and less power consuming systems has become the primary goal for Internet of Things applications.

In this thesis, a security application on Bluetooth Low Energy based Internet of Things system was analysed. This analysis aimed at comparing hardware and software implementations of AES encryption method. CC254x BLE module that supports hardware implementation of AES encryption was used. Also, application has software implementation of AES encryption on same development kit. The proposed system provides security between objects and gateways with AES ciphering.

When comparing hardware and software applications, it is seen that hardware implementation is faster and consumes less power than software implementation. It is believed that this study would be useful for wireless communication security.

Year : 2018

Number of Pages : 69

Keywords : aes encryption, bluetooth low energy, internet of things, security

TEŞEKKÜR

Tez çalışmam boyunca emeği geçen, bilgi ve tecrübeleriyle bana yol gösteren ve destek veren saygıdeğer danışman hocam Dr. Öğr. Üyesi Deniz TAŞKIN'a tüm katkılarından dolayı teşekkür ederim.

Yüksek Lisans Tez Savunma Sınavı Jüri üyeliği yapan, yapıcı yorumlarıyla tez çalışmama katkıda bulunan çok değerli hocalarım Dr. Öğr. Üyesi Tarık YERLİKAYA ve Dr. Öğr. Üyesi M. Olcay ÖZCAN'a teşekkür ederim. Ayrıca değerli hocam Doç. Dr. M. Tolga Sakallı'ya çalışmama katkılarından dolayı teşekkür ederim.

Yüksek Lisans öğrenimim boyunca her zaman destek olan ve anlayışlı davranan çalışma arkadaşlarıma ve tüm öğrenim hayatım boyunca ve her zaman bana destek olan aileme teşekkür ederim.

İÇİNDEKİLER

ŞEKİLLER LİSTESİ ..... **vi**
ÇİZELGE LİSTESİ ..... **viii**
SİMGELER VE KISALTMALAR ..... **ix**
SİMGELER ix
KISALTMALAR ix
1. GİRİŞ **1**
2. NESNELERİN İNTERNETİ **3**
2.1. Nesnelerin İnterneti Mimarisi 5
2.1.1. Nesne 6
2.1.2. Ağ Geçidi 9
2.1.3. Bulut/Büyük Veri 14
2.1.4. Uygulamalar 15
2.2. Nesnelerin İnternetinde Güvenlik 16
2.2.1. Güvenlik Açıkları 17
2.2.2. Güvenlik Gereklilikleri 21
3. BLUETOOTH DÜŞÜK ENERJİ **23**
3.1. Bluetooth Düşük Enerji Teknolojisi ve Gelişimi 23
3.2. BLE Cihaz Tipleri 25
3.3. BLE Yazılım Akışı 26
3.3.1. İşletim Sistemi Soyutlama Katmanı 27
3.3.2. Donanım Soyutlama Katmanı 27
3.3.3. BLE Yığıtı 28
4. MATERYAL VE METOD **33**
4.1. İleri Şifreleme Standardı 33
4.1.1. SubByte Dönüşümü 36
4.1.2. ShiftRows Dönüşümü 38

4.1.3. MixColumn Dönüşümü 39
4.1.4. Anahtar Ekleme 40
4.1.5. Anahtar Planlama 40
4.1.6. Anahtar Genişletme 40
4.2. Uygulama Ortamı 43
4.3. AES Şifreleme Metodunun Yazılımsal Uygulaması 47
4.4. AES Şifreleme Metodunun Donanımsal Uygulaması 48
5. SONUÇLAR VE TARTIŞMA 53
5.1. Şifreleme ve Şifre Çözme Sonuçları 53
5.2. Şifreleme ve Şifre Çözme İşlemlerinin Güç Tüketim ve Performans Analizi 55
5.3. Değerlendirme 64
KAYNAKLAR 65
ÖZGEÇMİŞ 68
TEZ ÖĞRENCİSİNE AİT TEZ İLE İLGİLİ BİLİMSEL FAALİYETLER 69

ŞEKİLLER LİSTESİ

Şekil 1.1. Nesnelerin İnterneti ve Tehditler 2
Şekil 2.1. Nesnelerin İnterneti İlk Örnekler 4
Şekil 2.2. Nesnelerin İnterneti Genel Yapısı 5
Şekil 2.3. Nesne Kavramının Basit Bir Şeması 6
Şekil 2.4. MQTT Protokolü Örnek Bağlantısı-I 12
Şekil 2.5. MQTT Protokolü Örnek Bağlantısı-II 12
Şekil 2.6. Geçmişten Geleceğe Nesnelerin İnterneti 17
Şekil 3.1. BLE Yazılım Akış Diyagramı 27
Şekil 3.2. BLE Yığıtı 28
Şekil 3.3. BLE Protokolü Frekans Aralığı 29
Şekil 3.4. İzleyici (Observer) Modu 29
Şekil 3.5. Merkez (Central) Mod 30
Şekil 3.6. BLE Paket Formatları 32
Şekil 4.1. AES Algoritmasının İşleyişi 34
Şekil 4.2. AES Algoritmasının Bir Döngüsü 34
Şekil 4.3. AES Algoritmasının Akış Şeması 35
Şekil 4.4. SubByte Dönüşümü Temsili 36
Şekil 4.5. Satır Kaydırma İşlemi (Forouzan, 2008) 38
Şekil 4.6. MixColumn Dönüşümü 39
Şekil 4.7. Anahtar Ekleme Aşamasında XOR İşlemi 40
Şekil 4.8. Anahtar Genişletme 41
Şekil 4.9. AES Şifresi Ana Fonksiyonu 43
Şekil 4.10. AES CBC Modu 43
Şekil 4.11. BLE Durumlarına Göre Harcanan Güç 45
Şekil 4.12. İşlem Bitiminde Harcanan Güç 45
Şekil 4.13. BLE Cihaz Konfigürasyonları 46

Şekil 4.14. AES Yazılım Akış Şeması 47
Şekil 4.15. AES Algoritması Yazılım Uygulaması Ana Fonksiyonu 48
Şekil 4.16. AES Donanım Uygulaması Akış Şeması 49
Şekil 4.17. AES Algoritması Donanım Uygulaması Ana Fonksiyonu 51
Şekil 5.1. CC2540-RS232 Bağlantısı 53
Şekil 5.2. AES Yazılım Uygulaması Şifreleme İşlemi 54
Şekil 5.3. AES Yazılım Uygulaması Şifre Çözme İşlemi 54
Şekil 5.4. AES Donanım Uygulaması Şifreleme İşlemi 55
Şekil 5.5. AES Donanım Uygulaması Şifre Çözme İşlemi 55
Şekil 5.6. Test Ortamı Blok Şeması 56
Şekil 5.7. Test Altındaki BLE Cihazı 56
Şekil 5.8. Yazılım Uygulamasının Güç ve Zaman Analizi 57
Şekil 5.9. BLE Durumları ve Donanım Uygulaması Güç Performansı 58
Şekil 5.10. AES Donanım Uygulamasının Güç ve Zaman Analizi 59
Şekil 5.11. BLE Durumları ve Donanım Uygulaması Güç Performansı 59

ÇİZELGE LİSTESİ

Çizelge 2.1. Nesnelerde (MCU ve Algılayıcılar Arasında) Kullanılan Haberleşme Protokolleri 7
Çizelge 2.2. Kablosuz Haberleşme Protokolleri 8
Çizelge 2.3. Ağ geçidi - Bulut Bağlantı Protokolleri 11
Çizelge 2.4. Büyük Veri İçin Bazı Veri Tabanları 15
Çizelge 3.1. Bluetooth Klasik-BLE Karşılaştırması 24
Çizelge 3.2. Protokollerin Sürümlere Göre Hız Karşılaştırmaları 25
Çizelge 3.3. Farklı Cihaz Tiplerinin Haberleşmesi 26
Çizelge 4.1. AES S-Kutusu Dönüşüm Çizelgesi 37
Çizelge 4.2. AES Ters S-Kutusu Dönüşüm Çizelgesi 38
Çizelge 4.3. AES-128 Bit için Rcon Değerleri 42
Çizelge 4.4. CC2540 Teknik Özellikleri 44
Çizelge 4.5. ENCCS Yazacı İçin Bit Dağılımı 52
Çizelge 4.6. ENCDI Yazacı İçin Bit Dağılımı 52
Çizelge 4.7. ENCDO Yazacı İçin Bit Dağılımı 52
Çizelge 5.1. Yazılım Uygulaması İçin Zaman ve Akım Ölçümleri 61
Çizelge 5.2. Donanım Uygulaması İçin Zaman ve Akım Ölçümleri 61
Çizelge 5.3. Yazılım ve Donanım Uygulamaları Performans Analizi Karşılaştırması 63

SİMGELER VE KISALTMALAR

SİMGELER

K	Şifre Anahtarı
Nr	Döngü Sayısı
RCon	Anahtar Genişletme Döngü Sabiti
W	Şifre Kelimesi

KISALTMALAR

AES	İleri Şifreleme Standardı (Advanced Encryption Standard)
BLE	Bluetooth Düşük Enerji (Bluetooth Low Energy)
DES	Data Encryption Standard
IoT	Nesnelerin İnterneti (Internet of Things)
LAN	Yerel Alan Ağı (Local Area Network)
LE	Düşük Enerji (Low Energy)
MCU	Mikro Kontrolcü Birimi (Microcontroller Unit)
MDS	Maximum Distance Separable (Özel bir matris)
NFC	Near Field Communication (Yakın Alan İletişimi)
TLS	Taşıma Katmanı Güvenliği (Transport Layer Security)
WAN	Geniş Alan Ağı (Wide Area Network)

BÖLÜM 1

GİRİŞ

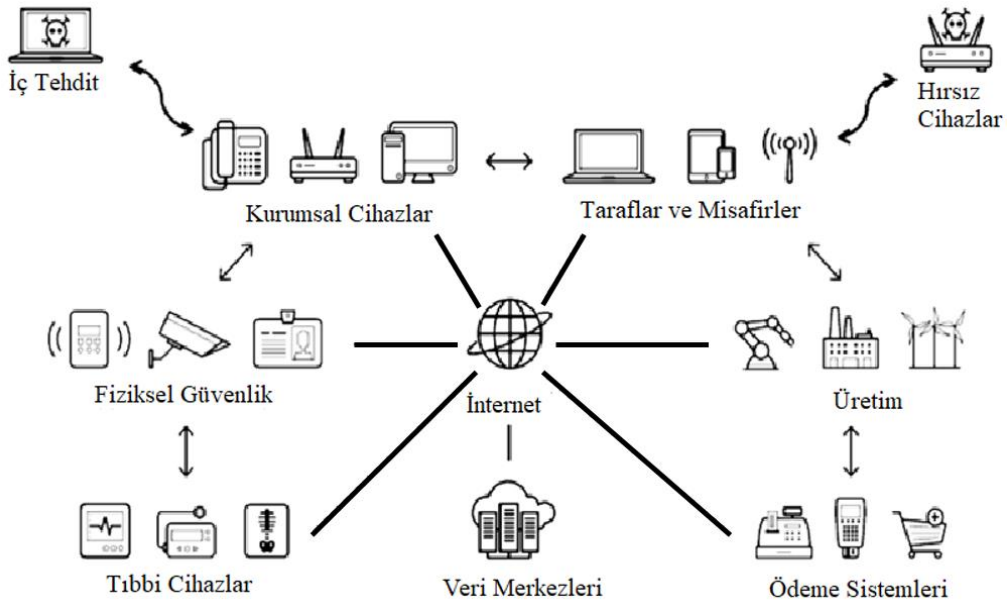
Gelişen teknolojiyle birlikte ihtiyaçlar üzerine akıllı sistemlerin üretimi ve kullanımı çok ciddi boyutlara ulaşmış durumdadır. Sensör ve aktüatörlerin başrolü paylaştığı bu sistemleri etrafımızda sürekli görmekteyiz.

Nesnelerin interneti (IoT) dünyası, RFID, WiFi, 4G, IEEE 802.15.x gibi kablosuz haberleşme teknolojilerinin gelişmesiyle her geçen gün genişlemekte ve günümüzde çok sayıda akıllı nesne ve sistemler üretilmektedir. Yapılan tahminlere göre 2020 yılı itibariyle yeryüzünde 50 milyar akıllı nesne kullanımda olacaktır (Cisco, 2016). Bu hızlı artışın sonucu olarak bu sistemler, birtakım ihtiyaçları beraberinde getirmektedir.

Sistemlerin verimli bir şekilde ve doğru çalışması oldukça önemlidir. Bu doğrultuda, çalışan sistemlerin enerji tüketimlerini azaltmak temel amaçlardan biri haline gelmiştir. Nesnelerin interneti kapsamında bu amaçla kullanılan teknolojilere örnek olarak; 6LowPAN, Zigbee, BLE, Wifi gibi teknolojiler verilebilir. Bu amaçla en çok tercih edilen teknolojilerden bir tanesi Bluetooth düşük enerji teknolojisidir. BLE ile birlikte çok daha az maliyetli, uzun ömürlü, düşük güç tüketimli, verimli sistemler geliştirilmektedir.

Nesnelerin interneti sistemlerinin çok sayıda olması ve her geçen gün hızla artması ile meydana gelen önemli bir sorun vardır: güvenlik sorunu. Nesnelerin interneti ya da düşük güç tüketimli cihazlar kolay saldırılabilecek sistemlerdir. Bu sistemlerin veri iletimlerinin nasıl şifrelendiği, hangi kullanıcının hangi sistem üzerinde yetkisi olduğu ya da olmadığı gibi konular iyi tanımlanmazsa ciddi veri sızıntıları ve diğer güvenlik problemlerinin oluşması kaçınılmazdır. Mesaj değiştirme/ekleme, trafik analizi, DoS saldırıları, izinsiz dinleme, izinsiz giriş gibi saldırılar, nesnelerin interneti kapsamındaki

sistemlerin karşı karşıya kaldığı bazı saldırı türleridir (Riahi Sfar, Natalizio, Challal, & Chtourou, 2018). Şekil 1.1'de internete bağlı cihazlar üzerindeki tehditlerin bir temsili görülmektedir (Wardekar & Ingole, 2013).



Şekil 1.1. Nesnelerin İnterneti ve Tehditler

Nesnelerin interneti kapsamında üretilen sistemlerin maliyetlerinin az olması, az güç tüketiyor olması ve güvenilir olması gerekmektedir. Güvenlik günümüz nesnelerin interneti uygulamalarında en önemli sorunlardan bir tanesidir. Çalışan sistemler sürekli olarak iç ve dış tehditlerle karşı karşıyadır. Güvenlik sorununu çözmek için genellikle iki faktörlü kimlik doğrulama, hafif sıklet şifreleme algoritmaları, AES, DES gibi algoritmalar kullanılmaktadır. Bu çalışmada Bluetooth düşük enerji ile iletim yapan bir geliştirme kitinin üzerinde yazılımsal ve donanımsal olarak AES algoritmasının uygulaması ve bunların tükettikleri güçlerin analizi yapılmaktadır. Çalışmada kullanılan CC254x geliştirme kiti üzerinde AES algoritması donanımsal olarak destekli gelmektedir. Çalışmanın amacı, donanımsal AES desteği olan bu kite, AES algoritması yazılımsal olarak da uygulandığında her iki uygulamanın güç tüketimleri, hızları ve performansları arasındaki farkı analiz etmektir.

BÖLÜM 2

NESNELERİN İNTERNETİ

Corser'a göre nesnelerin interneti, insan müdahalesi ile veya insan müdahalesi olmadan birbirleriyle haberleşebilen ve veri işleyebilen benzersiz bir tanımlama ile bağlanan cihazların kablolu veya kablosuz ağıdır (Corser, 2017).

Alaba ve diğerleri ise nesnelerin internetinin, insan müdahalesi olmadan çeşitli sensör ve nesneler arasında doğrudan iletişim kurduğundan dolayı son yıllarda kullanımı hızla artmakta olan bir sistem olduğunu düşünmektedir (Alaba, Othman, Hashem, & Alotaibi, 2017).

Nesnelerin interneti, kullanıcıların hayat kalitelerini geliştiren sistemlere yeni bir yön vermektedir. Kullanıcılar için nesnelerin interneti, enerji verimliliğinin kritik olduğu sistemlerde, sağlık, güvenlik, eğitim ve günlük hayatta karşılaşılabilecek çok sayıda farklı durum için çözümler oluşturabilmektedir. İşletmelerde ise; üretim, tarım, satış ve pazarlama gibi sektörlerde üretkenlik ve karar verme mekanizmalarını iyileştirmek için kullanılmaktadır (GSM, 2014).

Nesnelerin interneti, birbirleriyle haberleşebilen çeşitli sensörler, nesneler ve akıllı modüller içermektedir. Nesneler, diğer nesnelerle otomatik olarak bağlantı kurma fonksiyonuna sahiptirler. Nesnelerin interneti modülleri, küçük miktarlarda veriyi iletme, erişim ve veri toplama-veri alma işlemleri için bulut tabanlı kaynakları kullanma özelliklerine sahiptir. İnternet, dünyanın her köşesine yayılmış ve çeşitli yollarla insan hayatını etkilemiş durumdadır. Gün geçtikçe bu artış daha da hızlı bir şekilde devam etmektedir.

Nesnelerin interneti ile kişiler, sensörler, servisler ve nesnelerin haberleşmesi sağlanmıştır (Alaba v. d., 2017). Cihazlar, akıllı şebekelerden sağlık hizmetlerine, akıllı

ulaştırma sistemlerine kadar çok geniş bir alana uygulanmaktadır (Conti, Dehghantanha, Franke, & Watson, 2018).

ITU tanımına (Zennaro, 2017) göre ise nesnelerin interneti, bilgi toplumu için bilgi ve iletişim teknolojilerine dayanan nesneleri birbirine bağlayarak gelişmiş servisler sağlayan küresel bir alt yapı olarak görülebilir.

Nesnelerin internetine yönelik ilk uygulamalar 1980'lı yıllara kadar dayanmaktadır. Şekil 2.1.a'da görülen bir içecek otomatı (internetli kola makinesi) internet bağlantılı ilk uygulama olarak kabul edilmektedir. Benzer bir uygulama 1991 yılında Cambridge Üniversitesinde gerçekleştirilmiştir. Şekil 2.1.b'de görülen sistem, yaklaşık 15 akademisyenin ortak kullandıkları kahve makinesini ("Trojan room" kahve makinesi) görebilmek/izleyebilmek amacıyla kurulmuştur ve kahve makinasının görüntüsünü dakikada 3 defa bilgisayar ekranına göndermektedir. Her ne kadar internet bağlantısı kullanılmamasına rağmen, çevrimiçi ve gerçek zamanlı haberleşme özelliklerinden dolayı ilk uygulamalardan biri olarak kabul edilir.



a. İçecek Otomatı b. "xcoffee" Kahve Makinesi

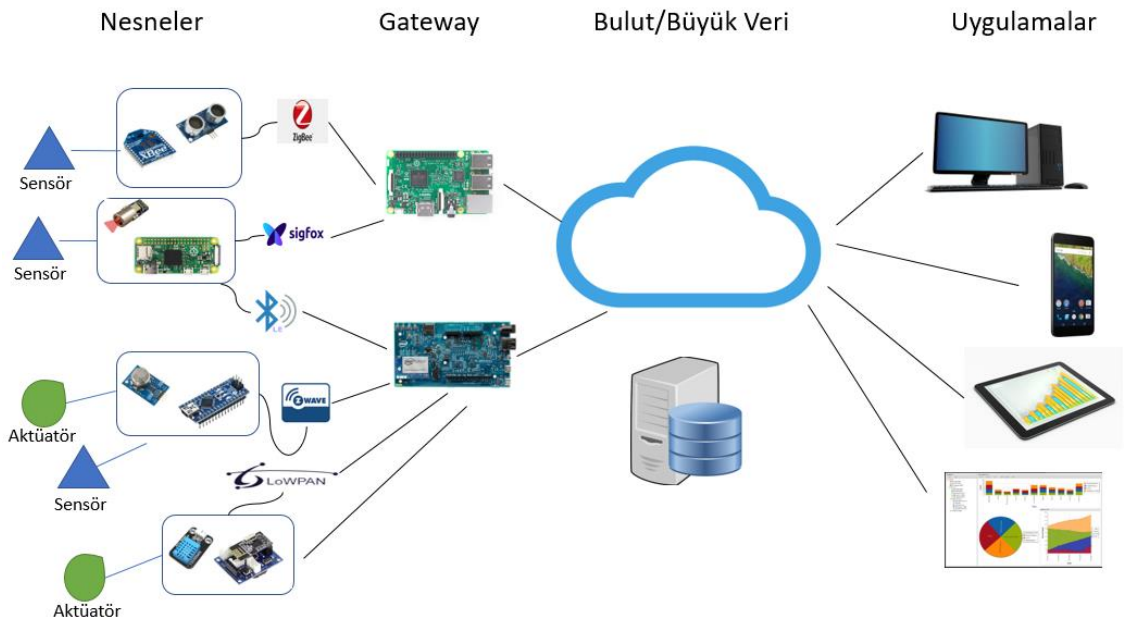
Şekil 2.1. Nesnelerin İnterneti İlk Örnekler

"Nesnelerin İnterneti" kavramı ilk kez 2000'li yılların başında Kevin Ashton tarafından MIT AutoID laboratuvarında ortaya atılmıştır.

Kevin Ashton'ın karşısına çıkan problem stoktaki ürünlerin takibinin ve yönetiminin zor olmasıydı. Bu amaçla bir çözüm geliştiren Ashton, RFID yongalarını

ürünlere yerleştirerek internet üzerinden takip edilmesini ve kolay yönetilebilmesini sağlamış oldu.

Günümüzde ise IoT mimarisi Şekil 2.2'deki gibi gösterilmektedir.



Şekil 2.2. Nesnelerin İnterneti Genel Yapısı

2.1. Nesnelerin İnterneti Mimarisi

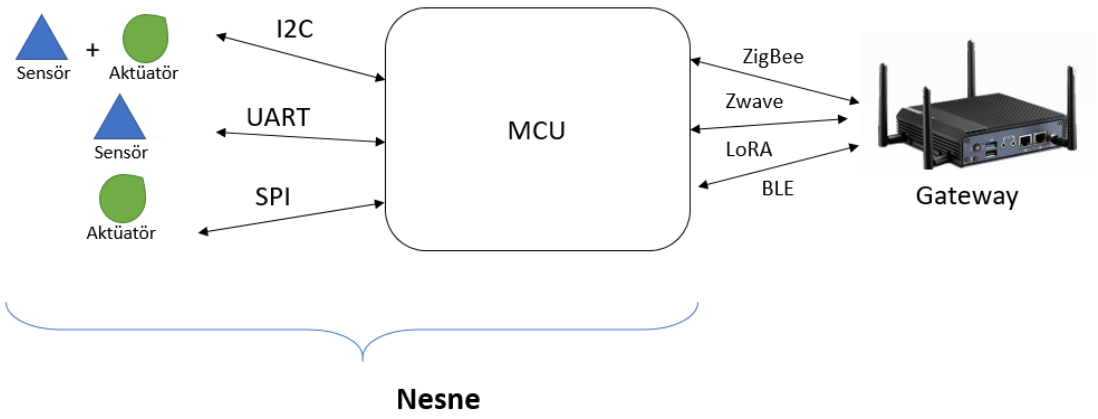
Şekil 2.2'de görüldüğü gibi nesnelerin interneti sistemlerini dört temel katmanda göstermek mümkündür; nesne, ağ geçidi (gateway), bulut ve büyük veri sunucuları, uygulama. Sınıflandırılan cihazlar, ağ üzerinde farklı katmanlarda çalışmaktadır ve uygulamaların gerçekleştirilebilmesi için yani en alt katmandan en üste çıkılabilmesi için bu katmanların her birinin kurulması gerekmektedir. Örneğin düşük enerjili bir protokolle çalışan bir nesnenin bulut katmanına çıkabilmesi için bir ağ geçidi cihazına ihtiyacı vardır.

