

**T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ**

**NESNELERİN İNTERNETİ İÇİN HAFİF SIKLET KRİPTOLOJİ
ALGORİTMALARINA DAYALI GÜVENLİ HABERLEŞME
MODELİ TASARIMI**

DOKTORA TEZİ

Uras PANAHI

Bilgisayar ve Bilişim Mühendisliği

ARALIK 2022

**T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ**

**NESNELERİN İNTERNETİ İÇİN HAFİF SIKLET KRİPTOLOJİ
ALGORİTMALARINA DAYALI GÜVENLİ HABERLEŞME
MODELİ TASARIMI**

DOKTORA TEZİ

Uras PANAHI

Bilgisayar ve Bilişim Mühendisliği

Tez Danışmanı: Prof. Dr. Cüneyt BAYILMIŞ

ARALIK 2022

Uras PANAHI tarafından hazırlanan “NESNELERİN İNTERNETİ İÇİN HAFİF SİKLET KRİPTOLOJİ ALGORİTMALARINA DAYALI GÜVENLİ HABERLEŞME MODELİ TASARIMI” adlı tez çalışması 07.12.2022 tarihinde aşağıdaki jüri tarafından oy birliği/oy çokluğu ile Sakarya Üniversitesi Fen Bilimleri Enstitüsü, Bilgisayar ve Bilişim Mühendisliği Doktora tezi olarak kabul edilmiştir.

Tez Jürisi

Jüri Başkanı : **Prof. Dr. Cüneyt BAYILMIŞ** (Danışman)
Sakarya Üniversitesi

Jüri Üyesi : **Doç. Dr. Sezgin KAÇAR**
Sakarya Uygulamalı Bilimler Üniversitesi

Jüri Üyesi : **Dr. Öğr. Üyesi Abdullah SEVİN**
Sakarya Üniversitesi

Jüri Üyesi : **Dr. Öğr. Üyesi Murat İSKEFİYELİ**
Sakarya Üniversitesi

Jüri Üyesi : **Dr. Öğr. Üyesi Zafer ALBAYRAK**
Sakarya Uygulamalı Bilimler Üniversitesi



ETİK İLKE VE KURALLARA UYGUNLUK BEYANNAMESİ

Sakarya Üniversitesi Fen Bilimleri Enstitüsü Lisansüstü Eğitim-Öğretim Yönetmeliğine ve Yükseköğretim Kurumları Bilimsel Araştırma ve Yayın Etiği Yönergesine uygun olarak hazırlamış olduğum “NESNELERİN İNTERNETİ İÇİN HAFİF SİKLET KRİPTOLOJİ ALGORİTMALARINA DAYALI GÜVENLİ HABERLEŞME MODELİ TASARIMI” başlıklı tezin bana ait, özgün bir çalışma olduğunu; çalışmamın tüm aşamalarında yukarıda belirtilen yönetmelik ve yönergeye uygun davrandığımı, tezin içerdiği yenilik ve sonuçları başka bir yerden almadığımı, tezde kullandığım eserleri usulüne göre kaynak olarak gösterdiğimi, bu tezi başka bir bilim kuruluna akademik amaç ve unvan almak amacıyla vermediğimi ve 20.04.2016 tarihli Resmi Gazete’de yayımlanan Lisansüstü Eğitim ve Öğretim Yönetmeliğinin 9/2 ve 22/2 maddeleri gereğince Sakarya Üniversitesi’nin abonesi olduğu intihal yazılım programı kullanılarak Enstitü tarafından belirlenmiş ölçütlere uygun rapor alındığını, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun ortaya çıkması halinde doğabilecek her türlü hukuki sorumluluğu kabul ettiğimi beyan ederim. (25/10/2022).

Uras PANAHI





Aileme, eşime ve öğlum Kaan'a



TEŞEKKÜR

Doktora eğitimim boyunca değerli bilgi ve deneyimlerinden yararlandığım, her konuda bilgi ve desteğini almaktan çekinmediğim, araştırmanın planlanmasından yazılmasına kadar tüm aşamalarında yardımlarını esirgemeyen, teşvik eden, aynı titizlikte beni yönlendiren değerli danışman hocam Prof. Dr. Cüneyt BAYILMIŞ'a teşekkürlerimi sunarım. Ayrıca tez çalışmasının gerçekleştirilmesi için özellikle maddi ve manevi desteklerini üzerimden eksik etmeyen değerli eşim ile aileme ve emeği geçen herkese teşekkür ederim.

Uras PANAHI



İÇİNDEKİLER

Sayfa

ETİK İLKE VE KURALLARA UYGUNLUK BEYANNAMESİ	v
TEŞEKKÜR	ix
İÇİNDEKİLER	xi
KISALTMALAR	xiii
SİMGELER	xv
TABLO LİSTESİ	xvii
ŞEKİL LİSTESİ	xix
1. GİRİŞ.....	1
1.1. Problem Tanımı ve Motivasyon	4
1.2. Literatür Taraması	4
1.3. Tezin Katkıları.....	10
2. NESNELERİN İNTERNETİ	13
2.1. IoT Yaygın Mesajlaşma Protokolleri	16
2.1.1. MQTT	16
2.1.2. AMQP	18
2.1.3. DDS.....	19
2.1.4. XMPP.....	20
2.1.5. CoAP	21
2.1.6. STOMP	21
2.1.7. WAMP	22
2.1.8. OPC UA	22
2.1.9. LwM2M	23
3. ŞİFRELEME ALGORİTMALARI.....	25
3.1. Kriptografi Çeşitleri	26
3.1.1. Simetrik Anahtar Şifreleme	26
3.1.2. Asimetrik Anahtar Şifreleme	27
3.1.3. Hash	27
3.2. Hafif Siklet Şifreleme Algoritmaları.....	27
3.3. Hafif Şifreleme Algoritmalarının Performans Değerlendirmesi	39
3.4. Bölüm Sonuçları.....	54
4. HAFİF SİKLET KRİPTOLOJİ ALGORİTMALARINA DAYALI İOT GÜVENLİ MESAJLAŞMA PROTOKOLÜ.....	57
4.1. Kullanımı Önerilen Hafif Siklet Kriptoloji Algoritmaları	58
4.2. MQTT-SN	59
4.2.1. Mesaj Formatı	61
4.2.2. MQTT-SN Kontrol Paket Tipleri.....	63
4.3. Önerilen Protokolün Benzetimi ve Performans Değerlendirmesi	63
5. MQTT-SN TEMELLİ GÜVENLİ AĞ PROTOKOLÜNÜN BENZETİMİ... 67	67
5.1. Riverbed Modeler Uygulaması	67
5.2. Contiki-Cooja Linux Uygulaması	77

5.3. Bölüm Sonuçları.....	83
6. SONUÇLAR VE ANALİZLER.....	85
KAYNAKLAR.....	89
EKLER.....	99
ÖZGEÇMİŞ.....	103



KISALTMALAR

AES	: Advanced Encryption System
AMQP	: Advanced Message Queuing Protocol
BER	: Bit Error Rate
CLBCSF	: Cross-Layer Based Comprehensive Security Framework
CLS-FTCM	: Cross-Layer Security-Based Fuzzy Trust Calculation Mechanism
CLS-IDS	: Cross Layer Security Based Intrusion Detection System
CoAP	: Constrained Application Protocol
DDS	: Data Distribution Service
DECRSA	: Dynamic EC-RSA
DES	: Data Encryption Standard
DoS	: Denial of Service
ECC	: Elliptic-Curve Cryptography
FS-SSL	: Four-Stage Security Service Level
GB	: Gigabyte
GHz	: Gigahertz
IoT	: Internet of Things
ISA	: Intelligent Security Agent
KAA	: Kablosuz Algılayıcı Ağlar
KB	: Kilobyte
KP/CP-ABE	: Key/Ciphertext Policy-Attribute Based Encryption
LMDH	: Logistic Map Diffie Hellman
LPM	: Low Power Mod
LwM2M	: Lightweight M2M Standard
LWSDAS	: Lightweight Secure Data Authentication Scheme
mW	: milliwatt
mA	: Milliampere
M2M	: Machine-to-Machine
MOM	: Message-Oriented Middleware
MQTT	: Message Queuing Telemetry Transport
MQTT-SN	: MQ Telemetry Transport for Sensor Networks

OPC UA	: OPC Unified Architecture
OWASP	: Open Web Application Security Project
PKC	: Public Key Cryptography
PRC	: Routed Remote Procedure Calls
QoS	: Quality of Service
RC	: Rivest Cipher
RSA	: Rivest–Shamir–Adleman
SPN	: Substitution-Permutation Network
SSO	: Social Spider Optimization
STOMP	: Simple (Or Streaming) Text Orientated Messaging Protocol
TTMP	: Text Oriented Messaging Protocol
UDGM	: Unit Disk Graph Medium
WAMP	: Web Application Messaging Protocol
XMPP	: Extensible Messaging and Presence Protocol
YTA	: Yazılım Tanımlı Ağ

SİMGELER

N_B	: Blok boyutu
T_e	: Şifreleme çalıştırma zamanı
T_d	: Şifre çözme çalıştırma zamanı
N_d	: Teslim edilen paketlerin sayısı
P	: Paket boyutu
I	: Mevcut akım [Birim]
T	: Zaman
Q	: Yük
V	: Voltaj [Birim]
E	: Enerji tüketimini [Birim]



TABLO LİSTESİ

Sayfa

Tablo 1.1. OWASP IoT Güvenlik Top 10.....	2
Tablo 1.2. KAA için güvenlik çözümleri karşılaştırması.....	9
Tablo 2.1. IoT’deki kullanılan iletişim teknolojilerin karşılaştırması.....	15
Tablo 2.2. IoT ’de yaygın olan mesajlaşma protokoller karşılaştırması.....	24
Tablo 3.1. Kriptografide kullanılan terimler.....	26
Tablo 3.2. Tez çalışması ve literatürdeki çalışmaların karşılaştırması (Raspberry Pi 3 ve Arduino Mega 256 için).....	36
Tablo 3.3. Seçili hafif siklet şifrelerin karşılaştırılması.....	39
Tablo 3.4. Raspberry Pi 3 ve Arduino Mega 2560’nın şifreleme işlemi için tüketilen enerji karşılaştırması.....	42
Tablo 3.5. Raspberry Pi 3 ve Arduino Mega 2560’nın şifre çözme işlemi için tüketilen enerji karşılaştırması.....	43
Tablo 3.6. Şifreleme RAM kullanımı için sıralanmış algoritmalar (Arduino Mega 2560, Raspberry Pi 3’e karşı).....	45
Tablo 3.7. Şifre çözme RAM kullanımı için sıralanmış algoritmalar (Arduino Mega 2560, Raspberry Pi 3’e karşı).....	46
Tablo 3.8. Şifreleme için ROM kullanım karşılaştırılması.....	47
Tablo 3.9. Şifre çözme modunda ROM kullanım karşılaştırılması.....	48
Tablo 3.10. Şifreleme çalışma süresine göre algoritmalar sıralaması.....	49
Tablo 3.11. Şifre çözme çalışma süresine göre algoritmalar sıralaması.....	51
Tablo 3.12. Ortalama şifreleme iş çıkarma oranı sıralaması.....	52
Tablo 3.13. Ortalama şifre çözmek iş çıkarma oranı sıralaması.....	53
Tablo 4.1. Kullanılan hafif siklet şifreler karşılaştırması (Kriptografik özellikler).....	59
Tablo 4.2. MQTT-SN kontrol paket tipleri.....	63
Tablo 4.3. Kullanıcının MAC katmanında seçebileceği güvenlik seviyeleri.....	64
Tablo 4.4. Katmanlar arası güvenlik seviyesi seçmek.....	65
Tablo 5.1. Ağ simülasyon parametreleri.....	68
Tablo 5.2. Cooja simülasyon parametreleri.....	78



ŞEKİL LİSTESİ

Sayfa

Şekil 1.1. IoT'ye yapılan saldırılar.	3
Şekil 2.1. IoT için bazı örnek katmanlı mimariler.	13
Şekil 2.2. Genel IoT ağ mimarisi.	14
Şekil 2.3. MQTT haberleşme modeli.	17
Şekil 2.4. MQTT-SN protokol mimarisi.	18
Şekil 2.5. AMQP publish/subscribe mekanizması.	19
Şekil 2.6. DDS kavramsal model.	20
Şekil 2.7. XMPP temel mimarisi.	20
Şekil 2.8. CoAP haberleşme mimarisi.	21
Şekil 2.9. STOMP haberleşme mimarisi.	22
Şekil 2.10. WAMP mesajlaşma mimarisi.	22
Şekil 2.11. OPC UA örnek haberleşme diyagramı.	23
Şekil 2.12. LwM2M örnek iletişim modeli.	24
Şekil 3.1. Kriptografi prosedürü.	25
Şekil 3.2. AES şifreleme şeması.	28
Şekil 3.3. RECTANGLE şifreleme işlemi.	29
Şekil 3.4. PRESENT hafif siklet şifresinin temel şifreleme şeması.	30
Şekil 3.5. Skipjack şifreleme işlemi.	30
Şekil 3.6. PRINCE şifreleme işlemi.	31
Şekil 3.7. SIMON round fonksiyonu.	32
Şekil 3.8. HIGHT şifreleme şeması.	32
Şekil 3.9. Piccolo şifreleme şeması.	33
Şekil 3.10. LBlock şifreleme işlemi.	34
Şekil 3.11. XTEA'de bir tur şifreleme şeması.	35
Şekil 3.12. Fantomas şifreleme işlemi.	35
Şekil 3.13. Camellia şifreleme (128-bit anahtar) işlemi.	36
Şekil 3.14. Test sistemleri için veri alışverişi şeması.	40
Şekil 3.15. Test edilen sistemin donanım düzeni (A, Arduino Mega kullanarak- B, Raspberry Pi kullanarak).	41
Şekil 3.16. Şifreleme için tüketilen enerji (Raspberry Pi 3).	42
Şekil 3.17. Şifreleme için tüketilen enerji (Arduino Mega 2560).	43
Şekil 3.18. Şifre çözme işleminde tüketilen enerji (Raspberry Pi 3).	44
Şekil 3.19. Şifre çözme işleminde tüketilen enerji (Arduino Mega 2560).	44
Şekil 3.20. Şifreleme RAM kullanımı (Raspberry Pi 3).	45
Şekil 3.21. Şifreleme RAM kullanımı (Arduino Mega 2560).	45
Şekil 3.22. Şifre çözme RAM kullanımı (Raspberry Pi 3).	46
Şekil 3.23. Şifre çözme RAM kullanımı (Arduino Mega 2560).	46
Şekil 3.24. Şifreleme ROM kullanımı (Raspberry Pi 3).	47
Şekil 3.25. Şifreleme ROM kullanımı (Arduino Mega 2560).	47
Şekil 3.26. Şifre çözme ROM kullanımı (Raspberry Pi 3).	48
Şekil 3.27. Şifre çözme ROM kullanımı (Arduino Mega 2560).	49

Şekil 3.28. Ortalama şifreleme çalışma süresi (Raspberry Pi 3).....	50
Şekil 3.29. Ortalama şifreleme çalışma süresi (Arduino Mega 2560).	50
Şekil 3.30. Ortalama şifre çözme çalışma zamanı (Raspberry Pi 3).....	51
Şekil 3.31. Ortalama şifre çözme çalışma süresi (Arduino Mega 2560).	51
Şekil 3.32. Ortalama şifreleme iş çıkarma oranı karşılaştırması (Raspberry Pi 3). ...	52
Şekil 3.33. Ortalama şifreleme iş çıkarma oranı karşılaştırması (Arduino Mega 2560).	53
Şekil 3.34. Ortalama şifre çözme iş çıkarma oranı karşılaştırması (Raspberry Pi 3). 54	
Şekil 3.35. Ortalama şifre çözme iş çıkarma oranı karşılaştırması (Arduino Mega 2560).....	54
Şekil 4.1. Önerilen protokolün iletişim mimarisi.....	57
Şekil 4.2. Önerilen yaklaşımın işleyişi.....	58
Şekil 4.3. MQTT-SN mesaj alışveriş şeması.	60
Şekil 4.4. Önerilen MQTT-SN güvenli mesaj gönderme sıralama diyagramı.....	61
Şekil 4.5. MQTT-SN mesaj formatı.....	62
Şekil 4.6. MQTT-SN değişken paket formatı.	63
Şekil 4.7. Zigbee genel MAC çerçeve formatı.....	64
Şekil 5.1. MICAz algılayıcı.	69
Şekil 5.2. Riverbed Modeler' de örnek 50 düğümlük ağ yapısı.	69
Şekil 5.3. Riverbed Modeler' da bir düğüm için örnek özellik menüsü.....	70
Şekil 5.4. Open-ZB düğüm modeli.	70
Şekil 5.5. Ortalama çalışma süreleri.....	71
Şekil 5.6. Farklı senaryolar arasında iş çıkarma oranı karşılaştırması.....	72
Şekil 5.7. Ortalama enerji tüketimi karşılaştırması.....	74
Şekil 5.8. Bellek (RAM ve ROM) kullanımları.....	76
Şekil 5.9. Ortalama uçtan uca gecikme karşılaştırması.	77
Şekil 5.10. Örnek 50 düğüm Cooja simülasyonu.....	78
Şekil 5.11. Ortalama çalışma süreleri.....	79
Şekil 5.12. Farklı senaryolar arasında iş çıkarma oranı karşılaştırması.....	80
Şekil 5.13. Ortalama güç tüketim karşılaştırması.	81
Şekil 5.14. Bellek (RAM ve ROM) kullanımı.....	82
Şekil 5.15. Ortalama uçtan uca gecikme karşılaştırması.	83

NESNELERİN İNTERNETİ İÇİN HAFİF SİKLET KRİPTOLOJİ ALGORİTMALARINA DAYALI GÜVENLİ HABERLEŞME MODELİ TASARIMI

ÖZET

Nesnelerin İnterneti, algılayıcılar, işleme yeteneği olan ve diğer cihazlarla (İnternet veya diğer şebekeler) veri alışverişi yapan teknolojilere sahip fiziksel nesneleri tanımlar. IoT ağların bir alt kümesi olan kablosuz algılayıcı ağları, verileri algılayan, toplayan ve ileten düğümlerden oluşur. Bu ağlardaki düğümler arasında algılanan ve aktarılan verilerin yüksek hacmi, verilerin güvenli ve el değmeden tutulmasını gerekli kılmaktadır. Bir IoT ağında düğümlerin kaynakları kısıtlı olduğundan, tipik şifreleme algoritmalarının uygulanması maliyetlidir ve verimli değildir. Birçok çözüm, kaynak kısıtlaması olan kablosuz algılayıcı ağlarının güvenlik taleplerini karşılar ve hafif siklet şifreler, uçtan uca güvenlik sağlamak için sofistike çözümlerdir. Bu tez çalışmanın amacı, kablosuz algılayıcı ağlar için uygulama türü, güvenlik düzeyi ve bit hata oranını dikkate alarak geliştirilmiş bir güvenli haberleşme mekanizması sunmaktır.

Tez çalışmasında, hafif siklet şifreleme çalışmalarında kullanılmak üzere; AES, PRESENT, LBlock, Skipjack, SIMON, XTEA, PRINCE, Piccolo, HIGHT ve RECTANGLE hafif siklet algoritmaların performans değerlerini şifreleme ve şifre çözme modlarında, bellek kullanımı (RAM ve ROM), enerji ve güç tüketimi, iş çıkarma oranı ve yürütme süresi gibi temel faktörler açısından test edilip ve karşılaştırılmıştır. Cihazlar arası veri iletimi bulut üzerinden şifreleme ve şifre çözme modları dahil olmak üzere gerçekleştirilmiştir. Performans analizi, IoT ortamında Raspberry Pi 3 ve Arduino Mega 2560 kullanarak yapılmıştır. Tez çalışmasında ayrıca kablosuz algılayıcı ağlar için uygulama, güvenlik düzeyi ve bit hata oranını dikkate alarak geliştirilmiş bir güvenlik mekanizması sunulmuştur.

Kullanıcıya güvensiz veya güvenli haberleşme modlar arasında seçim yapma yeteneği vermek için Zigbee MAC başlığında ayrılmış çerçeve kontrol alanı bitleri kullanılmıştır. Belirli koşullar altında çapraz katmanlı etkileşim mekanizmasını kullanan ağ, AES'den özel kriterler nedeniyle mevcut diğer algoritmalara (RECTANGLE, Fantomas veya Camellia) geçebilir. AES performansı, seçilen algoritmaların performansı ile donanım (MICAz mote) ve simülatör (Riverbed Modeler) ortamlarında farklı senaryolar için, bellek kullanımı, iş çıkarma oranı, batarya tüketimi ve uçtan uca gecikme metrikleri baz alarak, ölçülmüş ve karşılaştırılmıştır. Ayrıca, MQTT-SN mesajlaşma protokolünde uçtan uca şifrelemek için MQTT-SN mesaj paketi formatında 3 rezerve MessageType bit alanı kullanılarak adı geçen 3 algoritma (RECTANGLE, Fantomas ve Camellia) MQTT-SN mesaj paketinde tanımlanıp ve performansları AES'e karşı kıyaslanmıştır. Performans değerlendirmeler, Zolertia Z1 algılayıcı ve Contiki (Cooja) simülatör üzerinden farklı senaryolar için bellek kullanımı, iş çıkama oranı, pil tüketimi ve uçtan uca gecikmeye dayalı olarak karşılaştırılmıştır.



DESIGN OF A LIGHTWEIGHT CRYPTOGRAPHY-BASED SECURE COMMUNICATION MODEL FOR THE INTERNET OF THINGS

SUMMARY

The Internet of Things describes physical objects with technologies that have sensors, processing capability, and exchange data with other devices (Through internet or other communication networks). Wireless sensor networks, as a subset of IoT networks, consist of nodes that sense, collect and transmit data. The high volume of data detected and transmitted among nodes in these networks makes it necessary to keep data secure and untouched. Since in an IoT network, nodes have restricted resources, implementing typical encryption algorithms are costly and inefficient. Many solutions meet the security demands of resource-constrained wireless sensor networks, and lightweight block ciphers are sophisticated solutions for providing end-to-end security. The aim of this thesis study is to present a secure communication mechanism developed for wireless sensor networks, taking the application type, security level and bit error rate into account.

To be used in light weight encryption studies; Performance values of AES, PRESENT, LBlock, Skipjack, SIMON, XTEA, PRINCE, Piccolo, HIGHT and RECTANGLE lightweight algorithms were tested and compared for both encryption and decryption modes in terms of key factors such as memory usage (RAM and ROM), energy and power consumption, throughput and execution time. In the selection of the mentioned algorithms, the criteria of block length, key sizes, popularity, internal structures and application possibilities on software environments were taken into consideration. Some algorithms, such as PRESENT, were hardware-oriented in the early versions, but were later developed to be installed on software platforms. Except for AES which has a block length of 128 bits, all other algorithms have 64 bits block lengths and a key size of 80 or 128 bits. Data transmission among devices has been done over the Dropbox cloud for encryption and decryption modes. Performance analysis was done using Raspberry Pi 3 and Arduino Mega 2560 in IoT environment.

An improved security mechanism is also presented for wireless sensor networks, taking the application, security level and bit error rate into account. Reserved frame control field bits are used in the Zigbee MAC header to give the user the ability to choose between insecure or secure communication modes. Under certain conditions, using the cross-layer interaction mechanism the network can switch from AES to other available algorithms (RECTANGLE, Fantomas, and Camellia) due to special criteria. Camellia is certified by the ISO/IEC organization and has comparable security levels to AES. RECTANGLE and Fantomas have high-speed software applications using the bit-slice technique, which makes them a good choice for online applications. AES performance was measured and compared with the performance of the selected algorithms based on memory usage, efficiency, battery consumption, and end-to-end latency for different scenarios in hardware (MICAz mote) and simulator (Riverbed Modeler) in Windows 10. In addition, to make an end-to-end encryption over MQTT-SN messaging protocol, using the 3 reserved MsgType bit fields in the MQTT-SN

message packet format, the three algorithms (RECTANGLE, Fantomas, and Camellia) were defined in the MQTT-SN message packet and their performances were compared against AES. Performance evaluations were conducted for different scenarios using the Zolertia Z1 mote and the Contiki (Cooja) simulator on Linux. The evaluations were based on metrics such as memory usage, throughput, battery consumption, and end-to-end latency.



1. GİRİŞ

Nesnelerin İnterneti (Internet of Things, IoT), insan müdahalesi olmadan verilerin algılanması ve toplanmasını, paylaşılmasını sağlayan, internete bağlı ve veri işleme yeteneğe sahip küresel bir ağıdır. Cihazlar arasındaki iletişim, genellikle kısa menzili (Wi-Fi, bluetooth gibi) veya uzun menzilli (3G, 4G, LTE ve 5G gibi mobil ağlarla) ağlarla gerçekleşmektedir.

Teknoloji ilerlemeye devam ettikçe, her şey IoT'nin bir parçası haline getirilebilir. IoT cihazları farklı sektörlerde ve endüstride örneğin sağlık, ticaret, tüketici ve devlet uygulamalarında kullanılmaktadır [1]. Güvenilir ve birlikte çalışabilir bir IoT ekosistemi olmadan, ortaya çıkan IoT uygulamaları yüksek talebe ulaşamaz ve tüm potansiyellerini kaybedebilir. Bu kapsamda, sınırlı kaynaklara sahip IoT cihazlarının haberleşmesinin daha güvenli bir hale getirilmesi için var olan haberleşme tekniklerinin güçlenmesi, yeni uygulamalar ve güvenli haberleşme teknikleri geliştirilmesi önem arz etmektedir [2]. IoT cihazlar arasında, güvenli haberleşme sunan çalışmaların literatürde yeterli olmadığı görülmektedir [3].