Bu bölümde nesnelerin interneti mimarisinde bulunan elemanlar anlatılacaktır.

2.1.1. Nesne

En alt katman olan algılama katmanında bulunan "Nesne" kavramı çok geniş bir kavramdır. Algılama, komut verme, çevreden veri toplama gibi işleri yapan işlemcisi, işletim sistemleri, bellenimi olabilen donanımlardır. Tek başlarına yapabilecekleri sınırlı olduğundan gerekli protokolleri, standartları ve teknolojileri kullanabilirler.

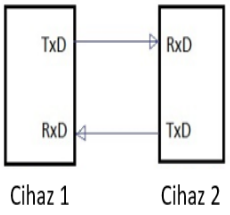
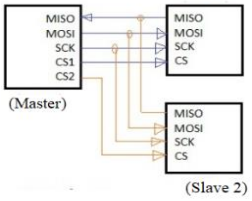
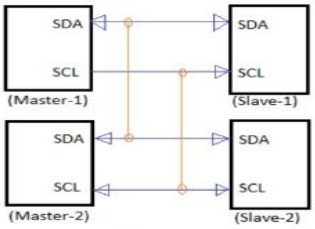
Nesnelerde çift taraflı bir bağlantı görülmektedir: sensör veya aktüatör gibi algılayıcılar ile kurulan bağlantı ve ağ geçidi cihazı ile internete çıkmak için kurulan kablosuz bağlantı. Bu katmanda sensörler ile dışarıdan veri toplanır ya da aktüatörler ile sistemlere komut gönderilebilir. Fakat bu sensör ve aktüatör gibi cihazlar tek başına çalışan cihazlar değildirler. Örneğin cep telefonlarında, kamera, GPS, akselerometre gibi çok sayıda sensör mevcuttur fakat telefon sadece bir sensör değildir. Şekil 2.3'te nesne olarak ifade edilen donanımların basit bir blok şeması görülmektedir.



Şekil 2.3. Nesne Kavramının Basit Bir Şeması

Algılayıcıların bağlanması gereken en azından bir mikro kontrolcü (MCU) bulunmalıdır. Algılayıcılar ile MCU arasında iletişimin sağlanması için kullanılabilecek protokoller Çizelge 2.1'de verilmiştir.

Çizelge 2.1. Nesnelerde (MCU ve Algılayıcılar Arasında) Kullanılan Haberleşme Protokolleri

Özellikler	**UART**	**SPI**	**I2C**
Arayüz Diyagramı			
Pin Tanımları	TxD: Veri Taşıma RxD: Veri Alma	SCLK: Seri Saat MOSI: Master Çıkış, Slave Giriş MISO: Master Giriş, Slave Çıkış SS: Slave Seçim SDA: Seri Veri SCL: Seri Saat	SDA: Seri Veri SCL: Seri Saat
Aktarım Hızı	230-460Kbps	10-20Mbps	100Kbps/400Kbps/3.4Mbps
Haberleşme Tipi	Asenkron	Senkron	Senkron
Protokol	8 bit veri için bir başlama ve bir bitiş biti kullanılır.	Kullanıcıya göre farklılık gösterir.	Başlama ve bitiş bitlerini kullanır. Verinin her 8 biti için ACK biti kullanır.
Avantajları	Hemen her cihazda 9 pinli UART desteği olduğundan kullanımı çok olan basit bir protokoldür.	Basit bir protokol olup büyük işlem yükleri gerektirmez. Çift yönlü iletişimi destekler. SPI daha yüksek veri iletimine ve daha uzun mesafelere imkân tanır. I2C'den daha az güç tüketir.	İletişim için 2 kablodan daha azına ihtiyaç vardır. I2C adresleme basit bit işlemdir, böylelikle veri yoluna yeni cihaz eklemek kolaylaşır. Akış kontrolü mevcuttur.
Dezavantajları	Sadece iki cihaz arasında iletişim kurmada iyidir. Başlangıçta iletişimden önce cihazlar arasında kararlaştırılan sabit veri hızını destekler, aksi halde verilerde hatalar meydana gelir.	Slave cihazların sayısı artarsa pinlerin de sayısı artacağından donanımsal bir karmaşa meydana gelecektir. Akış kontrolü yoktur.	Master ve slave cihazların sayısı arttıkça karmaşıklık artar. I2C, yarı çift yönlü iletime izin verir. İşlem yükü fazladır.

Algılayıcılardan elde edilen verilerin ağ geçidine aktarılması sırasında kullanılan protokoller ise Çizelge 2.2'de verilmiştir.

Çizelge 2.2. Kablosuz Haberleşme Protokolleri

Protokol	**ZigBee**	**BLE**	**Z-Wave**	**NFC**	**Wi-Fi**	**LoRaWAN**
IEEE Standart	802.15.4	802.15.1	ITU-T	ISO 13157	IEEE 1901-2010	LoRaWAN
Frekans	2.4 GHz	2.4-2.5 GHz	908.42 MHz	13.56 MHz	2.4 GHz; 5 GHz	Değişken
Aralık	10m	~50m	~30m	~5m	100m	2-5Km
Şifreleme	AES	AES	AES	-	RC4/AES	64-128 bit şifreleme
Ağ Tipi	WPAN	WPAN	WPAN	P2P	WPAN/ P2P	LPWAN
Güç Tüketimi	~40mA	~12.5mA	2.5mA	~50mA	~116mA	~20mA
Veri Oranı	250 kbps	1/2/3 Mbps	40 kbps	424 kbps	0.1-54 Mbps	< 50 kbps

Z-wave, ev otomasyonları ve özellikle akıllı ev IoT sistemlerinin iletişimi için kullanılan düşük güç tüketimli ortam erişim kontrolü (MAC) protokolüdür. 30 metrelik noktadan noktaya iletişimleri sağlar ve IoT uygulamalarında küçük mesajlaşmalara (ışık kontrolü, enerji kontrolü, giyilebilir sağlık bakım kontrolü gibi) uygun bir protokoldür. Ana cihazın bağlanan cihazları kontrol ettiği, onlara komutlar gönderdiği sahip/köle (master/slave) mimarisini takip eder (Gerber, 2017a).

ZigBee, akıllı evler, uzaktan kontrol sistemleri, sağlık bakım sistemleri gibi IoT uygulamaları için tasarlanmış, geniş mesafeli, akıllı enerji tüketimli bir protokoldür. Yıldız, uçtan-uca, küme-ağaç gibi çok sayıda ağ topolojisini desteklemektedir. ZigBee protokolü iki yığıt tipi tanımlamaktadır: ZigBee ve ZigBee Pro. Bu yığıt tipleri karmaşık ağları destekler ve düşük bellek ve işlem gücü olan uygulamalarla çalışma fırsatı sunar. ZigBee Pro simetrik şifreler kullanarak daha güvenli iletişimi sağlamaktadır (Gerber, 2017a).

6LowPAN, IP tabanlı bir protokol olup açılımı "IPv6 Düşük Güçlü Kablosuz Kişisel Alan Ağı" dır. Bluetooth ve Zigbee gibi bir nesnelerin interneti uygulama

protokolü olmak yerine başlık sıkıştırma mekanizmasına sahip bir ağ protokolüdür. Düşük güç tüketimi, IP tabanlı modül ve geniş örgü ağı desteği bu protokolü nesnelerin interneti için önemli bir seçenek haline getirmektedir (Olsson, n.d.).

NFC, Yakın alan iletişimi (Near Field Communication), yakın mesafeli ve temassız bir iletişim protokolüdür. En temel amaçlarından biri kullanıcılara global olarak kısa mesafeli temassız bir iletişimi sağlamaktır. Her zaman bir başlatıcı ve bir hedef tarafı bulunur ve başlatıcı aktif olarak pasif bir hedefe güç sağlayabilecek bir RF alanı oluşturur. Bu işlem, NFC hedeflerinin, batarya gerektirmeyen etiketler, çıkartmalar, anahtarlıklar veya kartlar gibi çok basit form faktörlerini almasını sağlar. Bazı kullanım alanlarına, akıllı posterlerden müzik ya da video indirmek, temassız olarak otobüs ya da trende ücret ödeme, uzaktayken yazıcıdan bir görüntü yazdırma, temassız kredi kartları gibi alışverişlerde telefondan ödeme yapma gibi örnekler verilebilir (Wardekar & Ingole, 2013).

IoT sistemlerinde kullanılabilecek çok çeşitli nesne ve bu nesnelerde kullanılan çok çeşitli donanımlar bulunmaktadır. Gloria ve diğerleri yaptıkları çalışmada sensörlerle veri topladıkları aşamada Arduino Uno yongasını nesne olarak kullanmışlardır (Glória, Cercas, & Souto, 2017). Yapılan başka bir çalışmada Mathur ve diğerleri, nesne olarak OpenMote platformu kullanmıştır (Mathur v. d., 2017). Yong ve diğerlerinin yaptığı bir akıllı fitness sistemi çalışmasında ise algılayıcılarla iletişim kurmak için daha az maliyetli olması amacıyla ESP8266 yongası kullanılmıştır (Yong v. d., 2018).

2.1.2. Ağ Geçidi

Ağ geçidi katmanı, nesneleri internet ile bağlayan önemli bir ağ katmanıdır. Algı (nesne) katmanında olduğu gibi bu katmanda da çift taraflı bağlantı söz konusudur. Bir taraftan nesnelerin bulunduğu algı katmanı ile belirli protokoller sayesinde bağlantı kurarken diğer taraftan bu nesnelerin bulut katmanı ile iletişime geçebilmesi için TCP/IP protokolünden faydalanmaktadır.

Ağ geçidi cihazı, nesneler ile bulut arasında köprü görevi görmektedir. Nesneler bir ağ geçidi cihazı ile konuşur; ağ geçidi cihazı da bu nesnelerden veriyi alıp bulut katmanına çıkarır. Nesneler ve bulut arasında iletişim için ekstra böyle bir katman olması,

nesnelerin interneti uygulamalarına çok büyük avantaj sağlamaktadır. Öncelikle nesneler uzaktan bağlantı kurduklarından yani uzun mesafeli bağlantıya ihtiyaç duyduklarından güç tüketimi ve maliyet artmaktadır. Bu durum, sınırlı bataryaya sahip küçük IoT nesneleri için problem oluşturmaktadır. Ağ geçidi cihazı kullanılarak kısa mesafeli aktarım gerçekleştirildiği takdirde daha az enerji harcanmış olacaktır. Bununla birlikte, IoT sistemlerinde çok sayıda farklı nesne kullanılmaktadır. Ağ geçidi cihazları nesneler ile çeşitli protokoller üzerinden haberleşebilirler ve sonrasında aldıkları veriyi buluta iletmek için MQTT gibi standart protokollere çevirebilirler.

Sensörler ve diğer nesneler algı katmanında bazen saklanamayacak boyutlarda veri toplamaktadırlar. Bu verilerin bir kısmı işe yarar verilerken bir kısmı ise kullanılmayacak verilerdir. Örneğin güvenlik kameralarından alınan görüntülerde sürekli boş olan koridorun görüntülerinin gönderilmesi gerekli değildir. Ağ geçidi cihazları, nesnelerden üretilen verileri ön işleme ve filtreleme gibi aşamalardan geçirerek iletim, işleme ve saklama aşamalarının yükünü azaltmış olur.

Ayrıca nesneler doğrudan buluta bağlanmaya kalkarlarsa bulut tarafının yoğunluğundan ve anında cevap veremeyişinden dolayı gecikmeler meydana gelebilmektedir. Bu durum özellikle zamanın kritik olduğu sağlık uygulamaları, akıllı arabalar gibi IoT sistemlerinde problemlere sebep olacaktır (Iotforall, n.d.).

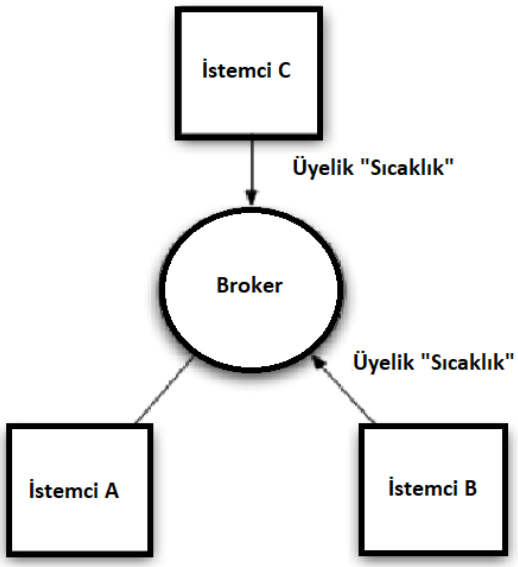
Bu katmanda bulut ile bağlantı için kullanılabilecek protokoller Çizelge 2.3'te verilmiştir.

Çizelge 2.3. Ağ geçidi - Bulut Bağlantı Protokolleri

Protokoller	**Taşıma**	**Mimari**	**Güvenlik**	**Hesaplama Kaynağı**
CoAP	UDP	İstek/Cevap	DTLS	10Ks/RAM Flash
MQTT	TCP	Yayın/Üye	TLS/SSL	10Ks/RAM Flash
XMPP	TCP	İstek/Cevap Yayın/Üye	TLS/SSL	10Ks/RAM Flash
RESTFUL	HTTP	İstek/Cevap	HTTPS	10Ks/RAM Flash
AMQP	TCP	Yayın/Üye	TLS/SSL	10Ks/RAM Flash
Web Soket	TCP	İstemci/Sunucu Yayın/Üye	TLS/SSL	10Ks/RAM Flash
SMQTT	TCP	Yayın/Üye	Kendi Güvenliği	10Ks/RAM Flash
DDS	TCP/UDP	Yayın/Üye	TLS/SSL	100 Ks/RAM Flash+++

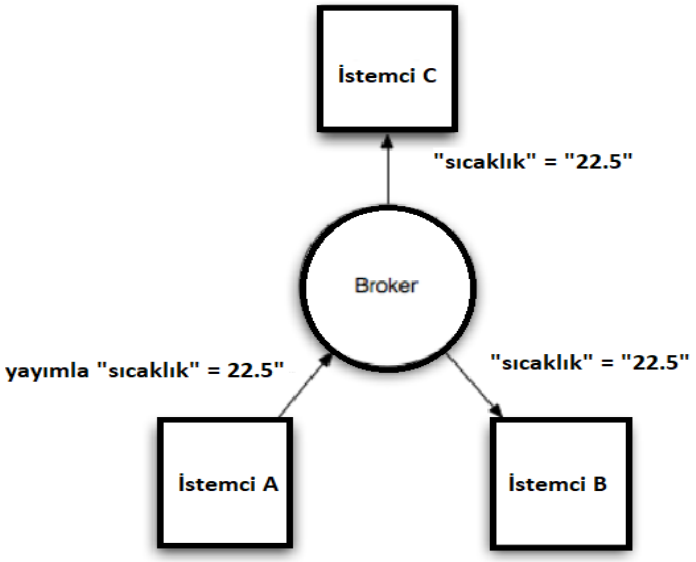
MQTT(Message Queue Telemetry Transport), hafifsıklet (Lightweight) M2M haberleşmesi için tasarlanan bir yayın/üyelik mesaj protokolüdür. IBM tarafından geliştirilmiş olup günümüzde açık standart haline gelmiştir.

MQTT, her sensörün bir istemci olduğu, TCP üzerinden broker ismi verilen bir sunucuya bağlandığı sunucu/istemci modeline sahiptir. Mesaj tabanlı bir protokoldür. Her mesaj "başlık" olarak bilinen bir adres yayar. İstemciler birden fazla başlık için üye olabilirler.



Şekil 2.4. MQTT Protokolü Örnek Bağlantısı-I

Örnek olarak Şekil 2.4'te gösterildiği gibi üç istemci ve bir brokerin olduğu basit bir ağ yapısı verilebilir. Üç istemci, broker ile TCP bağlantısı açar, istemci B ve C "Sıcaklık" başlığına üye olur. Bir süre sonra istemci A, Sıcaklık başlığı için 22.5 değerini yayımlar. Broker ise bu değeri, kendine bağlı istemcilere gönderir. Bu işlemler, MQTT istemcilerine bire-bir, birden-çok'a ve çoktan-bire bağlantılarını kurmayı sağlar (Jaffey, 2014).



Şekil 2.5. MQTT Protokolü Örnek Bağlantısı-II

COAP (Constrained Application Protocol), HTTP protokolü gibi CoAP protokolü de dosya taşıma protokolüdür, HTTP aksine kaynak kısıtlı olan cihazlarda kullanılmaktadır. CoAP paketleri, HTTP - TCP akışından çok daha küçük boyutlardadır. Paketlerin üretimi basittir ve fazladan RAM kullanmadan dönüştürülebilir (Jaffey, 2014).

CoAP, TCP değil UDP protokolü üzerinde çalışır. İstemciler ve sunucular bağlantısız datagramlar üzerinde haberleşirler. Datagram tabanlı bir protokoldür. Datagram, karşı tarafa iletilmesi kesin olmayan, iletim süresi belli olmayan aktarım ünitesidir.

CoAP, istemci/sunucu modelini izler. İstemciler sunuculara istek gönderirler, sunucular geri cevap döndürürler. İstemciler, kaynaklar üzerinde GET(Al), PUT(Bırak), POST(Gönder) ve DELETE(Sil) gibi işlemler yapabilirler.

Her CoAP mesajı UDP datagramlarından bir bölümü kaplar. CoAP mesajları CoAP tarafları arasında asenkron şekilde değiştirilir. Paketler sipariş ve tekrar iletim işlemleri yazılım yığıtlarına bağlıdır (Gerber, 2017a).

AMQP (The advanced message queuing protocol), TCP üzerinden ya da diğer taşıma protokolleri ile iletişim için yayın/üye yaklaşımını asenkron olarak gerçekleştirir. Güvenilirliği aşağıdaki 3 teslim seviyesi ile esneklik kazanır:

- Bir teslimat ya da hata gerçekleştiğinde bir mesaj sadece bir kez gönderilir.
- Bir ya da daha fazla mesajın teslimatı kesindir.
- Bir mesajın sadece bir kez gönderilecek oluşu kesindir.

AMQP protokolünde güvenlik TCP üzerinden TLS/SSL kullanılarak sağlanmaktadır (Eduonix, n.d.).

*XMPP (Extensible Messaging and Presence Protocol), g*ünümüz IoT dünyasında en çok kullanılan iletişim ve mesajlaşma protokollerinden biri olup IETF tarafından standartlaştırılmıştır. Bu protokol, tüm ağlarda geniş kullanımıyla tanınan bir protokoldür. IoT ihtiyaçlarından olan küçük mesajlar ve düşük gecikmeler XMPP tarafından desteklendiğinden bu protokol IoT iletişim ve mesajlaşması için iyi bir seçenektir.

XMPP protokolü istek/cevap ve yayın/üye modellerinin her ikisini de desteklemektedir: istek/cevap modeli çift yönlü iletişime izin verirken yayın/üye modeli

çok yönlü iletişimi sağlar (veri al/ver). Bu protokolün çok sayıda uzantısı bulunmaktadır, bu uzantılar da küçük ortamlarda çalışmaya olanak sağlar.

XMPP protokolü metin iletişimi için XML kullanmaktadır, bu durum ağ trafiğine ek yüke neden olabilmektedir fakat XML sıkıştırma ile bu durum çözülebilmektedir. (Shatnawi, 2016)

Nesnelerin interneti sistemlerinde kullanılabilecek çok çeşitli ağ geçidi bulunmaktadır. Örnek olarak, bir çalışmada ağ geçidi cihazı olarak Raspberry Pi 3 cihazı kullanılmıştır (Glória v. d., 2017). Başka bir çalışmada ağ geçidi cihazı olarak yine Raspberry Pi entegre sistemi kullanılmıştır (Kang & Choo, 2018). Mathur ve diğerlerinin yaptığı çalışmada MQTT ve UART bağlantılarını aynı anda kurabilecek Intel Edison Arduino Breakout entegresi kullanılmıştır (Mathur v. d., 2017).

2.1.3. Bulut/Büyük Veri

Bulut katmanı ise verinin işlenmesi ve saklanması gibi aktivitelerin cihazın kendisinde değil de bulut üzerinde olmasından dolayı IoT için önemli bir katmandır. Bulut sistemini kullanmak verileri birleştirmek ve bu verilerden bilgi edinmek gibi avantajlar sağlar. Örneğin, dünyanın farklı yerlerinde bulunan iki sensörden alınan verileri karşılaştırmak bulut üzerinden çok kolay biçimde gerçekleştirilebilirken bulut kullanımı olmadan bu karşılaştırmayı yapmak oldukça zor ve maliyetlidir.

Bulut kullanımı aynı zamanda yüksek ölçeklendirilebilme imkânı sağlamaktadır. Yüzlerce, binlerce ve hatta milyonlarca sensöre sahip olunduğunda, her bir sensör üzerinde büyük miktarlarda enerji harcamak yerine sensörlerden toplanan veriler buluta aktarılarak işlemler gerçekleştirilmektedir. IoT sistemlerinde işleme, komut verme, analitik gibi işlemler bulut üzerinden meydana gelir. Yani asıl akıllı işlemler bu katmanda gerçekleşir (Iotforall, n.d.).

Bir nesnelerin interneti uygulamasında çok sayıda farklı cihaz, protokol vs. kullanıldığından, bulut katmanına sürekli olarak çok büyük miktarlarda veri gelmektedir. Aynı katmanda görülen “büyük veri (big data)” sunucuları da bu sebeple oluşturulmaktadır.

Verilerin çok fazla olması, beraberinde depolama ve yönetim sorununu getirmektedir. Geleneksel veri tabanları (MsSQL, Oracle vb...) ile bu işlemlerin yapılması zordur. Verilerin saklanması ve yönetilebilmesi amacıyla kullanılabilecek bazı veri tabanları Çizelge 2.4'te verilmiştir (Infoivy, n.d.).

Çizelge 2.4. Büyük Veri İçin Bazı Veri Tabanları

Veritabanı	Geliştiricisi	Saklama Tipi	Özellikleri	Avantajları
MongoDB	10gen	Belge	• Tutarlılık • Bölünebilme toleransı • Sürdürülebilirlik	• Dinamik sorgular • İstatistiksel veri • Sık yazma ve nadiren okuma yapıldığı sistemlerde verimlidir.
SimpleDB	Amazon	Belge	-	• Basit veritabanı uygulaması
Couchbase	Couchbase	Belge	• Yüksek Erişilebilirlik • Sürdürülebilirlik • Tutarlılık	• Oturum depolama • Kullanıcı profili depolama • İçerik depolama
Hbase	Apache	Sütun	• Tutarlılık • Bölünebilme toleransı • Sürdürülebilirlik	• Büyük veritabanları için rastgele okuma/yazma
Cassandra	Apache	Sütun	• Yüksek Erişilebilirlik • Bölünebilme Toleransı • Sürdürülebilirlik	• Sık yazma ve az okuma işlemlerinde verimlidir.

2.1.4. Uygulamalar

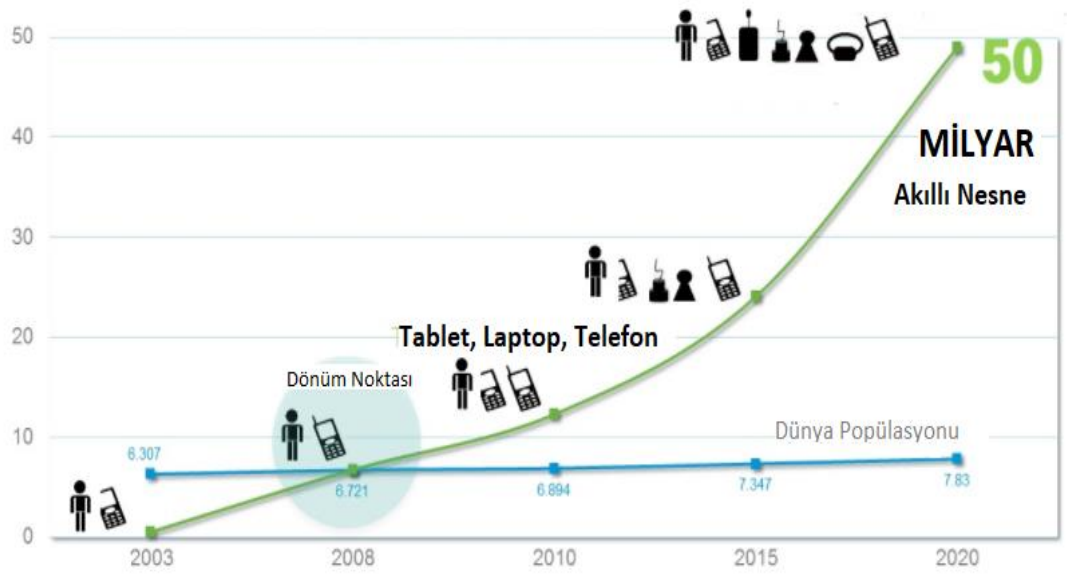
Alt katmanlarda toplanan, filtrelenen ve işlenen veriler son katman olan uygulama katmanına gelirler. Sistem artık kullanıcıya telefon, tablet, bilgisayar ve bazı gömülü sistemler vasıtasıyla kullanıma açılabilir. Bu uygulamalar günümüz nesnelerin interneti dünyasında akıllı evler, akıllı çiftlikler, fabrikalar, sağlık sisteminde kullanılan akıllı cihazlar, akıllı arabalar şeklinde örneklendirilebilir. Bu uygulamalar, dünya üzerinde

kişisel ağlardan (PAN) geniş alan ağlarına (WAN) her yerde karşılaşılabilecek uygulamalardır.

Özet olarak; bir IoT uygulamasını gerçekleştirirken veri aktarımı için öncelikle en alt katman olan algı katmanından başlanır. Bu katmanda nesneler yardımıyla gerekli veriler toplanır. Bu verilerin kullanıcıya erişmesi için öncelikle internete çıkması gerekmektedir. Bu işlem, ilgili ağ geçidi cihazı ile sağlanmaktadır. Bu katman iki ayrı bağlantı yapmalıdır: algı katmanı-ağ geçidi bağlantısı ve ağ geçidi-bulut katmanı bağlantısı. Ağ geçidi katmanına, algı katmanında kullanılan düşük enerji tüketimli bir protokol yardımıyla, veri geçişi sağlanır. Bu geçiş işlemi kullanılan algı katmanı protokolüne göre değişiklik gösterir. Ağ geçidi cihazına geçen veri için artık gerekli protokoller kullanılarak ve IP alınarak buluta geçişi sağlanmalıdır. Bulut/büyük veri sunucusu katmanına geçen veriler bu katmanda da işlenerek kullanıcı katmanı için hazır hale getirilir. Son olarak kullanıcı katmanına geçen veriler telefon, bilgisayar, tablet ve diğer IoT uygulamaları gibi cihazlardan erişilebilirdir.

2.2. Nesnelerin İnternetinde Güvenlik

Nesnelerin interneti uygulamalarıyla kullanılan cihaz sayısı her geçen gün artmaktadır. 2020 yılına kadar bu bağlı cihaz sayısının 50 milyona ulaşması beklenmektedir (Evans, 2011). Şekil 2.6'da de görüldüğü gibi cihaz sayısı dünya popülasyonunu aştığından dolayı 2008 yılı önemli bir dönüm noktasıdır. Bu alana adaptasyon süreci elektrik ve telefona olan adaptasyon sürecinden 5 kat daha hızlıdır. Tahmin edilen 50 milyar cihaza ulaşıldığında kişi başına ortalama olarak 6 cihaz düşecektir (Frahim, Pignataro, Apcar, & Morrow, 2016).



Şekil 2.6. Geçmişten Geleceğe Nesnelerin İnterneti

Bu artışın olumlu sonuçlarının yanında birtakım problemler de ortaya çıkmaktadır: çok sayıda ve çözümü karmaşık güvenlik açıkları, fiziksel güvenlik yetersizlikleri... Dışarıdan gelecek saldırılar, sistemlerde üretilen verilere erişme amacındadırlar. Bunun yanında donanımsal olarak gerçekleşebilecek güvenlik problemleri de mevcuttur: açık portlar, güvenliksiz haberleşme kanalları gibi. Sistemlerin çoğu, insan müdahalesi olmadan otonom olarak çalıştıklarından bu saldırıların belirlenmesi oldukça güçtür. Kimlik doğrulama işlemlerinin yetersizliği, güvenliksiz ağ servislerinin kullanımı, kişisel verilerin gerekmediği halde toplanması, fiziksel olarak yeterli güvenlik önlemlerinin alınmayışı, cihaz güncellemelerinin eski kalması gibi çok sayıda güvenlik açığı sayılabilmektedir.

2.2.1. Güvenlik Açıkları

- Kaynak kısıtlı cihazlarda güvenlik

Çoğu IoT cihazı sınırlı büyüklüklerde hafızaya, saklama kapasitesine ve işlem yapma yeteneğine sahiptir ve sıklıkla düşük güç ile işlem yapmaya ihtiyaç duyarlar. Ağırlıklı olarak şifrelemeye dayalı güvenlik yaklaşımları bu kısıtlı cihazlar için pek uygun değildir çünkü gerçek zamanlı veri iletimi güvenliğini sağlamada karışık şifreleme

ve şifre çözme işlemlerini hızlı bir şekilde yerine getirebilecek kapasiteye sahip değildir. Bu cihazlar sıklıkla ters mühendislik için kullanılabilen güç analiz saldırısı gibi yan kanal saldırılarına maruz kalırlar. Bu sebeplerden, kaynak kısıtlı cihazlar genellikle hızlı ve hafifsıklet şifreleme algoritmaları kullanmaktadır.

- Cihazlarda kimlik doğrulama ve yetkilendirme

Bir nesnelerin interneti sisteminde potansiyel hata noktalarına sahip çoğu cihazın güvenliği için kimlik doğrulama ve yetkilendirme kritik bir aşamadır. Cihazlar uygulamalara, üst servislere ve veri yollarına erişmeden önce kendi kimliklerini kurmak zorundadır. Fakat çoğu cihaz basit parola ile ya da ilk parolayı değiştirmeden kullanarak kimlik doğrulama yaptığından başarısızlığa uğramaktadır.

Bir IoT platformunda güvenliğin sağlanması için güçlü parola ile giriş, sertifikalar ya da iki faktörlü kimlik doğrulama yöntemleri gibi daha kalıcı çözümler bulunmalıdır.

- Cihaz güncellemelerinin yönetimi

IoT cihazları üzerinde çalıştırılan yazılımlara güvenlik ile ilgili yamaları da içeren güncellemelerin uygulanması, beraberinde riskler getirir. Tüm cihazlar uzaktan erişim ile güncellemeyi destekleyemez ya da çalışması durdurulmadan güncellenemezler, bu yüzden fiziksel erişim ile güncellenmeye gerek duyabilirler. Benzer şekilde, güncellemeler tüm cihazlar için ulaşılabilir olmayabilir; eski cihazlar ya da üreticisinin destek vermeyi bıraktığı cihazlar. Hatta güncellemeler ulaşılır olsa bile cihazın sahibi güncellemeyi devre dışı bırakmayı seçebilir. Bu sebeplerden, geliştirilen her bir cihazın versiyonunun takip edilmesi gerekmektedir. Cihaz yönetme sistemleri genellikle otomatik olarak güncellemeyi iptal etme desteğine sahiptir. Bu sayede güncelleme sırasında meydana gelebilecek hatalar karşısında sistemi eski haline getirebilmektedir.

- Veri güvenliği ve bütünlüğünde kesinlik

Veri iletiminden sonra verinin nereye ulaştığının, nerede saklandığının ve nerede işlendiğinin güvenli olup olmaması önemli bir konudur. Artık gerekli olmayan veriler

güvenli bir şekilde yok edilmelidir ve eğer veriler saklanırsa, düzenleyici ve yasal çerçevelerle uyumunun onarılması da önemli bir sorundur.

Verinin değiştirilmediğinden emin olmak için dijital imza ya da kontrol toplamı gibi yöntemler, veri bütünlüğünü sağlamada kullanılmalıdır.

- Güvenli web, mobil ve bulut uygulamaları

Web, mobil, bulut uygulamaları ve servisleri yönetim, erişim, nesnelerin interneti cihazlarının ve verilerinin işlenmesinde kullanılmaktadır. Bu yüzden IoT güvenliğinin çok katmanlı yaklaşımının parçası olarak güvenli olmak zorundadırlar.

IoT uygulamaları geliştirilirken, güvenlik açıklarından kaçınmak için güvenli mühendislik pratiklerinin uygulanması gerekmektedir. Cihazlarda ve uygulamalarda olduğu gibi güvenli kimlik doğrulama, uygulama ve kullanıcıları için kullanılan iki faktörlü kimlik doğrulama, yenilenen şifre uygulamaları gibi yöntemler kullanılmalıdır.

- Yüksek kullanılabilirlik garantisi

Günlük hayatta nesnelerin interneti uygulamaları kullanımı arttıkça, IoT geliştiricilerinin, IoT verilerine ve veriye dayalı web ve mobil uygulamalarına erişime aynı zamanda IoT sistemleri tarafından yönetilen fiziksel nesnelere erişime dikkat etmeleri gerekmektedir. Bağlantı kopukluğu, cihaz ile ilgili bir hata ya da DoS saldırısı gibi durumlar sonucunda oluşacak potansiyel bir aksaklık zor bir durumdan çok daha fazlasıdır. Bazı uygulamalarda oluşabilecek erişim ile ilgili bir problem kazanç kaybına, ekipmanların bozulmasına ve hatta hayat kayıplarına sebebiyet verebilir.

Örneğin, altyapısında sağlık hizmetleri, trafik kontrol sistemleri, kalp pili ve insülin pompaları gibi cihazları içeren sistemler bulunan akıllı şehirlerde erişilebilirlik önemli bir boyut kazanmıştır. Yüksek erişilebilirliği sağlamak amacıyla bağlı nesneler fiziksel müdahaleler kadar siber saldırılara karşı da korunmalıdır.

- Zayıflık ve kazaları tespit etme

İyi performanslara rağmen, güvenlik zayıflıkları ve ihlaller kaçınılmazdır. Bağlı cihaz sayısı açısından sistemlerin karmaşıklığı ve çeşitli cihaz, uygulama, servis ve iletişim protokolleri içeren sistemlerin varlığı, herhangi bir arıza meydana geldiğinde hangi cihazda sorun olduğunun belirlenmesini zorlaştırır. Zayıflık ve ihlallerin belirlenmesinde uygulanan stratejiler içinde, ağ iletişiminin ve sıra dışı durumlar için aktivite geçmişlerinin görüntülenmesi, veri sızıntılarını ortaya çıkarmak için sızma testlerinin yapılması, hata meydana geldiğinde tanımlayıp bildirmek için güvenlik istihbaratı ve analizi uygulanması yöntemleri sayılabilir.

- Zayıflıkların yönetimi

Nesnelerin interneti sistemlerinin karmaşıklığı, bir güvenlik açığının ya da ihlallerin yönetimini zorlaştırmaktadır. Güvenlik açıkları, hangi cihazların etkilendiğini, hangi veri ya da servislere erişildiği ya da bunlardan sızıntılar olduğunu, hangi kullanıcının zarar gördüğünün belirlenmesini ve bu tür durumlar karşısında çözüm için nasıl hareket edildiğinin ortaya çıkarılmasını içerir.

Aygıt yöneticisi, herhangi bir neden dolayı devre dışı kalacak olan cihazlar yerine geçici olarak kullanılabilecek cihazların kaydını tutar. Bu özellik, özellikle zarar verme potansiyeli olan kilit cihazların hareketlerini sınırlandırmak açısından önemlidir.

- Güvenlik konusunun öngörülmesi ve öncelik verilmesi

Nesnelerin interneti sistemlerinde güvenlik uygulamalarını, sadece bir güvenlik açığı meydana geldiğinde bu açığı belirleyip önlemek değil, aynı zamanda oluşabilecek güvenlik açıklarını öngörüp öncesinde hareket etmek için tasarlamak gerekmektedir. Tehdit modelleme, güvenlik sorunlarını tahmin etmede kullanılan bir yaklaşımdır. Diğer yaklaşımlar arasında, olayları ilişkilendirmek amacıyla izleme ve analiz yapma yöntemleri ile çalışma zamanında ortaya çıkan tehditleri görselleştirme yöntemleri bulunmaktadır (Gerber, 2017b).

2.2.2. Güvenlik Gereklilikleri

İnternet, herkesin erişip okuyabileceği, değişiklik yapabileceği ya da bilgi aktarabileceği karşılıklı bir güven ortamı olarak başlamıştır. Nesnelerin interneti ise, bireylerin ve devlet güvenliğinin ve gizliliğinin yüksek oranda tutulması beklenen fakat güvenli olmayan bir ortamda ortaya çıkmıştır. Aynı zamanda kişisel verilerin, şirket sırları, güvenliğin kritik olduğu altyapıların korunması konusunda beklentiler bulunmaktadır (Ericsson, 2017).

- Cihazların gelen bağlantı noktaları sorunu

Bir cihaz veri iletimi yapacağı zaman veriyi göndereceği diğer cihazı dinlemesi gerekir. Geleneksel modele göre, dinlenen cihaz gelen (inbound) bağlantı noktasını açar ve veri iletimi için bekler. Bu süreç işlerken, sürekli açık olan bağlantı noktası çok büyük bir risk oluşturur. Bu durumda ortaya çıkabilecek güvenlik sorunları içinde; zararlı yazılımların bulaşması, verinin değiştirilmesi ya da çalınması, DOS saldırıları bulunmaktadır.

- Uçtan uca şifreleme

Ulaşım katmanı güvenliği (TLS), uçtan uca güvenliği sağlamak amacıyla AES şifreleme ile eşleşen geniş alan ağı (WAN) üzerinden şifreli veri gönderen bir standart iletişim katmanıdır. TLS/SSL, şifreli veriyi cihazdan cihaza aktarırken veri akışına yüksek düzeyde koruma sağlar. Fakat veri iletimi ile ilgili güvenlik sağlansa da ITO cihazlarından üretilen veri hala güvenlik sorunu taşımaktadır. Bunun için bir şifreleme işlemine tabi tutulması gerekmektedir.

- Anahtar Tabanlı Erişim Kontrolü

AES ve TLS/SSL veri transferi esnasında şifreleme yapmak amacıyla kullanılmaktayken diğer önemli bir konu ise veri gönderme ve alma işlemleri kim/ne tarafından yapılacağıdır. Potansiyel olarak milyonlarca cihazın, doğru kanalı bulmaya

çalışırken, üyesi olmadıkları kanalları ve son cihazları dinlemeleri verimsiz ve güvenli olmayan bir durumdur. Bunun yerine ağ, bu görevi üstlenmelidir. Yayın/üye yaklaşımı ile cihazların belirli bir kanala erişimi için anahtarları ayırmada, anahtar tabanlı erişim kontrol yöntemi kullanılabilir. Bu yaklaşım, hangi anahtarların oluşturulacağı, hangi cihazın bu anahtarları alacağı ve hangi verilerin bu anahtarlara erişeceğini ince bir ayarla gerçekleştirmiş olur.

- Cihaz Durumu Görüntüleme

Nesnelerin internetinde, cihazların açıklık/kapalılık durumlarını (cihazın varlığını) görüntülemek kritiktir. Ev güvenlik sistemleri, yağ alan sensörleri, ev uygulamaları gibi cihazlar veri gönderme ya da almayı durdurdukları zaman, kendileri ya da izledikleri sistem hakkında bilgi edinilmeye ihtiyaç duyulur. Kapalı bir cihaz, bir müdahaleye maruz kaldığının ya da güç veya internet kesintisinin meydana geldiğinin habercisi olabilir (PubNub, 2015).

BÖLÜM 3

BLUETOOTH DÜŞÜK ENERJİ

Nesnelerin interneti uygulamalarında, kısıtlı kaynaklara sahip olduğundan, en kritik konu enerji tüketimidir. Bu amaçla kullanılabilecek birçok protokol mevcuttur. Uygulamaların amacına göre bu protokollerden en uygun olan seçilmelidir. Bluetooth düşük enerji (BLE) bu teknolojilerden en çok tercih edilenlerindendir. Çünkü BLE donanımları hem algılayıcılarla seri bağlantıyı kurmak için SPI, I2C, UART gibi seri bağlantı protokollerini sunarken, diğer taraftan klasik bluetooth' un düşük enerjili sürümü olan BLE teknolojisini sunmaktadır.

3.1. Bluetooth Düşük Enerji Teknolojisi ve Gelişimi

1990'lı yılların sonunda SIG tarafından çıkarılan Bluetooth Klasik (Bluetooth BR/EDR) kulaklıklarda, hoparlörlerde, araba setleri vb. çeşitli yerlerde kullanılmıştır. Bluetooth teknolojisi ile metin, fotoğraf, video gibi dosyaları elektronik cihazlar arasında iletmek mümkün olmuştur. Bağlantı tabanlı bir teknoloji olup, veri iletiminden önce ilgili cihazların bağlantı kurması gerekmektedir.

Çeşitli sürümleri bulunan Bluetooth teknolojisinin 2009 yılında çıkarılan Bluetooth v4.0 sürümü ile ise farklı bir Bluetooth teknolojisi ortaya konulmuştur. Bluetooth 4.0, "Bluetooth Düşük Enerji (BLE)" ya da "Akıllı Bluetooth" olarak adlandırılmaktadır. BLE ile asıl hedeflenen, düşük güç tüketimi ile uzun yıllar çalışan sistemler oluşturmaktır. Klasik Bluetooth ile BLE arasındaki benzerlik ve farklar Çizelge 3.1'de verilmiştir (Bluetooth SIG Inc., 2018).

Çizelge 3.1. Bluetooth Klasik-BLE Karşılaştırması

Özellik	**Bluetooth Düşük Enerji**	**Bluetooth Klasik**
İşleyiş	Aralıklı veri iletimi	Sürekli veri akışı
Frekans Bandı	2.4 GHz	2.4 GHz
Kanallar	2 MHz genişlikte 40 kanal (3 tanıtım/37 veri kanalı)	1 MHz genişlikte 79 kanal
Ağ Topolojisi	Noktadan-noktaya (P2P), Yayın, Karmaşık (Mesh)	Noktadan-noktaya (P2P)
Transfer Hızı	1-3Mbps	1Mbps
Güvenlik	128-bit AES	56-128 bit şifreleme
İletimde Gecikme	< 6ms	100+ ms
Mesafe	~10-100 metre	~10-100 metre

BLE, düşük güç tüketimi, yayın modunun oluşu, mobil platformlardaki desteği gibi temel özellikleriyle, özellikle nesnelerin interneti uygulamalarında daha avantajlıdır. Özellikle enerji depolama aletleri veya bataryalar gibi güç kaynaklarında çalışan aletler için güç verimliliği açısından çok faydalıdır. Bluetooth düşük enerjili bir alet, çoğunlukla uyku modunda çalışır; sadece bir bağlantı başlatıldığında uyku modundan çıkar. Böylelikle güç tüketimi düşük seviyede kalır çünkü asıl bağlantı süresi sadece birkaç milisaniyedir.

BLE teknolojisi dünya çapında bilgisayar ve iletişim cihazları tarafından geniş kapsamda her sektörde kullanılabilmektedir. Bluetooth 3.0 ile birlikte yüzlerce Mbps hız ile fiziksel veri transferi gerçekleştirildi. Fakat BLE tamamen farklı bir yöntem tercih etti; veri transfer hızını arttırmak yerine mümkün olduğu kadar düşük enerji tüketimli bir sistem oluşturmak. Veri iletim hızını arttıramayan bu teknoloji, daha az güç tüketimi ile sistemlerin çok daha uzun ömürlü olmasını hedefledi (Wei & Ye, 2016).

BLE, klasik bluetooth ile tasarlanabilecek en düşük güç tüketimli kablosuz teknolojiyi bir araya getiren yeni bir teknolojidir. Bluetooth markası ile anılsa da BLE farklı bir teknoloji, farklı bir tasarım amacı ve farklı market segmenti olarak değerlendirilmelidir. BLE tamamen farklı bir yön getirdi. Veri iletim hızını arttırmak

yerine, ultra düşük enerji tüketimi için optimizasyon gerçekleştirdi. Çok sayıda kablolu ve kablosuz iletişim teknolojilerinde bu hız arttırılmaya çalışılmıştır fakat BLE, Çizelge 3.2'de de görüldüğü gibi tam tersi bir yöntem uygulamıştır. Bu farklı yaklaşım klasik Bluetooth teknolojisinde olmayan, düşük güç tüketiminin gerekli olduğu düğme pillerle beslenen cihazlarda verimliliği arttırmıştır (Heydon, 2012).