Nesnelerin İnterneti mimarisinde küçük ve düşük güçlü cihazlar kullanımı yaygındır. Karmaşıklıkları nedeniyle, geleneksel şifreleme yöntemleri ve algoritmaları doğrudan uygulamak zordur. Şifrelemek ve kodunu çözmek için çok sayıda tur gerektirir ve cihazlarda bulunan sınırlı enerjiyi boşa harcar. Bu nedenle, akademik ve ticari IoT çalışmalar düşük maliyetli, yüksek verimli güvenlik çözümlerine odaklanmıştır [1-3]. Düşük maliyetli güvenlik çözümleri seçerken çeşitli saldırılara karşı dayanıklılığı, daha az enerji tüketimi, daha küçük paket boyutu ve daha az hesaplama yüküne sahip olan farklı kriterler dikkate alınmalıdır.

Bu tez çalışmasında, literatürdeki çalışmalara katkı sağlamak ve yukarıda açıklanan IoT güvenlik problemlerine bir çözüm sağlanması açısından, IoT nesnelerin çalışması dikkate alınarak, yeni ve özgün bir haberleşme teknolojisi geliştirmesine odaklanılmıştır.

Günlük hayatta kullanılan akıllı cihazlar hassas ve özel veriler üretmektedir. Bilginin kötü niyetli kişilerin eline geçmesi kullanıcı için oldukça kötü sonuçlar doğurabilir.

Bunu engellemek için, Nesnelerin İnternetinde güvenlik ve gizliliğin sağlanması oldukça fazla önem arz eder. Konuyla ilgili önemli bir araştırma OWASP (Open Web Application Security Project) tarafından gerçekleştirmiştir (Tablo 1.1 [4]). OWASP Nesnelerin İnterneti projesi, üreticilerin, geliştiricilerin ve tüketicilerin Nesnelerin İnterneti ile ilişkili güvenlik sorunlarını daha iyi anlamalarına yardımcı olmak ve herhangi bir bağlamda kullanıcıların IoT teknolojilerini oluştururken, dağıtırken veya değerlendirirken daha iyi güvenlik kararları almalarını sağlamak için tasarlanmıştır.

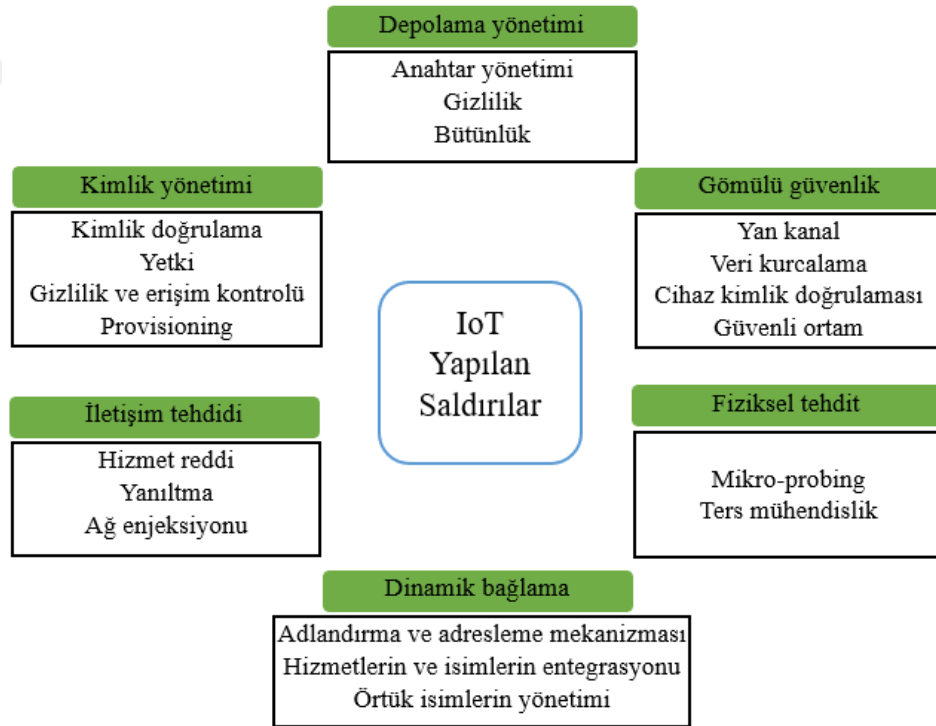
Tablo 1.1. OWASP IoT Güvenlik Top 10.

OWASP IoT TOP 10	Uygulama Güvenliği	Ağ Güvenliği
I1- Zayıf, Tahmin Edilebilir veya Sabit Kodlanmış Şifreler	Evet	Biraz
I2- Güvensiz Ağ Servisleri	Biraz	Evet
I3- Güvenli olmayan ekosistem Arayüzleri	Evet	Biraz
I4- Güvenli Güncelleme Mekanizmalarının Olmaması	Evet	Biraz
I5- Güvensiz veya Güncel Olmayan Bileşenlerin Kullanımı	Evet	Evet
I6- Yetersiz Gizlilik Koruması	Evet	Evet
I7- Güvensiz veri İletilmesi	Evet	Evet
I8- Cihaz Yönetimi Eksikliği	Evet	Evet
I9- Güvensiz Varsayılan Ayarlar	Biraz	Biraz
I10- Yetersiz Fiziksel Güvenlik	Hayır	Biraz

OWASP IoT Top 10 kategorisinin derinlemesine araştırılması IoT güvenlik açıklarının genel olarak üç kategoride sınıflandırılabileceğini öne sürüyor: Yazılım, sistem ve donanım. Yazılım açıkları, IoT sisteminde farklı katmanlarda çalışan uygulamalarla ilişkili güvenlik sorunlarını ifade eder. Bunun yerine, sistem güvenlik açıkları, dağıtılan sistemin yapılandırmasının yanı sıra aygıtların bellekimi veya işletim sistemleriyle ilgili güvenlik sorunlarına atıfta bulunur. Son olarak, donanım güvenlik açıkları, donanım bileşenleri ve içinde çalıştıkları fiziksel ortamla ilişkilendirilir. Tez çalışması, OWASP IoT Top 10’da (Tablo 1.1) verilen Nesnelerin İnterneti alanında güvenlik durumlarının iki tanesine doğrudan çözüm sunmaktadır. Bunlar “I2- Güvensiz Ağ Servisleri” ve “I7- Güvensiz veri İletilmesi” riskleridir. Bu iki risk hem

uygulama (Yazılım) güvenliğini hem de Ağ (Sistem) güvenliğini tehdit etmektedir. Güvensiz Ağ Servisleri, saldırganın cihazdaki veri veya kontrollere erişmek için ayrıntılı izinlerin eksikliğini kullanmasına yol açar. Saldırgan, şifre seçeneklerinin eksikliğini diğer saldırıları gerçekleştirmek için de kullanabilir. Güvensiz veri İletilmesi, ağ bağlantısı üzerinden erişime sahip olan herkesi ilgilendirir. Saldırgan, ağ üzerinden iletilen verileri görüntülemek için ağ trafiğinde şifreleme eksikliğini kullanır. Bu zafiyette, güvenlik zayıflığı açısından yaygın bir zafiyettir. Ağ trafiğinde şifreleme eksikliği veri kaybına neden olabilir ve maruz kalmış verilere bağlı olarak cihazın veya kullanıcı hesaplarının tamamen ele geçirilmesine neden olabilir. İletişimde güvenli bir haberleşmenin kullanımı, sunulan tez çalışmasının motivasyonunu oluşturmaktadır.

Konuyla ilgili diğer bir araştırma IoT cihazlarına yapılan saldırılar Şekil 1.1’de [5] sınıflandırılmıştır. Bu sınıflandırmada beş farklı sınıftaki saldırıların iki tanesi (Şifreleme ve Ağ saldırıları) doğrudan haberleşme ile ilgilidir. Sonuç olarak tez çalışması IoT teknolojisi açısından önemli bir ihtiyacı karşılayacaktır.



Şekil 1.1. IoT’ye yapılan saldırılar.

1.1. Problem Tanımı ve Motivasyon

Güvenli olmayan ağın güvenlik açıklarına ve istenmeyen hasarlara yol açabileceği, açıktır. Verileri istenmeyen erişimlerden korumanın olası yollarından biri, onları güçlü algoritmalarla şifrelemektir. Kaynak sınırlı cihazlar için, RSA, SHA ve diğerleri gibi tipik şifreleme algoritmaları ölçeklenemez. Klasik şifreleme algoritmaları masaüstlerinde, telefonlarda, tabletlerde ve sunucularda kullanılmak üzere tasarlanırken, hafif siklet şifreler, gömülü sistemler ve algılayıcı ağları üzerinde uçtan uca güvenlik sağlamak için ideal çözümlerdir. Gömülü sistemler çoğunlukla 4, 8, 16 ve 32 bit mikro denetleyiciler kullanırlar. Burada bahsedilen tüm özellikler, klasik yöntemler yerine hafif siklet kriptografi kullanımına yol açmaktadır. Tezin amacı IoT alt kümesi olarak kabul edilen Kablosuz Algılayıcı Ağları (KAA) için hafif siklet şifreleme algoritmaları kullanarak, güvenli veri iletiminin sağlanmasıdır. Farklı güvenlik taleplerine göre, hafif siklet şifreleme algoritmalar, uçtan uca şifreleme için önemli ölçüde tercih edilen çıkar yoludur. Hafif siklet algoritmalar hızlı bir şekilde uygulanabilir ve kaynak kısıtlamalı algılayıcı düğümleri için uygundur.

1.2. Literatür Taraması

Kablosuz Algılayıcı Ağların zafiyetleri s-spesifik modeline dayalı olarak [6]'de tartışılmıştır. Adı geçen çalışmada, algılayıcı düğümlerinin güvenlik özelliklerini artırmak için güvenilir bir Kablosuz Algılayıcı Düğüm çerçevesi sunulmuştur. Çerçeve, güvenlik geliştirme ve güvenilir kimlik doğrulama protokolu sağlamaktan sorumlu iki ana platform içerir. KAA destekli IoT'deki güven modelleri (Algı güveni, iletişim güveni ve veri birleştirme güveni), mevcut zorluklar ve ilerlemeler üzerine bir araştırma [7]'de incelenmiştir. IoT tabanlı kablosuz algılayıcı ağlarında yeterli bir güvenlik derecesi sağlamak için Deebak ve Fadi [14], karmaşık (Hybrid) güvenli yönlendirme (Routing) sağlamak için Two-Fish simetrik anahtar yöntemi önermişler. IoT ve KAA, temel özellikleri, uygulamaları, teknolojileri, KAA saldırıları, IoT ve KAA özelliklerini anlamak için iyi bir referans olan mevcut çözümler hakkında kapsamlı bir özet sunulmuştur [15]. Geleneksel bir KAA'nın güvenlik açıkları, KAA için bilinen güvenlik saldırıları, IoT Güvenlik mimarisi, IoT zorlukları ve farklı katmanlar üzerindeki güvenlik sorunları [16]'de açıklanmıştır. Çalışmada ayrıca Blokzincir (Blockchain) teknolojisinin IoT üzerinden uygulanmasına da değinmişler. Veri ve kullanıcı gizliliğine dayalı KAA erişim kontrol modelleri, esneklik, acil erişim

desteđi ve eriřim kontrolü cözümleri için Eliptik Curve Kriptografisi [17]'da sunulmuřtur. [18]'deki calıřmada, aęgözlü (Greedy) yönlendirme tekniđini ve RUT řemasını kullanarak KAA tabanlı bir IoT ađı üzerinden güvenli aktarım verileri sađlamak için SAVEER adlı bir protokol önerilmiřtir. SAVEER, verileri iç ve dıř tehditlerden koruyacaktır. Poongavanam [19], Kablosuz Algılayıcı Ađlarında güvenlik zorlukları ve saldırı türlerini ele almıřtır. Ayrıca, KAA yönlendirme protokolleri için mevcut saldırıları kategorilere ayırmıřlardır. Nabaa [20], anahtar yönetim sürecini iyileřtirmek için Lojistik Harita Diffie Hellman (LMDH) algoritmasını kullandı. Ayrıca kimlik dođrulama ve Ortadaki Adam (Man-in-the-middle) ve tekrar oynatma (Replay) gibi yaygın siber saldırılara karřı önlemler için subMAC kullandılar. Yazarlar ayrıca calıřma süresini azaltmak ve güvenliđi artırmak için KAA varsayılan AES 128 řifreleme algoritması yerine RSA algoritmasını kullandılar. Kablosuz algılayıcı ađlara yapılan saldırılar, aktif ve pasif saldırılara ayrılmıřtır. Gizli dinleme, trafik analizi ve kamuflej, pasif saldırılara iyi örneklerdir. Node Capture Attack, Sybil Attack ve Denial of Service (DoS) en yaygın aktif saldırı örnekleridir. Ghani et al. [21], KAA için Li ve ark. [22] tarafından IoT ortamında önerilen kimlik dođrulama protokolünü genişletmiřtir. İzlenebilirlik ve calıntı dođrulamayı saldırı (Stolen-verifier attack) güvenlik açıklarını ele geçirmek için simetrik kriptografik kimlik dođrulama yöntemini kullandılar. Hafif siklet řifreleme kullanımına yönelik bir diđer calıřma [23]'de sunulmuřtur. Calıřmada, trafik sıklıklađının ve park yönetiminin akıllı izlenmesi için bir IoT altyapısı aracılıđıyla KAA üzerinden aktarılan verilerin güvenliđini sađlamak için karıřık bir hafif siklet řifreleme (PRESENT-GRP) kullanılmıřtır. Önerilen sistem, akıllı bir park izleme sistemi içindeki düđümlerin hesaplama maliyetini ve enerji tüketimini ölçmek için kullanılır. Dört hafif siklet řifrelemenin (AES-128, XXTEA, Skipjack ve RC5) enerji tüketimi, [24]'te bir MICAZ KAA modülü üzerinden XMAC, CMAC, HMAC ve CBC-MAC teknikleri ile ölçölmüřtür. Calıřmada, řifresiz (No-Security) ve řifreli (Enciphered) olmak üzere iki mod üzerinde deđerlendirmelerde bulunulmuřtur. Zirem [25], gercek zamanlı KAA ve IoT uygulamalarını güvence altına almak için 8 bitlik bir mikro denetleyici üzerinden kaos tabanlı bir sistem kullanmayı önermiřtir. Önerilen sistem iki ayrı lojistik harita (Tek boyutlu) kullanmaktadır. Bu sistemin zaman alıcı hesaplamalardan kaçınması gibi bazı avantajları vardır. Yazılım Tanımlı Ađ (YTA) mimarisi, güvenlik zorlukları, YTA ve IoT uygulamaları için cözümler ve YTA'nın IoT ve KAA ile entegre edilmesi olasılıđı [26]'te analiz edilmiřtir. Kablosuz algılayıcı verilerinin bulut üzerinden

aktarılması da özellikle akademik amaçlar için hızlı bir büyüme göstermektedir. Dwivedi et al. [27], bulut ve KAA üzerinden güvenlik sorunları ve farklı saldırı türleri hakkında bir özet yazmıştır. Düğümler arasında güvenli iletişim sağlamak için yazarlar [28], kimlik doğrulama ve anahtarların yönetimi için kullanılmak üzere Eliptik Curve Kripto Sisteminin (ECC) kullanılmasını önerdiler. Ağ dağıtımı, komşuları tanıma, şifreleme/şifre çözme süreçleri ve imzaları doğrulama, KAA ve IoT'de ECC'yi uygulamanın temel adımlarıdır. KAA üzerinden büyük veri uygulamaları için güvenlik hususlarının üstesinden gelmek için Smys [29], enerjiye duyarlı bir yönlendirme protokolü önermiştir. Bahsedilen algoritma üç adımda uygulanmaktadır: Yol tanımlama, bilgi aktarma (Şifreleme/şifre çözme) ve bakım prosedürü. Önerilen yönlendirme protokolü performansı, enerji tüketimi, iş çıkarma oranı, gecikme, paket teslim oranı ve ağ ömrü açısından LCRP ve LEACH protokolleriyle karşılaştırılmıştır. Shafiqul Abidin [30], kötü niyetli (Malicious node) düğümleri tespit etmek ve bilgileri güvende tutmak için hafif uzman sistemler konseptini uygulamıştır. KAA'ı güvenlik saldırılarından korumak için ADIOS ve CLIPS motorlarını kullandı. KAA güvenlik yönleriyle ilgili bir özet [31-34]'te incelenmiştir. [35]'te, Zigbee modülü CC2530 ve Cortex A-9 işlemcileri üzerinde bir LED hafif siklet algoritması uygulanmıştır. Farklı veri uzunluğu için LED algoritmasının enerji tüketim karşılaştırması AES'e göre değerlendirilmiştir. Ayrıca tekrar oynatma, dinleme ve bilinen anahtar saldırıları için iki algoritma arasında güvenlik analizi yapılmıştır. KAA'ı güvenceye almak için Sumalatha [36] tarafından, katmanlar arası tabanlı bir güvenlik tanıtım sistemi (CLS-IDS) ve katmanlar arası bulanık tabanlı güven hesaplama mekanizması (CLS-FTCM) tanıtılmıştır. İlki ağ performansını ve ağ ömrünü iyileştirebilir, enerji tüketimini azaltabilir ve QoS gereksinimlerini karşılayabilir. İkincisi, KAA için daha iyi güvenlik sağlar. IoT tabanlı Kablosuz Algılayıcı Ağlarda izinsiz girişlere karşı bir savunma mekanizması sağlamak için [37]'de ağ ömrünü artıracak güvenli, enerji açısından verimli bir protokol önerilmiştir. Önerilen algoritma üç aşamada çalışmaktadır: Küme oluşturma, güvenli yönlendirme ve küme başlıklarını güncelleme. Sharma [38] çalışmasında, KAA'da güvenliğini sağlamak için çapraz katmanlı entegre çerçeve önerilmiştir. Bu yapıda akıllı bir güvenlik aracı (ISA) bulunmaktadır. Bu aracı, katmanlar arası etkileşimlerden ve güvenlik seviyelerinin değerlendirilmesinden sorumludur. Gawdan [39] çalışmasında katmanlar arası kapsamlı bir güvenlik çerçevesi (CLBCSF) kullanımını önerilmiştir. Adı geçen modelde, Kablosuz Algılayıcı Ağlar için hiyerarşik bir güvenlik çerçevesi sunulmaktadır. Bu model, yeni

bir anahtar yönetim modülü, güvenli iletişim protokolü (Ağ tabanlı) ve Hiyerarşik küme oluşturma bölümü içermektedir. KAA için çapraz katman tasarım yaklaşımları ve zorlukları hakkında kapsamlı bir araştırma, [40-42]'de sunulmuştur. Khashan [43], KAA düğümleri için şifreleme parametrelerinin seçimini (Her düğümün kaynakları nedeniyle) otomatikleştiren bir FlexCrypt şeması tasarlamıştır. Ayrıca, güvenli veri alışverişi sağlamak için çalışmada yeni kimlik doğrulama ve anahtar-yönetim yöntemler önerilmiştir. [44]'te asimetrik kriptografi algoritması (RSA) ve ECC algoritmalarına dayalı bir karmaşık kriptografik şema (DECRSA) önerilmiştir. KAA için yüksek bir güvenlik seviyesi sağlamak için, ECC özelliklerinden faydalanıp ve RSA sistemini kullanarak önerilen yöntem geliştirilmiştir. Algılayıcı bilgi sistemlerinde geliştirilmiş güç verimli toplama algoritması [45]'te ele alınmıştır. Çalışmada, iletim sırasında enerji tüketimini azaltmak için en uygun iletişim mesafelerini belirlenmiştir. Bu yöntemde, düğümler arasındaki enerji tüketimini dengelemek için mobil alıcı düğüm teknolojisini kullanarak, önerilen algoritmanın enerji tüketimi, ağ gecikmesi ve kullanım ömrü açısından iyi bir performans sergilemiştir. Tabatabaei [46]'te algılayıcı düğümlerini kümelemek için sosyal örümcek optimizasyonu adlı bir algoritma önermiştir. Çalışmada, koloninin biyolojik kurallarına göre bir grup örümcek, ağ ömrünü artırmak için birbirleriyle etkileşime girmektedir. Wang [47], büyük ölçekli kablosuz algılayıcı ağları için bir zamanlama yöntemi önermiştir. Çalışmada, optimum kapsama oranına sahip park konumlarını bulmak için parçacık sürüsü optimizasyonunu mutasyon operatörüyle birleştirilmiştir. [48]'de, enerji dağılımını ve optimal küme başı seçimini iyileştirmek için Cuckoo araştırması baz alınarak parçacık sürüsü optimizasyon tekniği kullanılmıştır. [49]'de, tarım alanı izlemede kullanılacak KAA için matematiksel bir güç tüketimi modeli sunulmuştur. Atarraya ağ simülatörünü kullanarak, birkaç protokolü ve hafif siklet şifrelerini değerlendirilmiştir. Carlos Anders [50] bir FPGA üzerinden Encrypt-then-MAC jenerik kompozisyonunun iş çıkarma oranını, gecikmesini ve enerji tüketimini değerlendirilmiştir. AES 128, PRESENT 128, SHA 256 ve SPO 88'i temel şifreleme algoritma olarak baz almıştır. Mahato [51], değiştirilmiş ECC algoritmasını kullanan bir sistem önermiştir. Paket göndermek için yürütme sürelerini karşılaştırarak geleneksel ve önerilen ECC algoritmalarını değerlendirmiştir. Kalyane [52], ECC algoritmasına dayalı hafif bir güvenli veri doğrulama şemasını (LWSDAS) tanıtmıştır. LEACH, SLEACH ve önerdikleri algoritma çeşitli kriterlere göre karşılaştırılmıştır. IoT uygulamaları için blok şifrelemeye dayalı hafif siklet kriptoloji algoritmalarını

[53]'de incelemiştir. IoT ağ uygulamalarında sıklıkla kullanılan mikro denetleyiciler, mobil cihazlar, akıllı kartlar gibi kısıtlı kaynaklara sahip cihazlar üzerinde en sık kullanılan kriptografik algoritmaların performans değerlendirmesini [54]'te ele alınmıştır. LICITUS çerçevesi [55], bir IEEE 802.15.4 ağında 2. katman güvenlik hizmetlerini etkinleştirir, yapılandırır ve yönetir. Bu işlemlerin tümü dağıtılmış bir şekilde yürütülür. Üretici tarafından depolanan veya sistem yöneticisi tarafından güncellenen bir dizi kriptografik malzeme ve konfigürasyon değişkeninden başlayarak, aslında, her düğüm, herhangi bir uzaktan kumandanın kontrolüne gerek duymadan güvenlik hizmetlerini bağımsız olarak önyükleyebilir ve komşularıyla bağlantı düzeyinde anahtarlar üzerinde anlaşabilir. IoT için DTLS ve CoAP entegrasyonu (Lithe) [56]'te ele alınmıştır. Lithe ayrıca, 6LoWPAN standardından yararlanarak enerji tüketimini önemli ölçüde azaltmayı amaçlayan yeni bir DTLS başlık sıkıştırma şeması önermektedir. En önemlisi, önerilen DTLS başlık sıkıştırma şema, DTLS tarafından sağlanan uçtan uca güvenlik özelliklerinden ödün vermez. Aynı zamanda, DTLS standart uyumluluğunu korurken iletilen bayt sayısını önemli ölçüde azaltır. Singh [57], MQTT ve MQTT-SN için güvenliği artırılmış SMQTT ve SMQTT-SN haberleşme protokoller önermiştir. Önerilen çözüm, Key/Ciphertext Policy-Attribute Based Encryption (KP/CP-ABE) yöntemine dayanmaktadır. ABE kullanmanın avantajı, tek bir şifreleme ile mesajı birden fazla kullanıcıya teslim edilir ve dolayısıyla IoT uygulamaları için uygun olan doğal tasarımından kaynaklanmaktadır. [58]'de MQTT protokolünü kullanarak güvenli bilgi alışverişi ve tüm IoT altyapısını güvence altına almak için yeni bir yaklaşım önerilmiştir. MQTT protokolüne ve ortak anahtar altyapısı "PKI"ye dayalı olarak iletişim ve güvenlik anahtarını yönetmek için yeni bir yaklaşım kullanılmıştır. Şifreleme/şifre çözme aşamasında, her birinin avantajını ve faydasını sağlamak için iki farklı yaklaşım (RSA ve Eleptic Curve) kullanılmıştır. PRESENT ve LBlock algoritmaların IoT ortamında performans değerlendirmesi [59-60]'de karşılaştırılmıştır. Nesnelerin İnternet'inin bir alt alanı olan kablosuz algılayıcı ağlarda sistem kaynaklarını minimum kullanarak güvenlik seviyesini arttırmaya yönelik olarak düşük güçlü şifreleme algoritmaları ile kaotik şifrelemenin kullanımının önerildiği çalışmalar da gerçekleştirmişlerdir [61-64]. Tablo 1.2'de, literatürdeki KAA için güvenlik çözümlerine ilişkin bazı ilgili çalışmaların özeti tez kapsamında yapılan ve ölçülen faktörler ile bir karşılaştırması sunulmaktadır.

Tablo 1.2. KAA için güvenlik çözümleri karşılaştırması.

Makale	Platform	Değerlendirme Metrikleri	Algoritmalar	Karşılaştırma Yöntemi
Othman [24]	MICAz mote	Enerji tüketimi	Skipjack, XXTEA, RC5, AES-128	Yükler (8, 16, 24, 32)
Khashan [43]	Contiki (Cooja) ağ simülâtör	Ağ ömrü, güç tüketimi, şifreleme süresi	AES, TEA, FlexCrypt, FlexenTech	Blok boyutu (4, 8, 16, 32, 64, 128), tur sayısı
Tabatabaei [46]	OPNET Modeler 11.5	Protokoller (DCRRP, SSFBCA, NODIC), enerji tüketimi, orta erişim gecikmesi, iş çıkarma oranı, uçtan uca gecikme	Social Spider Optimizasyon (SSO)	50 düğüm
Radosavljević [49]	Atarraya ağ simülâtör	Protokoller (A3, A3 kapsama, EECDS, CDS rule K), protokol başına güç tüketimi	AES, Noekeon, LED 128, PRESENT, Prince, Piccolo, TWINE, Simon 64/96, KATAN 64	Algoritmaların blok uzunlukları (64, and 128 bayt)
Carlos Anders [50]	FPGA	Güç tüketimi, gecikme, iş çıkarma oranı, FPGA kaynakları	AES, PRESENT, SHA-256, SPO-88	FPGA units of slices (SLC), Look-Up Tables (LUT), and Flip-Flops (FF)
Anisha Mahato [51]	NS simülâtör	Yürütme zaman farkı	Önerilmiş ECC, ECC	Yükler (4, 8, 16, 32 bits)
Kalyane [52]	NS-2 simülâtör	Enerji tüketimi, paket teslim oranı, uçtan uca gecikme, ek yük	LWSDAS, LEACH, SLEACH	Düğüm sayısı (10, 20, 30, 40, 50, 60, 70, 80, 90, 100)
Tez çalışması	1. aşama [13]	RAM, ROM, çalışma süresi, çıktı, birleşik metrik, enerji tüketimi, döngü/bayt	AES, PRESENT, LBlock, Skipjack, SIMON, XTEA, PRINCE, Piccolo, HIGHT, RECTANGLE	Yükler (8, 16, 32, 64, 128, 256, 512, 1024, ve 2048 bayt), şifreleme ve şifre çözme modları
	2. aşama [66]	Ortalama çalışma süresi, iş çıkarma oranı, enerji tüketimi, bellek kullanımı (RAM ve ROM), uçtan uca gecikme	AES, RECTANGLE, Fantomas, Camellia	Yükler (16, 32, 64, 128, 256, 512, 1024 bayt) ve Düğüm sayısı (5, 10, 25, 50)
	3. aşama	Ortalama enerji tüketimi, bellek kullanımı (RAM ve ROM), iş çıkarma oranı, uçtan uca gecikme, çalışma süresi	AES, RECTANGLE, Fantomas, Camellia	Yükler (16, 32, 64, 128, 256, 512, 1024 bayt) ve Düğüm sayısı (5, 10, 25, 50)

1.3. Tezin Katkıları

Tezin ana katkıları aşağıdaki gibi sınıflandırılmıştır:

- IoT nesnesi gibi kısıtlı donanımlarda kullanılabilecek hafif siklet kriptoloji algoritmaları ayrıntılı olarak incelenmiştir.
- Raspberry Pi 3 ve Arduino Mega 2560 platformlar üzerinde hafif siklet kriptoloji algoritmalarının RAM, ROM kullanımı, çalışma süresi, enerji tüketimi gibi metriklerle göre başarımlar değerlendirilmesi gerçekleştirilmiştir.
- IoT uygulama geliştiriciler için donanımlara uygun algoritma seçimi için yol gösterilmiştir. MICAz algılayıcı (8-bit Atmel ATmega128L mikro denetleyici) ve ağ simülatörü (Riverbed (OPNET) Modeler) üzerinden üç hafif siklet şifreleme artı AES-128'in (Kriptanaliz özelliklerine odaklanarak) önerilmesi ve yazılım uygulaması.
- Hafif siklet kriptoloji algoritmalarının uygulamaya özgü seçilebildiği, MQTT-SN protokolüne dayalı IoT güvenlik protokolü önerilmiştir.
- Kullanıcı istekleri veya artırılmış BER gibi çeşitli uygulama güvenlik gereksinimlerine göre Zigbee ağları için dört aşamalı bir güvenlik düzeyi (FS-SSL) önerilmiştir. Kullanıcıya güvensiz veya güvenli modlar arasında seçim yapma yeteneği vermek için Zigbee MAC başlığında ayrılmış rezerve çerçeve kontrol alanı bitleri kullanılmıştır. Kullanıcı tarafından talep edildiğinde veya bit hata oranında bir artış olduğunda, fiziksel katmandan MAC katmanına geri bildirim göndermek için güvenli veri transferinde çapraz katmanlı mimari kullanılmıştır. MAC katmanı daha sonra verileri daha düşük ağırlıklı bir algoritma ile yeniden şifreleyecektir (Bu, kod içinde daha düşük çalışma süresi veya daha düşük bellek kullanımı gibi çeşitli ilkeler uygulanarak tanımlanabilir). Alternatif bir algoritma ile şifreleme, veri gizliliğini artıracak ve özellikle kaynak kısıtlaması IoT uygulamaları için verilerin yetkisiz erişime karşı daha iyi korunmasına, ağ ömrünün artmasına ve enerji tüketiminin azalmasına yardımcı olacaktır.
- Önerilen şema, KAA'lar için yazılım tabanlı bir güvenlik çözümü olarak adil bir ölçüm ve karşılaştırma yapmaktadır.
- MQTT-SN mesajlaşma standardını kullanarak uçtan uca seçilen dört kriptoloji algoritması kullanılarak şifreleme yapıldı. Simülasyonlar Contiki (Cooja)

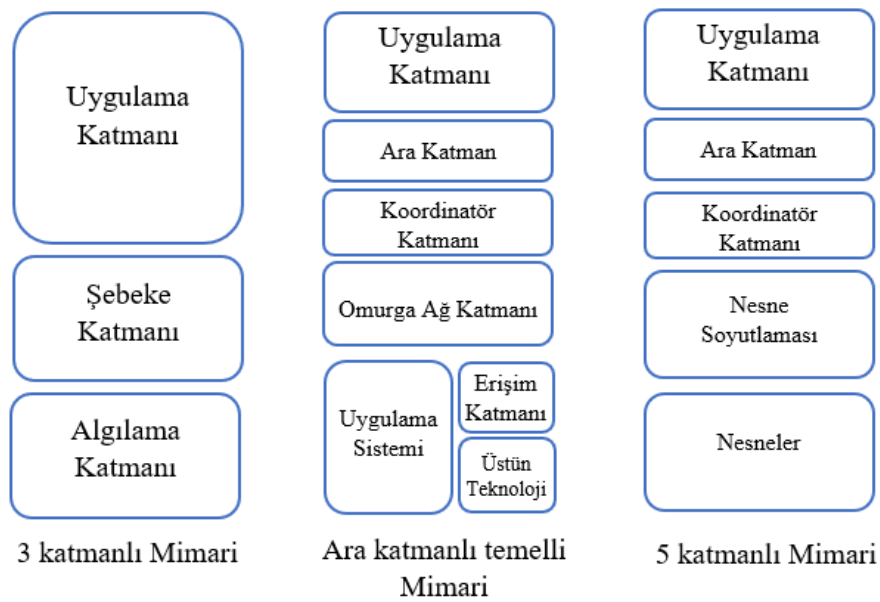
üzerinde gerçekleştirildi. MQTT-SN Kontrol Paketinde rezerve olan bitleri kullanarak, kullanıcı şifresiz veya şifreli modlarda tanımlanan hafif siklet algoritmalarından birini seçebilir. Performans değerlendirmeleri, Zolertia Z1 algılayıcı ve Contiki (Cooja) üzerinden çoklu senaryolar için bellek, iş çıkarma oranı, pil tüketimi ve uçtan uca gecikmeye dayalı ve ortalama çalışma zamanı olarak ölçülüp ve karşılaştırılmıştır.





2. NESNELERİN İNTERNETİ

Günümüzde, Nesnelerin İnterneti birçok uygulaması insan yaşamının benzersiz yönlerinde görülmektedir. Nesnelerin İnterneti, algılayıcılar, işleme yeteneği olan ve diğer cihazlarla (İnternet veya diğer şebekeler) veri alışverişi yapan teknolojilere sahip fiziksel nesneleri tanımlar. Akıllı evler, giyilebilir cihazlar, endüstriyel akıllı izleme sistemleri, IoT'nin bazı örnekleridir. Kevin Ashton ilk kez 1999 yılında bir RFID projesi kapsamında nesnelerin interneti terimini tanıttı. Nesnelerin İnterneti teriminin ilk kez kablolu veya kablosuz olarak birbirine bağlanabilen, birbiriyle ilişkili bir cihaz grubu olarak ortaya çıktığı 2009 yılına bakılırsa, şimdiye kadar teknolojiler, altyapı ve IoT kullanımı için artan bir ilerleme görebilir. IoT'nin genel büyümesine iki ana faktör yardımcı olmaktadır: Hızlı internet bağlantıları ve bulutu kullanmak. Hızlı internet bağlantıları açısından, 5G ağlarına yönelik eğilim etkisini gösterecektir. Bulut tabanlı çözümler, ölçeklenebilir bir IoT ağı oluşturmak ve sürdürmek için hayati önem taşıyan veri mobilitesi ve iş birliği sağlar. IoT düğümlerinin sınırlı pil koşulları nedeniyle belirli bir fiziksel boyutu, işlem gücü ve bellek kapasitesi vardır. İşlemsel teknolojiler, bilgi teknolojiler ve akıllı nesneler, nesnelerin internetin ağ bileşenleridir. Şekil 2.1 [1], IoT için bazı önerilen katmanlı mimarileri göstermektedir.

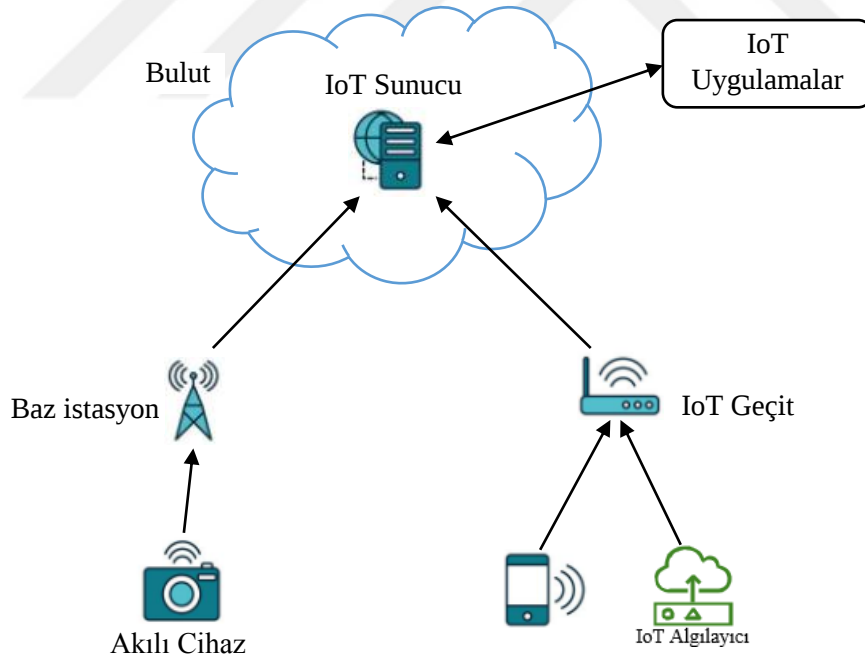


Şekil 2.1. IoT için bazı örnek katmanlı mimariler.

IoT ekosistemindeki yenilik, gerçek ve dijital dünya arasındaki boşluğu kapatmaktadır. IoT'deki en büyük teknik zorluklar aşağıda listelenmiştir:

- Teknoloji
- Standardizasyon eksikliği
- Güvenlik
- Gizlilik
- Büyük veri yönetimi
- Birlikte çalışabilirlik

Şekil 2.2 [3] genel bir IoT ağ mimarisini göstermektedir. Cihazlar geçit veya baz istasyon vasıtasıyla buluta bağlanmışlar. Ağ geçidi, bulut ile kontrolörler, algılayıcılar ve akıllı cihazlar arasında bağlantı noktası görevi gören fiziksel bir cihaz veya yazılım programıdır. Buluttaki IoT sunucular farklı uygulamalar için veri toplama, işleme, kaydetme, bildirim ve yayınlama için geliştirilmiş standart veri yönetimi fonksiyonlarını donatır.



Şekil 2.2. Genel IoT ağ mimarisi

Öte yandan IoT’de birçok ağ teknolojisi kullanılmaktadır. Tablo 2.1 [67], kullanılan IoT teknolojileri, avantajları, dezavantajları ve örnek kullanım alanlarını göstermektedir.

Tablo 2.1. IoT’deki kullanılan iletişim teknolojilerin karşılaştırması.

Ağ	Bağlantı	Avantaj (+), Dezavantaj (-)	Popüler kullanım alanları
Ethernet	Kablolu, kısa menzil	+ Yüksek hız + Güvenli - Kısıtlı hareket imkânı	Video kameralar, oyun konsolları
Wifi	Kablosuz, kısa menzil	+ Yüksek hız + Mükemmel uyumluluk - Kısıtlı menzil - Yüksek güç tüketimi	Akıllı evler, kolayca şarj edilebilen cihazlar
NFC	Kablosuz, ultra kısa menzil	+ Güvenilirlik + Düşük güç tüketimi - Kısıtlı menzil - Müsaitlik eksikliği	Ödeme sistemleri, akıllı evler
Bluetooth	Kablosuz, kısa menzil	+ Yüksek hız + Düşük güç tüketimi - Düşük bant genişliği - Yüksek gecikme	Küçük ev aletleri, giyilebilir cihazlar
LPWAN	Kablosuz, uzun menzil	+ Uzun menzil + Düşük güç tüketimi - Düşük bant genişliği - Yüksek gecikme	Akıllı evler, akıllı şehirler, akıllı tarım
Zigbee	Kablosuz, kısa menzil	+ Düşük güç tüketimi + Ölçeklenebilirlik - Kısa menzil - Uyumluluk sorunları	Ev otomasyonu, sağlık ve sanayi siteleri
Hücreli ağlar	Kablosuz, uzun menzil	+ Neredeyse küresel kapsama + Yüksek hız + Güvenilirlik - Maliyetli - Yüksek güç tüketimi	Video ve görüntü gönderen dronlar

Tipik bir Kablosuz Algılayıcı Ağı (KAA), belirli fiziksel alanın koşullarını (Ses, basınç, hareket ve sıcaklık gibi) izlemek ve toplamak için entegre algılama teknolojisi ve kablosuz iletişim içerir. KAA'ler, askeri, endüstri ve sağlıktan çevresel ve kentsel kullanıma kadar düşük maliyetli ve hızlı dağıtım çözümleri sunmalarıyla ünlüdür. Zigbee (802.15.4) standardı, çoğu kablosuz algılayıcı ağı için temel altyapıdır. KAA, genellikle nesnelerin interneti sistemi içindeki bir teknolojidir. KAA düğümlerini internete bağlamak, kullanıcı avantajlarını artırır ve akıllı çevrimiçi uygulamaları güçlendirir. KAA'ler için geleneksel güvenlik mekanizmaları zayıftır veya etkin bir şekilde kullanılamaz.

IoT düğümleri ve veri merkezleri arasında aktarılan yüksek trafik yoğunluğuna bakıldığında, güvenlik sağlama adı verilen başka bir çaba var. Güvenli olmayan bir ağın güvenlik açıklarına ve istenmeyen hasarlara yol açabileceği açıktır. Verileri istenmeyen erişimlerden korumanın olası yollarından biri, onları güçlü algoritmalarla şifrelemektir. Algılayıcı ağlar farklı teknolojiler, güç zamanlama gereksinimleri, geçit (Gate) sayısı ve kriptografik fonksiyonların uygulanması ile ilişkilendirilir.

2.1. IoT Yaygın Mesajlaşma Protokolleri

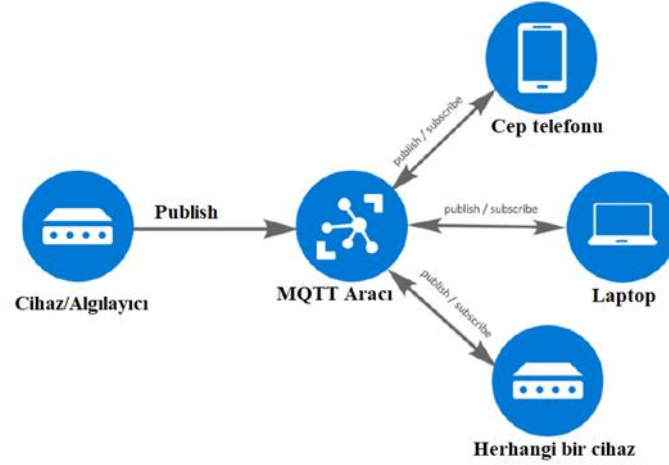
IoT cihazları, birbirleriyle iletişim kurmak için her katmanda çeşitli mesajlaşma ve iletişim protokolleri kullanır. Bir IoT cihazı oluştururken tipini, işlevselliğini ve katmanını akılda tutmak önemlidir. Piyasada çok çeşitli mesajlaşma ve iletişim protokolleri mevcuttur. Teknoloji uzmanları, IoT ekosistemlerine hizmet etmek için bir ağ oluştururken birden fazla iletişim protokolü arasından seçim yapılabilir. En yaygın olan mesajlaşma protokolleri aşağıda tartışılmaktadır.

2.1.1. MQTT

MQTT [68], hafif yayımla/abone ol mesajlaşma aktarımı nedeniyle yaygın olarak popülerlik kazanan bir IoT iletişim protokolüdür. Çeşitli makineler arasında veri aktarımı yapan bir protokoldür. Nesnelerin internetinde önemli protokollerden biri haline gelmiştir. Tüm cihazlar bir aracı (Broker) veya sunucu (Server) aracılığıyla iletişim kurar. İstemciler (Client), aracından veya yalnızca belirli konulara (Topic) abone olarak bilgi alır veya kabul eder. Ayrıca aracıya belirli konu mesajlarını da yayımlayabilirler. Tüm cihazlar, aracı aracılığıyla birbirleriyle iletişim kurar. Konular, gönderilebilecek mesajların türünü sınıflandırır. İstemciler belirli bir konuya abone olurlar ve yalnızca bu konulardan mesajlar alırlar. Aracı bu mesajları alır ve onları o konuya abone olan diğer cihazlara iletir. MQTT’de üç farklı QoS (Hizmet kalitesi) türü mevcuttur:

- ✓ QoS0: Aracıdan gelen herhangi bir geri bildirimden bağımsız olarak mesajlar bir kez gönderilir.
- ✓ QoS1: Aracıdan bir onay alana kadar mesajlar tekrar tekrar gönderilir.
- ✓ QoS2: Gönderilen her mesaj için aracı bir onay mesajı geri gönderir.

MQTT, boyut ve veri gücü iletimi açısından hafiftir ve bu nedenle çok sayıda cihazda mevcuttur. MQTT, verilerini esas olarak TCP/IP protokolü aracılığıyla iletir. Şekil 2.3 [68], MQTT haberleşme modelini göstermektedir.



Şekil 2.3. MQTT haberleşme modeli.

2.1.1.1. MQTT-SN

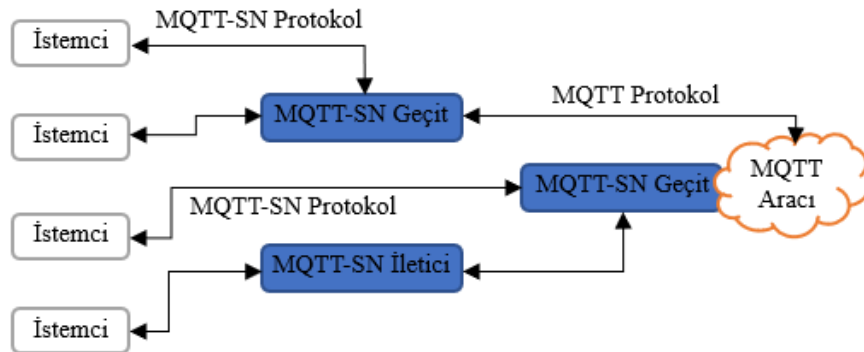
MQTT-SN (Algılayıcı ağları için MQTT) [69], kablosuz ağlarda çalışmak ve mümkün olduğunca MQTT ile aynı şekilde çalışmak üzere özel olarak tasarlanmıştır. Aynı yayınlama/abone olma modelini kullanır ve MQTT'nin bir sürümü olarak kabul edilebilir. MQTT-SN, taşıma protokolü UDP kullanmasına olmasına rağmen mümkün olduğunca MQTT ile aynı tasarıma sahiptir. Bu bakımdan MQTT-SN, mesaj gönderip alabilmesi için genellikle aracıyla bağlantı kurulmasını gerektirir. Bu bağlantı aslında sanal bir bağlantıdır. MQTT-SN mimarisi üç elemandan oluşmaktadır: MQTT-SN istemci, MQTT-SN geçit ve MQTT-SN iletilci (Forwarder). MQTT-SN istemcileri, MQTT-SN protokolünü kullanarak bir MQTT-SN geçit aracılığıyla kendilerini bir MQTT sunucusuna bağlar. MQTT-SN geçit, bir MQTT sunucusuyla entegre olabilir veya olmayabilir. Bağımsız bir geçit olması durumunda, MQTT sunucusu ile MQTT-SN geçit arasında MQTT protokolü kullanılmaktadır. Ana işlevi, MQTT ve MQTT-SN arasındaki çeviridir. MQTT ve MQTT-SN arasındaki temel farklar şunlardır:

- Mesaj yükünün boyutu MQTT-SN'de küçülmüştür.
- UDP'yi taşıma protokolü olarak kullanarak kalıcı bağlantı ihtiyacını ortadan kaldırma.

MQTT-SN'nin iki ağ geçidi (Gateway) türü vardır:

- Transparent (Şeffaf) bir ağ geçidi, her MQTT-SN bağlantısının karşılık gelen bir MQTT bağlantısına sahip olmasıdır. Bu, uygulanması en kolay türdür.

- Aggregation (Toplama) ağ geçidi, birden çok MQTT-SN bağlantısını tek bir MQTT bağlantısı ile paylaşmaktadır. Şekil 2.4 [69], MQTT-SN protokol mimarisini göstermektedir.



Şekil 2.4. MQTT-SN protokol mimarisi.

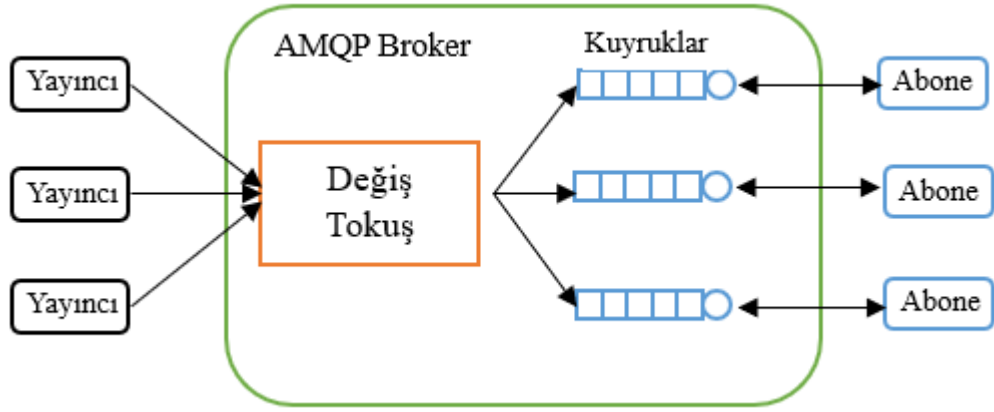
2.1.2. AMQP

Advanced Message Queuing Protocol (AMQP) [70], çeşitli uygulamalar ve şirketler arasında iş mesajlarını aktarır. Bu protokol, özellikle IOT uygulamaları için oluşturulmamıştır, ancak nesnelerin internetinde geniş bir kullanım alanına sahiptir. Ancak mesaj iletişiminin aktarılmasında etkili bir şekilde çalışır. AMQP, sistemi birbirine bağlar, gerekli bilgilerle sisteme hizmet eder ve gerekli hedeflere ulaşmak için bilgileri iletir. AMQP protokolünün tamamını yöneten üç bölüm vardır:

Exchange: Yayıncılardan gelen mesajları alır ve müsaitlik durumuna göre bu mesajları mesajlaşma kuyruklarına yönlendirir.

Message Queue: Bu mesajları bir uygulama tarafından kullanıma alınana kadar veri tabanlarında saklar.

Bağlama: Bir Exchange ve Message Queue arasındaki ilişkiyi ayırt eder ve mesaj yönlendirme kriterlerini verir. Şekil 2.5 [71], AMQP publish/subscribe mekanizmasını göstermektedir.

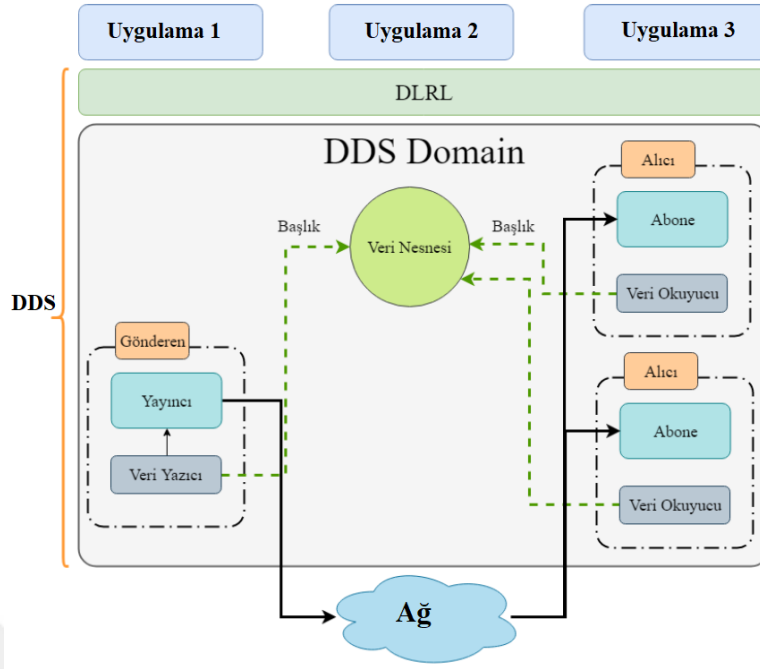


Şekil 2.5. AMQP publish/subscribe mekanizması.