Çizelge 3.2. Protokollerin Sürümlere Göre Hız Karşılaştırmaları

Versiyonlara Göre Hız Karşılaştırması

Modemler	Ethernet
V.21: 0.3kbps	802.3i: 10Mbps
V.22: 1.2kbps	802.3u: 100Mbps
V.32: 9.6kbps	802.3ab: 1000Mbps
V.34: 28.8kbps	802.3an: 10000Mbps
Wi-Fi	**Bluetooth**
802.11: 2Mbps	V1.1: 1Mbps
802.11b: 11Mbps	V2.0: 3Mbps
802.11g: 54Mbps	V3.0: 54Mbps
802.11n: 135Mbps	V4.0: 0.3Mbps

Klasik bluetooth 54 Mbps hız ile transfer yaparken BLE ile bu hız 0.3 Mbps'e kadar düşmüştür. Klasik Bluetooth 15 mA akım ile çalışırken BLE çok daha düşük güç ile işlemlerini gerçekleştirmektedir (Taskin, Taskin, & Cetintav, 2017).

3.2. BLE Cihaz Tipleri

BLE, iki tip cihaz tasarımı sunar: çift-mod ve tek-mod cihazlar. Bir çift-mod cihaz, hem klasik bluetooth hem de BLE teknolojilerini destekleyen bir Bluetooth cihazıdır. Tek-mod bir cihaz ise sadece BLE teknolojisini desteklemektedir.

Klasik Bluetooth desteği olduğundan, çift-mod cihazlar mevcut olan milyarlarca Bluetooth cihazı ile haberleşebilmektedir. Mevcut sistemlerin BLE desteği için güncellenmesi mümkün olmamaktadır. Bu cihazlar BLE teknolojisini de desteklediğinden yeni kurulması ya da yenilenmesi gereken cihazlardır. Kontrolörlerinde

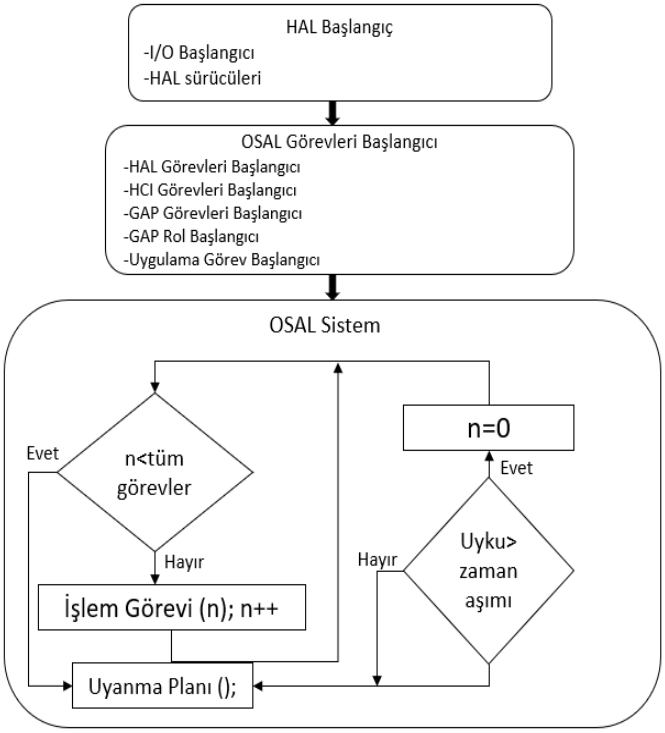
yeni donanım ve bellenim (firmware), istemci cihazlarında ise yeni yazılımlara ihtiyaç duymaktadır. Çoğu çift-mod cihazın kontrolörlerini yenilemek oldukça kolay bir işlemdir. Bu da, kullanıcılara, klasik Bluetooth teknolojisine sahip telefonların, bilgisayarların ve diğer cihazların kontrolörlerini çift-mod kontrolörler ile hızla değiştirilmesini sağlar. Çünkü tek-mod cihazlar klasik Bluetooth teknolojisini desteklemez, bir BLE tek-mod cihazı mevcut Bluetooth cihazları ile haberleşemez, fakat diğer tek-mod ve çift-mod cihazlar ile iletişim kurabilmektedir. Bu yeni tek-mod cihazlar, ultra-düşük güç tüketimi için yüksek optimizasyon sağlamaktadır. Tek-mod çalışan Bluetooth cihazlarının, BLE yüksek hızda veri aktarımı gerçekleştiremediği ve kulaklıklar ve stereo müzik için ses desteği olmadığından, çok fazla sayıda kullanılamayacaktır. Çizelge 3.3, hangi cihazların hangi cihazlar ile ne şekilde haberleşeceğini göstermektedir. Tek-mod cihazlar diğer tek-mod cihazlarla düşük enerji (LE) kullanarak haberleşir. Tek-mod cihazlar çift-mod cihazlarla aynı şekilde düşük enerji kullanarak haberleşmektedir. Çift-mod cihazlar diğer çift modlu cihazlar ile veya klasik cihazlar ile Bluetooth klasik (BR/EDR) ile haberleşmektedir. Tek modlu cihazlar, klasik cihazlar ile iletişim sağlayamazlar (Heydon, 2012).

Çizelge 3.3. Farklı Cihaz Tiplerinin Haberleşmesi

	Tek-Mod	**Çift-Mod**	**Klasik**
Tek-Mod	LE	LE	Yok
Çift-Mod	LE	Klasik	Klasik
Klasik	Yok	Klasik	Klasik

3.3. BLE Yazılım Akışı

BLE geliştirme aracı kullanarak yazılım geliştirme 5 temel aşamadan oluşmaktadır: OSAL, HAL, BLE protokol yığıtı, profiller ve uygulama. OSAL ve HAL tam kaynak olarak sağlanırken, BLE protokol yığıtı nesne kodu olarak sağlanmaktadır. Şekil 3.1'de yazılım akış diyagramı görülmektedir (Texas, 2010).



Şekil 3.1. BLE Yazılım Akış Diyagramı

3.3.1. İşletim Sistemi Soyutlama Katmanı

BLE protokol yığıtı, profiller ve tüm uygulamalar İşletim Sistemi Soyutlama Katmanı (OSAL) katmanı üzerine kurulmaktadır. OSAL, bir işletim sistemi değil yazılımsal işlemlerin yürütülmesini sağlayan bir kontrol döngüsüdür. Bu tip bir kontrole ihtiyaç duyan her yazılım katmanı için bir görev tanımlayıcı (ID) oluşturulmalı, bir görev başlatma rutini tanımlanmalı ve OSAL başlatma işlemine eklenmeli ve bir işlem rutini tanımlanmalıdır. İsteğe bağlı olarak bir mesaj işleme rutini de tanımlanabilir. Örnek olarak BLE yığıtın bazı katmanları OSAL görevlerindendir. Görev yönetimine ek olarak OSAL, mesaj geçişleri, bellek yönetimi ve zamanlayıcılar gibi çeşitli servisler sağlamaktadır. Tüm OSAL kodu tam bir kaynak sağlar. Hazırlanan uygulamalara göre OSAL hizmetlerinde artış gözlenebilmektedir.

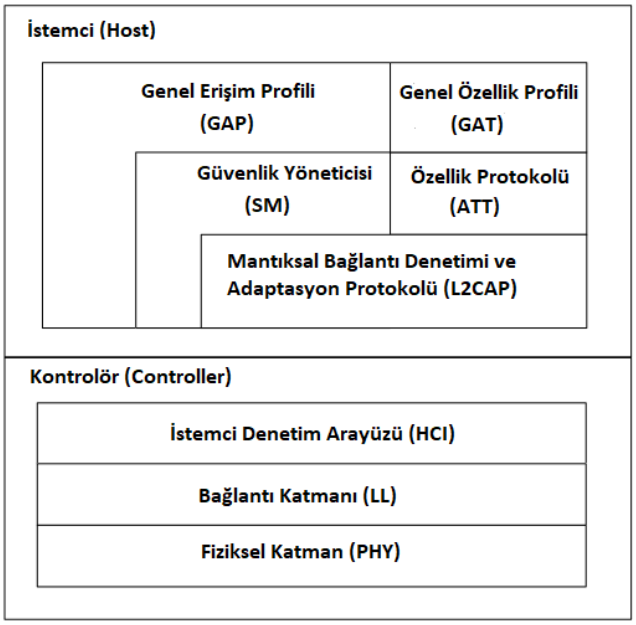
3.3.2. Donanım Soyutlama Katmanı

Bu katman, fiziksel donanım elemanlarıyla uygulama ve protokoller arasında soyutlama sağlayacak bir arayüzdür. Böylelikle yeni bir donanım geliştirme aşamasında,

protokollerde veya uygulama kaynak kodunda bir değişiklik yapmaya gerek kalmayacaktır. UART ile haberleşme arayüzü için yazılımı içermektedir. Bu kod, kullanıcının herhangi bir platforma uyacak şekilde değiştirmesine imkân tanımaktadır.

3.3.3. BLE Yığıtı

BLE haberleşme protokolü Şekil 3.2'de görüldüğü gibi katmanlar halinde tasarlanmıştır (Texas, 2010). Bu protokol yığıtı (stack) iki bölümden oluşmaktadır: kontrolör ve istemci. Bu kontrolör ve istemci bölümlerinin ayrı olması, iki bölümün ayrı uygulandığı standart Bluetooth BR/EDR cihazlara benzemektedir. Profil ve uygulamalar yığıtın GAP ve GATT katmanlarının en üstünde konumlanırlar.



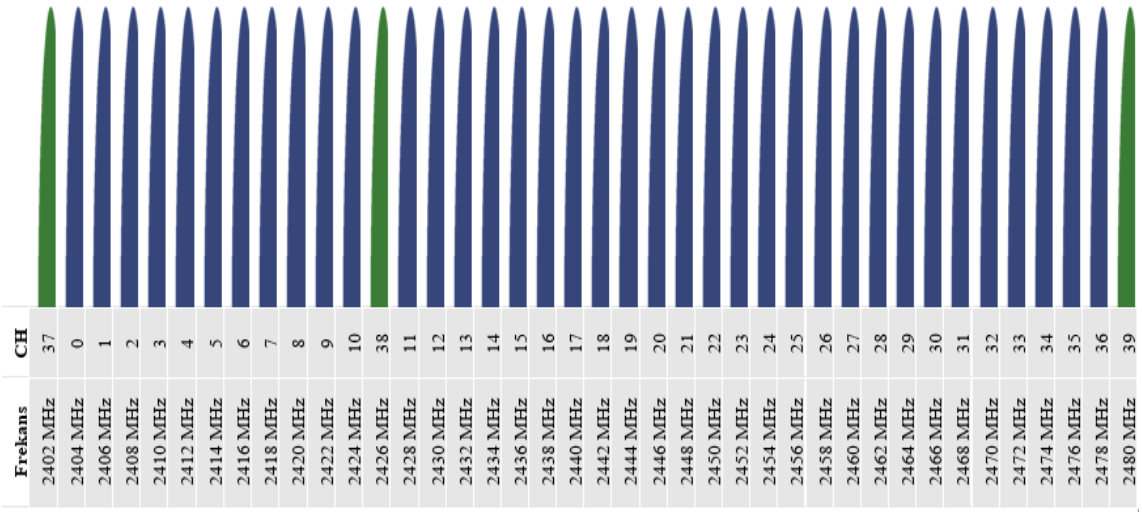
Şekil 3.2. BLE Yığıtı

Genel Erişim Profili (GAP), Bluetooth cihazlarının bağlantıları ile ilgili genel prosedürleri tanımlar. Bununla birlikte, bu profil, kullanıcı arayüzü seviyesinde erişilebilen parametreler için gerekli olan genel formatları da içermektedir.

Genel erişim profili, dört farklı modda işlem yapabilmektedir:

- Yayın (Broadcaster) Modu: Verilerin yayın yoluyla iletildiği moddur. Bir gönderici ve olası alıcılar bulunmaktadır. Tek yönlü ve bağlantısız bir iletişimdir.

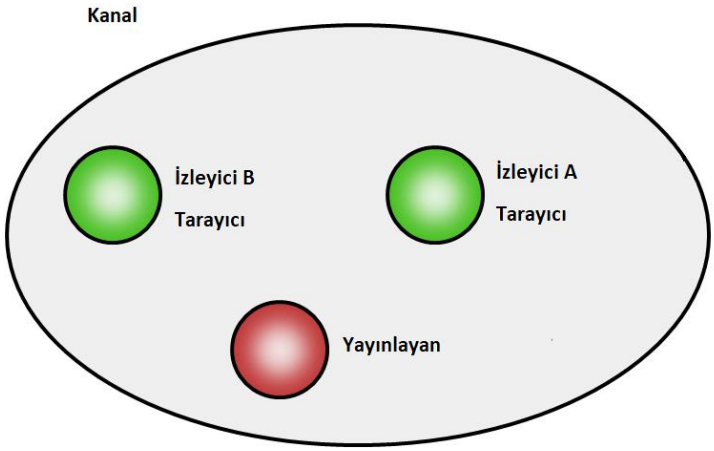
Bluetooth düşük enerji, 2.4 GHz frekansta işlem yapmaktadır ve bu frekans bandı 40 kanaldan oluşmaktadır. Bu kanalların 37 tanesi veri transferi yaparken kalan 3 kanal ise yayın için ayrılmaktadır. Şekil 3.3'te görüldüğü gibi 37, 38 ve 39. kanallar bu işlem için ayrılmaktadır.



Şekil 3.3. BLE Protokolü Frekans Aralığı

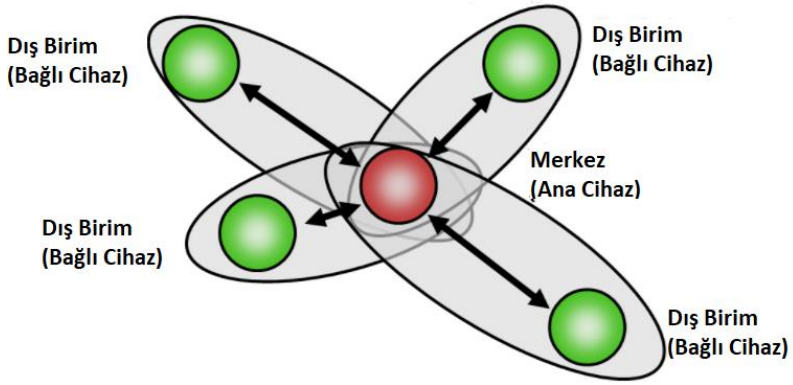
Broadcast modunda çalışan bir cihaz, verilerle birlikte yayın paketleri de iletir. Bu veri paketlerinin formatı Şekil 3.6'da verilmiştir.

- İzleyici (Observer) Modu: Yayınlanan verileri tarayıp bu verileri dinleyen moddur. LL katmanının "tarayıcı" rolünü kullanır. Bir alıcı ve olası bir gönderici mevcuttur.



Şekil 3.4. İzleyici (Observer) Modu

- Dış Birim (Peripheral) Modu: Çift yönlü ve bağlantı tabanlıdır. LL katmanındaki köle (slave) rolünü kullanmaktadır. Bu modda çalışan cihaz bir dış birimmiş gibi davranır. Bu dış birim hem gönderici hem de alıcı taraf olabilir.
- Merkez (Central) Mod: Merkez rolünde çalışan bir cihaz LL katmanında "ana (master)" rolündedir. Bağlantıyı kurmak ve yönetmekten sorumludur. Aynı anda birden fazla cihazla bağlantı kurabilmektedir ("Bluetooth® Low Energy GAP Roles - Developer Help," n.d.).



Şekil 3.5. Merkez (Central) Mod

Genel Özellik Profili (GATT), BLE bağlantısı üzerinden verilerin nasıl düzenleneceğini ve değiştirileceğini belirler. Bu profil ATT protokol katmanını kullanarak bir genel servis yazılım çerçevesi (framework) tanımlamaktadır. Bu yazılımlar prosedürleri ve servislerin formatlarını belirlemektedir. Belirtilen prosedürler, okuma, yazma, bildirim gönderme ve yayın (broadcast) ayarları gibi işlemleri içermektedir.

GATT Rolleri:

- GATT İstemci: Veri isteyen cihazdır. GATT sunucudan istek yapmak için komutları başlatır, sunucu tarafından gönderilen veriler için bildirimler alabilmektedir.
- GATT Sunucu: Veriler bu cihazda bulunmaktadır ve istemciden gelen komut ve istekleri kabul eder, geri cevap ve bildirim gönderir.

Not: Bir BLE yığıtı aynı anda iki rolün de çalışmasını desteklemektedir.

Not: GATT ve ATT katmanlarındaki “Nitelik (Attribute)”, GATT Sunucu, düzenlenen verileri “Nitelik (Attribute)” formunda saklar. Bir nitelik, etiketlenmiş, adreslenebilir bir veri parçası ya da bir metadır.

- Sunucuda saklanır.
- İstemci tarafından erişilebilir.

Nitelik (Attribute) Protokolü (ATT): BLE mantıksal taşıma kanalı üzerinde istemci/sunucu mimarisini tanımlamaktadır. Bu protokol, GATT sunucusu olan cihazın bir dizi nitelik yayması ve bunların istemci cihaz ile ilişkilendirilmesini sağlar. GATT sunucu tarafından yayılan bu nitelikler GATT istemci tarafından tespit edilebilir, okunabilir, yazılabilir.

Güvenlik Yönetim Protokolü (SMP): Cihazlar arasında eşleşme, kimlik doğrulama, şifreleme gibi işlemlerin yönetimi için gerekli prosedürlere sahip protokoldür.

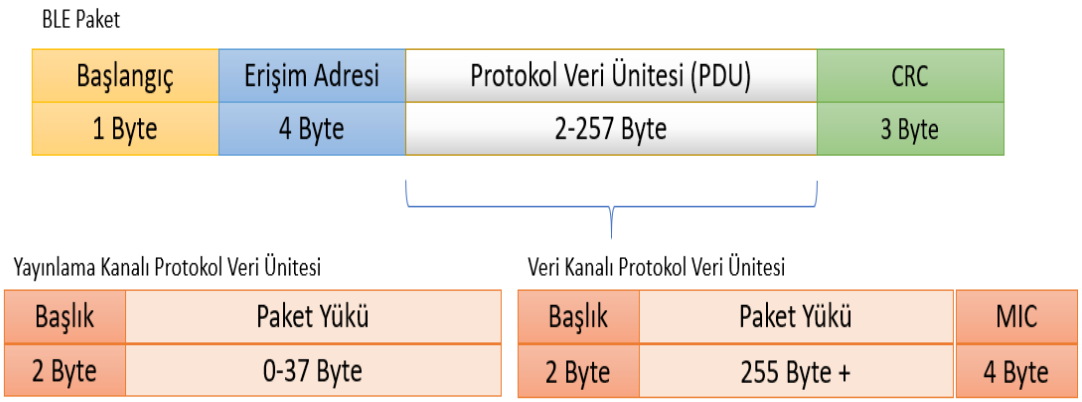
L2CAP (Mantıksal Bağlantı Denetim Uyum Protokolü), bağlantısız bir veri kanalı sağlar. Özellikleri:

- Kanal çoklama (Üç kanalı yönetmektedir. İki kanal ATT, SMP gibi daha üst katmanlar için ayrılmıştır. Diğer kanal kendi kullanımı içindir).
- BLE kontrolcünün yönetebileceği maksimum paket büyüklüğüne uygun paketlere ayırma ve tekrar birleştirme işlemini gerçekleştirir.
- BLE ile daha büyük veriler gönderebilmede etkilidir.

İstemci Denetim Arayüzü (HCI); GAP, L2CAP ve SMP gibi daha üst katmanlardan LL katmanına erişim için gereken arayüzü sağlar.

Bağlantı Katmanı (LL): Doğrudan PHY katmanı ile bağlantılıdır. Cihazlar arasındaki fiziksel BLE bağlantısını yöneten protokoldür. Yayın, tarama, başlatma ve bağlanma gibi tüm LL durumlarını desteklemektedir. Şifreleme, bağlantı güncelleme, kanal güncelleme gibi prosedürleri uygulamaktadır.

**Veri Paket Formatları: Bağlantı katmanı yayın kanalı paketi ve veri kanalı paketi olarak kullanılabilen tek bir paket formatı kullanır:



*MIC (Mesaj Bütünlük Kontrolü)

Şekil 3.6. BLE Paket Formatları

Başlangıç (Preamble), senkronizasyon ve göndericinin zamanlama tahmini için kullanılan 1 byte'lık veridir. Yayın paketi için her zaman 0xAA olacaktır. Erişim adresi de aynı şekilde, yayın paketi için sabittir. Protokol veri ünitesi (PDU) ise bir başlık ve paket yükü içermektedir. Başlık, paket tipini tanımlar. PDU tipi ise cihazın amacını belirlemektedir. Paketin son bölümü olan CRC, gönderilen paketin herhangi bir değişikliğe uğrayıp uğramadığını doğrulayan bir hata-bulucu koddur.

Fiziksel Katman (PHY): Bu katman, dijital sembollerin transferinden sorumlu olan analog haberleşme devreleri içermektedir. Protokol yığıtının en düşük katmanıdır ve LL için servisler sağlamaktadır ("Bluetooth® Low Energy GAP Roles - Developer Help," n.d.).

BÖLÜM 4

MATERYAL VE METOD

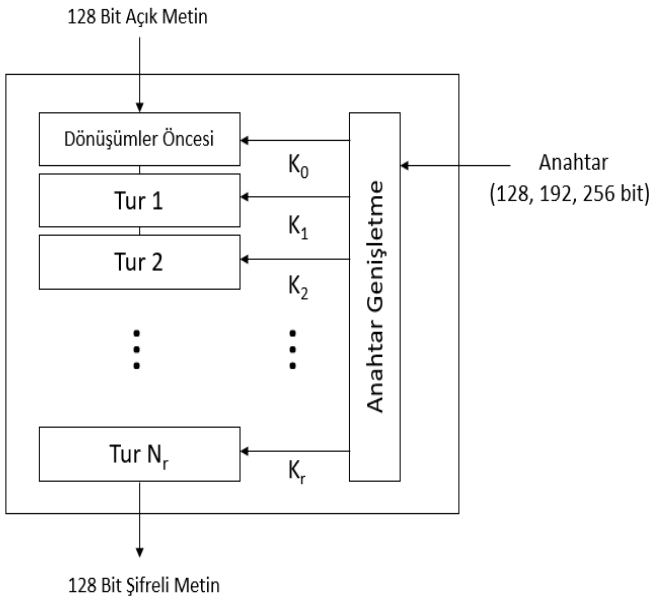
Bu bölümde, AES şifreleme metodunun yapısından bahsedilmekte ve anlatılacak olan test ortamı üzerinde geliştirilmiş olan BLE temelli IoT nodu üzerinde yazılımsal olarak koşan AES şifreleme metodu ile CC254x yonga tarafından sunulmakta olan donanımsal AES metodu uygulamaları sunulmaktadır.