2.1.3. DDS

Data Distribution Service (DDS) [72] protokolü, bir ağdaki veri tabanları ve kullanıcı uygulamaları arasında bir köprü görevi görür ve bu nedenle bir ara katman protokolüdür. Bu protokol, bir sistemin parçalarını bir araya getirir. Protokol düşük bilgi işlem verisi kullanır, oldukça verimli ve güvenilirdir ve mimarisi son derece genişletilebilirdir. DDS bir ara katman yazılımı olduğundan, görevi etkin iletişim ve kolay veri paylaşımını sağlamaktır. Yorumcu ve kafa karıştırıcı iletişim yollarını yönetme işini halleder ve geliştiricilerin uygulamaları oluşturmaya odaklanmasına olanak tanır.

DDS, asıl amacı M2M (Makineden makineye) iletişim olan hayati bir protokoldür. Veri alışverişi, yayımla-abone ol metodolojisi aracılığıyla gerçekleşir. DDS veri merkezlidir ve bu nedenle nesnelerin interneti teknolojisinde yaygın olarak kullanılmaktadır. Şekil 2.6 [72], DDS kavramsal modelini göstermektedir.

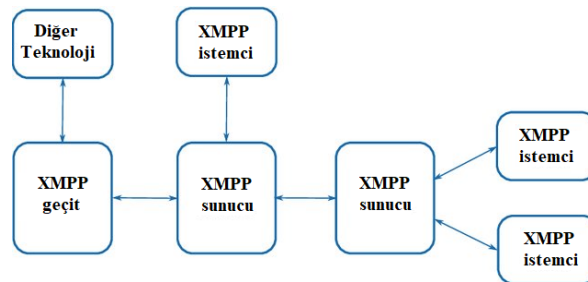


Şekil 2.6. DDS kavramsal model.

2.1.4. XMPP

Extensible Messaging and presence protocol (XMPP) [73] protokolü, ölçeklenebilirliği nedeniyle uzun mesafeli mesajlaşma için geçerlidir ve insan varlığını veya insan müdahalesini içerir. XMPP, genişletilebilir biçimlendirme dili olan XML'den kaynaklanır ve XML, web sayfaları oluşturmak için kullanılan bir protokol olan HTML'den kaynaklanır. Her ikisi de işaretleme dilleridir.

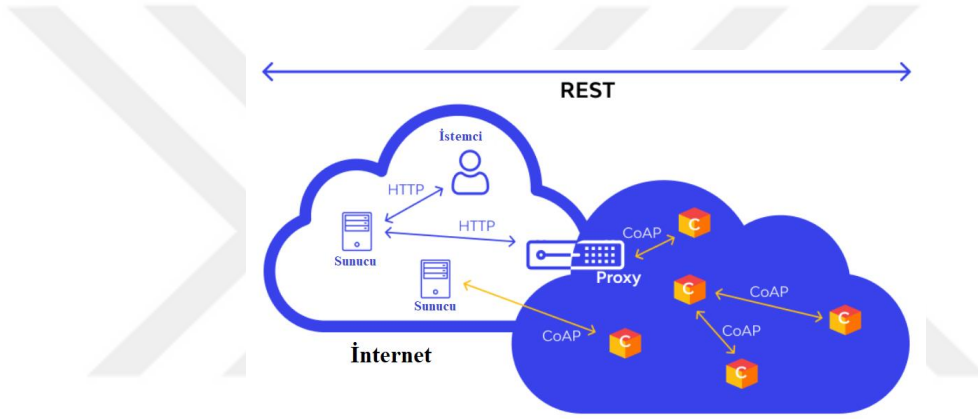
XMPP, genişletilebilirliği nedeniyle birbirleriyle iletişim kurabilen geniş bir kullanım alanına sahiptir. Standart internet iletişim protokolünü kullanır ve bu onu evrensel olarak iletilebilir kılar ve ayrıca HTTP üzerinden iletişim kurar. XMPP ağları, diğer protokollere bağlanmak için ağ geçitleri içerir. TCP bağlantısı üzerinden anlık mesaj aktarımını destekleyecek şekilde tasarlanmıştır. Şekil 2.7 [73], XMPP temel mimarisini göstermektedir.



Şekil 2.7. XMPP temel mimarisi

2.1.5. CoAP

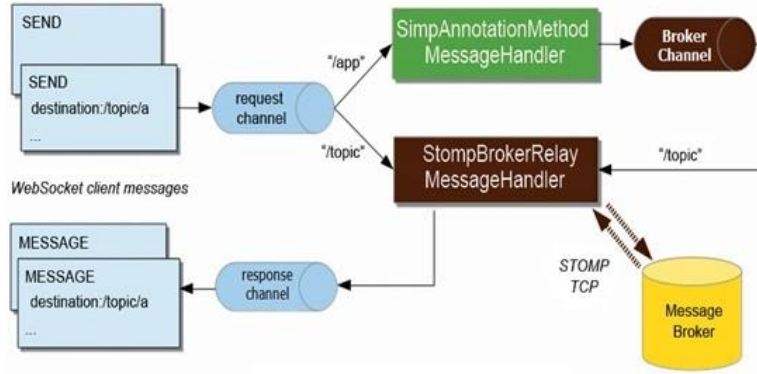
Constrained Application Protocol (CoAP) [74], UDP üzerinde çalışır ve bu nedenle bağlantısız bir protokoldür. Verimli bir protokoldür ve 4 baytlık bir başlığı (Header) vardır. CoAP, HTTP'ye benzer istek/yanıt protokolünü sahiptir. Her istek belirli bir yanıt üretmelidir. HTTP'ye benzer şekilde CoAP, GET, POST ve PUT yöntemlerini destekler. HTTP'den farklı olarak CoAP, bu yöntemleri UDP ile eş zamansız olarak işler. CoAP, HTTP ara yüzünü kolaylaştıracak ve aynı zamanda kısıtlı düğümler ve ağlarla kolayca uyumlu olacak şekilde oluşturulmuştur. CoAP, 4 çeşit mesajı destekler: Confirmable, non-confirmable, acknowledgement ve rest. Confirmable ve non-confirmable mesajlar, isteği ve yanıtı aktarır. Şekil 2.8 [74], CoAP örnek haberleşme mimarisini göstermektedir.



Şekil 2.8. CoAP haberleşme mimarisi.

2.1.6. STOMP

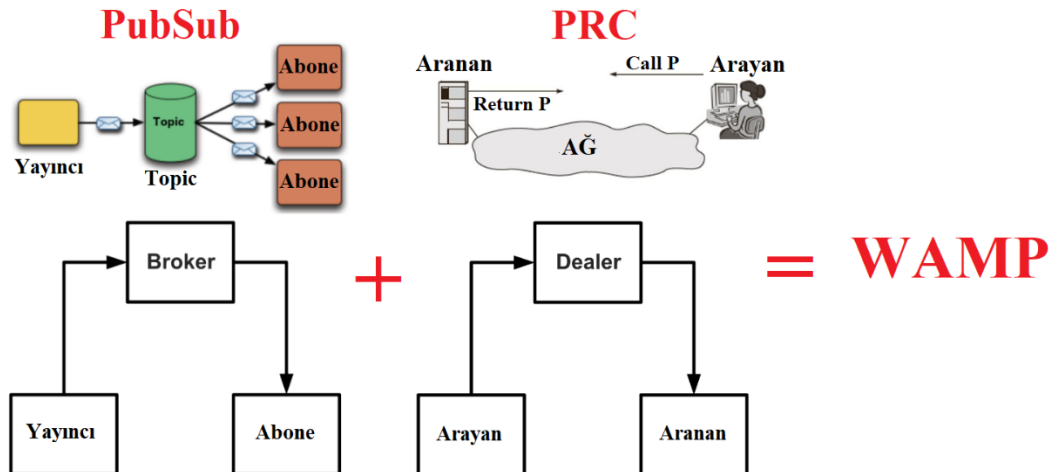
Simple (Or Streaming) Text Orientated Messaging Protocol (STOMP) [75] önceden eskiden Text Oriented Messaging Protocol (TTMP) olarak da bilinen adıyla Message-Oriented Middleware (MOM) ile çalışmak için tasarlanmış basit bir metin tabanlı Protokoldür. STOMP İstemcilerinin protokolü destekleyen herhangi bir mesaj aracı ile konuşmasına izin veren birlikte çalışabilir bir Wire Protocol formatı sağlar. Protokol genel olarak http yapısına benzer bir yapıya sahiptir ve TCP üzerinden çalışır. İstemci-sunucu arasındaki iletişim, birkaç satırdan oluşan bir "çerçeve" aracılığıyla gerçekleşir. Şekil 2.9 [76], örnek STOMP haberleşme mimarisini göstermektedir.



Şekil 2.9. STOMP haberleşme mimarisi.

2.1.7. WAMP

Web Application Messaging Protocol (WAMP) [77], açık uygulama düzeyinde bir protokoldür. İki çeşit mesajlaşma modeli sağlamaktadır: Routed Remote Procedure Calls (PRC) ve yayımcı (Publish) & abone (Subscribe). Mesaj kodlaması için farklı sertleştiriciler kullanır. WebSocket alt protokolü, Raw TCP Socket, Unix Domain soketi gibi farklı aktarımlar üzerinden çalışabilir. WAMP, bileşen ve mikro hizmet tabanlı uygulamaların tüm mesajlaşma gereksinimleri için kullanılır, teknoloji yığını karmaşıklığını ve ek yükü azaltır, uygulamaların güvenebileceği yetenekli ve güvenli bir temel sağlar. Şekil 2.10 [77], WAMP mesajlaşma mimarisini göstermektedir.

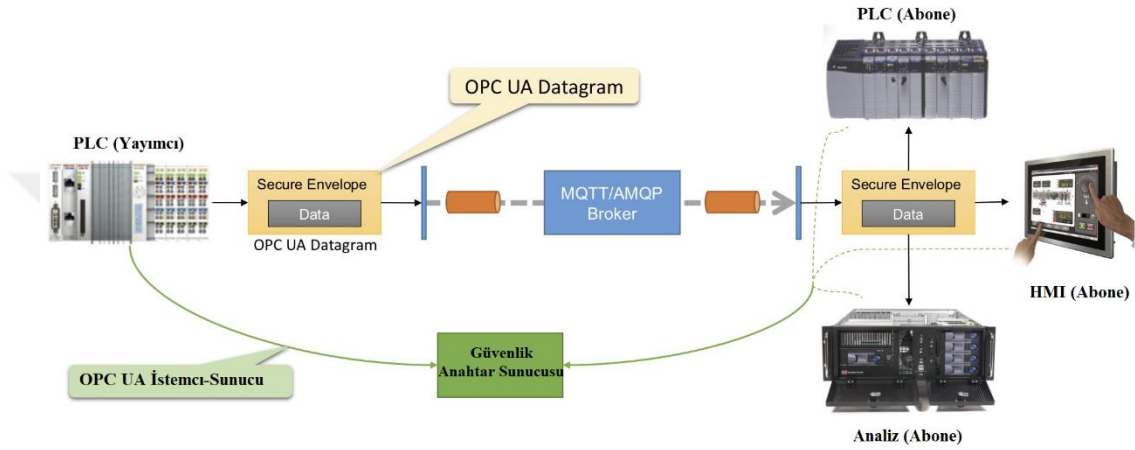


Şekil 2.10. WAMP mesajlaşma mimarisi.

2.1.8. OPC UA

OPC Unified Architecture (OPC UA) [78], endüstriyel otomasyon için kullanılan ve OPC Foundation tarafından geliştirilen bir makineden makineye iletişim protokolüdür.

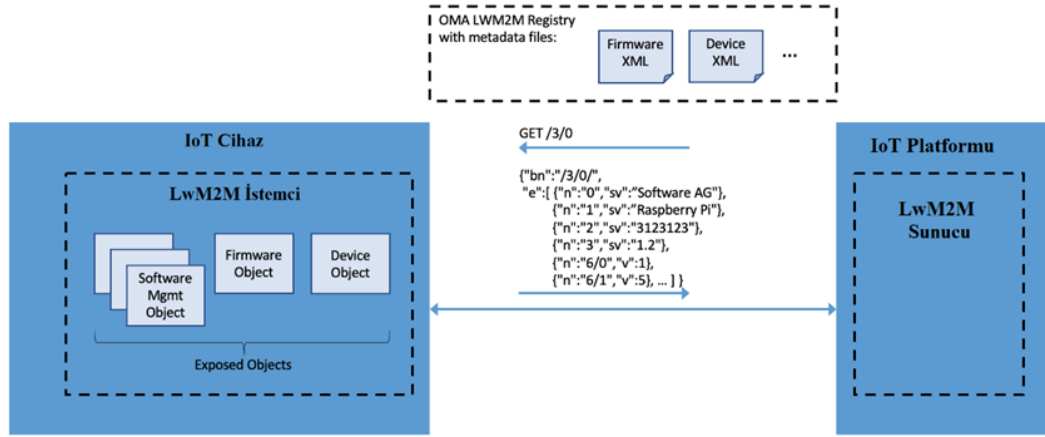
OPC UA, bireysel OPC Classic özelliklerini genişletilebilir bir çerçeveye entegre eden, platformdan bağımsız hizmet odaklı bir mimariye sahiptir. OPC UA, güvenli ve platformdan bağımsız bir standart kullanarak tüm cihazları, otomasyon sistemlerine ve yazılım uygulamalarına entegre edebilmek için endüstriyel bağlantıyı basitleştirir. OPC UA'yı bulut aracılığıyla IoT'ye bağlayarak cihaz tanılama, varlık yönetimi, izleme, raporlama ve diğer uygulamaların çalışmasını mümkün kılar. Şekil 2.11 [78], OPC UA örnek haberleşme diyagramını göstermektedir.



Şekil 2.11. OPC UA örnek haberleşme diyagramı.

2.1.9. LwM2M

Lightweight M2M standard (LwM2M) [79], Open Mobile Alliance tarafından geliştirilen LwM2M hala oldukça genç bir protokoldür. LwM2M ilk olarak Şubat 2017'de tanıtıldı. LwM2M, CoAP'a dayalıdır ve nesnelerin nternetinde cihaz yönetimini basitleştirmeyi ve standartlaştırmayı amaçlar. Bu nedenle yalnızca mesajlaşma için tasarlanmamıştır, aynı zamanda cihaz yönetimi için de öğeler içerir. Cihaz ve platform arasındaki bilgi alışverişi için LwM2M, ilgili cihaz üreticisinden bağımsız standartlaştırılmış bir veri modeli kullanır. Model hiyerarşik olarak üç nesne düzeyine bölünmüştür: Nesne, nesne örneği ve kaynak. Şekil 2.12 [79], LwM2M örnek iletişim modelini göstermektedir.



Şekil 2.12. LwM2M örnek iletişim modeli.

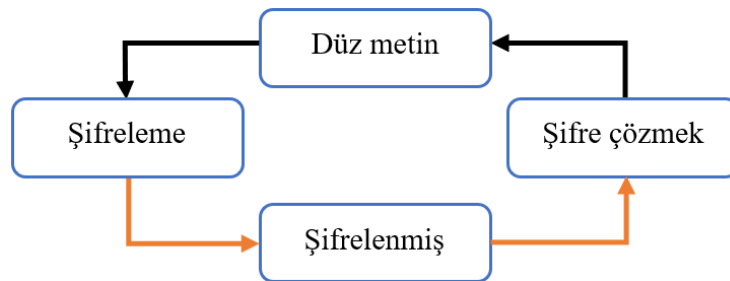
Tablo 2.2, IoT 'de yaygın olan mesajlaşma protokoller karşılaştırmasını kullanılan amaç, mimariler, özellikler, tanımlanmış roller, ve keşif açısından karşılaştırmaktadır.

Tablo 2.2. IoT 'de yaygın olan mesajlaşma protokoller karşılaştırması.

Protokol	Ana amaç	Tanımlanmış roller	Mimariler	Keşif	Gerçek zamanlı	Spesifik özellikler
MQTT	M2M ve IoT küçük kod ayak izi, sınırlı bant genişliği ve yüksek gecikmeli ağlar	Yayımcı, abone, broker	İstemci-broker-istemci	Konular	Kısmi	Güvenilir kuyruk
MQTT-SN	Kısıtlı cihazlar	Yayımcı, abone, broker	İstemci-broker-istemci	Konular, ağ geçidi	Kısmi	Sınırlı yük boyutu
CoAP	Kısıtlı cihazlar, Kayıplı ağlar	Sunucu, istemci, pub/subm mekanizma	P2P, istemci-sunucu, master-slave	Kaynaklar	Evet	RESTful
STOMP	Basit veri alışverişi	Yayımcı, abone, sunucu	İstemci-m-sunucu-istemci	--	Kısmi	Uygulanması kolay
XMPP	Genelleştirilmiş XML verilerinin yönlendirmesi	İstemci, sunucu	İstemci-m_router-istemci	İstemciler	Sınırlı	Kullanıcının listesi ve iletişim durumu
WAMP	Mesajlaşma ve RPC	Yayımcı, abone, broker, aranan, arayan, dealer	İstemci-m_router-istemci, İstemci-RPC_router-istemci	Konular ve RPC	Evet veya kısmen	Farklı serileştiriciler kullanır
AMQP	Kurumsal ortamlar	Yayımcı, abone, broker	İstemci-m_router-istemci, İstemci-broker-istemci	Veri alışverişi	Kısmi	Tek bağlantıda birden çok bağlantı
LwM2M	Genel M2M iletişimi	İstemci, sunucu	İstemci, sunucu	Cihazlar	Evet veya kısmen	İletişim modeli CoAP yöntemlerine dayanmaktadır
OPC UA	Endüstriyel uygulamalar	İstemci, sunucu, yayımcı, abone	İstemci-sunucu, yayımcı- abone, sunucular-aggregatorclients, p2p,	Uygulama profilleri, nesne yöntemleri ve değişkenler	Evet veya kısmen	Alana özgü bilgi modelleri

3. ŞİFRELEME ALGORİTMALARI

Son zamanlarda uygarlığın ilerlemesi ile tüm dünyamız internete ve uygulamalarına (İletişim, veri iletimi, dosyalar, videolar vb.) büyük ölçüde bağımlıdır. Bu verilerin internet üzerinden iletilmesi sırasında, verilerin kopyalanmasını ve bu verilerin izinsiz veya yetkisiz kişiler tarafından yeniden dağıtılmasını önlemek için koruma yöntemlerine ihtiyaç vardır. Ağ güvenliği, verileri güvenli ve saldırılara karşı dirençli hale getirmek için mesaj dönüştürme bilimi olan kriptografi ile gerçekleştirilebilir. Bu yöntem, bir mesajın içeriğinin çok gizli bir şekilde iletilmesini ve değiştirilmemesini sağlamak için kullanılır. Şifrelemenin ana konsepti, verileri gizli kodla kodlamak ve eğer ele geçilirse bile bilgisayar korsanları veya yetkisiz kişiler için anlaşılabilir hale getirmektir. Kriptografi, milyarlarca insan tarafından, çoğu kullandıklarının farkında olmasalar bile, verileri ve bilgileri korumak için evrensel olarak düzenli olarak kullanılmaktadır. Kriptografinin tarihsel kökleri, eski Mısırlıların “gizli” hiyeroglifleri kullandığı MÖ 2000 yılına kadar uzanmaktadır. Kriptografinin diğer tarihsel kanıtları, antik Yunanistan'daki gizli yazıları veya antik Roma'nın ünlü Sezar şifresini içerir. Şifreleme uygulamalar bilgisayar şifrelerinde, finansal işlemlerde, e-ticaret ve birçok farklı ortamda kullanılmaktadır. Kriptografi, verileri yalnızca hırsızlıktan veya değişiklikten korumakla kalmaz, aynı zamanda kullanıcı doğrulaması için de kullanılabilir. Şekil 3.1 [80], kriptografi mekanizmasının genel işleyişini göstermektedir.



Şekil 3.1. Kriptografi prosedürü.

Yaygın olarak kullanılan üç tür şifreleme planı vardır: Gizemli anahtar şifreleme (Simetrik), açık anahtar (Asimetrik) şifrelemesi ve karma (Hash) çeşitleri. Kriptografide kullanılan terimler Tablo 3.1’de [80] açıklanmıştır.

Tablo 3.1. Kriptografide kullanılan terimler.

Terim	Açıklaması
Anahtar	Sayısal veya alfanümerik bir metindir veya benzersiz bir Şekil olabilir.
Düz metin	Gizlice göndermek istediğimiz asıl mesajdır. Düz metin herkesin anlayabileceği bir biçimde yazılır. Mesajın internet üzerinden hedeflenen alıcılara gönderilmeden önce bazı güvenlik önlemlerinin alınması gerekir.
Şifreli metin	Kimsenin anlayamayacağı mesaj veya amaçsız bir mesaj, şifreli içerik dediğimiz şeydir.
Şifreleme	Şifreleme, düz metni, anlamsız ve rastgele bir bit dizisi gibi görünen şifreli metne dönüştürme işlemidir.
Şifre çözmek	Şifreli metni tekrar düz metne dönüştürme işlemine şifre çözme denir.
Kriptanaliz	Kriptografinin nasıl tersine çevrileceğinin incelenmesidir. Kriptosistem bilgisi olmadan mesajların şifresini çözmekle ilgilenir.
Kriptoloji	Kriptanaliz ve kriptografi çalışması kriptoloji olarak bilinir.
Kriptosistem	Şifreleme veya şifre çözme algoritması uygulayarak düz metni şifreli metne veya şifreli metni düz metne dönüştüren bir sistemdir. Şifreleme ve şifre çözme algoritmaları için anahtar üretimi de bir şifreleme sisteminin parçasıdır.
Cipher	Şifreleme sisteminde kullanılan bir algoritmadır.

3.1. Kriptografi Çeşitleri

3.1.1. Simetrik Anahtar Şifreleme

Simetrik anahtar şifrelemesi, mesajları şifrelemek ve şifresini çözmek için gönderici ve alıcı tarafından tek ortak anahtarın kullanıldığı bir şifreleme türüdür. Bu sistem aynı zamanda özel veya gizli anahtar kriptografisi olarak da bilinir. AES (Advanced Encryption System), yaygın olarak kullanılan bir simetrik anahtar şifreleme yöntemidir. Simetrik anahtar sistemi, şifreleme ve şifre çözme işlemi için yalnızca tek bir anahtar olduğundan, iki tarafın gizli bir şekilde anahtarı değiş tokuş etmesi gerektiği gibi önemli bir dezavantaja sahiptir. DES, Triple DES, RC2, RC4, RC5, IDEA, Blowfish, Stream cipher, vb. yaygın olarak kullanılan simetrik anahtar şifreleme tekniklerinden bazılarıdır [80].

3.1.2. Asimetrik Anahtar Şifreleme

Aynı zamanda açık anahtarlı şifreleme olarak da adlandırılır. Bilgi aktarımı için farklı türde bir koruyucu yöntem kullanır. Bu sistemde her kullanıcı şifreleme ve şifre çözme işlemi için iki anahtar veya bir çift anahtar (Özel anahtar ve genel anahtar) kullanır. Özel anahtar her kullanıcıyla gizli tutulur ve genel anahtar ağ üzerinden dağıtılır, böylece herhangi bir kullanıcıya mesaj göndermek isteyen herkes bu ortak anahtarları kullanabilir. Açık anahtar herkes tarafından bilinse bile, amaçlanan alıcı, yalnızca özel anahtarı bildiği için kodu çözebilir. Simetrik anahtardan daha güvenlidir. RSA, en yaygın olarak kullanılan asimetrik anahtar şifrelemesidir. DSA, PKC'ler, Eliptik Curve teknikleri vb. yaygın asimetrik anahtar şifreleme türleridir [80].

3.1.3. Hash

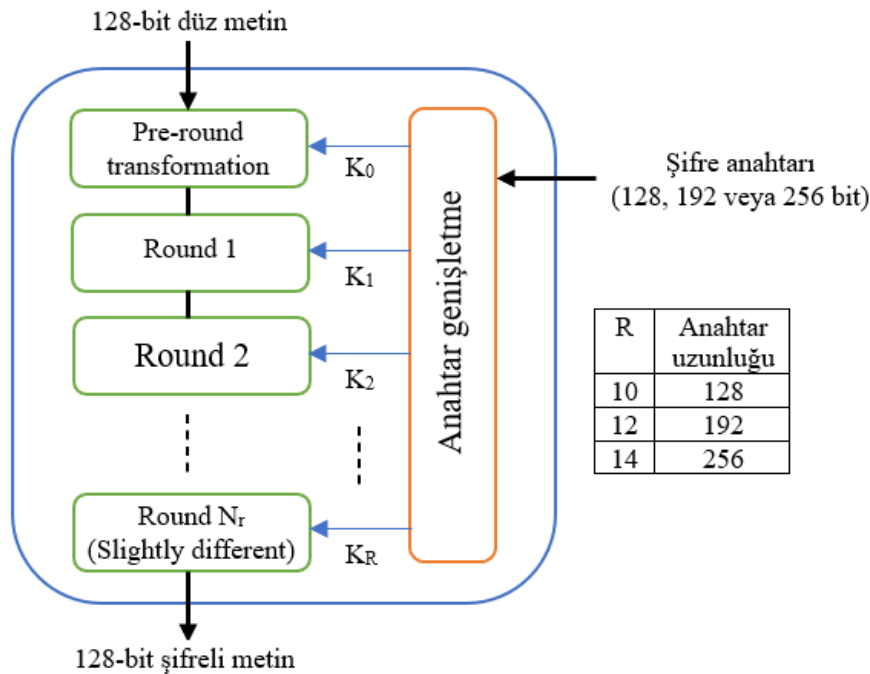
Hashing, herhangi bir rastgele boyuttaki veriyi bir hash fonksiyonu kullanarak sabit uzunluklu bir değere eşleme işlemidir. Teknik olarak, hashing tersine çevrilebilir, ancak şifresini çözmek için gereken hesaplama gücü, şifre çözmeyi imkânsız hale getirir. Hashing'in kullanımlarından biri, büyük miktarda veriyi karşılaştırmaktır. Hash değerlerinin karşılaştırılması, daha özlü oldukları için büyük veri yığınlarından çok daha kolaydır. Hash, dijital imzalarda ve veri tabanlarında da verilerin tekrarlanmasını önlemek için rastgele dizeler oluşturmak için kullanılır. Hash işleminin tersine çevrilmesi son derece olanaksız olduğundan, parolalarda hash algoritmaları kullanılacak, parolayı daha kısa ve saldırganlar tarafından keşfedilemez hale getirir [80].

3.2. Hafif Siklet Şifreleme Algoritmaları

Hafif şifreleme, bellek kullanımını, güç tüketimini optimize etmeye ve yeterli düzeyde güvenlik sağlamaya odaklanan tercih edilen bir şifreleme yöntemidir. Hafif şifreler donanım ve yazılım platformlarında uygulanabilir. İlki (Donanım), çalışan döngülerin ve bellek alanı sayısını azaltmaya odaklanır ve diğeri (Yazılım) enerji tüketimini ve bellek kullanımını azaltmayı hedefler. Güç ve hız açısından donanım çözümü daha iyi sonuçlar sağlar. Literatürü araştırdığımızda, üç grup hafif siklet şifre görebiliriz; Substitution-Permutation Network (SPN), Feistel Ağları ve diğer tasarımlar. Feistel, SPN'nin asimetrik yapısına karşı simetrik (Şifreleme ve şifre çözme için aynı anahtar) kullanır. SPN'nin round fonksiyonu (F) tersine çevrilebilir olmalıdır ve Feistel ağının

aksine nispeten daha yüksek güvenlik sağlar. Diğer yandan, SPN daha fazla kaynak tüketmektedir. Bahsedilen şifrelerin doğal özelliklerinin birleştirilmesi, hybrid (Karışık) adı verilen yeni blok şifre kategorilerinin ortaya çıkmasına neden olmuştur. Bellek alanı (Footprint), hafif bir blok şifresi tasarlamamanın en büyük zorluklarından biridir, bu nedenle, blok şifrelerinin çoğunda S-Box yoktur veya küçük (Örneğin 4-bayt) S-Box kullanılmıştır. S-Box, saldırılara karşı direnmede kilit rol oynar. Bu bölümde, tezde kullanılan hafif siklet şifrelerin kısa bir incelemesini sunulmuştur. Tablo 3.2, hafif siklet şifreler farklı kriterler açısından karşılaştırmaktadır.

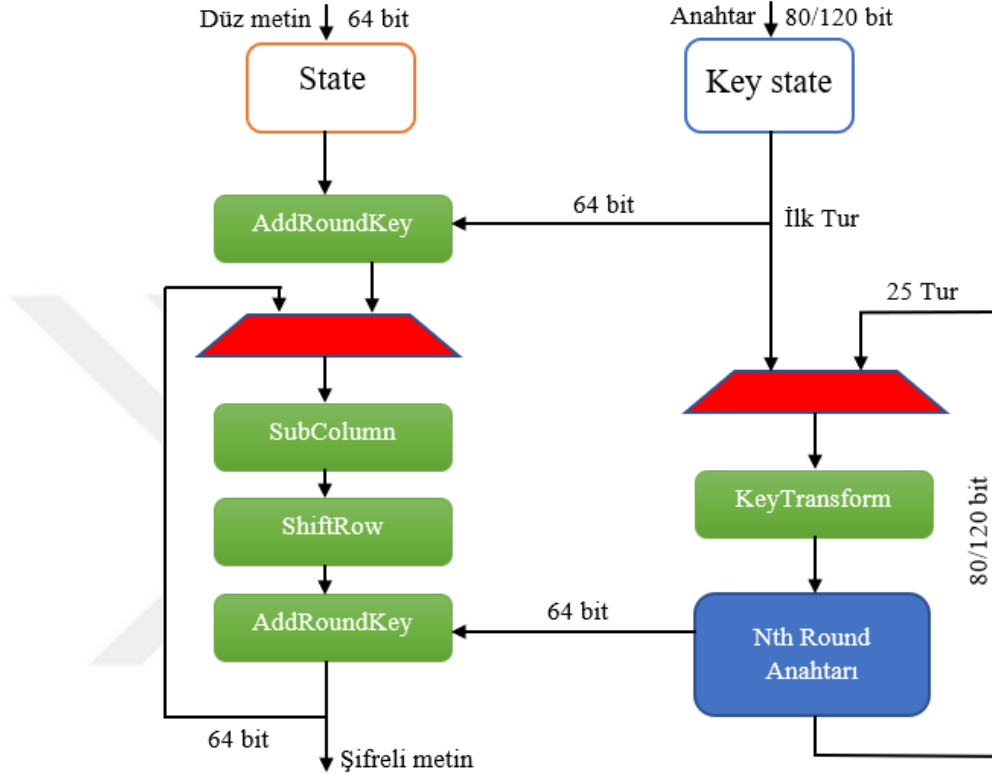
AES [81]- Farklı şifreleme paketlerinde bulunan bir SPN şifresidir. Bu şifrenin S-Box dörde dört bir matristir. Anahtar genişletmesinden sonra, tercih edilen anahtar uzunluğuna bağlı olarak, düz metin üzerinde round fonksiyonu uygular ve dört işlemden oluşmaktadır; SubBytes (Dört baytlık bir sözcük, SubByte dönüş değeridir), ShiftRows (Durum satırları, farklı uzaklıklar üzerinden çevrimsel olarak kaydırılır.), MixColumns (Durum sütunları, GF (28) üzerinden polinomlar olarak ifade edilir ve bir sabitle modulo $x^4 + 1$ ile çarpılır. polinom $c(x)$) ve AddRoundkey (Basit bir bit düzeyinde EXOR kullanarak, duruma bir round anahtarı uygulanır). Şekil 3.2 [81] AES şifreleme şemasını göstermektedir.



Şekil 3.2. AES şifreleme şeması.

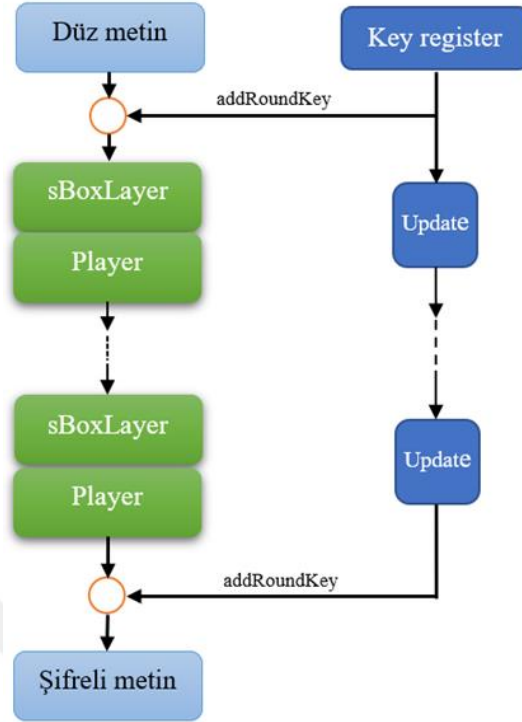
RECTANGLE [9]- Donanım ve yazılım ortamlarına uygundur ve SPN yapısına sahiptir. Bu şifrede şu adımları içeren 25 tur vardır: AddRoundKey (Ara duruma

uygulanan basit bir bitisel-XOR (Yuvarlak alt anahtarın), SubColumn (Aynı sütundaki dört bite S-Box paralel uygulaması), ShiftRow (Farklı ofsetler üzerinden her satıra bir sola döndürme uygulanmaktadır). On iki temel mantıksal işlem dizisi bu şifre için S-Box oluşturmaktadır. P-katmanı üç rotasyondan oluşmaktadır. Şekil 3.3 [9] RECTANGLE şifreleme işlemini gösteriyor.



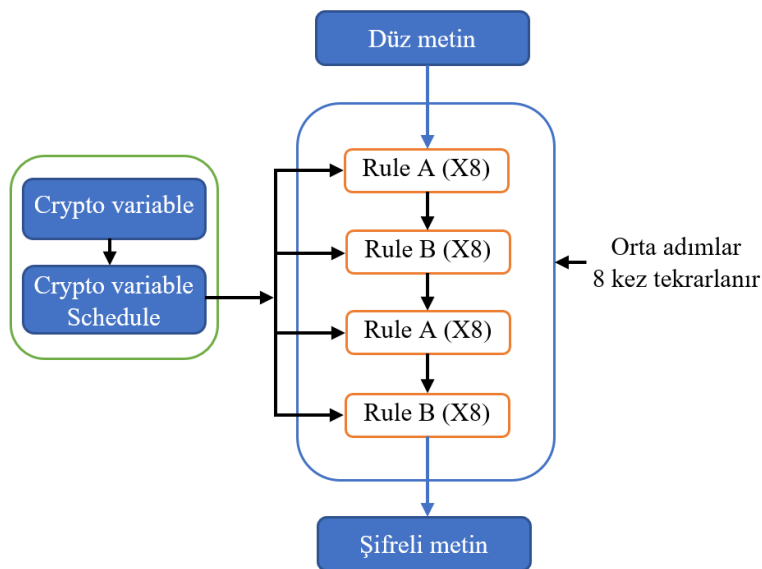
Şekil 3.3. RECTANGLE şifreleme işlemi.

PRESENT [82]- Altyapısından ilham alan birçok blok şifre vardır. Bit odaklı bir SPN yapısıdır. Round anahtarlar oluşturduktan sonra, algoritma her bir 31 tur için şu adımları uygular: addRoundKey, sBoxLayer, pLayer. Dört bite dört bit S-Box kullanır ve donanım platformlarında kullanılması önerilir. PRINCE; FX yapısına dayanmaktadır ve bu algoritmanın anahtar zamanlama mekanizması şu şekilde tanımlanmaktadır; 64 bitlik anahtarlarından ikisini, 128 bitlik whitening anahtardan elde eder ve şifreleme sırasında üçüncü anahtarı dahili durumda XOR'lanır. Her tur şu adımları birleştirir: Anahtar ekleme, bir S-Box katmanı, doğrusal katman, round sabit ekleme. Şekil 3.4 [82] PRESENT hafif siklet şifresinin temel şifreleme şemasını göstermektedir.



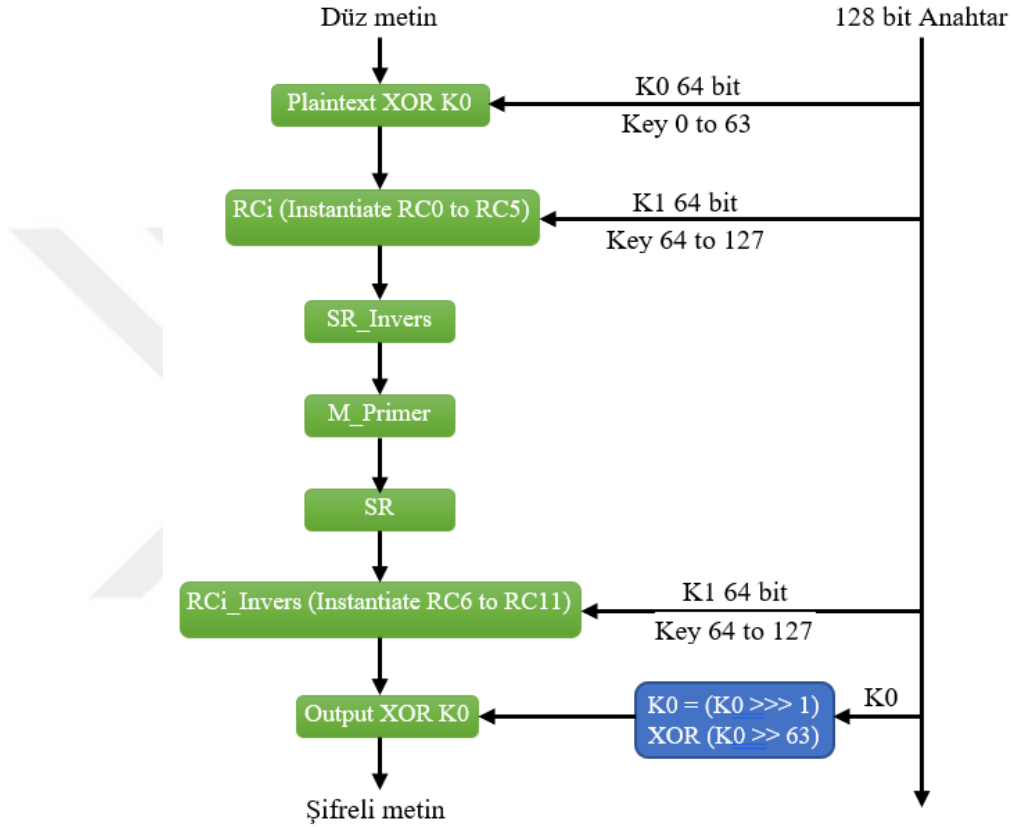
Şekil 3.4. PRESENT hafif siklet şifresinin temel şifreleme şeması.

Skipjack [65]- 80 bitlik anahtar uzunluğundan dolayı yaygın saldırılara karşı zaaf göstermektedir (Anahtar boyutu nedeniyle). Skipjack'in blok uzunluğu 64 bit ve tur sayısı 32'dir ve dengesiz (Unbalanced) bir Feistel ağı olarak sınıflandırılır. 8x8 S-Box sahip basit bir şifreleme algoritmasıdır. Şekil 3.5 [65] Skipjack şifreleme işlemini gösteriyor.



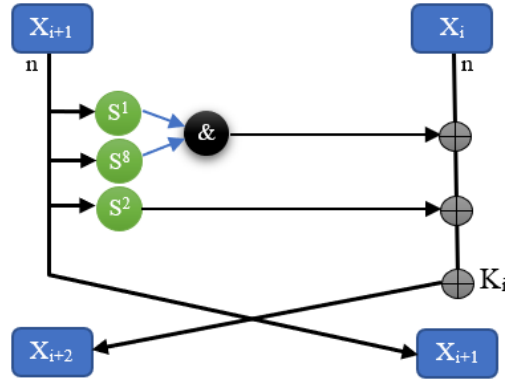
Şekil 3.5. Skipjack şifreleme işlemi.

PRINCE [83]- FX yapısına dayanmaktadır ve bu algoritmanın anahtar zamanlama mekanizması şu şekilde tanımlanır; 64 bitlik anahtarlarından ikisini, 128 bitlik whitening anahtardan elde eder ve şifreleme sırasında üçüncü anahtarı dahili durumda XOR'lanır. Her tur şu adımları birleştirir: Anahtar ekleme, bir S-Box katmanı, doğrusal katman, round sabit ekleme. Şekil 3.6 [83] PRINCE algoritmanın şifreleme işlemini göstermektedir.



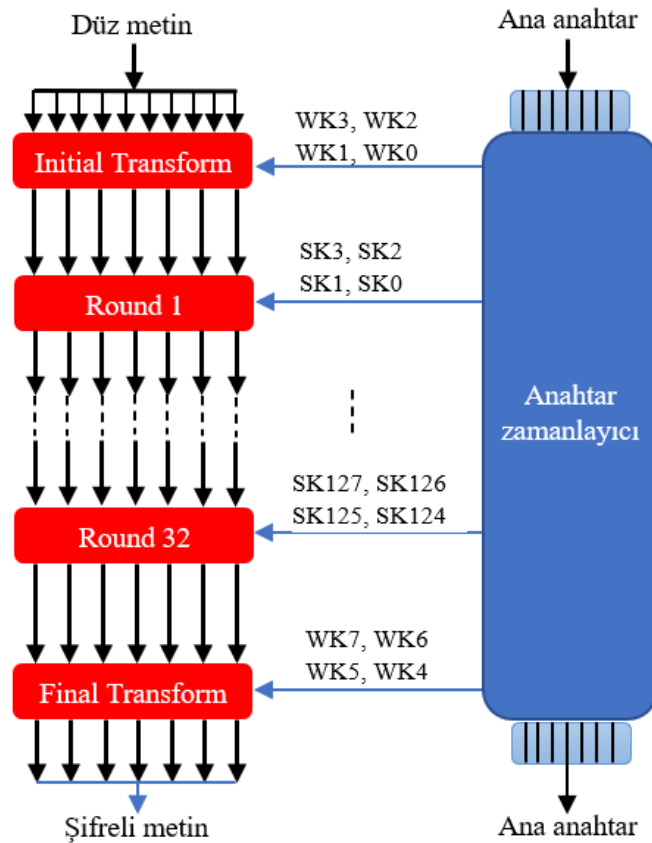
Şekil 3.6. PRINCE şifreleme işlemi.

SIMON [84]- Dengeli bir Feistel yapısına sahiptir ve donanım odaklı sistemlerde kullanım için optimize edilmiştir. Bitsel XOR, bitsel AND ve sola dairesel kaydırma round fonksiyonu oluşturmak için kullanılmaktadır. Anahtar zamanlaması dengeli olabilir veya olmayabilir. Şekil 3.7 [84] SIMON round fonksiyon yapısını göstermektedir.



Şekil 3.7. SIMON round fonksiyonu.

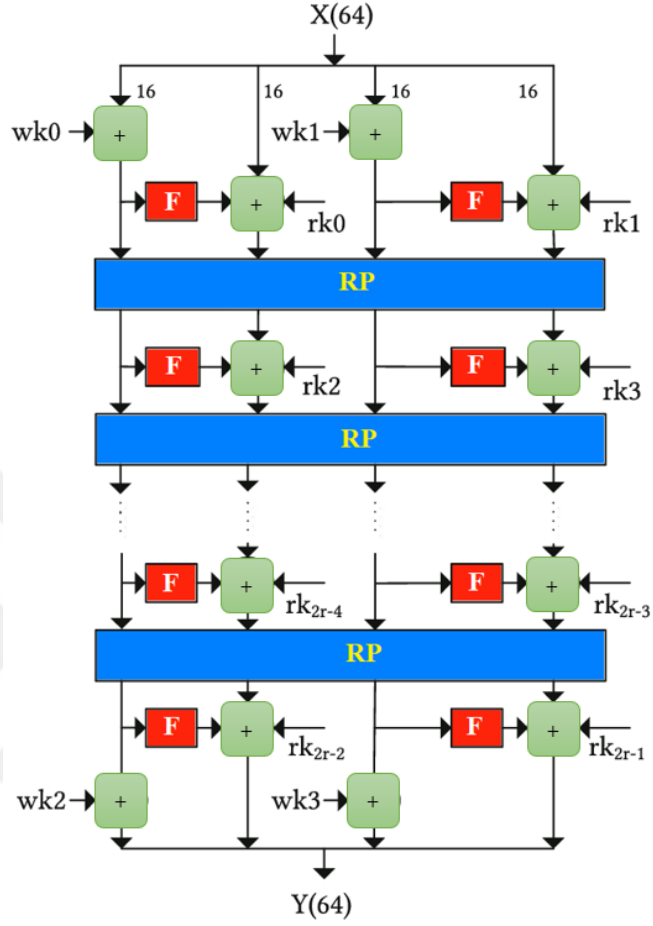
HIGHT [85]- ARX tabanlı geliştirilmiş bir Feistel ağıdır. HIGHT, basit işlemlere dayanmaktadır: XOR, ek mod 28, bit yönünde döndürme (Soldan). Anahtar zamanlaması için iki algoritma vardır: WhiteningKeyGeneration ve SubKeyGeneration. Donanım ve yazılım platformları için uygundur. Şekil 3.8 [85] HIGHT algortimanın şifreleme şemasını göstermektedir.



Şekil 3.8. HIGHT şifreleme şeması.

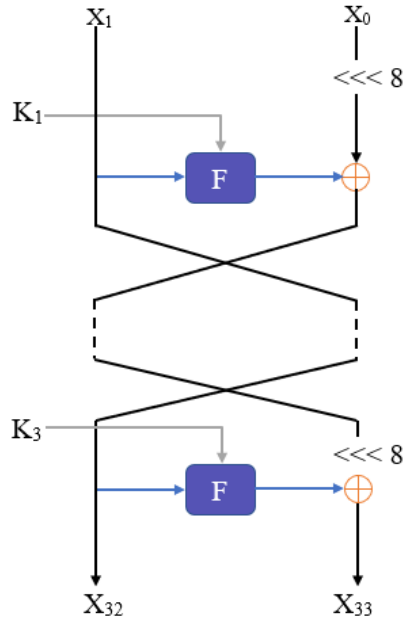
Piccolo [11]- Donanım odaklı bir blok şifredir ve anahtar boyutları Piccolo-80 ve Piccolo-128'de mevcuttur. Difüzyon tabakası için özel yarım kelime permütasyon ve

whitening tekniği kullanmaktadır. 4x4 S-Box'a sahiptir ve sonraki işlemleri kullanılarak uygulanmaktadır: Dört NOR, üç XOR ve bir XNOR geçit. Şekil 3.9 [11] Piccolo algortmanın şifreleme şemasını göstermektedir.



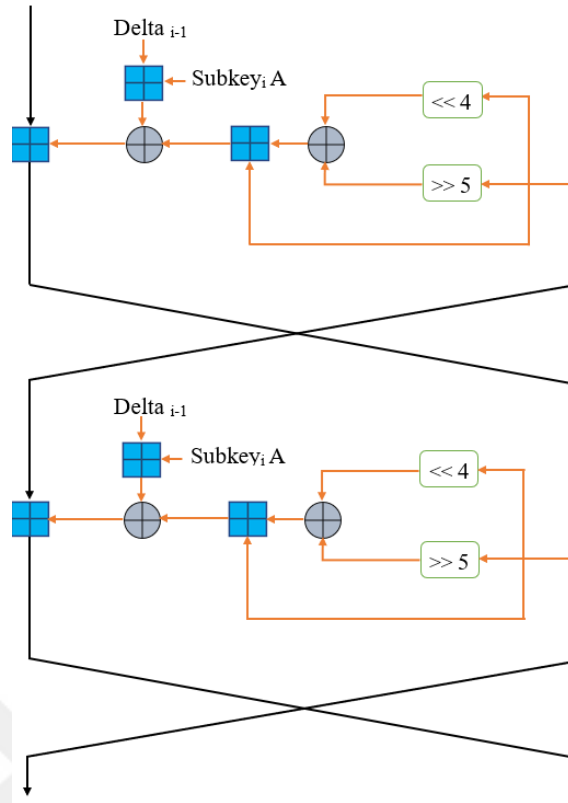
Şekil 3.9. Piccolo şifreleme şeması.

LBlock [86]- Yazılım ve donanım ortamlarında uygulama yeteneğine sahiptir. LBlock'ta 4x4 bir S-Box var. Anahtar zamanlama mekanizması iki ek S-Box uygulamaktadır. Şifreleme işlemi 32 turlu yinelenmeli bir yapı içerir ve şifre çözme, şifrelemenin ters işlemidir. Şekil 3.10 [86] LBlock şifreleme işlemi göstermektedir.



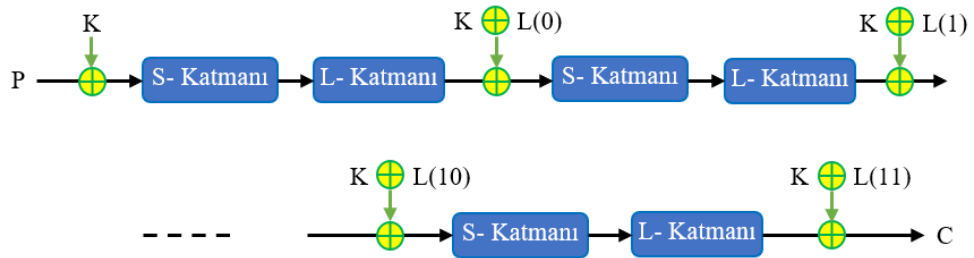
Şekil 3.10. LBlock şifreleme işlemi.

TEA ve halefi XTEA [11]- Feistel mimarisine sahiptir. Her iki Şifre de 128 bit anahtar uzunluğuna ve 64 bit blok uzunluğuna sahiptir. TEA, karmaşık olmayan bir anahtar zamanlama mekanizmasını temsil eder. XTEA ile karşılaştırıldığında, shift işlemlerin yeniden düzenlenmesi, XOR'ler ve ekleme işlemleri gibi bazı iyileştirmeler yapılmıştır. Ayrıca, bunun için daha karmaşık bir anahtar zamanlama mekanizması kullanılmıştır. Her ikisi de 128 bit anahtar uzunluğu ve 128 bit blok uzunluğuna sahiptir. Şekil 3.11 [11] XTEA'de bir tur şifreleme şemasını göstermektedir.



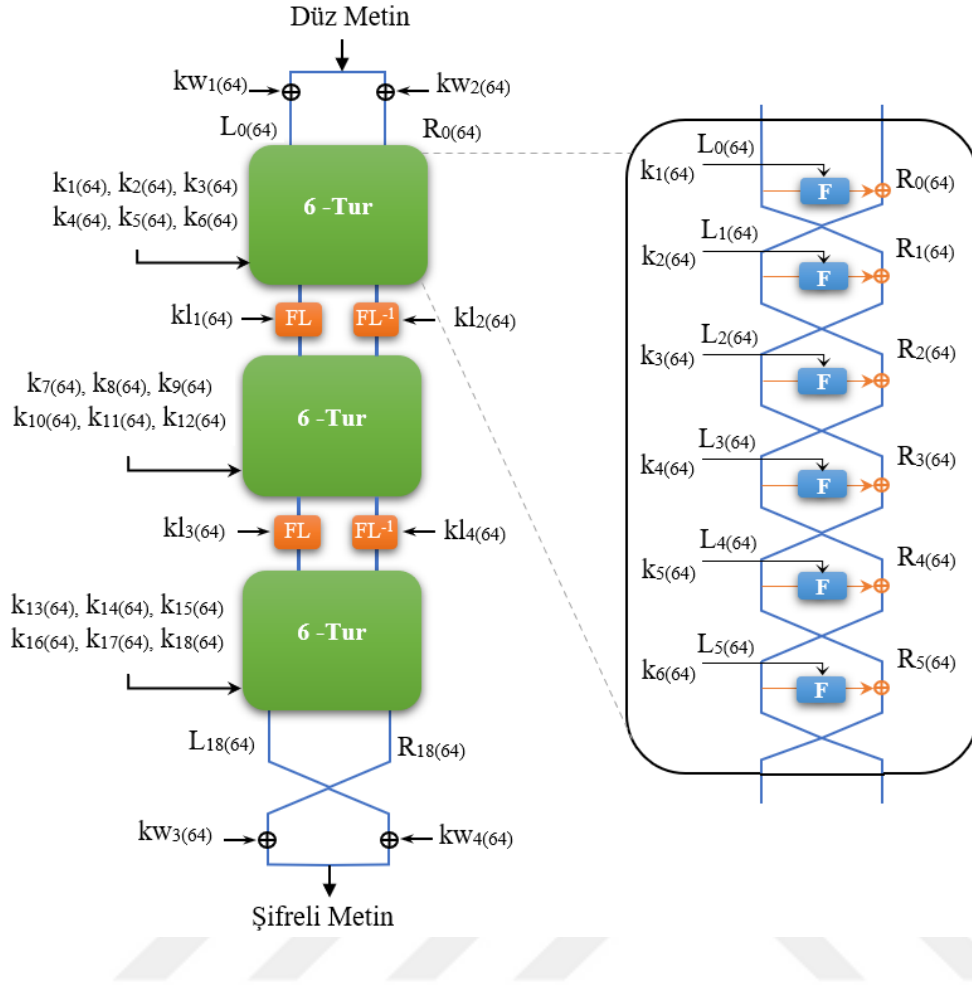
Şekil 3.11. XTEA'de bir tur şifreleme şeması.