4.1. İleri Şifreleme Standardı

Simetrik bir şifreleme algoritmasıdır. Artan hesaplama gücü ile birlikte güvenlik ataklarına karşı güçlü bir algoritmadır. AES (Rijndael) algoritması 128 bit veri bloklarını 128, 192, 256 bit anahtar seçenekleri ile şifreler. Anahtar uzunluğuna uygun döngü sayısında işlem yapmaktadır (Tutorialspoint).

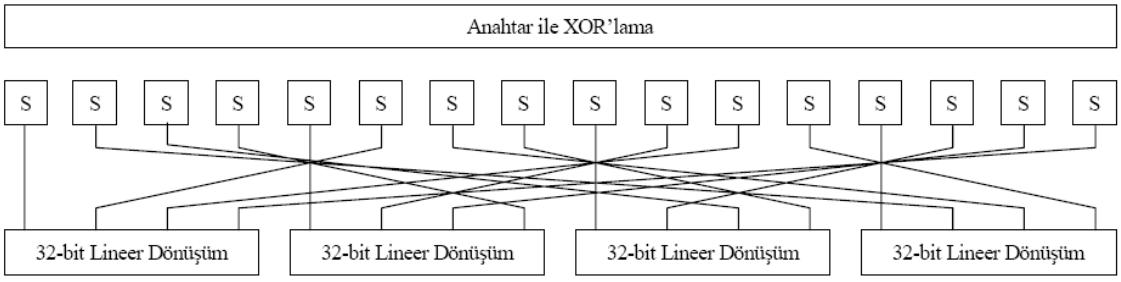
AES özelliklerinden bazıları şu şekildedir:

- 128-bit veri, 128/192/256 bit anahtar
- Tam teknik özellik ve tasarım detayları
- C ve Java gibi dillerle yazılım uygulaması (Tutorialspoint)



Şekil 4.1. AES Algoritmasının İşleyişi

Öncelikle 128 bit veri 4×4 byte matris haline getirilir. Her döngüde sırasıyla byte'ların değiştirilmesi, satırların kaydırılması, sütunların değiştirilmesi ve anahtar planlama evresinden gelen o döngü için belirlenen anahtar ile XOR'lama işlemleri yapılır. Byte'ların yer değiştirilmesinde 16 byte değerinin her biri 8 bit girişli ve 8 bit çıkışlı S kutusuna sokulur. S kutusu değerleri, Galois Alanı'nda (Galois Field - GF) GF(2^8), 8 bitlik polinom için ters alındıktan sonra lineer bir dönüşüme sokularak elde edilmiştir. Satırların ötelenmesi işleminde 4×4 byte matrisinde satırlar ötelenmiş ve sütunların değiştirilmesi işleminde herhangi bir sütun için o sütundaki değerler değiştirilmiştir. Döngünün son katmanında ise o döngüye ait anahtar ile XOR işlemi yapılmaktadır. Şekil 4.2, tek döngülük AES algoritmasını göstermektedir (Aslan v. d., 2012).

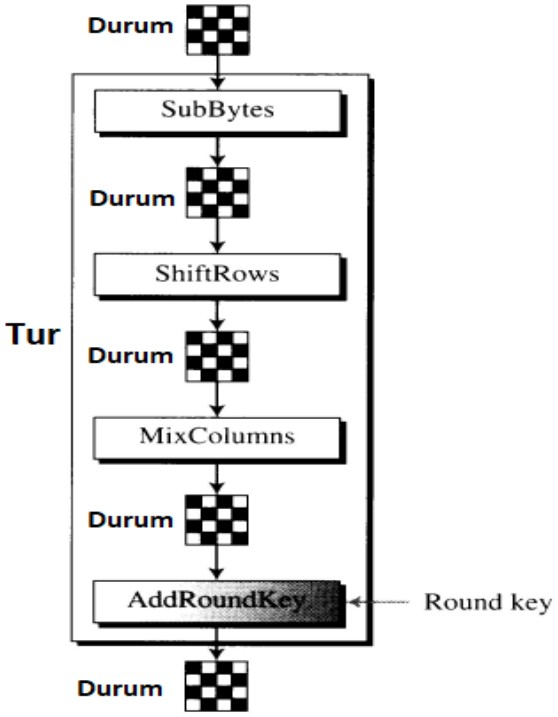


Şekil 4.2. AES Algoritmasının Bir Döngüsü

AES birden fazla döngü işlemi yapar. Her döngü de birden fazla aşamadan oluşmuştur. Şifrenin başında ve sonunda veri bloğu terimi kullanılır. Her aşamadan önce ve sonra veri bloğu durum (state) adını alır.

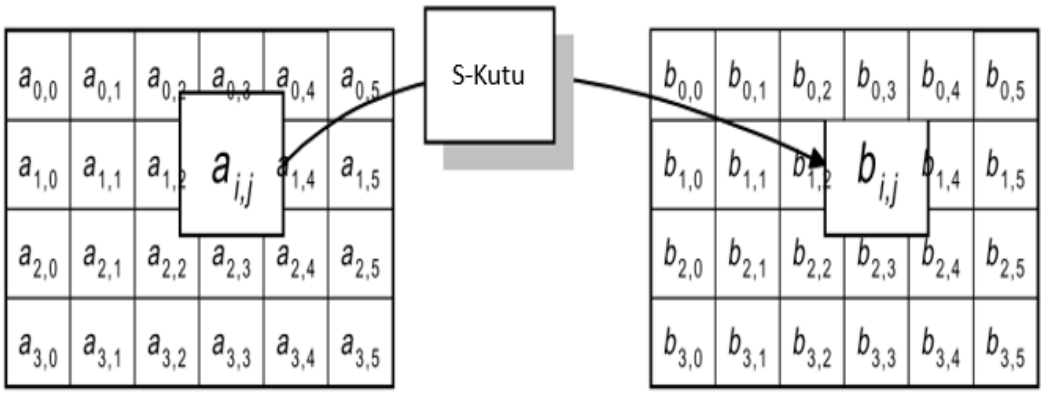
Her Döngünün Yapısı

1- Her döngü tersi alınabilir dönüşümler kullanır.
2- Döngülerde 4 dönüşüm kullanılır: SubBytes, ShiftRows, MixColumns ve AddRoundKey. Son döngü MixColumns dönüşümünü atlar.
3- Döngüler farklı anahtar materyali kullanır.
4- Farklı anahtar materyalleri, anahtar planlama işlemi ile bulunur.
5- Şifre çözme işleminde ise bu aşamaların tersi işlemleri yapılır: InvSubByte, InvShiftRows, InvMixColumns ve AddRounKey.



Şekil 4.3. AES Algoritmasının Akış Şeması

4.1.1. SubByte Dönüşümü



Şekil 4.4. SubByte Dönüşümü Temsili

SubByte dönüşümü, şifreye doğrusal olmama özelliğini katmak amacıyla her bir durum byte'ı üzerinde ayrı ayrı gerçekleştirilen ve her byte'a karşılık farklı bir byte getirilen işlemdir. Yer değiştirme çizelgesi (S-box) tersi alınabilen ve iki dönüşümün sonucunda oluşturulan bir çizelgedir. S-kutusu $GF(2^8) = Z_2(x)/(x^8 + x^4 + x^3 + x + 1)$ sonlu cisminde $x \to x^{-1}$ ters haritalama işleminden sonra aşağıdaki verilen doğrusal dönüşüm uygulanarak elde edilmiştir.

$$L_A(x) = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \quad (4.1)$$

Bu doğrusal dönüşümün eklenmesi S-kutusunun kriptografik özellikleri açısından herhangi bir iyileştirme yapmamakla beraber cebirsel ifadesini daha karmaşık hale getirmektedir. GF(2^8) de cebirsel ters alma işleminin cebirsel ifade x^{-1} veya x^{254} olarak

ifade edilirken doğrusal dönüşüm eklenerek bu ifade Denklem (4.2)'de görülen şekli almıştır.

$$S(x) = \text{"05"}x^{254} + \text{"09"}x^{253} + \text{"f9"}x^{251} + \text{"25"}x^{247} + \text{"f4"}x^{239} + \text{"01"}x^{223} + \text{"b5"}x^{191} + \text{"8f"}x^{127} + \text{"63"} \quad (4.2)$$

Diğer bir ifadeyle S-kutusunun cebirsel ifadesi geliştirilmiştir. Yine S-kutusunun sonlu bir cisimde ters alma işlemi ile tasarlanmasındaki amaç doğrusal kriptanaliz ve diferansiyel kriptanalize karşı dayanıklı bir S-kutusu seçme amaçlıdır. Çizelge 4.1'de AES S-kutusu çizelgesi verilmiştir.

Çizelge 4.1. AES S-Kutusu Dönüşüm Çizelgesi

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Şifre çözme işleminde kullanılacak olan InvSubBytes dönüşümü Çizelge 4.2'de verilmiştir.

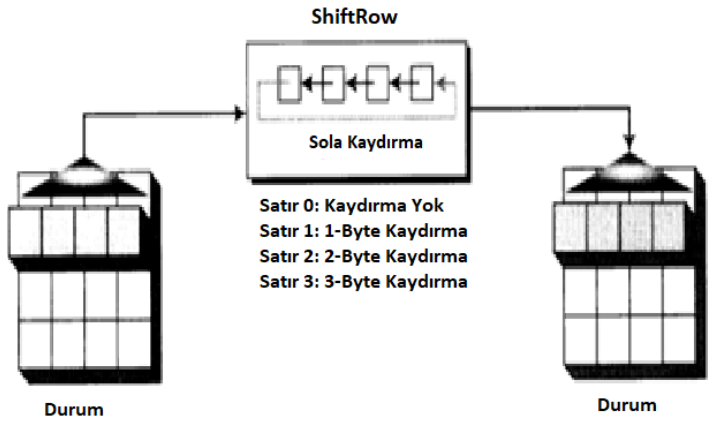
Ters SubByte işlemi için Çizelge 4.2'de verilen çizelge kullanılmalıdır. Bu çizelgedeki değerler ters Affine haritalama ve çarpım tersi yöntemleri ile elde edilmektedir (Daemen & Rijmen, 2003).

Çizelge 4.2. AES Ters S-Kutusu Dönüşüm Çizelgesi

		y														
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

4.1.2. ShiftRows Dönüşümü

Bu dönüşümde, durum matrisinin satırları farklı sayılarda dairesel olarak kaydırma işlemine tabi tutulurlar. 4 byte'lık bloklar için; 0. satır kaydırılmaz, 1. satır 1 kez, 2. satır 2 kez ve 3. satır 3 kez sola doğru dairesel kaydırılır.



Şekil 4.5. Satır Kaydırma İşlemi (Forouzan, 2008)

4.1.3. MixColumn Dönüşümü

32 bit üzerinde doğrusal dönüşümün gerçekleştirildiği dönüşümdür. Bir MDS matris tabanlı olan bu doğrusal dönüşüm aşağıdaki gibi gösterilebilir.

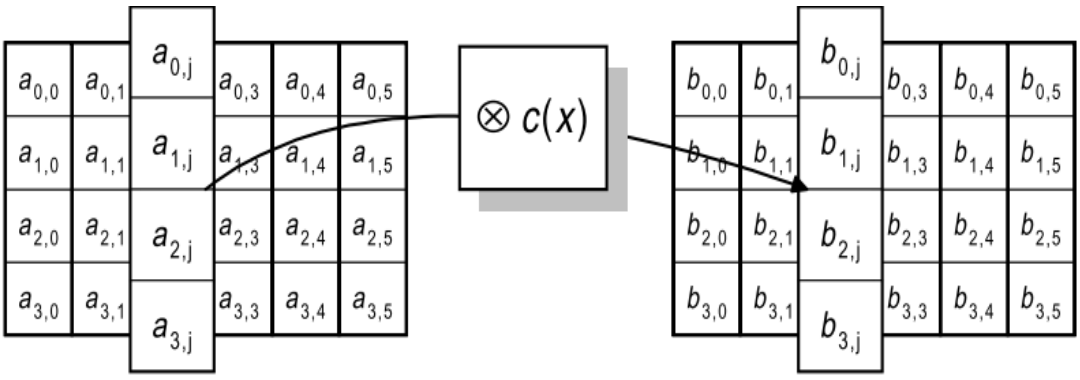
$$\begin{bmatrix} a & b & 1 & 1 \\ 1 & a & b & 1 \\ 1 & 1 & a & b \\ b & 1 & 1 & a \end{bmatrix}$$

(I) numaralı matriste a=x ve b=x+1 olarak seçilmiştir.

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} \quad (4.3)$$

Durum matrisinin sütunları GF(2^8)'de birer polinom olarak kabul edilir ve x^4+1 ile modu alınmış c(x) fonksiyonu ile çarpılır (Daemen & Rijmen, 2003; Heron, 2009).

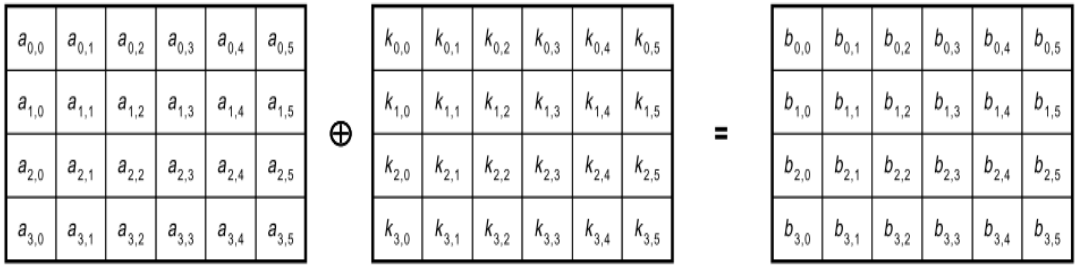
$c(x)= \text{'03'}\ x^3 + \text{'01'}x^2 + \text{'01'}\ x + \text{'02'}$



Şekil 4.6. MixColumn Dönüşümü

4.1.4. Anahtar Ekleme

Bulunulan turdaki geçerli anahtar, bit bazında basit bir XOR işleminin durum matrisine uygulanması işlemidir. Bu anahtar şifre anahtarından elde edilmektedir ve uzunluğu blok uzunluğuna eşittir.



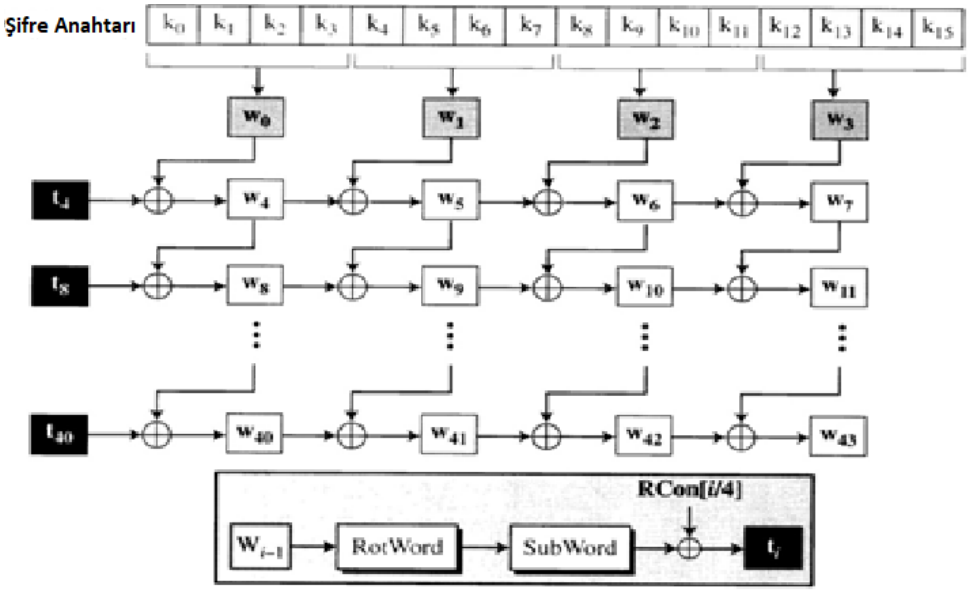
Şekil 4.7. Anahtar Ekleme Aşamasında XOR İşlemi

4.1.5. Anahtar Planlama

AES (Rijndael) algoritması 128 bit veri bloklarını 128, 192, 256 bit anahtar seçenekleri ile şifreleyen bir algoritmadır. Döngü sayısı anahtar genişliğine göre değişmektedir. 128 bit anahtar kullanılan AES şifrelemede döngü sayısı 10, 192 bitlik anahtar uzunluğunda 12 ve 256 bitlik anahtar uzunluğunda 14 olmaktadır. Her döngü için döngü anahtarı yaratmak için AES bir anahtar genişletme işlemi kullanır. Eğer döngü sayısı N_r ise anahtar genişletme işlemi N_r +1 128 bit döngü anahtarını tek bir 128 bit şifre anahtarından elde eder. İlk anahtar döngü başlamadan önce kullanılırken geri kalan döngü anahtarları her döngünün sonundaki son dönüşüm olarak kullanılır (Forouzan, 2008).

4.1.6. Anahtar Genişletme

AES şifrelemede anahtar genişletme algoritması girdi olarak 4 kelimelik (word) bir şifre alır ve 44 kelimelik doğrusal bir dizi üretir. Her turda bu kelimelerden 4 tanesi kullanılır. Her kelime 32 bit içerdiğinden her alt anahtar 128 bit boyutundadır. Şekil 4.8'de anahtar genişletme işlemi adım adım gösterilmiştir (Forouzan, 2008).



Şekil 4.8. Anahtar Genişletme

İlk 4 kelime (word) (w_0, w_1, w_2, w_3) şifre anahtarından elde edilir. İlk 4 byte (k_0 dan k_3'e) w_0, ikinci 4 byte (k_4'ten k_7'ye) w_1 ve bu şekilde diğer kelimeler w_2 ve w_3'te şifre anahtarının kelimeler şeklinde yan yana konması ile elde edilir.

Diğer kelimeler (w_i i=4 den 43'e kadar) aşağıdaki şekilde elde edilir:

a- Eğer i (mod 4) ≠ 0 ise $w_i = w_{i-1} \oplus w_{i-4}$

b- Eğer i (mod 4) = 0 ise $w_i = t \oplus w_{i-4}$

(Forouzan, 2008).

$$t = SubWord\ (RotWord\ (w_{i-1})) \oplus RCon_{i/4}. \quad (4.4)$$

Anahtar geliştirme algoritmasının 3 temel aşaması bulunmaktadır:

- RotWord: Kelime üzerinde 1 byte sola dairesel kaydırma işlemi gerçekleştirilir. [b_0, b_1, b_2, b_3] olan kelime [b_1, b_2, b_3, b_0] olarak düzenlenmiş olur.
- SubWord: S-Kutu (S-Box)'ları olarak tanımlanmış olan çizelgeye göre kelimelerin her byte'ına karşılık gelen byte ile değiştirilmesi işlemidir.
- Son adımda, RotWord ve SubWord işlemlerinden geçen sonuçlar, döngü(tur) sabiti olan Rcon[j] ile Xor işlemine tabi tutulur.

Her döngü sabiti, RCon, 4 byte değerinde ve en sağdaki 3 byte'ı 0 olan bir değerdir. Çizelge 4.3, AES-128 için (10 döngü) değerleri göstermektedir.

Çizelge 4.3. AES-128 Bit için Rcon Değerleri

Tur	**Sabit (Rcon)**	**Tur**	**Sabit (Rcon)**
1	$(\underline{01}\ 00\ 00\ 00)_{16}$	**6**	$(\underline{20}\ 00\ 00\ 00)_{16}$
2	$(\underline{02}\ 00\ 00\ 00)_{16}$	**7**	$(\underline{40}\ 00\ 00\ 00)_{16}$
3	$(\underline{04}\ 00\ 00\ 00)_{16}$	**8**	$(\underline{80}\ 00\ 00\ 00)_{16}$
4	$(\underline{08}\ 00\ 00\ 00)_{16}$	**9**	$(\underline{1B}\ 00\ 00\ 00)_{16}$
5	$(\underline{10}\ 00\ 00\ 00)_{16}$	**10**	$(\underline{36}\ 00\ 00\ 00)_{16}$

Anahtar genişletme rutini word değerlerini hesaplarken ya yukarıdaki çizelgeyi ya da en soldaki byte'ı dinamik olarak hesaplamak GF(2^8) cismini kullanır. En soldaki byte, RC_i, x^{i-1} dir ve i değeri döngü sayısıdır. AES, $x^8+x^4+x^3+x+1$ indirgenemez polinomunu kullanır.

1. RotWord bir kelime üzerinde sola kaydırma işlemi yapar. Bunun anlamı girdi verisi olan [b0, b1, b2, b3] kelimesi [b1, b2, b3, b0] haline dönüşür.
2. SubWord girdi kelimesinin her bir byte'ı üzerinde daha önceden tanımlanmış S-kutu formuna göre dönüşüm yapar.
3. İlk iki adım sonucunda Rcon[j] döngü sabiti ile XOR işlemi gerçekleştirilir (Daemen & Rijmen, 2003).

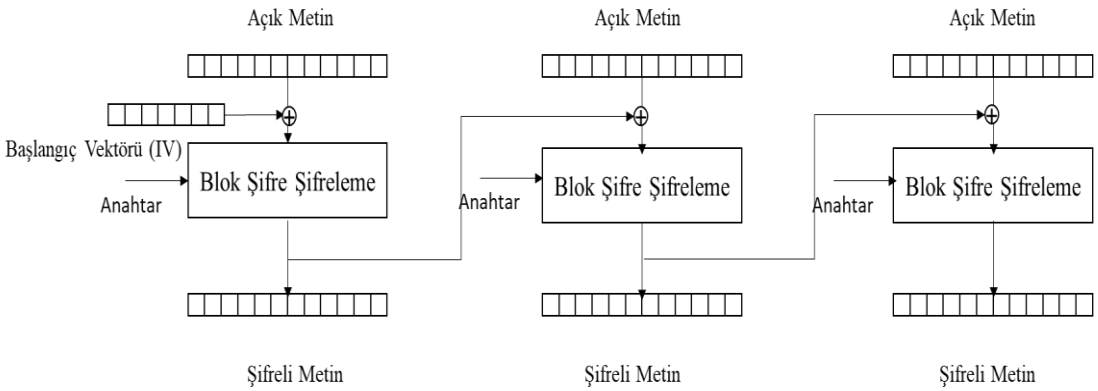
Şekil 4.9'da AES şifreleme algoritmasının ana fonksiyonu verilmiştir.

AES blok şifresinin beş farklı modu vardır: ECB, CBC, CFB, OFB ve CTR. Bu çalışmada CBC (Şifre Blok Zincir) modu kullanılmaktadır.