Fantomas/Robin [87]- Her ikisi 'de LS-tasarım üyesidir ve dahil edici şifre Robin ve dahil edici olmayan şifre Fantomas olarak kategorize edilmiştir. Her iki şifredeki round fonksiyonu şu katmanları içerir: S-Box katmanı, L-Box katmanı ve KC katmanı. Her iki şifrenin S-Box'ı sekize sekiz bit ve L-Box'ı on altıya on altı bitten oluşmaktadır. Şekil 3.12 [87] Fantomas şifreleme işlemini göstermektedir.



Şekil 3.12. Fantomas şifreleme işlemi.

Camellia [10]- Şifreyi daha güvenli hale getirmek için şifrelemek ve şifreye engellemek için anahtar whitening tekniğini kullanmaktadır. Ayrıca her altı döngüde bir FL ve FL^{-1} mantıksal fonksiyonlar sisteme dahil edilmiştir. Verimliliği yazılım ve donanım platformlarında dikkat çekicidir. Şekil 3.13 [10] Camellia şifreleme (128-bit anahtar) işlemini göstermektedir.



Şekil 3.13. Camellia şifreleme (128-bit anahtar) işlemi.

Hafif sıklet şifreler karşılaştırması, Tablo A.3’de sunulmaktadır [13].

Tablo 3.2. Tez çalışması ve literatürdeki çalışmaların karşılaştırması (Raspberry Pi 3 ve Arduino Mega 256 için).

Makale	Platform	Değerlendirme Metrikleri	Algoritmalar	Karşılaştırma yöntemi	Mod
Gaurav B [88]	ARM 7 LPC2129	Flash, RAM, Çalışma süresi, iş çıkarma oranı, döngü sayısı	PRESENT, LED, BORON, KLEIN, HUMMINGBIRD2, SIMON, SPECK, PICCOLO, TWINE, CLEFIA	Her algoritmanın blok uzunlukları	Şifreleme
Rutuja S [89]	ARM 7 LPC2129	Hafıza kullanımı, güç tüketim, iş çıkarma oranı ve Çalışma süresi	PRESENT, TWINE, PICCOLO, AES, CLEFIA, NUCLEAR	Her algoritmanın blok uzunlukları	Şifreleme
Jaber H [90]	Atmega128 (AVR simülatör)	Güç tüketim ve hafıza kullanımı (Flash ve RAM)	SIMON, SPECK, TWINE, KLEIN, Piccolo	Her algoritmanın blok uzunlukları	Şifreleme
Kedar D [91]	Raspberry Pi 3, Beagle Bone Black	CBC ve EBC modlarında çalışma süresi	AES, DES, TDES, Blowfish, Twofish, RC2	MB büyüklüğünde dosyalar (1, 2, 4, 8, 16, 32, 64, 128 MB)	Şifreleme

Tablo 3.2. (Devamı) Tez çalışması ve literatürdeki çalışmaların karşılaştırması (Raspberry Pi 3 ve Arduino Mega 256 için).

Makale	Platform	Değerlendirme Metrikleri	Algoritmalar	Karşılaştırma yöntemi	Mod
Michael A [92]	8-bit mikro denetleyici (4 ve 16 MHZ)	RAM, döngüler, iş çıkarma oranı, enerji Flash, SRAM, iş çıkarma oranı	KATAN, TEA, PRESENT, SEA, AES SIMON, SPECK, PRESENT, SEA, AES	Her algoritmanın blok uzunlukları	Şifreleme
Johann G [93]	StrongARM SA-1100 (200 MHZ)	Güç tüketim, iş çıkarma oranı, kod boyutu, hafıza (RAM & ROM) alanı	RC6, Rijndael, Serpent, Twofish, XTEA	Her algoritmanın blok uzunlukları	Şifreleme + Şifre çözmek
Miroslav B [94]	Atmel ATmega128RFA1	Enerji tüketimi, zaman ve iş çıkarma oranı	XTEA, AES	Yükler (1, 15, 16, 31, 32, 47, 48, 63, 64, 79, 80, 95, 96, 104 bayt)	Şifreleme + Şifre çözmek
Mickael C [12]	MSP430	Döngü sayısı, döngü/bayt, RAM, ROM, birleşik metrik	AES, CLEFIA, DESXL, HIGHT, IDEA, Noekeon, KATAN, KLEIN, KTANTAN, LBlock, LED, MCRYPTON, MIBS, PRESENT, Piccolo, SEA, Skipjack, TEA, TWINE, XTEA	Her algoritmanın blok uzunlukları	Şifreleme + Şifre çözmek
C.A. Lara N [95]	Qualcomm MSM8926 ve Snapdragon 400 (D2303, D2306)	Güç tüketimi, çalışma süresi, iş çıkarma oranı, hafıza kullanımı	AES, PRESENT	Her algoritmanın blok uzunlukları	Şifreleme + Şifre çözmek
S. Kotel [96]	8-bit AVR ve 16-bit MSP mikro denetleyici	Kod büyüklüğü, RAM, Çalışma süresi, efficiency İş çıkarma oranı,	SIMON, SPECK, AES, PRESENT	Her algoritmanın blok uzunlukları	Şifreleme + Şifre çözmek
William D [97]	Xilinx Kintex-7 FPGA	TP/A, döngüler/blok, döngüler/bayt Hafıza kullanımı, hız, kayan nokta sayıları karşılaştırması	AES, SIMON, SPECK, PRESENT, LED, TWINE	Her algoritma	Yazılım ve Donanım
Margi [98]	Arduino Uno, MATLAB	Hafıza kullanımı, döngüler/bayt, hız, çalışma süresi	PRESENT, AES, HIGHT, Klein	Her algoritma	Yazılım
Deepti S [99]	Intel (R) core (TM) i5-2430M CPU @ 2.40 GHz	Hafıza kullanımı, döngüler/bayt, hız, çalışma süresi	BRIGHT, RoadRunneR, SPECK, HIGHT, SPNRX	Blok şifreler	Şifreleme + Şifre çözmek
M. Razvi [100]	Sim-Panalyzer	Enerji tüketimi, çalışma süresi, hız	AES, RC5	Yükler (16, 64, 256, 1024, 8192 bayt)	Şifreleme + Şifre çözmek
Lejla [101]	Cadence Encounter RTL Compiler & ModelSIM simülator	Şifreleme time, güç (Statik ve dinamik), enerji	AES, CLEFIA, KLEIN, LED, mCrypton, PRESENT, KATAN	Blok uzunlukları	Şifreleme
Ray B [84]	Atmel ATmega128	Flash, RAM, hız, enerji	SIMON, SPECK	Blok uzunlukları	Şifreleme

Tablo 3.2. (Devamı) Tez çalışması ve literatürdeki çalışmaların karşılaştırması (Raspberry Pi 3 ve Arduino Mega 256 için).

Makale	Platform	Değerlendirme Metrikleri	Algoritmalar	Karşılaştırma yöntemi	Mod
W.K.LEE [102]	ODROID-XU3	Çalışma süresi, enerji iş çıkarma oranı, güç tüketimi	AES, CLEFIA, SIMON, SPECK, PRESENT AES, Camellia, IDEA, NOEKEON, KASUMI, GOST, HIGHT, PRESENT, CLEFIA, HB, Piccolo, TEA, XTEA, SEA, mCrypton, MIBS, TWINE, LED, Klein, KATAN, KTANTAN, ITUbee, SIMON, SPECK, PRINTcipher, LBlock, PRINCE, PRIDE, Zorro, Robin, Fantomas, LEA, DES Güç tüketimi (AES, DES, TDES), Çalışma süresi (AES, Blosfish, Camellia, CAST 5/6, DES, DESede, GOST28147, IDEA, Noekeon, RC 2/4/5, Rijindal, SEED, Serpent, Skipjack, TEA, XTEA, Twofish)	64 MB düz metin	Şifreleme
George H [103]	8-bit, 16-bit, 32-bit mikro denetleyici	RAM, ROM, gecikme, enerji, iş çıkarma oranı, Yazılım verimliliği	mCrypton, MIBS, TWINE, LED, Klein, KATAN, KTANTAN, ITUbee, SIMON, SPECK, PRINTcipher, LBlock, PRINCE, PRIDE, Zorro, Robin, Fantomas, LEA, DES Güç tüketimi (AES, DES, TDES), Çalışma süresi (AES, Blosfish, Camellia, CAST 5/6, DES, DESede, GOST28147, IDEA, Noekeon, RC 2/4/5, Rijindal, SEED, Serpent, Skipjack, TEA, XTEA, Twofish)	Blok Anahtar ve Blok Boyutu	Yazılım ve donanım karşılaştırması
Lukas [104]	Samsung Galaxy S i9000	Güç tüketimi, çalışma süresi	CAST 5/6, DES, DESede, GOST28147, IDEA, Noekeon, RC 2/4/5, Rijindal, SEED, Serpent, Skipjack, TEA, XTEA, Twofish)	64, 128, ve 512-bit mesajlar	Şifreleme + Şifre çözmek
Tasnime O [105]	Raspberry Pi 2 ve Arduino UNO	RAM, ROM, çalışma süresi, Saat/döngü	PRESENT, SIMON, SPECK, RECTANGLE, PRINCE, Pride, LBlock	Her algoritmanın blok uzunlukları	Şifreleme + Şifre çözmek, anahtar zamanlama
Levent E [106]	STM32F401RE mikro denetleyici	İş çıkarma oranı, RAM&ROM, enerji tüketimi, çalışma süresi	CLEFIA, PICCOLO ve TWINE	512, 1024, 2048, 3072 bayt yükler	Şifreleme
Mojtaba A [107]	Atmel ATtiny45 mikro denetleyici	RAM, ROM, enerji tüketimi	TEA, HIGHT, KATAN, KLEIN	Her algoritmanın blok uzunlukları	Şifreleme
Murat Ç [108]	Atmel Atmega128 mikro denetleyici	RAM, ROM, çalışma süresi, iş çıkarma oranı	AES, SERPENT, Camellia, Cast5, MARS	Her algoritmanın blok uzunlukları	Şifreleme + Şifre çözmek

Tablo 3.2. (Devamı) Tez çalışması ve literatürdeki çalışmaların karşılaştırması (Raspberry Pi 3 ve Arduino Mega 256 için).

Makale	Platform	Değerlendirme Metrikleri	Algoritmalar	Karşılaştırma yöntemi	Mod
Pal-Stefan M [109]	S08, S12, S12Z, RL78 D1A, TC1782, TC1797, MSP430, MPC5606B, iMX6, RH850 G3K, RH850 G3M	ROM, çalışma süresi	MD5, SHA1, SHA2, SHA3, Blake2, AES, KATAN, PRESENT, SPECK	8, 64, 576, 1536, 4096, uzun mesajlar	Şifreleme
Tez çalışması	Raspberry Pi 3 ve Arduino Mega 2560 (Atmel ATmega 2560)	RAM, ROM, çalışma süresi, iş çıkarma oranı, enerji tüketimi	AES, PRESENT, LBlock, Skipjack, SIMON, XTEA, PRINCE, Piccolo, HIGHT, RECTANGLE	16, 64, 128, 256, 512, 1024, 2048 bayt yükler	Şifreleme + Şifre çözmek

3.3. Hafif Şifreleme Algoritmalarının Performans Değerlendirmesi

Bu bölümde Raspberry Pi 3 ve Arduino Mega 256 üzerinde testleri gerçekleştirmek için on hafif siklet şifresi seçilmiştir. Adı geçen iki test cihazı özelliklerinden dolayı IoT projelerinde önde gelen cihazlardır. Algoritmaların seçiminde, blok uzunluğu, anahtar boyutları, popüleritesi, iç yapıları ve yazılım ortamları üzerinde uygulama olanakları kriterleri göz önüne alınmıştır. PRESENT gibi bazı algoritmalar ilk sürümlerde donanım odaklı olsa da daha sonra yazılım platformlarına kurulmak üzere geliştirilmiştir. 128 bitlik blok uzunluğuna sahip algoritmalar (AES hariç), diğer tüm algoritmalar 64 bit uzunluğunda ve anahtar boyutu açısından 80 ve 128 bit anahtar boyutuna sahipler. LBlock ve Skipjack hariç, diğerleri 128 bit anahtar boyutuna sahipler. Tablo 3.3, seçilen on hafif siklet şifre algoritmanın karşılaştırılmasını göstermektedir.

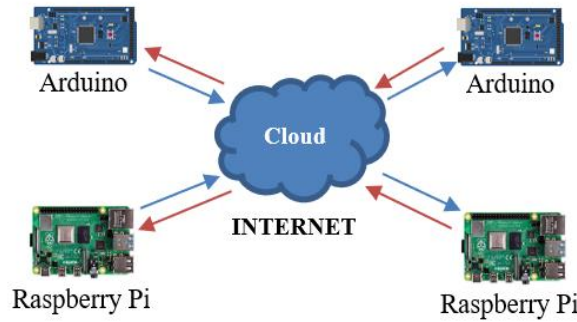
Tablo 3.3. Seçili hafif siklet şifrelerin karşılaştırılması.

Ad	Blok uzunluğu (Bit)	Anahtar uzunluğu (Bit)	Yapı
LBlock	64	80	Feistel
Skipjack	64	80	Feistel
XTEA	64	128	Feistel
HIGHT	64	128	Feistel
Piccolo	64	128	Feistel
SIMON	64	128	Feistel
RECTANGLE	64	128	SPN
PRESENT	64	128	SPN
PRINCE	64	128	SPN
AES	128	128	SPN

Bahsedilen on blok şifreyi bir IoT platformu üzerinden değerlendirmek için Raspberry Pi 3 ve Arduino Mega 2560 cihazları kullanılmıştır. Şifreli dosya değişim platformu olarak Dropbox bulut hizmeti seçilmiştir. Daha kesin sonuçlar elde etmek için her deney beş kere tekrarlanmıştır. Düz metin, gönderici tarafında bir dosyaya şifrelenir ve dosya buluta aktarılır ve hedefte şifre çözme için cihaz dosyayı buluttan okur ve şifresini çözmeye başlar. Gecikme ve ağ hızı gibi ağ parametreleri testler sırasında değişebileceğinden, buluta yükleme ve indirme sürelerini göz ardı edilmiş ve bu tezdeki tüm ölçülen değerler, gönderici veya alıcıdaki yerel şifreleme ve şifre çözme sürelerine dayanmaktadır. Şekil 3.14, veri alışveriş sisteminin genel bir görünümünü gösterir ve Şekil 3.15, sırasıyla değerlendirilen sistemin donanım düzenini gösterir. Şekil 3.15'te görüldüğü gibi Raspberry Pi 3, 10400 mA bir güç kaynağı (Power bank) ile güç verilmiş ve Arduino Mega, laptop USB portuna bağlanmıştır. Komutları kontrol etmek ve yürütmek için Raspberry Pi ve Arduino ethernet ve USB bağlantı noktaları aracılığıyla MacBook'a bağlanmıştır. Farklı senaryolarda güç tüketimini ölçmek için USB güç ölçer kullanılmıştır. Pi 3, 1.2 GHz hızında çalışır ve 1 GB Ram kullanır ve 64-bit ARM Cortex A53 işlemci ile donatılmıştır. 5V mikro USB veya güç bankası ile çalıştırılabilir. Arduino Mega 2560 ise 16 MHz ATmega2560, 8 bit mikro denetleyici, 256 KB Flash bellek, 8 KB SRAM ve 4 KB EEPROM'a sahiptir. Arduino için önerilen giriş voltajı 7-12V'dir.

$$\text{Akım akışı (A)} * \text{Süre (Saniye)} = \text{Şarj (C)} \quad (3.1)$$

$$\text{Şarj (C)} * \text{Gerilim (V)} = \text{Enerji (J)} \quad (3.2)$$



Şekil 3.14. Test sistemleri için veri alışveriş şeması.



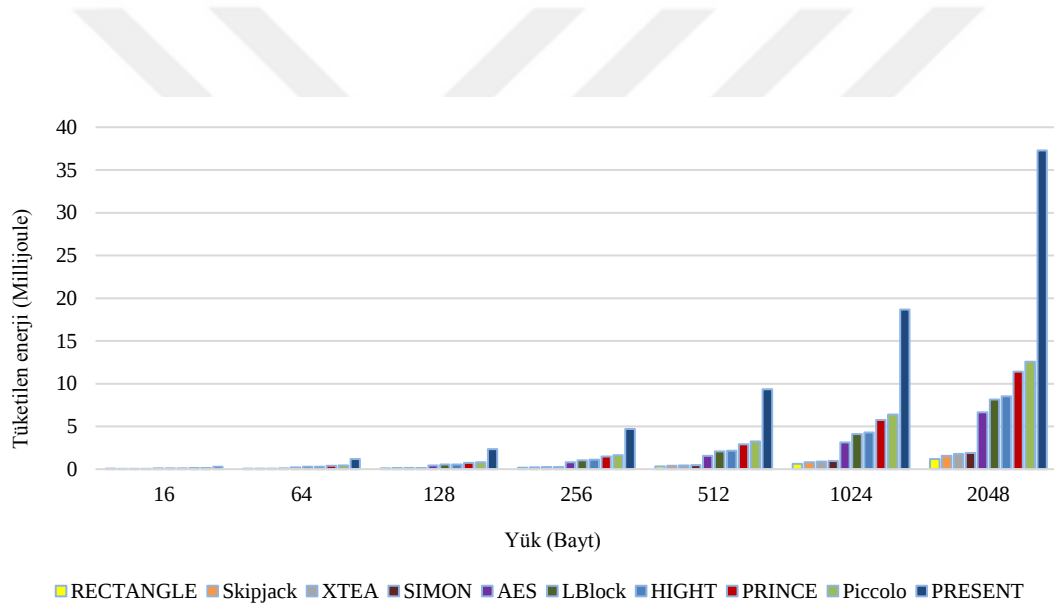
Şekil 3.15. Test edilen sistemin donanım düzeni (A, Arduino Mega kullanarak- B, Raspberry Pi kullanarak).

Şekil 3.16 ve Şekil 3.17, her iki cihazın şifreleme işlemi için enerji tüketim davranışını temsil etmektedir. Raspberry Pi 3 ve Arduino Mega 2560, diğer blok şifrelere kıyasla PRESENT algoritması için dikkat çekici şekilde maksimum enerji tüketir. Bunun nedeni, bu blok şifrenin aslında bir donanım dostu olarak tasarlanmış olmasıdır. Tablo 3.4, iki cihaz için tüketilen enerji karşılaştırmasını göstermektedir. İlgili sütunlardaki hafif siklet şifreler, en düşükten en yükseğe doğru sıralanmıştır. XTEA, Skipjack ve RECTANGLE, Raspberry Pi 3 için minimum enerji tüketimine sahiptir. XTEA ve Skipjack'in basit yapıları enerji tüketimlerinin neden az olduğunu açıklamak için önemli bir nedendir. RECTANGLE bir bit dilim (Bit-slice) mimarisine sahiptir. Bu özellikler, yazılım uygulamalarında daha az güç tüketilmesine yardımcı olur. Arduino Mega 2560 için Piccolo ve PRESENT, ölçülen en yüksek güç tüketimi miktarına sahiptir. Bahsedilen algoritmalar donanım platformlarında kullanılmak üzere tasarlanmıştır ve bu da yazılım uygulamalarında neden maksimum enerji tüketildiğini açıklamaktadır. Skipjack, RECTANGLE ve HIGHT, Arduino Mega 2560 için en az tüketen algoritmalar. HIGHT, donanım platformları için tasarlanmış olmasına rağmen, şifreleme güç tüketimi açısından iyi bir performansı temsil etmektedir. Arduino Mega 2560, Raspberry Pi 3'e göre daha düşük işlemci ve hafıza özelliklerine sahiptir ve bu, Tablo 3.5'teki algoritmanın sırasını etkiler. Arduino Mega 2560 şeması için, PRESENT algoritması değerleri diğer blok şifrelerinden önemli ölçüde daha yüksek olduğundan, grafik ölçeğini 250'yi sınırlandırılmıştır. Bu durumda PRESENT algoritma enerji tüketimi, 512, 1024 ve 2048 bayt için eşit görülmektedir. Bir karşılaştırma olarak, bahsedilen faydalı yükler için PRESENT algoritmasının gerçek

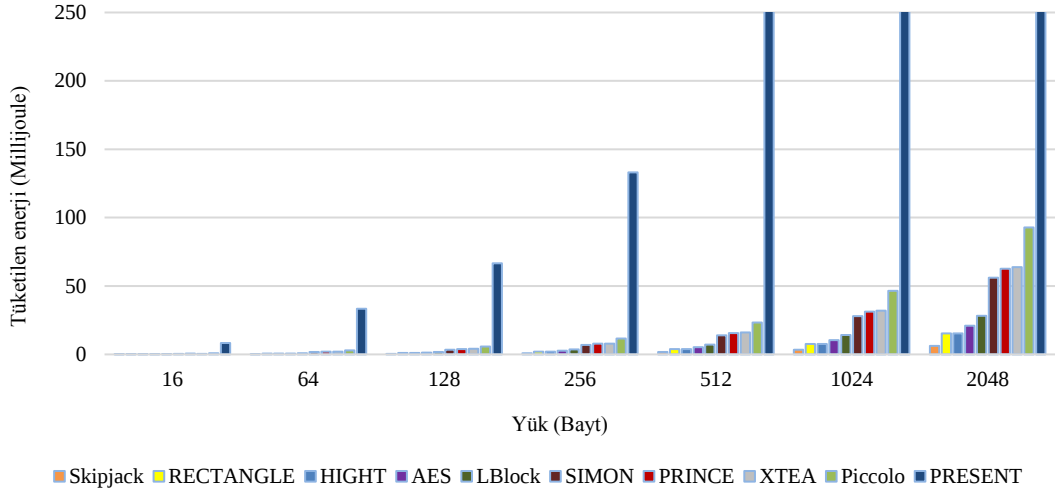
değerleri, aynı faydalı yükler için Piccolo algoritmasının enerji tüketiminden dokuz kat daha büyüktür.

Tablo 3.4. Raspberry Pi 3 ve Arduino Mega 2560'nın şifreleme işlemi için tüketilen enerji karşılaştırması.

Cihaz	Yükler	Algoritmalar sırası (En düşükten en yükseğe)
Raspberry Pi 3	16	XTEA, Skipjack, SIMON, RECTANGLE, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
	64-128	RECTANGLE, XTEA, Skipjack, SIMON, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
	256-2048	RECTANGLE, Skipjack, XTEA, SIMON, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
Arduino Mega 2560	16	Skipjack, HIGHT, RECTANGLE, AES, LBlock, SIMON, XTEA, PRINCE, Piccolo, PRESENT
	64-2048	Skipjack, RECTANGLE, HIGHT, AES, LBlock, SIMON, PRINCE, XTEA, Piccolo, PRESENT



Şekil 3.16. Şifreleme için tüketilen enerji (Raspberry Pi 3).

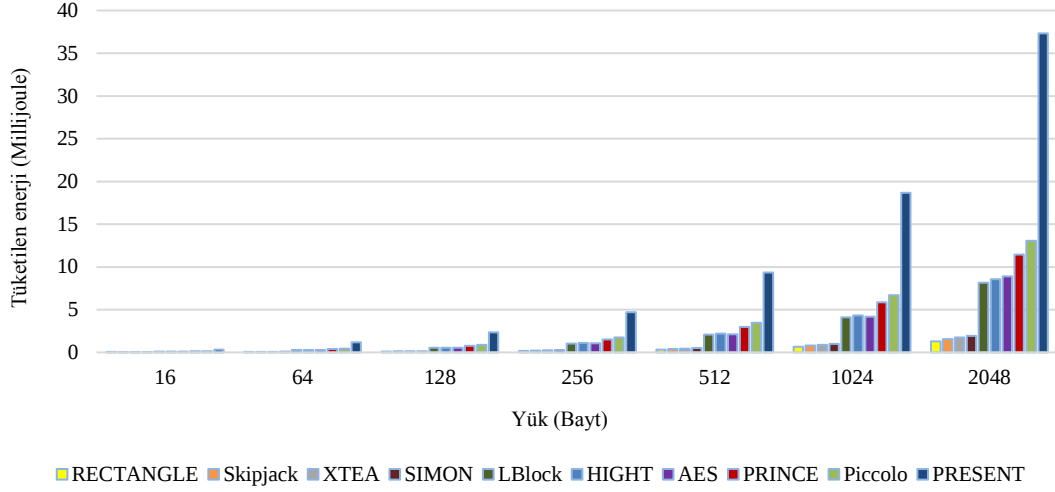


Şekil 3.17. Şifreleme için tüketilen enerji (Arduino Mega 2560).