CBC mod, açık metin bloklarının kendisinden önceki şifreli metin bloklarıyla XOR işlemi ile birleştirilmesi işlemidir. İlk açık metin bloğu ile birleştirilmek üzere bir IV değerine ihtiyaç vardır. IV değerinin gizli olması şart değildir fakat tahmin edilemez olmalıdır. Şekil 4.10 CBC modu için gereken işlemi özetlemektedir (Dworkin, 2007).

```
static void Encryption(state_t* state, uint8_t* RoundKey)
{
  uint8_t round = 0;
  AddRoundKeyF(0, state, RoundKey);
  for (round = 1; round < Nr; ++round)
  {
    SubBytesF(state);
    ShiftRowsF(state);
    MixColumnsF(state);
    AddRoundKeyF(round, state, RoundKey);
  }
   SubBytesF(state);
  ShiftRowsF(state);
  AddRoundKeyF(Nr, state, RoundKey);
}
```

Şekil 4.9. AES Şifresi Ana Fonksiyonu



Şekil 4.10. AES CBC Modu

4.2. Uygulama Ortamı

AES şifreleme ve şifre çözme işlemlerinin gerçekleştiği BLE tabanlı geliştirme aracı olarak CC2540 kullanılmıştır. CC2540, BLE uygulamalarında kullanılan düşük maliyetli, düşük güç tüketimli bir modüldür. Bu modül, çok az maliyetlerde güçlü BLE

nodları gerçekleştirebilmektedir. CC2540 bir RF alıcı-vericiyi 8051 MCU, flash bellek, 8 KB RAM ve diğer destekleyici dış birimlerle birlikte çalıştırır. CC2540 düşük güç tüketimi gerektiren uygulamalar için uygun bir modüldür.

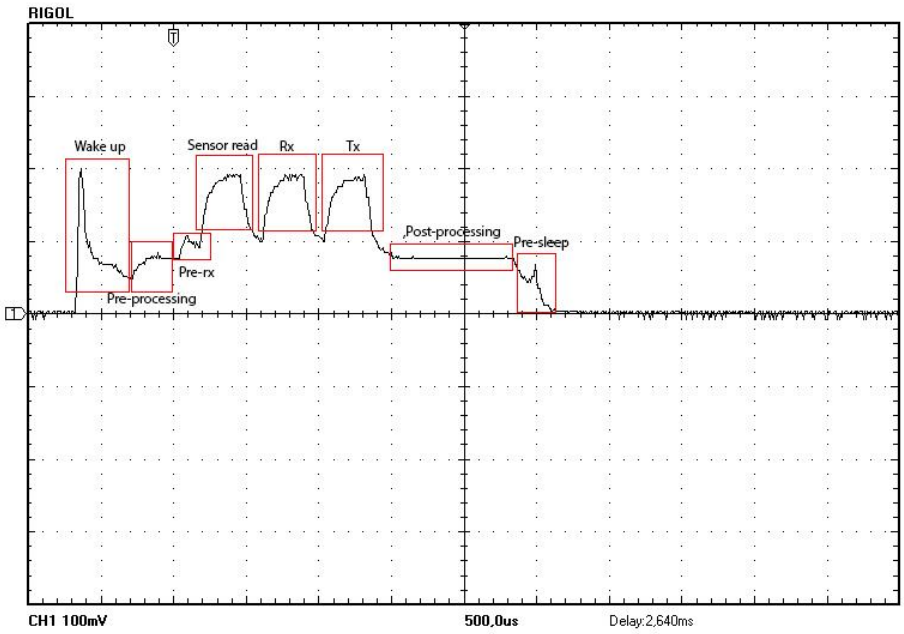
İki farklı CC2540 versiyonu bulunmaktadır: CC2540F128/F256, sırasıyla 128 KB ve 256 KB flash bellek. CC2540 ile ilgili özellikler Çizelge 4.4'te verilmiştir (Texas, 2013).

Çizelge 4.4. CC2540 Teknik Özellikleri

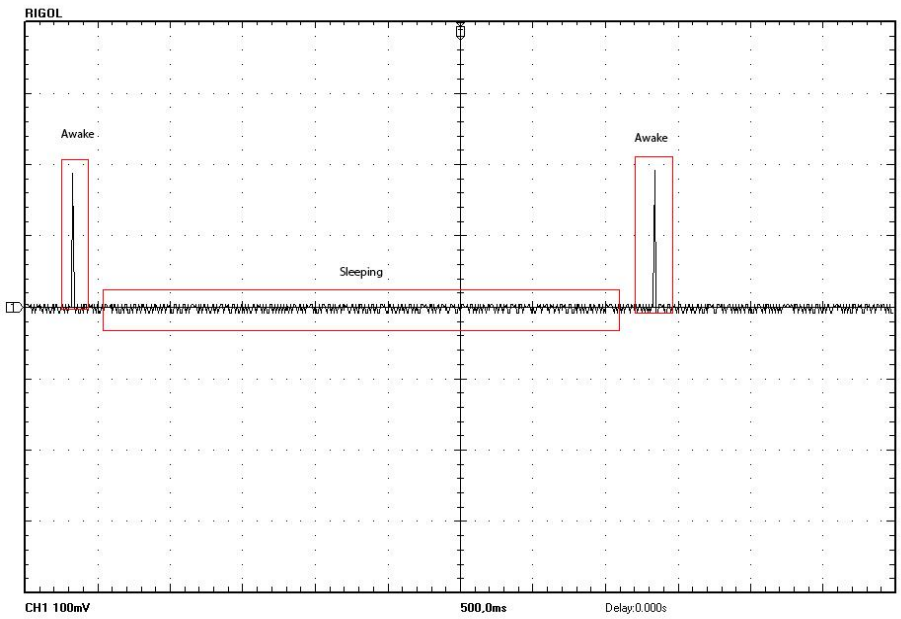
Boyut	• 6 mm x 6 mm
Düşük Enerji	• Aktif mod RX: 19.6 mA • Aktif mod TX: 24 mA • Güç modu 1 (3-µs uyanma): 235 µA • Güç modu 2 (Uyku modu): 0.9 µA • Güç modu 3 (Harici Kesmeler): 0.4 µA
Mikrodenetçi	• Yüksek-performans ve düşük enerjili 8051 mikrodenetçi çekirdeği, • 128 KB ya da 256 KB sistem içi programlanabilen (In-System-Programmable) Flash • 8-Kb SRAM
Dış Birimler	• 8 Kanallı 12-bit ADC • Entegre yüksek-performanslı op-amp • Genel-Amaçlı 8 ve 16 bitlik zamanlayıcılar • 21 Adet genel-amaçlı I/O pinleri (19.6 mA,2x20 mA) • 32 kHz uyku zamanlayıcı • Seri protokollerle desteklenen iki USART • USB arayüzü • IR jenerasyon devre • 5 Kanal DMA • AES Güvenlik yardımcı işlemci • Batarya ekranı ve ısı sensörü • Özgün 48-bit IEEE adresi

Yapılan bir çalışmada, CC2540 ile bir sensör uygulaması yapılarak sistemin çok düşük maliyetli ve düşük güç tüketimli bir uygulama olduğu kanıtlanmıştır. Ayrıca sistemdeki bataryanın yaklaşık olarak 2 yıl kesintisiz şekilde kullanılabileceği sonucuna

ulaşılmıştır (Taskin, Taskin, & Cetintav, 2017). Çalışma sonucunda zamana bağlı güç tüketimi sonuçları BLE durumlarına göre Şekil 4.11 ve şekil 4.12'de verilmektedir.



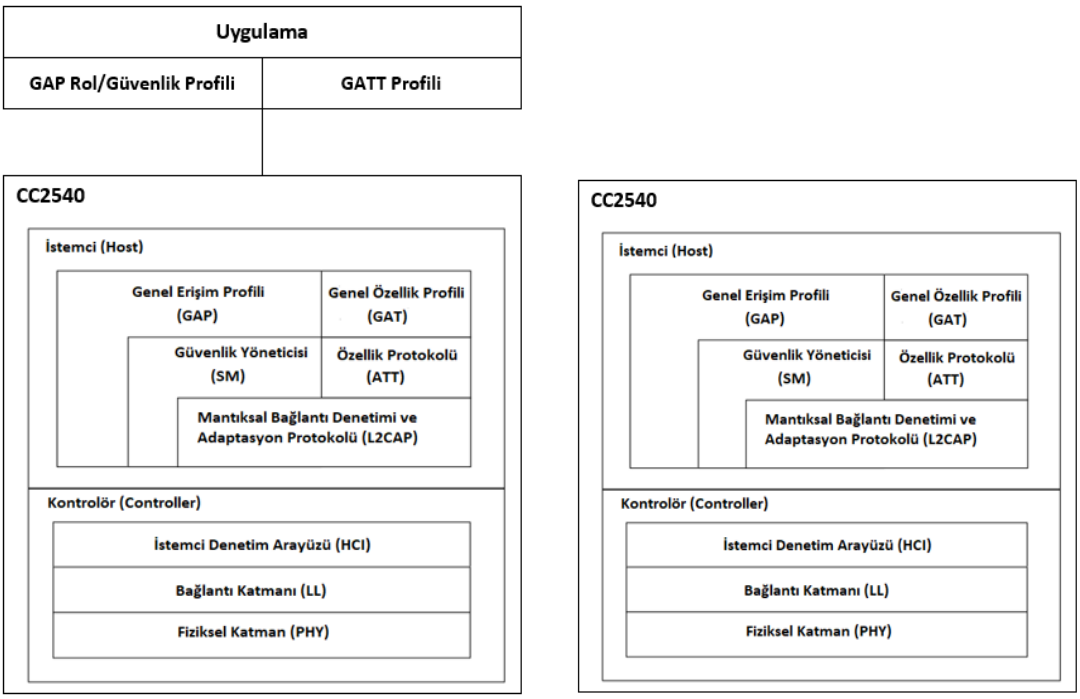
Şekil 4.11. BLE Durumlarına Göre Harcanan Güç



Şekil 4.12. İşlem Bitiminde Harcanan Güç

Bu çalışmada vurgulanan asıl konu CC2540 BLE modülünün konfigürasyon seçimidir. Burada karşımıza iki tip konfigürasyon çıkmaktadır: ağ işlemci konfigürasyonu ve tek çip konfigürasyonu.

a) Ağ İşlemci Konfigürasyonu: Bluetooth stack katmanları ve uygulama katmanının ayrı tasarlandığı konfigürasyondur. Bu ayrı katmanlar UART ara yüzü ile bağlantı kurabilirler. Eğer BLE modülü sadece BLE özelliği için kullanılacak ve ek bir uygulama donanımı kullanılacaksa bu konfigürasyon tercih edilebilir.
b) Tek Çip Konfigürasyonu: Bu tasarımda ise uygulama ve BLE stack katmanları birliktedir ve CC2540 modülü üzerinde birleştirilmiştir. Bu tasarım en basit ve en yaygın kullanılan tasarımdır. Maliyet ve enerji tüketimi açısından en verimli tasarımdır (Developer, 2015).

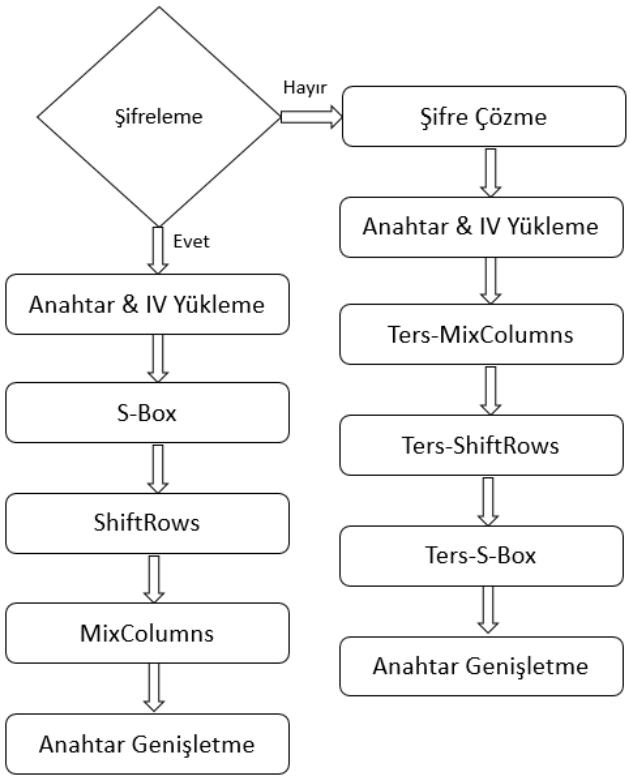


a. Ağ İşlemci Konfigürasyonu b. Tek Çip Konfigürasyonu

Şekil 4.13. BLE Cihaz Konfigürasyonları

4.3. AES Şifreleme Metodunun Yazılımsal Uygulaması

AES algoritmasının CC2540 geliştirme kiti üzerinde yazılımsal olarak uygulanması daha uzun ve karmaşık bir işlemdir. Algoritmaya uygun olarak tüm fonksiyonların, belirlenen programlama dilinde yazılıp derlendikten sonra kite aktarılması gerekmektedir. Şekil 4.14'te AES algoritması için yazılım akış şeması verilmiştir.



Şekil 4.14. AES Yazılım Akış Şeması

AES algoritması için yazılan temel fonksiyonlar:

- Anahtar Genişletme
- Döngüye Anahtar Ekleme
- SubBytes
- ShiftRows
- MixColumns

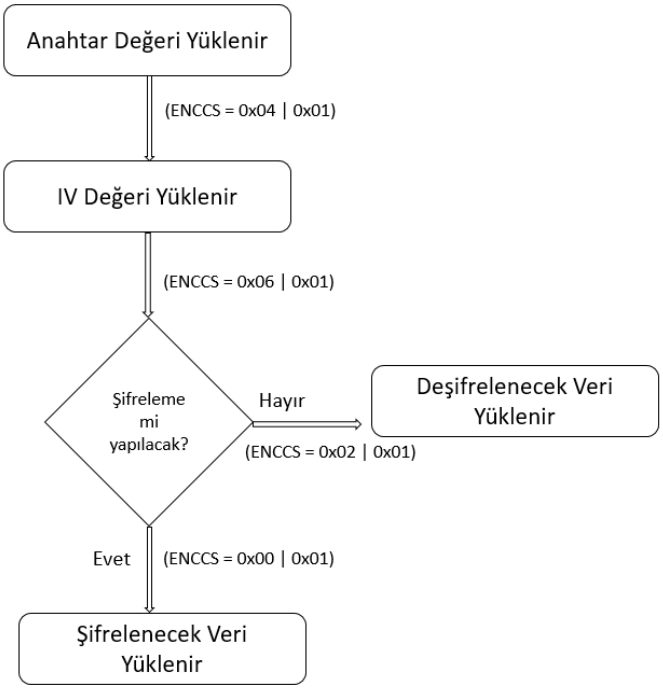
Yazılım uygulaması için yazılan temel fonksiyon Şekil 4.15'te verilmiştir.

```
void AES_ encrypt(uint8_t* out, uint8_t* in, uint32_t length, const uint8_t* key, const
uint8_t* iv)
{
  int i;
  uint8_t extra = length % BLOCKLEN;
  //Anahtar 0 ise anahtar genisletme fonksiyonunu atla
  if (0 != key)
  {
    Key = key;
    KeyExpansionF();
  }
  if (iv != 0)
  {
    Iv = (uint8_t*)iv;
  }
  //
  for (i = 0; i < length; i += BLOCKLEN)
  {
    memcpy(out, in, BLOCKLEN);
    XorWithIv(out);
    state = (state_t*) out;
    //Asil sifreleme fonksiyonuna git
    Encryption();
    //Seri konsola yazdir
    UartSendString((uint8*)state,sizeof(*state));
    Iv = out;
    in += BLOCKLEN;
    out += BLOCKLEN;
  }
  if (extra)
  {
    memcpy(out, in, extra);
    memset((out + extra), 0, (BLOCKLEN - extra));
    XorWithIv(out);
    state = (state_t*) out;
    Encryption();
  }
}
```

Şekil 4.15. AES Algoritması Yazılım Uygulaması Ana Fonksiyonu

4.4. AES Şifreleme Metodunun Donanımsal Uygulaması

Bir mesaj AES algoritması ile şifrelenmek istendiğinde gerekli olan temel aşamalar Şekil 4.16'da verilmiştir.



Şekil 4.16. AES Donanım Uygulaması Akış Şeması

AES algoritması 128 bitlik bloklar üzerinde çalışmaktadır. Verinin bir bloğu işlemciye yüklenir, şifreleme işlemi gerçekleşir ve sonuç, bir sonraki veri bloğu yüklenmeden hemen önce okunmalıdır.

Anahtar ve Başlangıç Vektörü

Anahtar ve başlangıç vektörü (IV) yüklenmeden önce bu işlem için uygun komutun işlemciye yüklenmesi gerekmektedir. IV yüklenirken doğru modu ayarlamak da önemlidir. Anahtar yükleme veya IV yükleme işlemi çalışabilecek herhangi bir işlemi iptal eder. Anahtar, yeni anahtar yüklenene kadar geçerli olarak kalır. IV ise her mesajdan önce (bloktan önce değil) yüklenmelidir.

Verinin Yüklenmesi

AES, 128 bitlik veri blokları üzerinde işlem gerçekleştirir. Eğer son blok 128 bitten kısa ise sonuna 128 bit uzunluğunu tamamlayacak kadar sıfır eklenir.

AES şifreleme algoritmasını donanımsal olarak CC2540 modülüne uygulamak için, modülde var olan yazaçlar kullanılmaktadır. Bu yazaçlar için gerekli değerler verildiğinde sadece küçük bir kod parçacığı ile ve hiçbir algoritma modülünü ayrı ayrı tanımlamaya gerek kalmadan şifreleme işlemi gerçekleştirilmektedir.

İşlemcinin yardımcı işlemci ile iletişim kurabilmesi için kullanılan üç temel yazaç bulunmaktadır (Texas, 2014):

- ENCCS: Şifreleme Kontrol ve Durum Yazacı
- ENCDI: Şifreleme Girdi Yazacı
- ENCDO: Şifreleme Çıktı Yazacı

ENCCS yazacı için Çizelge 4.5'te gösterilen bitlerden ST, başlangıç işlem ID'sini belirtir ve CMD işlem başlatma komutudur. ST=1 yapıldığında CMD ve RDY bitleri dört değer alabilir. Bu değerler sırasıyla "veri şifrele", "şifre çöz", "anahtar yükle" ve "IV yükle" karşılık gelir (Tongrang v. d., 2016).

Durum yazacına okuma veya yazma işlemi, doğrudan CPU tarafından yapılırken giriş/çıkış yazaçlarına erişim Doğrudan Bellek Erişim (DMA) yöntemi kullanılarak olmalıdır. DMA, AES yardımcı işlemcisiyle uygulandığında, biri giriş verisi diğeri çıkış verisi için olmak üzere iki DMA kanalı kullanılmalıdır. DMA kanalları, ENCCS yazacına "başla" komutu yazılmadan önce başlatılmalıdır. "Başla" komutu bir DMA tetikleyicisi üretir ve iletim başlar. Her blok işlemi gerçekleştikten sonra bir kesme işlemi üretilir. Kesme işlemi ENCCS yazacına yeni bir başlatma komutu yayınlamak için kullanılır.

AES algoritması donanımsal olarak CC2540 geliştirme kiti üzerinde uygulamak istenildiğinde belirtilen yazaçlar kullanılır. Böylelikle daha kısa bir programla ve daha kısa zamanda istenilen şifreleme işlemi gerçekleşmiş olacaktır. AES algoritmasının donanımsal olarak IAR IDE üzerinde C programlama dili ile yazılan kodu Şekil 4.17'deki gibidir (TI, Encryption, & Decryption, 2008).

```
void AesHardwareImplementation()
{
  unsigned short i, j;
  ENCCS = 0x04 | 0x01; //Anahtar Yükleme
  for (i = 0; i < SIZE_OF_AES_BLOCK; i++)
  {
    ENCDI = key[i];
  }
  while(!(ENCCS & 0x08));
  ENCCS = 0x06 | 0x01; //IV Yükleme
  for (i = 0; i < SIZE_OF_AES_BLOCK; i++)
  {
    ENCDI = iv_nonce[i];
  }
  while(!(ENCCS & 0x08));
  for (j = 0; j < SIZE_OF_ENC_PACKET/SIZE_OF_AES_BLOCK; j++)
  {
    ENCCS = 0x00 | 0x01; //Sifreleme Modu
     //Sifre çözme icin ENCCS=0x02|0x01
    for (i = 0; i < SIZE_OF_AES_BLOCK; i++)
    {
      ENCDI = aes_buffer_1[i+(j*SIZE_OF_AES_BLOCK)];
    }
    for (i = 0; i < SIZE_OF_AES_BLOCK; i++)
    {
      aes_buffer_2[i+(j*SIZE_OF_AES_BLOCK)] = ENCDO;
    }
    while(!(ENCCS & 0x08));
  }
  asm("NOP");
}
```

Şekil 4.17. AES Algoritması Donanım Uygulaması Ana Fonksiyonu

Kullanılan yazaçlar, bitlerine uygun değerler alarak şifreleme işlemini donanımsal olarak gerçekleştirmektedirler. Çizelge 4.5, çizelge 4.6 ve çizelge 4.7, yazaçlar için gerekli değerleri göstermektedir (Texas, 2014).

Çizelge 4.5. ENCCS Yazacı İçin Bit Dağılımı

Bit	İsim	Reset	R/W	Tanım
7	-	0	R0	Rezerve, her zaman 0 olarak okunur.
6:4	Mod[2:0]	000	R/W	Şifreleme/Şifre Çözme Modu 000: CBC 001: CFB 010: OFB 011: CTR 100: ECB 101: CBC MAC 110: Rezerve 111: Rezerve
3	RDY	1	R	Şifreleme ya da Şifre Çözme Hazır Durum 0: Şifreleme veya şifre çözme yürütülüyor. 1: Şifreleme veya şifre çözme tamamlandı.
2:1	CMD	0	R/W	ST=1 olduğunda yapılacak komutlar 00: Şifrele 01: Şifre Çöz 10: Anahtar Yükle 11: IV Yükle
0	ST	0	R/W1 H0	CMD ile İşlem başlatma komutu verilir. Her komut veya 128 bitlik veri blokları için verilmiş olmalı. Donanım tarafından sıfırlanır.

Çizelge 4.6. ENCDI Yazacı İçin Bit Dağılımı

Bit	İsim	Reset	R/W	Tanım
7:0	DIN [7:0]	0x00	R/W	Girdi verisinin şifrelenmesi

Çizelge 4.7. ENCDO Yazacı İçin Bit Dağılımı

Bit	İsim	Reset	R/W	Tanım
7:0	DOUT [7:0]	0x00	R/W	Çıktı verisinin şifrelenmesi

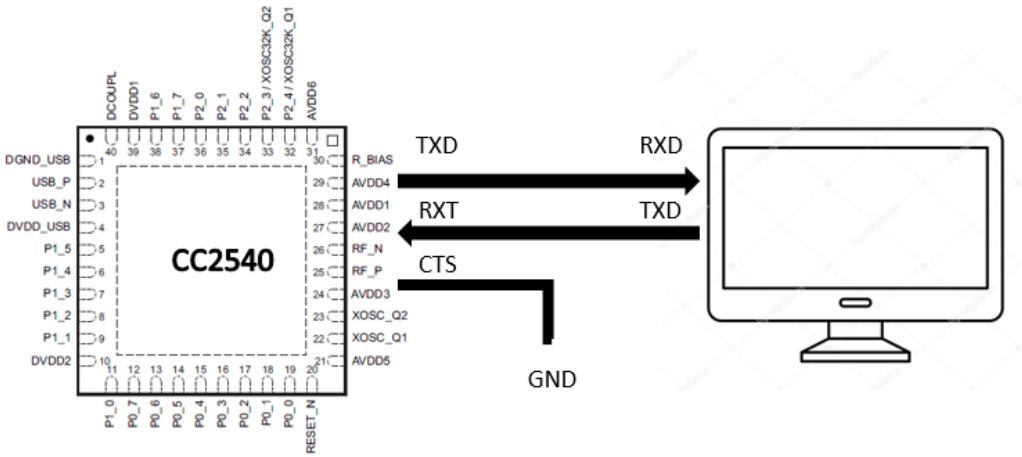
BÖLÜM 5

SONUÇLAR VE TARTIŞMA

128 bitlik AES algoritması Bölüm 4'te anlatıldığı gibi uygulanmıştır. Bu uygulamalarda yazılımsal şifreleme ve şifre çözme işlemi yapıldığında sonuçlar konsol ekranlarında görülmektedir. Seri konsol çıktıları ile birlikte güç ve zaman tüketimleri için yapılan analizlerin osiloskop görüntüleri bu bölümde verilmektedir. Ayrıca işlemlerin yapıldığı cihazın kesintisiz kullanım süresi bu bölümde hesaplanmaktadır.

5.1. Şifreleme ve Şifre Çözme Sonuçları

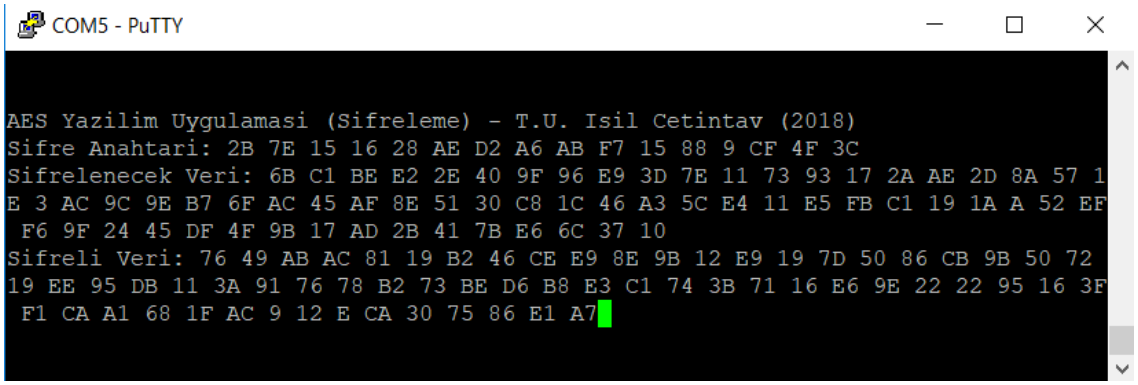
CC2540 geliştirme aracı geliştirme ortamına seri bağlantı yolu ile bağlanır. Donanımsal ve yazılımsal olarak yazılan AES kodları IAR Embedded Workbench üzerinde derlendikten sonra CC2540'a aktarılır. Şekil 5.1'de seri bağlantının blok şeması gösterilmiştir. Cihaz ile bağlantı RS232 üzerinden yapılmaktadır.



Şekil 5.1. CC2540-RS232 Bağlantısı

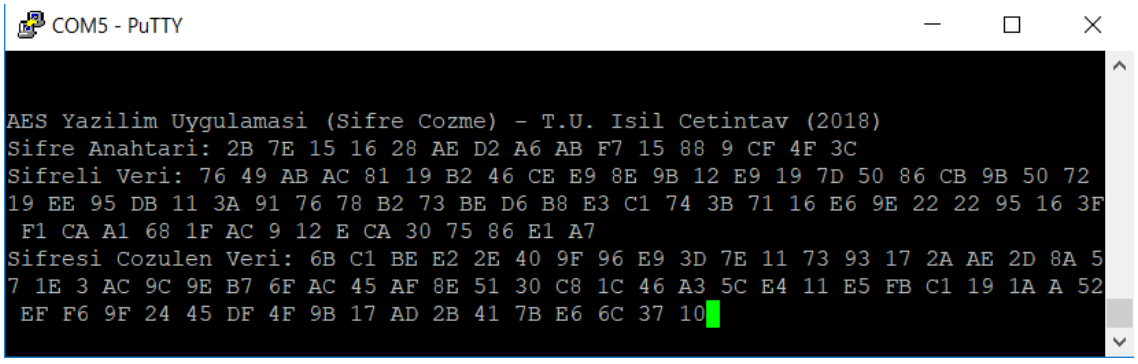
Uygulamada kullanılan anahtar değeri ve girdi-çıktı verileri konsol çıktılarında gösterildiği gibidir. Standart olarak kullanılan örnek veriler kullanılmıştır (Dworkin, 2007).

Şekil 5.2, AES algoritmasının yazılım uygulamasının şifreleme sonucunu seri konsol üzerinde göstermektedir.



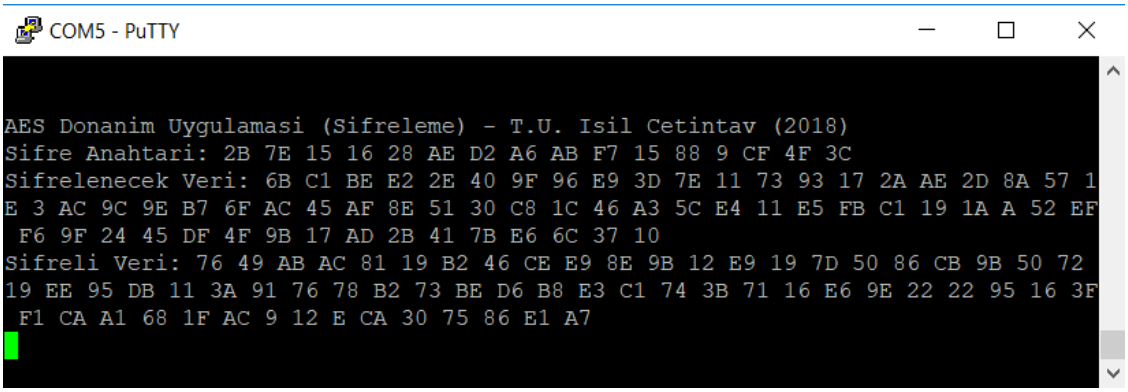
Şekil 5.2. AES Yazılım Uygulaması Şifreleme İşlemi

Şekil 5.3, uygulanan AES yazılım uygulamasının şifre çözme işlemi sonucunu göstermektedir.



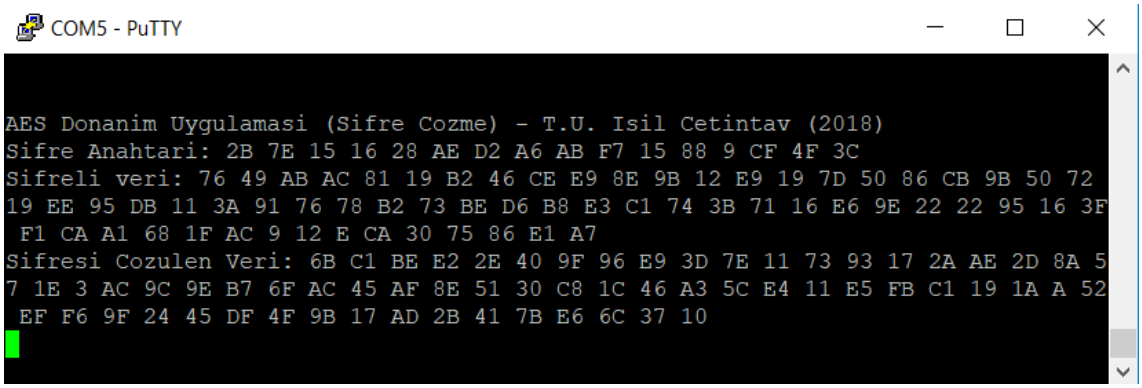
Şekil 5.3. AES Yazılım Uygulaması Şifre Çözme İşlemi

Şekil 5.4'de AES algoritmasının donanım uygulamasının şifreleme sonucu verilmiştir.



Şekil 5.4. AES Donanım Uygulaması Şifreleme İşlemi

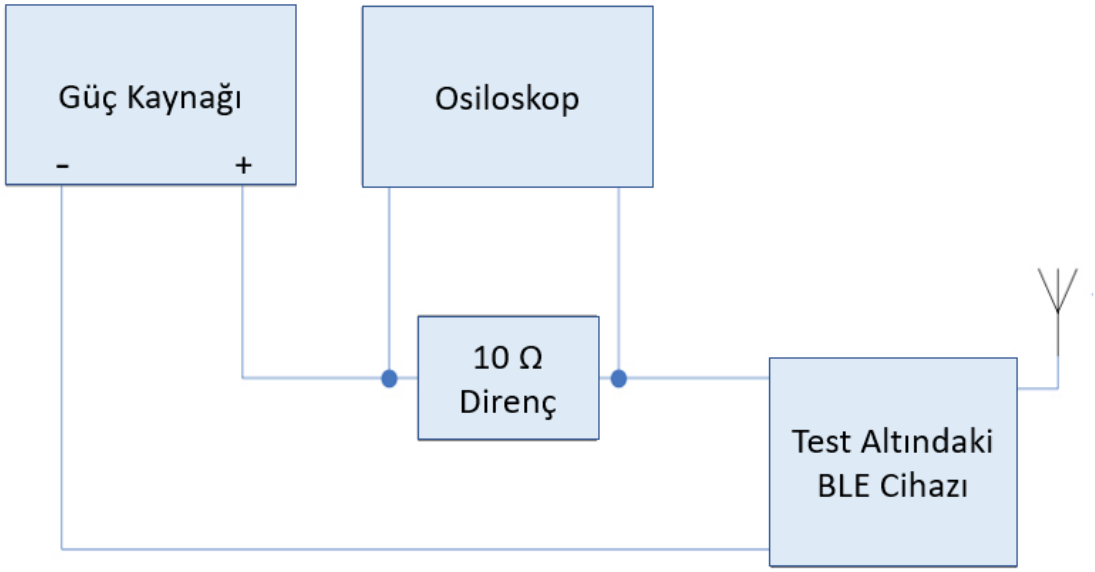
Donanım uygulamasında şifrelenen verinin, aynı algoritma üzerindeki ufak değişiklikler ile (Bölüm 4.4) şifresinin çözülmesi Şekil 5.5'te verilmiştir. Şifre anahtarı ve IV değerleri aynıdır.



Şekil 5.5. AES Donanım Uygulaması Şifre Çözme İşlemi

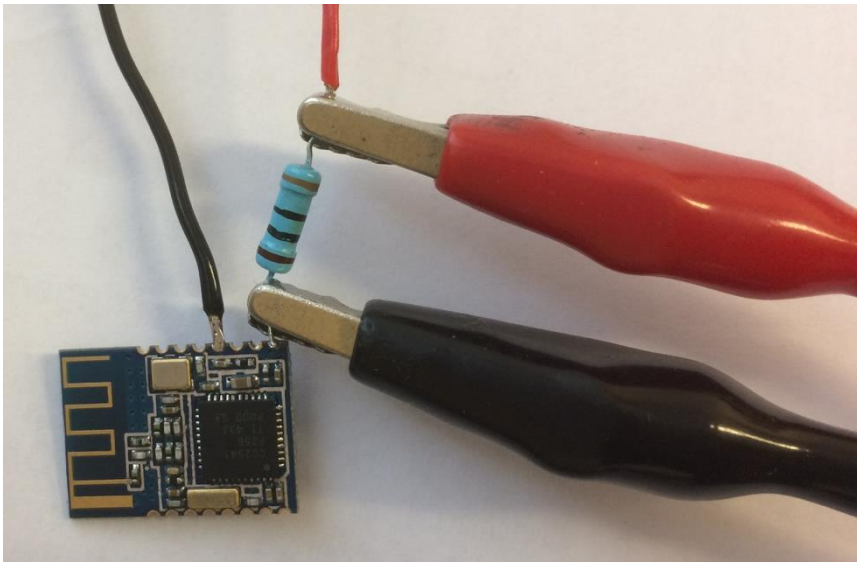
5.2. Şifreleme ve Şifre Çözme İşlemlerinin Güç Tüketim ve Performans Analizi

Bir IoT cihazı üzerinde BLE, uyku, bağlantı, uyanma, işlem gibi çeşitli durumlarda olabilmektedir. Bu yüzden yapılacak ölçümler de tek bir anı değil bir zaman aralığını göstermelidir. Alınan sonuç, cihazın uyku modu, ön işleme modu, şifreleme, şifre çözme ve uyku moduna dönüş aşamalarının zamana bağlı olarak tüm aralıklarda güç tüketimi görülmelidir. Ayrıca bu işlem ampermetre ile yapılamayacak kadar ani değişimlerin olduğu bir işlem olduğundan bu çalışmadaki ölçümler osiloskop ile yapılmıştır. Osiloskop ile ölçümlerin yan kanal analizi ile yapılan saldırılarda da sıklıkla kullanıldığı görülmektedir. Şekil 5.6'da ölçümlerin yapıldığı test ortamı görülmektedir.



Şekil 5.6. Test Ortamı Blok Şeması

Test altındaki cihaz ile güç kaynağı arasında 10 Ω direnç kullanılmıştır. Çekilen voltaj değeri direnç değeri olan 10'a bölünerek akım değeri kolaylıkla bulunabilmektedir.

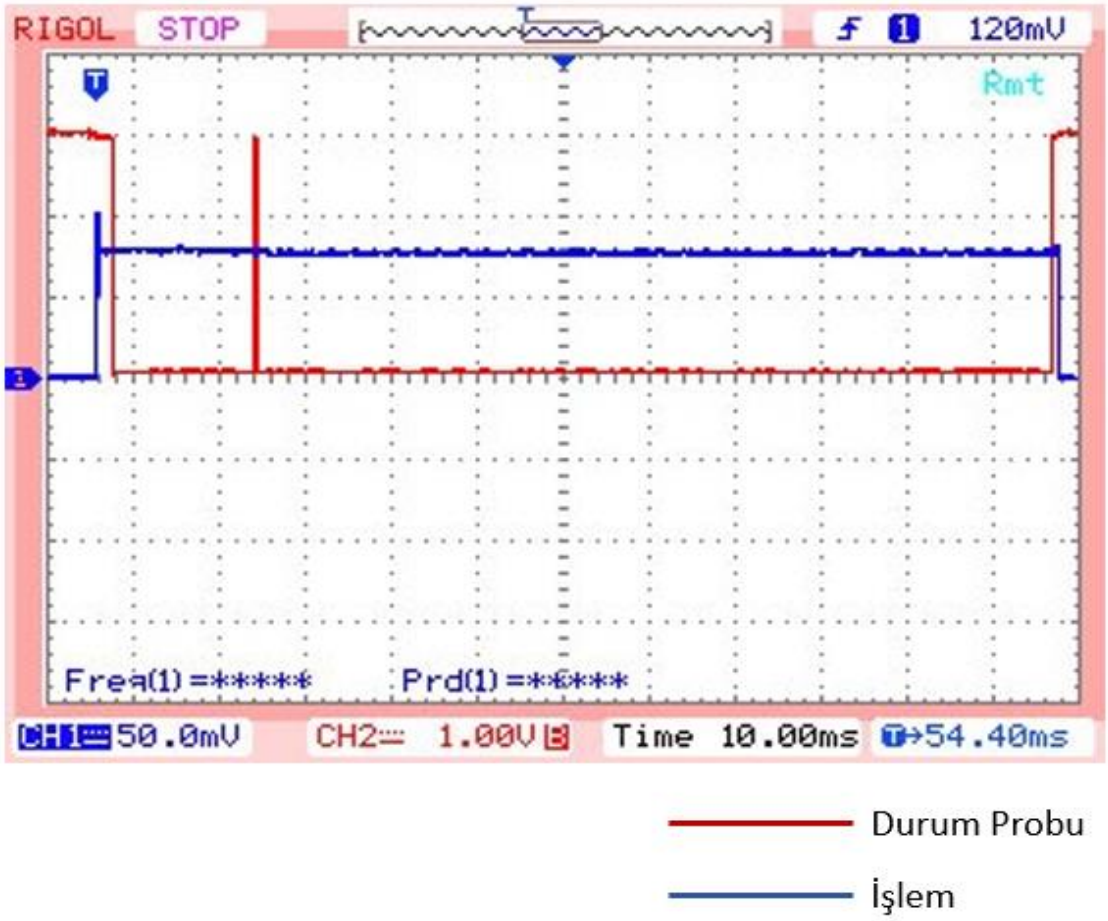


Şekil 5.7. Test Altındaki BLE Cihazı

Yapılan ölçümlerin sonuçları çift kanal üzerinden gösterilmektedir. Mavi ile gösterilen kanal 1 çıktısı, yapılan şifreleme ve şifre çözme işlemlerinin uyanma-tekrar uykuya geçme arasındaki zaman aralığı üzerinde zamana bağlı olarak güç tüketimini göstermektedir. Kırmızı çizgi ile gösterilen kanal 2 ise durum probu olup şifrelemenin

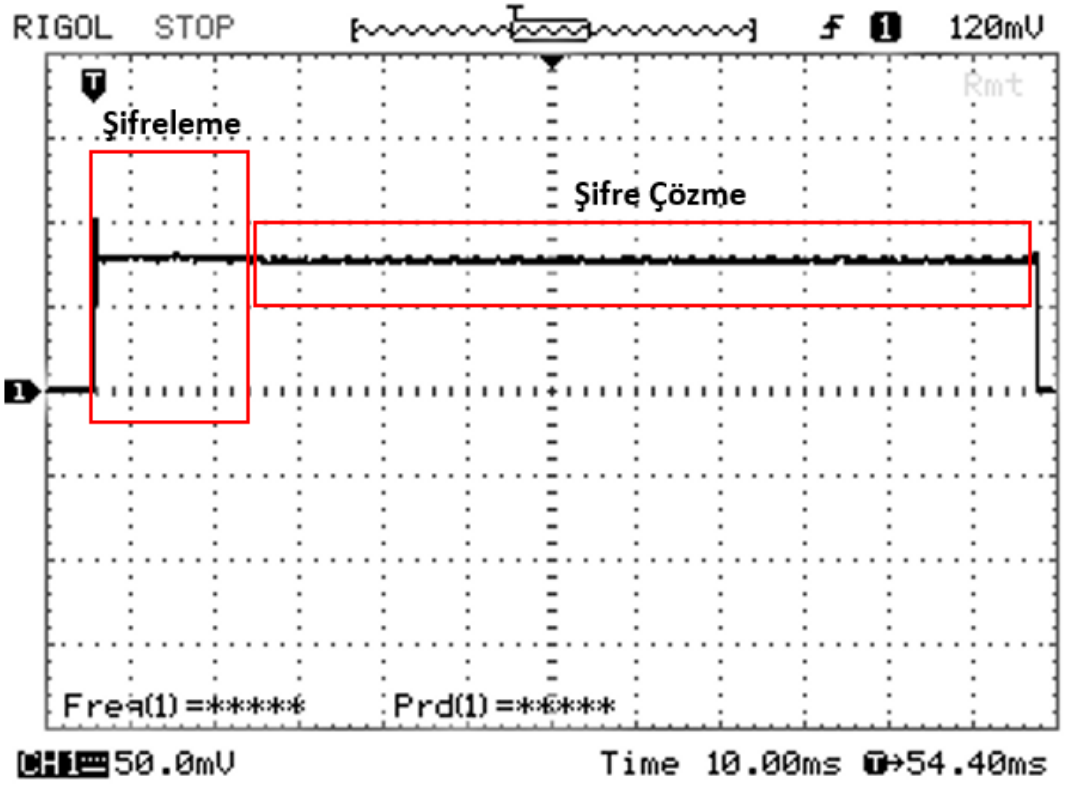
başlangıç-bitiş ve şifre çözme işleminin başlangıç-bitiş noktalarını tespit etmek için kullanılmaktadır.

Şekil 5.8 AES yazılım uygulamasının osiloskop üzerinde sonucunu göstermektedir.



Şekil 5.8. Yazılım Uygulamasının Güç ve Zaman Analizi

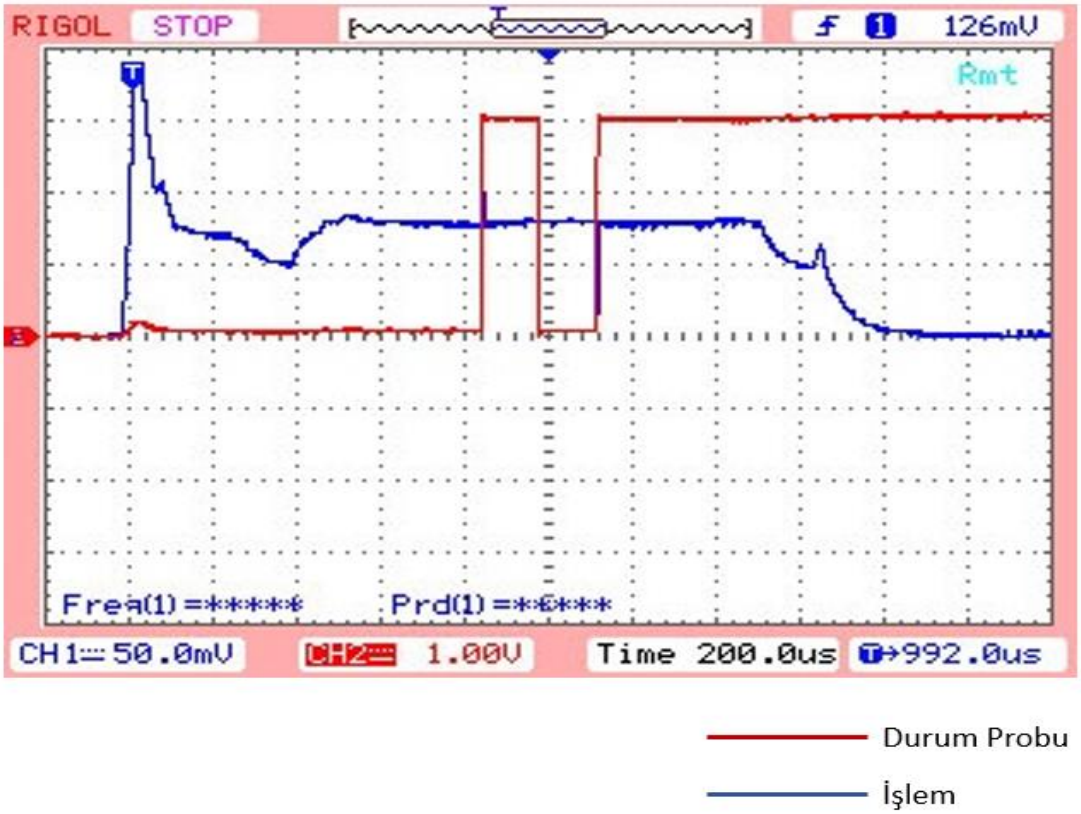
Yazılım uygulamasında şifreleme ve şifre çözme işlemleri 80 mV sabit gerilim ile yapılmaktadır. 10 ohm'luk direnç hesaba alındığında 80 mV/10 Ω= 8 mA'lik akım çekildiği görülmektedir. Şifreleme işlemi 16 ms sürerken şifre çözme işlemi toplamda 92 ms sürmektedir.