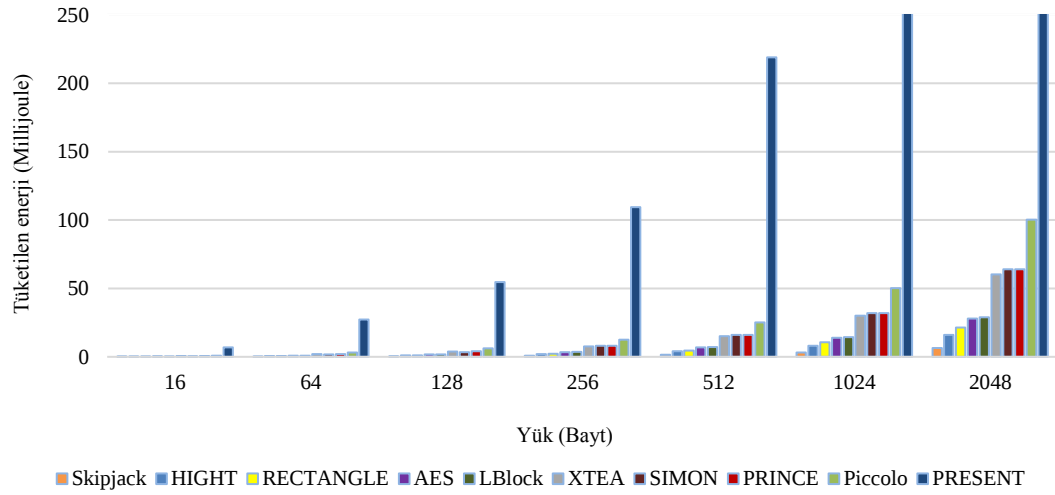
Şekil 3.18 ve Şekil 3.19, iki cihazın şifre çözme işlemi için ölçülen enerji tüketimini göstermektedir. Tablo 3.5, algoritmaların şifre çözme işlemini en düşükten en yükseğe doğru düzenlenmesinin daha iyi anlaşılmasına yardımcı olur. Şifre çözme modunda güç tüketimi, şifreleme işleminin istatistikleri benzer bir durum göstermektedir. Arduino Mega grafiği için dikey eksen, diğer algoritmaları daha iyi görselleştirmek için tıpkı şifreleme grafiği gibi 250 ile sınırlandırılmıştır.

Tablo 3.5. Raspberry Pi 3 ve Arduino Mega 2560'nın şifre çözme işlemi için tüketilen enerji karşılaştırması

Cihaz	Yükler	Algoritmalar sırası (En düşükten en yükseğe)
Raspberry Pi 3	16	XTEA, SIMON, Skipjack, RECTANGLE, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
	64	XTEA, Skipjack, RECTANGLE, SIMON, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
	128-2048	RECTANGLE, Skipjack, XTEA, SIMON, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
Arduino Mega 2560	16	Skipjack, HIGHT, RECTANGLE, LBlock, AES, SIMON, XTEA, PRINCE, Piccolo, PRESENT
	64-128	Skipjack, HIGHT, RECTANGLE, AES, LBlock, SIMON, XTEA, PRINCE, Piccolo, PRESENT
	256-2048	Skipjack, HIGHT, RECTANGLE, AES, LBlock, XTEA, SIMON, PRINCE, Piccolo, PRESENT



Şekil 3.18. Şifre çözme işleminde tüketilen enerji (Raspberry Pi 3).



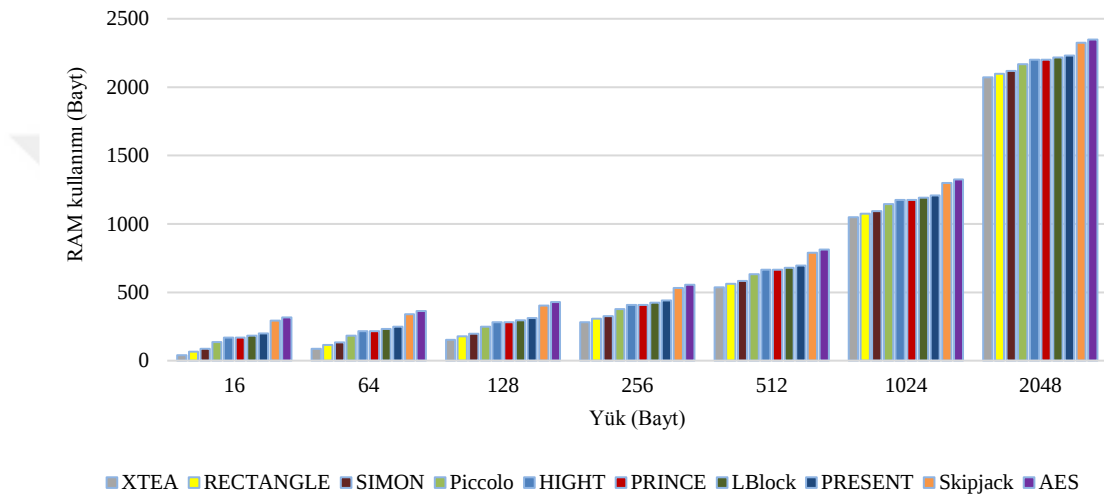
Şekil 3.19. Şifre çözme işleminde tüketilen enerji (Arduino Mega 2560).

Her blok şifre tarafından kullanılan bellek (RAM & ROM) miktarını analiz etmek için Microchip Studio 7 [110] kullanıldı. Şekil 3.20 ve Şekil 3.21, test edilen sistemler için şifreleme RAM kullanımını göstermektedir. Arduino Mega 2560, Raspberry Pi 3'ün 1 GB RAM ile kıyasla oldukça düşük SRAM boyutuna (8 KB) sahiptir. Tablo 3.6, iki cihaz arasındaki tüm faydalı yükler için şifreleme RAM kullanımını özetlemektedir. Raspberry Pi 3 için XTEA ve RECTANGLE, Arduino Mega 2560 için XTEA ve Piccolo'ya karşı minimum RAM kullanımına sahiptir. En yüksek RAM işgali göz önüne alındığında, Raspberry Pi 3 için Skipjack ve AES maksimum RAM kullanımına sahiptir. Arduino Mega için, HIGHT ve PRESENT maksimum değerlere ulaşmaktadır. Test edilen cihazların farklı donanım özellikleri ve test edilen

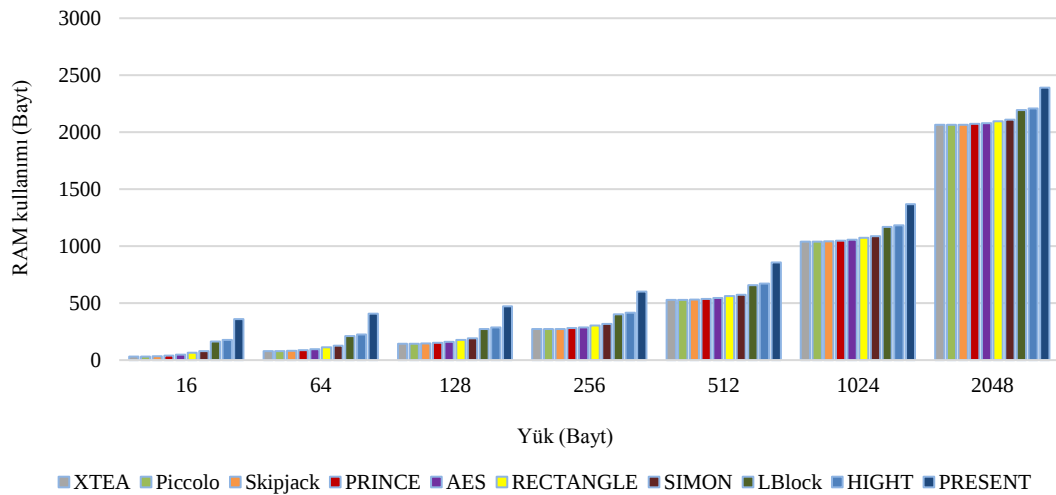
algoritmaların altyapısı ve tasarladıkları hedefler (Donanım veya yazılım sistemler), şifreleme ve şifre çözme modları için test sonuçlarını etkiler.

Tablo 3.6. Şifreleme RAM kullanımı için sıralanmış algoritmalar (Arduino Mega 2560, Raspberry Pi 3'e karşı).

Cihaz	Yükler	Algoritmalar sırası (En düşükten en yükseğe)
Raspberry Pi 3	16-2048	XTEA, RECTANGLE, SIMON, Piccolo, HIGHT, PRINCE, LBlock, PRESENT, Skipjack, AES
Arduino Mega 2560	16-2048	XTEA, Piccolo, Skipjack, PRINCE, AES, RECTANGLE, SIMON, LBlock, HIGHT, PRESENT



Şekil 3.20. Şifreleme RAM kullanımı (Raspberry Pi 3).



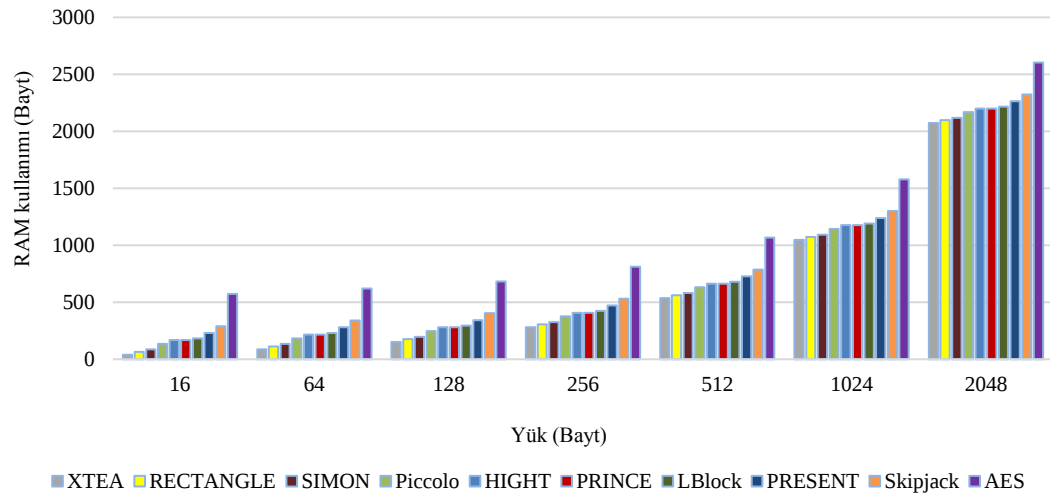
Şekil 3.21. Şifreleme RAM kullanımı (Arduino Mega 2560).

Şifre çözme RAM kullanımındaki değişiklikleri Şekil 3.22 ve Şekil 3.23'te gösterilmektedir. Tablo 3.7'de Raspberry Pi 3 için şifreleme moduna göre

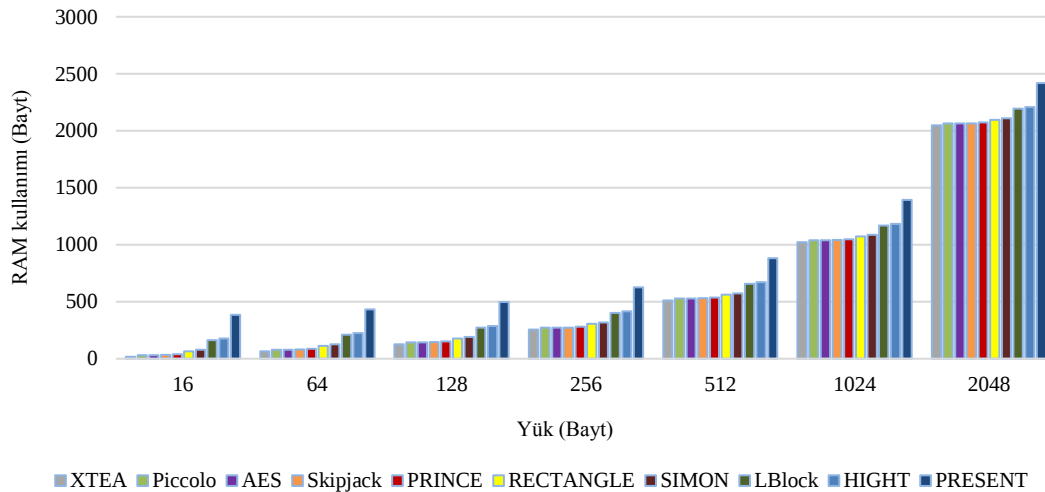
algoritmaların sıralamasında bir değişiklik bulunmamaktadır. Arduino Mega 2560 için XTEA, şifre çözme modu için minimum miktarda RAM kaplamıştır ve PRESENT, tıpkı şifreleme modu gibi maksimum değere ulaşır. Şifreleme işleminin aksine, Arduino Mega için algoritma sıralamasında bazı değişiklikler var.

Tablo 3.7. Şifre çözme RAM kullanımı için sıralanmış algoritmalar (Arduino Mega 2560, Raspberry Pi 3'e karşı).

Cihaz	Yükler	Algoritmalar sırası (En düşükten en yükseğe)
Raspberry Pi 3	16-2048	XTEA, RECTANGLE, SIMON, Piccolo, HIGHT, PRINCE, LBlock, PRESENT, Skipjack, AES
Arduino Mega 2560	16-2048	XTEA, Piccolo, AES, Skipjack, PRINCE, RECTANGLE, SIMON, LBlock, HIGHT, PRESENT



Şekil 3.22. Şifre çözme RAM kullanımı (Raspberry Pi 3).



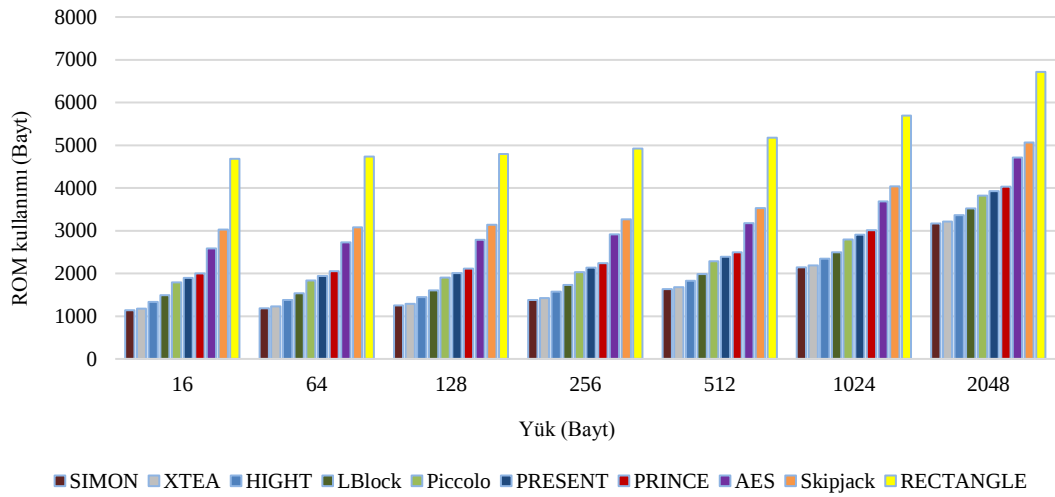
Şekil 3.23. Şifre çözme RAM kullanımı (Arduino Mega 2560).

Raspberry Pi 3'ün donanım özellikleri göz önüne alındığında Arduino Mega 2560'dan oldukça güçlüdür. Şekil 3.24 ve Şekil 3.25, iki cihaz arasında on hafif siklet şifreleme

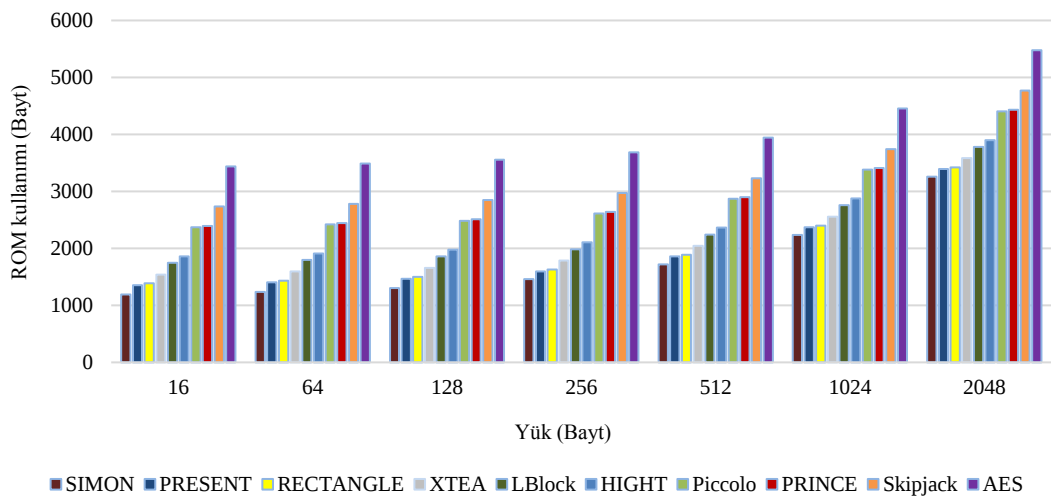
algoritması için iyi bir karşılaştırma sunmaktadır. Tablo 3.8, algoritmaların ROM kullanımını en düşükten en yükseğe doğru özetlemektedir. Şifreleme işlemi için bayt cinsinden en yüksek ROM kullanımı, Raspberry Pi 3 için RECTANGLE algoritmasına aittir ve Arduino Mega 2560 için AES en yüksek ROM'u işgal etmiştir. Raspberry Pi 3 için SIMON ve XTEA ve Arduino Mega için SIMON ve PRESENT en düşük ROM kullanımına sahipler.

Tablo 3.8. Şifreleme için ROM kullanım karşılaştırılması.

Cihaz	Yükler	Algoritmalar sırası (En düşükten en yükseğe)
Raspberry Pi 3	16-2048	SIMON, XTEA, HIGHT, LBlock, Piccolo, PRESENT, PRINCE, AES, Skipjack, RECTANGLE
Arduino Mega 2560	16-2048	SIMON, PRESENT, RECTANGLE, XTEA, LBlock, HIGHT, Piccolo, PRINCE, Skipjack, AES



Şekil 3.24. Şifreleme ROM kullanımı (Raspberry Pi 3).

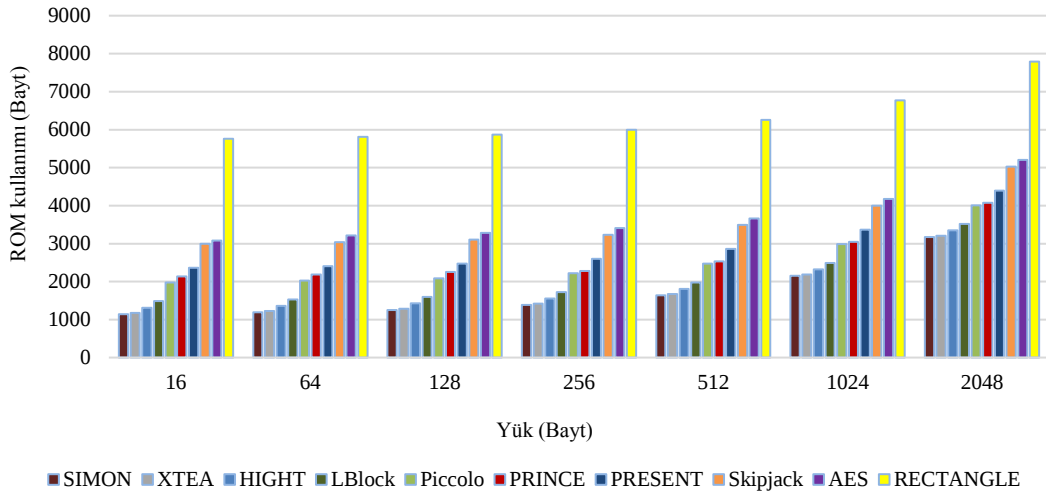


Şekil 3.25. Şifreleme ROM kullanımı (Arduino Mega 2560).

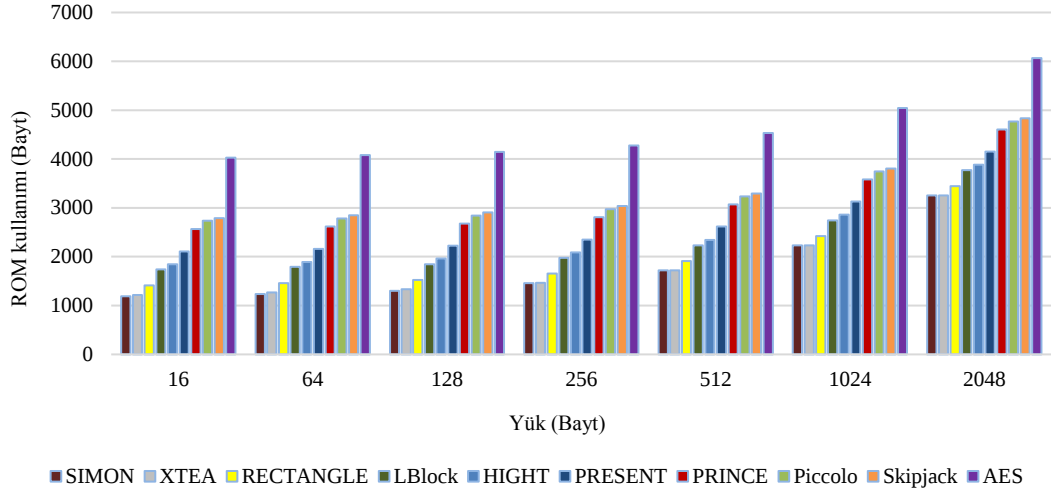
Şekil 3.26 ve Şekil 3.27, Raspberry Pi ve Arduino Meganın şifre çözme işlemi için farklı yüklere ilişkin on blok şifrenin ROM kullanımını göstermektedir. Şekil 3.26, RECTANGLE ve Şekil 3.27'de AES, başlangıçtan itibaren diğer blok şifrelere kıyasla önemli ölçüde daha yüksek miktarda ROM kullanımına sahiptirler. İlginç bir şekilde, ROM kullanımındaki artış oranı her iki cihaz için de belirli bir model izlemektedir. SIMON ve XTEA, şifre çözme modunda her iki cihaz için dahil, minimum ROM kullanımına sahiptir.

Tablo 3.9. Şifre çözme modunda ROM kullanım karşılaştırılması.

Cihaz	Yükler	Algoritmalar sırası (En düşüktten en yükseğe)
Raspberry Pi 3	16-2048	SIMON, XTEA, HIGHT, LBlock, Piccolo, PRINCE, PRESENT, Skipjack, AES, RECTANGLE
Arduino Mega 2560	16-2048	SIMON, XTEA, RECTANGLE, LBlock, HIGHT, PRESENT, PRINCE, Piccolo, Skipjack, AES



Şekil 3.26. Şifre çözme ROM kullanımı (Raspberry Pi 3).

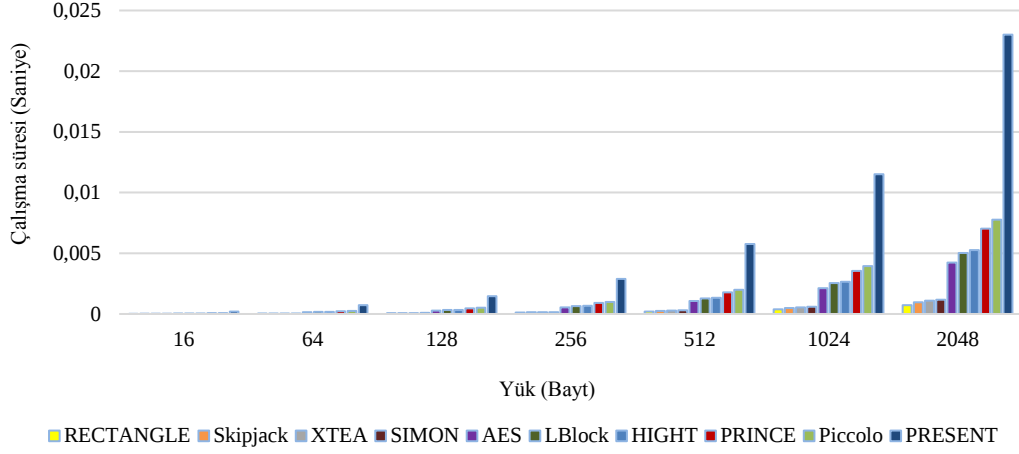


Şekil 3.27. Şifre çözme ROM kullanımı (Arduino Mega 2560).

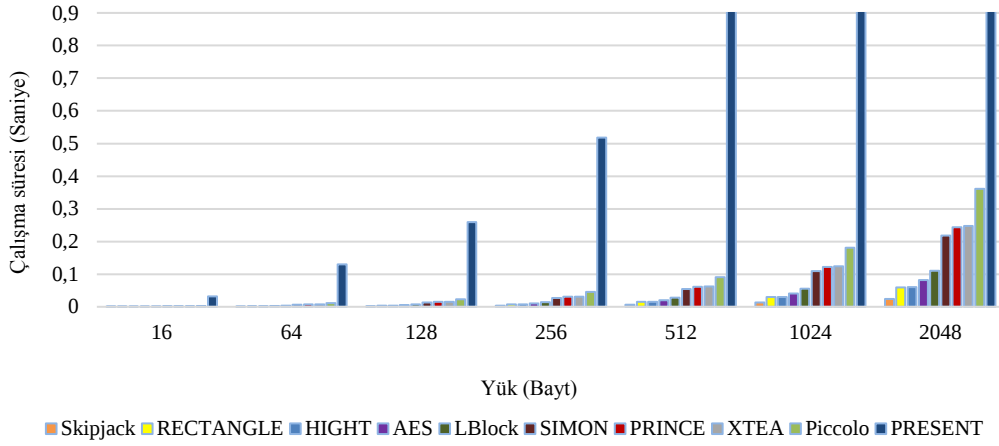
Raspberry Pi 3 ve Arduino Mega 2560 için ortalama şifreleme çalışma sürelerini Şekil 3.28 ve Şekil 3.29'de gösterilmiştir. Tüm faydalı yükler için PRESENT, her iki cihaz için de en yüksek şifreleme çalışma süresi ile ilk sıradadır. 512 bayttan sonra PRESENT çalışma süreleri yüksek olduğundan, Şekil 3.29'daki dikey eksen, diğer algoritmaların çalışma sürelerini net bir şekilde göstermek için 0,9 saniye ile sınırlandırılmıştır. Tablo 3.10'de her cihaz için ortalama şifreleme yürütme süresine göre artan on algoritma sıralanmıştır.