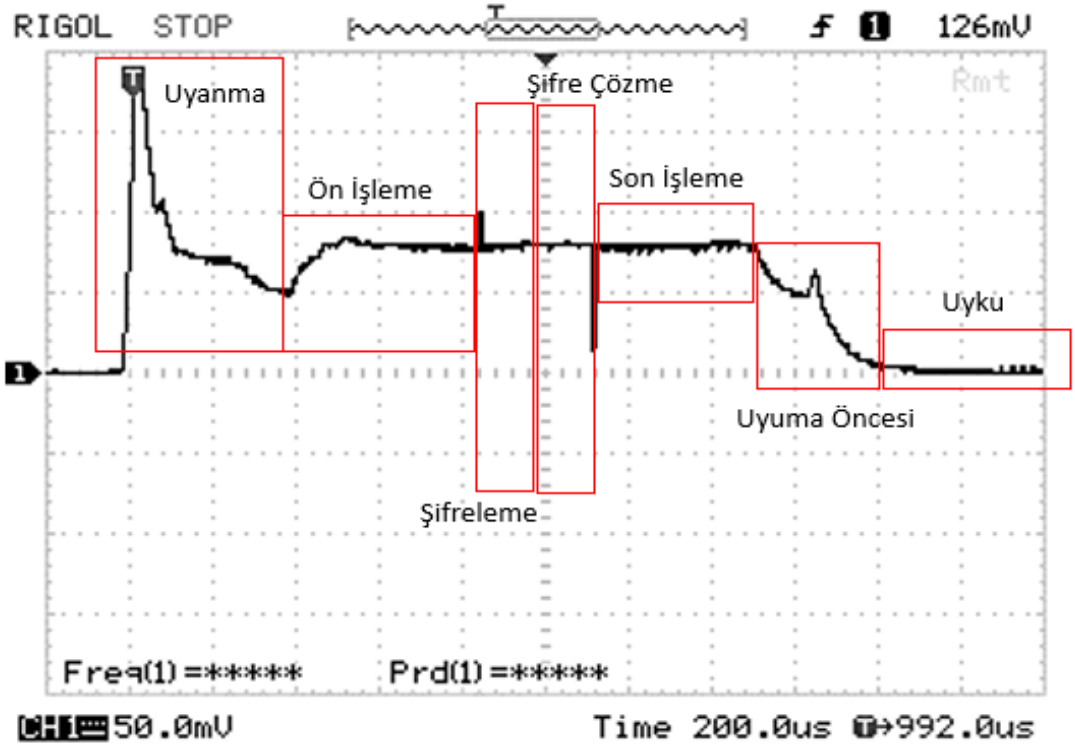
Şekil 5.9. BLE Durumları ve Donanım Uygulaması Güç Performansı

Şekil 5.9, AES yazılım uygulamasının tek kanal üzerinde hangi durumlarda ne kadar gerilimle çalıştığını belirtmektedir.

Şekil 5.10'da ise AES donanım uygulamasının osiloskop çıktısı verilmektedir. Bu çıktıda görüldüğü gibi yazılım uygulamasına göre çok daha az zaman harcanmaktadır. Çekilen akım ise aynıdır ve AES işlemleri boyunca güç tüketiminde bir artış olmamaktadır.

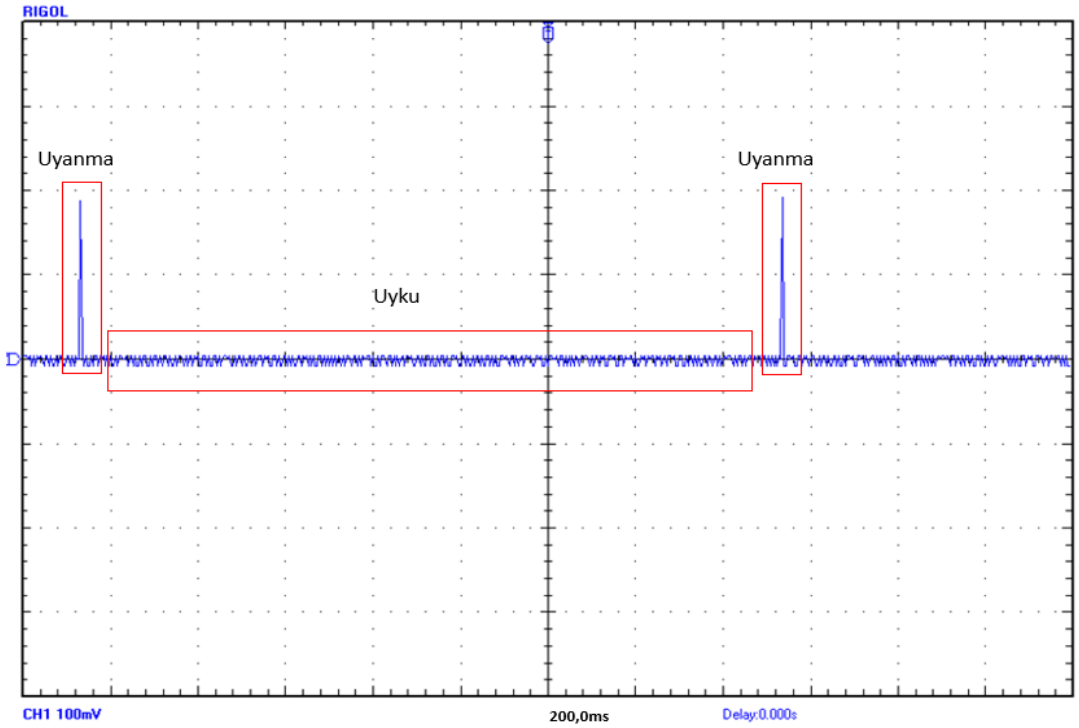


Şekil 5.10. AES Donanım Uygulamasının Güç ve Zaman Analizi



Şekil 5.11. BLE Durumları ve Donanım Uygulaması Güç Performansı

Şekil 5.11 ise AES donanım uygulamasının tek kanal üzerinde hangi durumlar olduğunu belirtmektedir. BLE cihazının bulunduğu durumlara göre uyanma işlemi ile başlayan zaman aralığı tekrar uyku moduna dönme ile sona ermektedir. İlk gerilim sıçraması yonganın uyanma işleminden dolayı 180 mV' tur. Bu işlem aralığında 400 µs için 180 mV/10 Ω= 18 mA akım kullanılmıştır. Sonraki aşamada cihaz, ön işleme durumuna gelir. Bu işlem yaklaşık 480 µs devam eder. Bu işlem sırasında cihaz, OSAL görevlerini hazırlar. Hazırlık işlemi bittiğinde şifreleme ve şifre çözme işlemleri yapılır. Uyku moduna tekrar geçmeden son işleme durumuna geçilir ve ardından uyku moduna girer. Cihaz uyku modundayken ise neredeyse hiç enerji tüketmemektedir.



Şekil 5.12. BLE Cihazının Enerji Tüketimi

Osiloskopun zaman aralığı 200 ms iken cihazın çektiği akım Şekil 5.12'de görülmektedir. Bu tez çalışmasında kullanılan BLE cihazı 1.6 saniye uykuda kalmakta ve bu süre boyunca çok az miktarda (0.001 mA) enerji tüketmektedir.

Zamana bağlı olarak BLE stack durumlarına göre zaman ve akım değerleri Çizelge 5.1'de yazılım uygulaması için ve Çizelge 5.2'de donanım uygulaması için verilmektedir.

Çizelge 5.1. Yazılım Uygulaması İçin Zaman ve Akım Ölçümleri

Durum	**Zaman**	**Akım**
Uyanma	400 µs	18 mA
Ön İşleme	480 µs	8.5 mA
Şifreleme	16 ms	8 mA
Şifre Çözme	92 ms	8 mA
Son İşleme	360 µs	8 mA
Uyuma Öncesi	280 µs	6 mA
Toplam Zaman	109.520 ms	

Çizelge 5.2. Donanım Uygulaması İçin Zaman ve Akım Ölçümleri

Durum	**Zaman**	**Akım**
Uyanma	400 µs	18 mA
Ön İşleme	480 µs	8.5 mA
Şifreleme	140 µs	8 mA
Şifre Çözme	140 µs	8 mA
Son İşleme	360 µs	8 mA
Uyuma Öncesi	280 µs	6 mA
Toplam Zaman	1800 µs	

Sistemin uyku modu dışında (aktif iken) tükettiği enerji aşağıdaki formül ile ölçülebilir:

$$\textbf{Aktif Enerji Tüketimi} = \frac{\sum(Durum\ n\ Zaman * Durum\ n\ Akım)}{Toplam\ Uyanık\ Kalma\ Zamanı} \quad (5.1)$$

Ortalama enerji tüketimi formülü ise aşağıdaki gibidir:

$$\textbf{Ortalama Enerji Tüketimi} = \frac{[Uyuma\ Süresi * Uyku\ Boyunca\ Akım] + [Aktif\ Zaman * Uyku\ Dışı\ Akım]}{Uyuma\ Süresi + Aktif\ Zaman} \quad (5.2)$$

Kullanılan cihaz 230 mAh kapasiteli CR2032 bataryasını kullanmaktadır. Tahmini batarya kullanım süresini hesaplamak için aşağıdaki formül kullanılır.

Beklenen Batarya Ömrü = Batarya Kapasitesi / Ortalama Enerji Tüketimi (5.3)

Yazılım uygulaması için enerji tüketimi hesapları aşağıdaki gibidir:

Aktif Enerji Tüketimi =
[(400 µs) * (18 mA) + (480 µs) * (8.5 mA) + (16 ms) * (8 mA) + (92 ms) * (8 mA) + (360 µs) * (8 mA) + (280 µs) * (6 mA)**]** / 109.520 ms = **8.03360 mA**

Ortalama Enerji Tüketimi =
[(1600 ms* 0.001 mA) + (109.52 ms * 8.03360 mA)] / 1709.52 = **0.515606 mA**

Ortalama enerji tüketimi yaklaşık olarak 0.515 mA olarak hesaplanmıştır. Bu sonuç ile sürekli çalışacak olan bu cihazın bataryasının beklenen ömrü ise aşağıdaki gibidir.

Beklenen Batarya Ömrü = (230 mAh) / (0.515606 mA) = **446.076 saat.**

Yazılım uygulamasının batarya kullanım ömrü yaklaşık olarak 446 saattir. Batarya yaklaşık olarak 18.5 gün sürekli çalışabilecektir.

Donanım uygulaması için enerji tüketimi hesapları aşağıdaki gibidir:

Aktif Enerji Tüketimi =
[(400 µs) * (18 mA) + (480 µs) * (8.5 mA) + (140 µs) * (8 mA) + (140 µs) * (8 mA) + (360 µs) * (8 mA) + (280 µs) * (6 mA)**]** / 1800 µs = **10.0444 mA**

Ortalama Enerji Tüketimi =

[(1600 ms* 0.001 mA) + (1.8 ms * 10.0444 mA)] / 3400 ms = **0.005788211 mA**

Ortalama enerji tüketimi yaklaşık olarak 0.005 mA (5 µA) olarak hesaplanmıştır. Bu sonuç ile sürekli çalışacak olan bu cihazın bataryasının ömrünün ne kadar olacağı tahmin edilebilecektir.

Beklenen Batarya Ömrü = (230 mAh) / (0.005788211 mA) = **39735.939 saat.**

Donanım uygulaması için tahmini batarya ömrü yaklaşık olarak 39735 saat (1655 gün) dolayısıyla yaklaşık 4.5 yıl olarak hesaplanmıştır.

Çizelge 5.3'te, yapılan iki uygulamanın süre ve enerji tüketimi açısından performans karşılaştırılması verilmiştir.

Çizelge 5.3. Yazılım ve Donanım Uygulamaları Performans Analizi Karşılaştırması

	Yazılım Uygulaması		**Donanım Uygulaması**	
	Süre	Çekilen Akım	Süre	Çekilen Akım
Uyanma	400 µs	18 mA	400 µs	18 mA
Ön İşleme	480 µs	8.5 mA	480 µs	8.5 mA
Şifreleme	16 ms	8 mA	140 µs	8 mA
Şifre Çözme	92 ms	8 mA	140 µs	8 mA
Son İşleme	360 µs	8 mA	360 µs	8 mA
Uyuma Öncesi	280 µs	6 mA	280 µs	6 mA
Aktif Enerji Tüketimi	8.03360 mA		10.0444 mA	
Ortalama Enerji Tüketimi	0.515606 mA		0.005788211 mA	
Beklenen Batarya Ömrü	**446.076 saat (18.5 gün)**		**39735.939 saat (4.5 yıl)**	

Her ne kadar yazılım uygulamasında aktif enerji tüketimi daha düşük gözükse de bunun sebebi bu değerin ortalama bir değer olmasıdır. Uyanma anında çekilen akımın yüksek olması, şifreleme ve şifre çözme işlemlerinde çekilen akımların çok daha az olması ve bu işlemlerin yazılım uygulamasında çok uzun sürmesi bu değeri

düşürmektedir. Fakat toplam süreler göz önüne alındığında toplam enerji tüketiminin yazılım uygulamasında çok daha fazla olduğu görülmektedir.

5.3. Değerlendirme

Bu uygulamalar zaman ve güç bakımından belirtilen test ortamında değerlendirilmiştir. Zaman ölçümlerine göre yazılım uygulaması AES işlemleri toplamda 108 ms sürerken, donanım uygulaması AES işlemleri toplamda 280 µs gibi çok kısa bir zamanda gerçekleşmiştir. Görüldüğü gibi donanım uygulaması yazılım uygulamasına göre yaklaşık olarak 385 kat daha hızlı sonuçlanmaktadır.

Cihazın batarya kapasitesine bakıldığında ise hesaplamalar sonucunda ortaya çıkan pil ömürleri, yazılım uygulaması için 18.5 gün iken donanım uygulaması için 4.5 yıldır. Donanım uygulamasının yapıldığı cihaz kesintisiz olarak yaklaşık 90 kat daha uzun süre çalışacaktır.

Bu tez çalışmasında donanım destekli şifreleme işleminin çok daha hızlı olduğu gözlenmiştir. Ayrıca bu işlemler yapılırken güç tüketimi sabit kalmıştır. Küçük boyutlu ve kaynak kısıtlı cihazlar için bu iki özellik büyük avantajdır. Bu doğrultuda kablosuz ağ iletişimi standartlarının şifreleme destekleri de göz önünde bulundurularak donanım destekli şifreli cihazların artacağı tahmin edilmektedir. Elde edilen analiz sonuçlarının olumlu olması ile bu tür cihazlarda şifrelemeye verilen önemin de artacağı öngörülmektedir. Ayrıca bu tez çalışması, yazılım ve donanım arasındaki belirgin fark açıkça gösterildiğinden, hafif sıklet algoritmaların geliştirilmesi aşamasında faydalı olacağı düşünülmektedir.

KAYNAKLAR

Alaba, F. A., Othman, M., Hashem, I. A. T., & Alotaibi, F. (2017). Internet of Things security: A survey. *Journal of Network and Computer Applications*, *88*, 10–28. https://doi.org/10.1016/j.jnca.2017.04.002

Aslan, Y., Sakalli, T., & Aslan, B. (2012). Önemli Blok Şifrelerde Kullanılan Doğrusal Dönüşümlerin İncelenmesi.

Bluetooth® Low Energy GAP Roles - Developer Help. (n.d.).

Bluetooth SIG Inc. (2018). Radio Versions | Bluetooth Technology Website. Retrieved from https://www.bluetooth.com/bluetooth-technology/radio-versions

Conti, M., Dehghantanha, A., Franke, K., & Watson, S. (2018). Internet of Things security and forensics: Challenges and opportunities. *Future Generation Computer Systems*, *78*, 544–546. https://doi.org/10.1016/j.future.2017.07.060

Corser, G. (2017). White Paper Internet of Things (IoT) Security, (May).

Daemen, J., & Rijmen, V. (2003). *The Rijndael Block Cipher: AES Proposal*. *Nist*. Retrieved from http://csrc.nist.gov/archive/aes/rijndael/Rijndael-ammended.pdf

Developer, S. (2015). Software Developer ' s, (September).

Dworkin, M. J. (2007). Recommendation for block cipher modes of operation : https://doi.org/10.6028/NIST.SP.800-38d

Eduonix. (n.d.). Learn the Application Layer Protocols used In IoT.

Ericsson. (2017). IoT SECURITY, (February).

Evans, D. (2011). The Internet of Things - How the Next Evolution of the Internet is Changing Everything. *CISCO White Paper*, (April), 1–11. https://doi.org/10.1109/IEEESTD.2007.373646

Frahim, J., Pignataro, C., Apcar, J., & Morrow, M. (2016). Securing the Internet of Things: A Proposed Framework - Cisco. *Cisco.Com*. Retrieved from http://www.cisco.com/c/en/us/about/security-center/secure-iot-proposed-framework.html

Gerber, A. (2017a). Connecting all the things in the Internet of Things A guide to selecting network technologies to solve your IoT networking challenges, 1–10.

Gerber, A. (2017b). Top 10 IoT security challenges - The developerWorks Blog. *IBM DeveloperWorks*. Retrieved from https://developer.ibm.com/dwblog/2017/iot-security-challenges/

Glória, A., Cercas, F., & Souto, N. (2017). Design and implementation of an IoT gateway to create smart environments. *Procedia Computer Science*, *109*, 568–575. https://doi.org/10.1016/j.procs.2017.05.343

GSM, A. (2014). Understanding the Internet of Things (IoT), (July).

Heron, S. (2009). Advanced Encryption Standard (AES). *Network Security*, *2009*(12), 8–12. https://doi.org/10.1016/S1353-4858(10)70006-4

Heydon, R. (2012). *Bluetooth Low Energy:The Developer's Handbook*. Retrieved from http://www.pearson.ch/1471/9780132888363/Bluetooth-Low-Energy-The-Developers.aspx

Infoivy. (n.d.). Big Data - News, Views and Reviews: NoSQL databases (MongoDB, SimpleDB etc) characteristics comparison. Retrieved from http://www.infoivy.com/2014/05/nosql-databases-mongodb-simpledb-etc.html

Iotforall. (n.d.). What is the Cloud_ How Does it Fit into the Internet of Things_ _ IoT For All.

Jaffey, T. (2014). MQTT and CoAP, IoT Protocols. *Eclipse*.

Kang, B., & Choo, H. (2018). An experimental study of a reliable IoT gateway. *ICT Express*, 0–3. https://doi.org/10.1016/j.icte.2017.04.002

Mathur, A., Newe, T., Elgenaidi, W., Rao, M., Dooly, G., & Toal, D. (2017). A secure end-to-end IoT solution. *Sensors and Actuators, A: Physical*, *263*, 291–299. https://doi.org/10.1016/j.sna.2017.06.019

Olsson, J. (n.d.). 6LoWPAN demystified.

PubNub. (2015). A New Approach to IoT Security.

Riahi Sfar, A., Natalizio, E., Challal, Y., & Chtourou, Z. (2018). A roadmap for security challenges in the Internet of Things. *Digital Communications and Networks*, *4*(2), 118–137. https://doi.org/10.1016/j.dcan.2017.04.003

Shatnawi, M. Q. (2016). Application Layer Protocols for the Internet of Things : A survey.

Taskin, D., Taskin, C., & Cetintav, I. (2017). BLE Single Device Configuration Usage: An Internet of Things Sensor Node. *Materials, Methods & Technologies*, *11*, 465–473.

Texas, I. (2010). Texas Instruments CC2540 Bluetooth ® Low Energy Software Developer ' s Guide. *Energy*.

Texas, I. (2013). 2.4-GHz Bluetooth ® low energy System-on-Chip, (June).

Texas, I. (2014). User ' s Guide, (April 2009).

TI, Encryption, A. E. S., & Decryption, E. (2008). Design Note DN108 Design Note DN108, 1–17.

Tongrang, F., Bingchao, H., Wenbin, Z., Xin, H., Tao, Y., & Tiedao, S. (2016). Research on Access Control and Encryption Transmission of 6LoWPAN, *10*(12), 1–12.

Tutorialspoint. (n.d.). Advanced encryption standard.

Wardekar, R., & Ingole, R. (2013). Wireless Communication Technology NFC in Mobile Computing – A Review Article, 1–9.

Wei, G., & Ye, X. (2016). A Design of Wireless Sensor Network based on BLE, (Aeece), 307–312.

Yong, B., Xu, Z., Wang, X., Cheng, L., Li, X., Wu, X., & Zhou, Q. (2018). IoT-based intelligent fitness system. *Journal of Parallel and Distributed Computing*, *118*, 14–21. https://doi.org/10.1016/j.jpdc.2017.05.006

Zennaro, M. (2017). Introduction to the Internet of Things.

ÖZGEÇMİŞ

Işıl ÇETİNTAV, 1994 yılında Edirne'de doğdu. İlköğrenimini Edirne'de, Ortaöğrenimini Kırklareli ilinde tamamladı. 2008 yılında başladığı Kırklareli Vize Anadolu Lisesi'nden 2012 yılında mezun oldu. Aynı yıl Lisans öğrenimi için başladığı Trakya Üniversitesi Bilgisayar Mühendisliği Bölümü'nden 2016 yılında mezun oldu. 2016 yılında Yüksek Lisans eğitimine Trakya Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda başladı. Araştırma Görevlisi olarak 2017 yılında Trakya Üniversitesi Bilgisayar Mühendisliği Bölümü'nde çalışmaya başladığı görevine halen devam etmektedir.

TEZ ÖĞRENCİSİNE AİT TEZ İLE İLGİLİ BİLİMSEL FAALİYETLER

1) Uluslararası Hakemli Dergilerde Yayınlanan Makaleler

- Taşkın D., Taşkın C., Çetintav I., "BLE Single Device Configuration Usage Example: An Internet of Things Application", Materials, Methods & Technologies, Volume 11, Pages: 465-473 (Ağustos 2017)

2) Uluslararası Bilimsel Toplantılarda Sunulan ve Bildiri Kitabında Basılan Bildiriler (Proceedings)

- Taşkın D., Taşkın C., Çetintav I., "A Key Distribution Method for Internet of Things Applications", European Conference on Science, Art&Culture, Prague (Ekim 2017)