Tablo 3.10. Şifreleme çalışma süresine göre algoritmalar sıralaması.

Cihaz	Yükler	Algoritmalar sırası (En düşüktten en yükseğe)
Raspberry Pi 3	16	XTEA, Skipjack, SIMON, RECTANGLE, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
	64-128	RECTANGLE, XTEA, Skipjack, SIMON, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
	256-2048	RECTANGLE, Skipjack, XTEA, SIMON, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
Arduino Mega 2560	16	Skipjack, HIGHT, RECTANGLE, AES, LBlock, SIMON, XTEA, PRINCE, Piccolo, PRESENT
	64-2048	Skipjack, RECTANGLE, HIGHT, AES, LBlock, SIMON, PRINCE, XTEA, Piccolo, PRESENT



Şekil 3.28. Ortalama şifreleme çalışma süresi (Raspberry Pi 3).

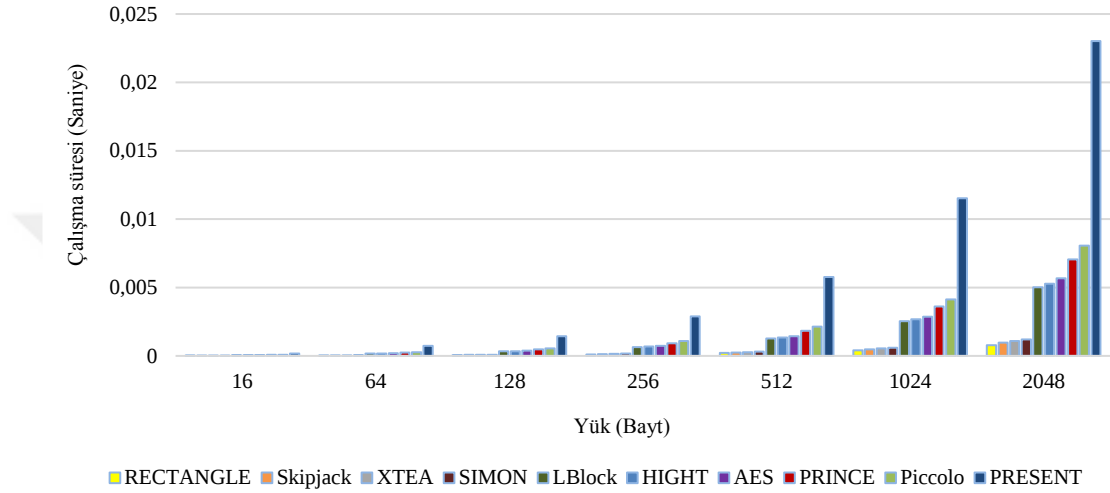
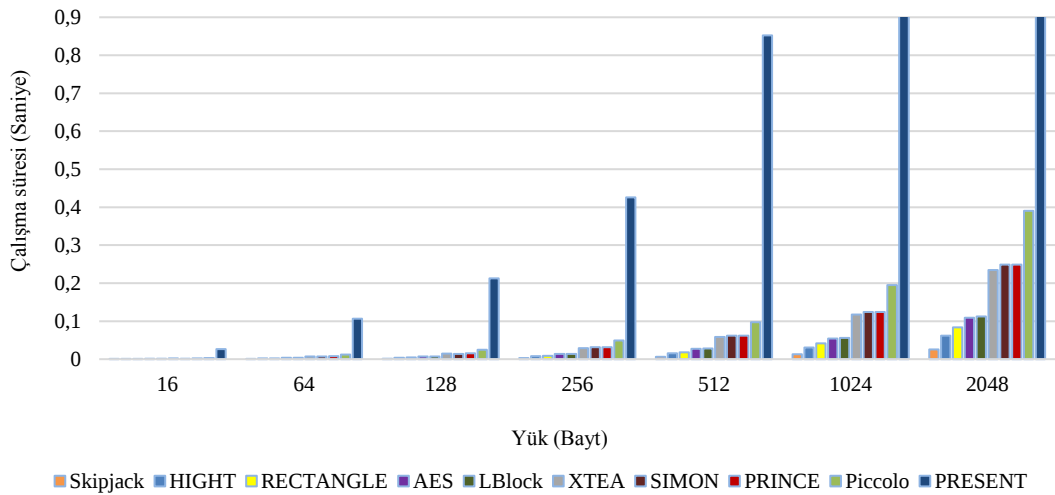


Şekil 3.29. Ortalama şifreleme çalışma süresi (Arduino Mega 2560).

Şekil 3.30 ve Şekil 3.31 test ortamında şifre çözme işlemi için harcanan ortalama süreyi temsil etmektedir. İki grafik için, özellikle 128 bayttan başlayan en yüksek çalışma süreleri değerlerine sahip PRESENT blok şifresi gözükmemektedir. Diğer algoritmaların çalışma sürelerini daha kolay görebilmek için ikinci grafikte yürütme süresi 0,9 saniye ile sınırlandırılmıştır. Tablo 3.11’de çalışma süresi dikkate alınarak algoritmanın şifre çözme işlem sırasını özetlemektedir. PRINCE, Piccolo ve PRESENT, donanım odaklı mimariye sahiptir ve yazılım uygulamalar için sırasıyla en yüksek çalışma sürelerine sahiptir. Diğer yandan Skipjack ve HIGHT basit bir yapıya sahip ve Arduino Mega için diğer algoritmalar arasında minimum çalışma sürelerine sahiptir. Raspberry Pi 3 için XTEA ve RECTANGLE diğer hafif sıklet şifrelerden daha hızlı olduğu tespit edilmiştir.

Tablo 3.11. Şifre çözme çalışma süresine göre algoritmalar sıralaması.

Cihaz	Yükler	Algoritmalar sırası (En düşükten en yükseğe)
Raspberry Pi 3	16	XTEA, SIMON, Skipjack, RECTANGLE, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
	64	XTEA, RECTANGLE, Skipjack, SIMON, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
	128-2048	RECTANGLE, Skipjack, XTEA, SIMON, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
Arduino Mega 2560	16	Skipjack, HIGHT, RECTANGLE, LBlock, AES, SIMON, XTEA, PRINCE, Piccolo, PRESENT
	64-2048	Skipjack, HIGHT, RECTANGLE, AES, LBlock, SIMON, XTEA, PRINCE, Piccolo, PRESENT

**Şekil 3.30.** Ortalama şifre çözme çalışma zamanı (Raspberry Pi 3).**Şekil 3.31.** Ortalama şifre çözme çalışma süresi (Arduino Mega 2560).

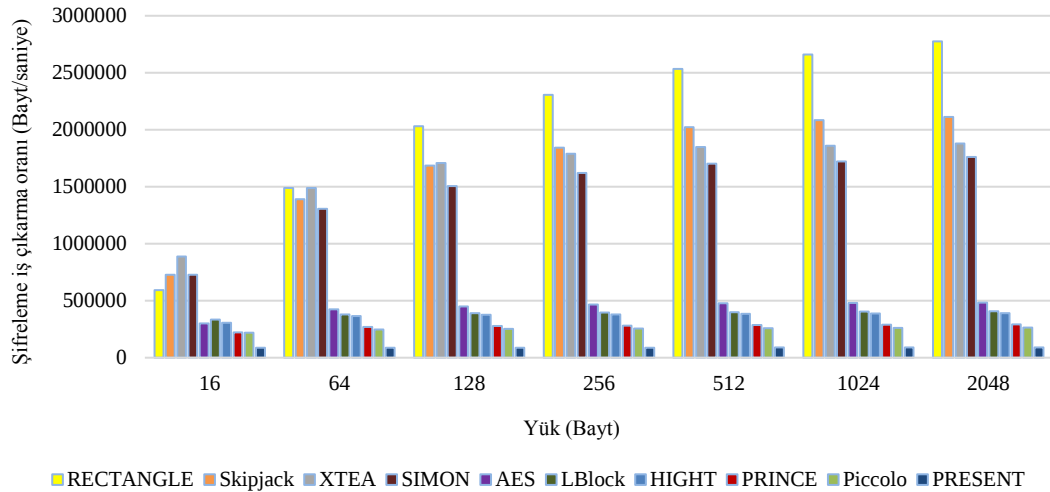
Raspberry Pi 3 ve Arduino Mega 2560 için ortalama şifreleme iş çıkarma oranı (Throughput) Şekil 3.32 ve Şekil 3.33'de gösterilmektedir. İş çıkarma oranı, aşağıdaki denklemi kullanılarak hesaplanmaktadır:

$$\text{İş çıkarma oranı} = \frac{\text{Bayt Sayısı}}{\text{Bitiş zamanı} - \text{Başlangıç zamanı}} \quad (3.3)$$

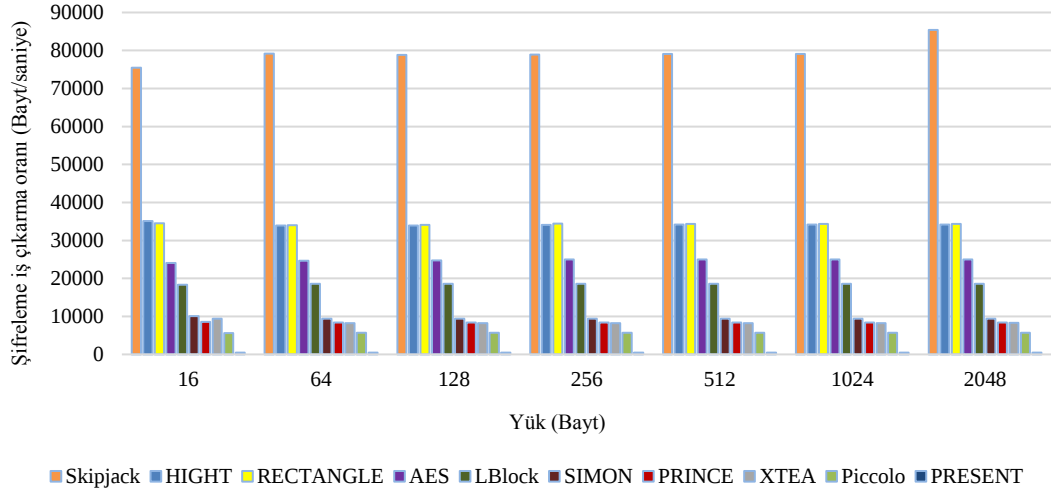
Şifreleme modunda, 16 bayt için, XTEA, Raspberry Pi 3’de diğer blok şifreler arasında en yüksek iş çıkarma oranı değeri elde etmektedir. 64 bayttan sonra, RECTANGLE, Raspberry Pi 3 için en yüksek iş çıkarma oranına sahiptir. Her iki cihaz için de PRESENT, en düşük şifreleme iş çıkarma oranı göstermektedir.

Tablo 3.12. Ortalama şifreleme iş çıkarma oranı sıralaması.

Cihaz	Yükler	Algoritmalar sırası (En düşükten en yükseğe)
Raspberry Pi 3	16	XTEA, SIMON, Skipjack, RECTANGLE, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
	64-128	RECTANGLE, XTEA, Skipjack, SIMON, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
	256-2048	RECTANGLE, Skipjack, XTEA, SIMON, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
Arduino Mega 2560	16-2048	Skipjack, HIGHT, RECTANGLE, AES, LBlock, SIMON, XTEA, PRINCE, Piccolo, PRESENT



Şekil 3.32. Ortalama şifreleme iş çıkarma oranı karşılaştırması (Raspberry Pi 3).

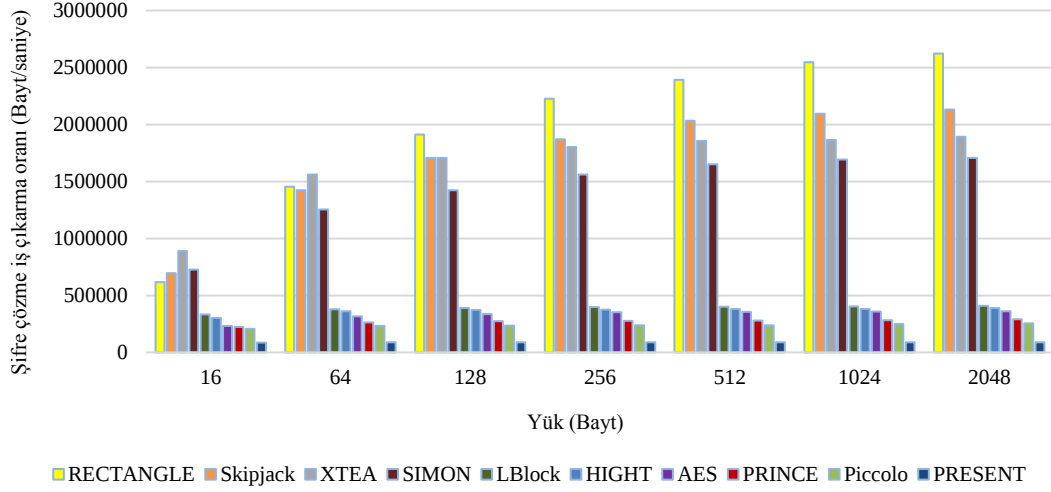


Şekil 3.33. Ortalama şifreleme iş çıkarma oranı karşılaştırması (Arduino Mega 2560).

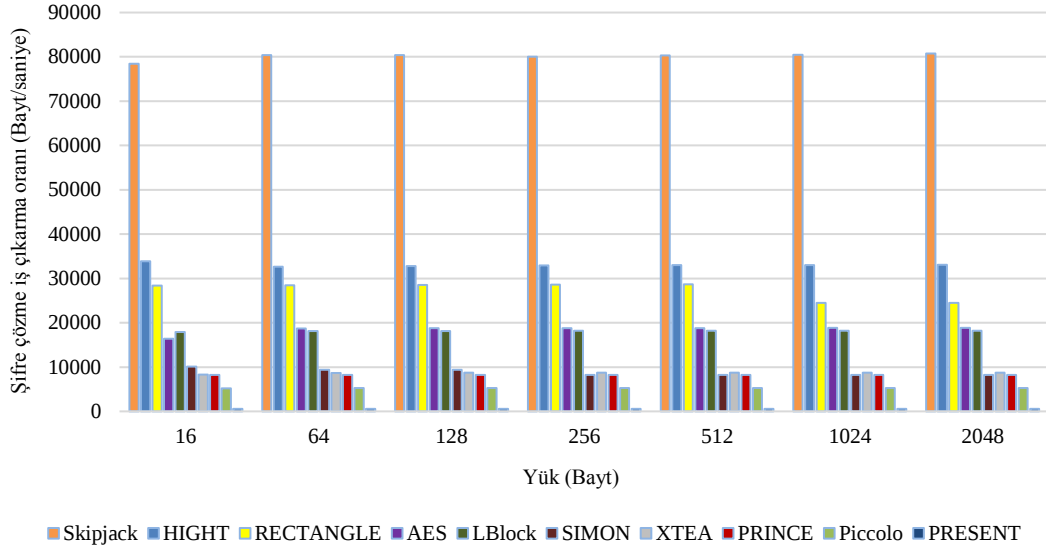
Şekil 3.34, Raspberry Pi 3 için ortalama şifre çözme iş çıkarma oranı değerlerini göstermektedir. RECTANGLE, Skipjack, SIMON ve XTEA, diğer algoritmalarından önemli ölçüde daha yüksek iş çıkarma oranlarına sahiptir. PRINCE, Piccolo ve PRESENT şifreleme iş çıkarma oranı değerleri, diğerleri arasında en küçük olanlardır. Arduino Mega 2560 için (Şekil 3.35), Skipjack şifre çözme iş çıkarma oranı değeri diğer algoritmalarından önemli ölçüde daha büyüktür, HIGH ve RECTANGLE ikinci ve üçüncü konumlardadır. Beklendiği gibi, Piccolo ve PRESENT, her iki cihaz için minimum şifre çözme iş çıkarma oranı değerlerine sahiptir.

Tablo 3.13. Ortalama şifre çözmek iş çıkarma oranı sıralaması.

Cihaz	Yükler	Algoritmalar sırası (En düşükten en yükseğe)
Raspberry Pi 3	16	XTEA, SIMON, Skipjack, RECTANGLE, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
	64	RECTANGLE, XTEA, Skipjack, SIMON, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
	128-2048	RECTANGLE, Skipjack, XTEA, SIMON, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
Arduino Mega 2560	16	Skipjack, HIGHT, RECTANGLE, LBlock, AES, SIMON, XTEA, PRINCE, Piccolo, PRESENT
	64-128	Skipjack, HIGHT, RECTANGLE, AES, LBlock, SIMON, XTEA, PRINCE, Piccolo, PRESENT
	256-2048	Skipjack, HIGHT, RECTANGLE, AES, LBlock, XTEA, PRINCE, SIMON, Piccolo, PRESENT



Şekil 3.34. Ortalama şifre çözme iş çıkarma oranı karşılaştırması (Raspberry Pi 3).



Şekil 3.35. Ortalama şifre çözme iş çıkarma oranı karşılaştırması (Arduino Mega 2560).

3.4. Bölüm Sonuçları

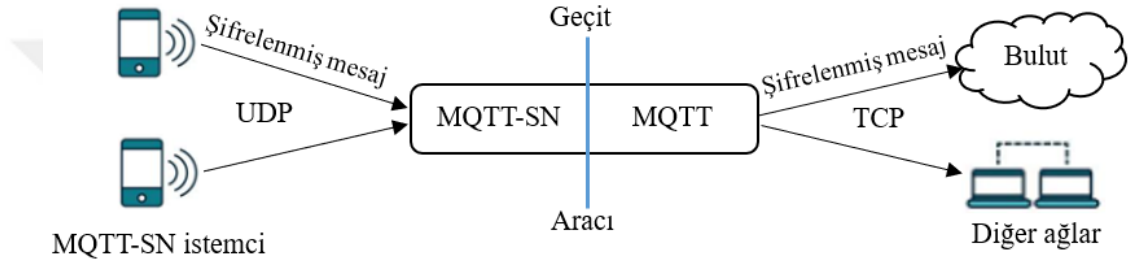
Bu bölümde literatürdeki hafif siklet kriptoloji algoritmaları incelenmiş ve kısıtlı kaynaklı IoT cihazlarında uçtan uca güvenlik sağlamak amacıyla kullanılacak seçilen şifreleme algoritmaları (AES, PRESENT, LBlock, Skipjack, SIMON, XTEA, PRINCE, Piccolo, HIGHT, RECTANGLE) blok uzunluğu, anahtar boyutları, popüleritesi, iç yapıları ve yazılım ortamları üzerinde uygulama olanakları kriterlere göre belirlenmiştir. Testler ve performans değerlendirmeleri Raspberry Pi 3 ve Arduino Mega 2560 üzerinde gerçekleştirilmiştir. Raspberry Pi 3 ve Arduino Mega

2560, diğer blok şifrelere kıyasla PRESENT algoritması için dikkat çekici şekilde maksimum enerji tüketir. Skipjack, RECTANGLE ve HIGHT, Arduino Mega 2560 için en az tüketen algoritmalarıdır. Raspberry Pi 3 için XTEA ve RECTANGLE, Arduino Mega 2560 için XTEA ve Piccolo'ya karşı minimum RAM kullanımına sahiptir. En yüksek RAM işgali göz önüne alındığında, Raspberry Pi 3 için Skipjack ve AES maksimum RAM kullanımına sahiptir. Arduino Mega için, HIGHT ve PRESENT maksimum değerlere ulaşmaktadır. Şifreleme işlemi için bayt cinsinden en yüksek ROM kullanımı, Raspberry Pi 3 için RECTANGLE algoritmasına aittir ve Arduino Mega 2560 için AES en yüksek ROM'u işgal etmiştir. Raspberry Pi 3 için SIMON ve XTEA ve Arduino Mega için SIMON ve PRESENT en düşük ROM kullanımına sahiptirler. Skipjack ve HIGHT basit yapılarıyla Arduino Mega için diğer algoritmalar arasında minimum çalışma sürelerine sahiptirler. Raspberry Pi 3 için XTEA ve RECTANGLE diğer hafif siklet şifrelerden daha hızlı oldukları tespit edilmiştir.



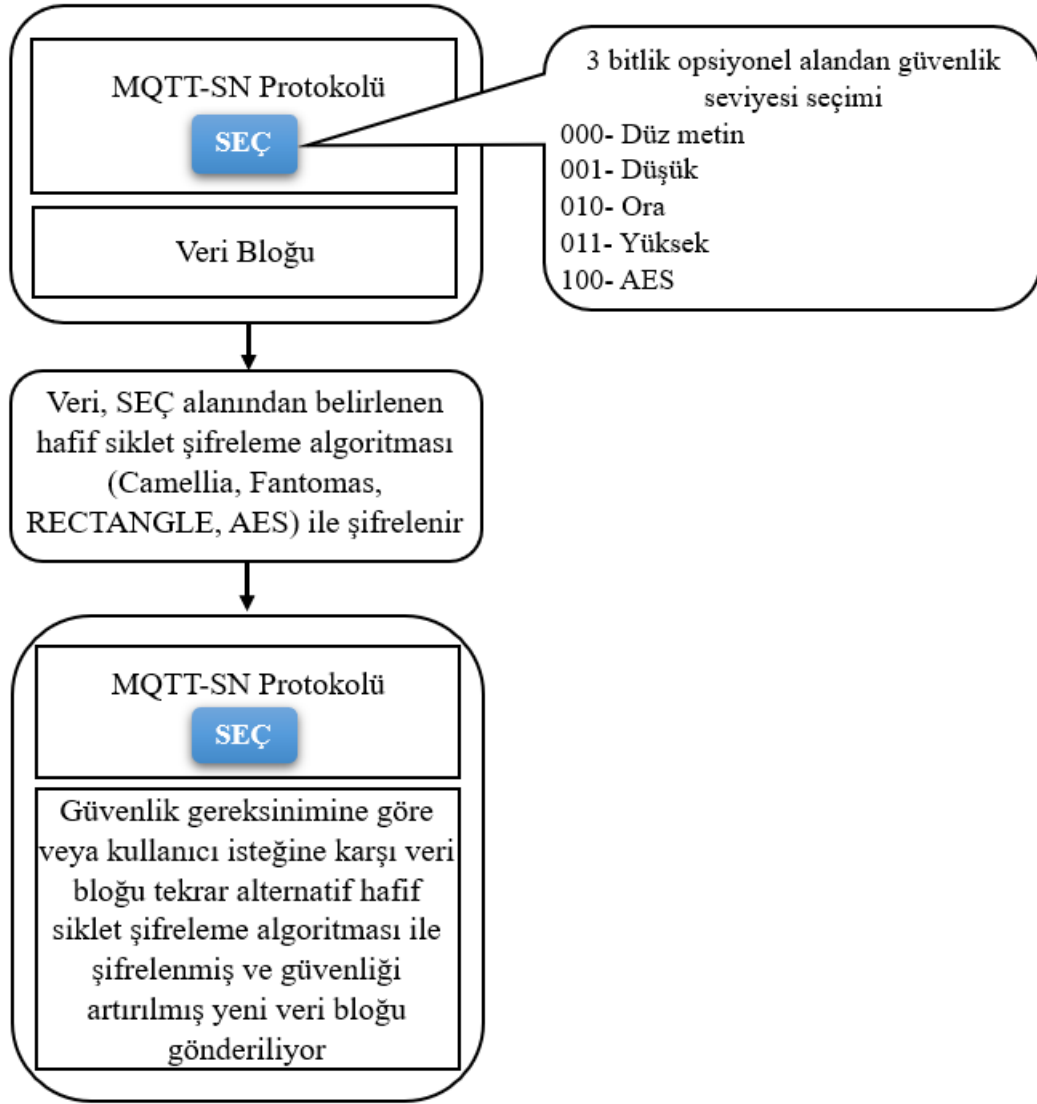
4. HAFİF SİKLET KRİPTOLOJİ ALGORİTMALARINA DAYALI IOT GÜVENLİ MESAJLAŞMA PROTOKOLÜ

MQTT-SN, özellikle IoT'nin bir alt kümesi olan kablosuz algılayıcı ağlarda çalışmak üzere tasarlanmıştır. Paketleri uçtan uca şifrelemek için güvenlik gereksinimlere göre dört hafif şifreleme algoritmadan biri seçilebilir. KAA'lara yönelik önerilen protokolün çalışacağı genel ağ mimarisi Şekil 4.1.'de verilmektedir.



Şekil 4.1. Önerilen protokolün iletişim mimarisi.

Şekil 4.2'de kullanıcı istekleri veya artırılmış BER gibi çeşitli uygulama güvenlik gereksinimlerine göre Zigbee ağları için dört aşamalı bir güvenlik düzeyi (FS-SSL) önerilmiştir. Kullanıcıya güvensiz veya güvenli modlar arasında seçim yapma yeteneği vermek için Zigbee MAC başlığında ayrılmış rezerve çerçeve kontrol alanı bitleri kullanılmıştır. Kullanıcı tarafından talep edildiğinde veya bit hata oranında bir artış olduğunda, fiziksel katmandan MAC katmanına geri bildirim göndermek için güvenli veri transferinde çapraz katmanlı mimari kullanılmıştır.



Şekil 4.2. Önerilen yaklaşımın işleyişi.

4.1. Kullanımı Önerilen Hafif Siklet Kriptoloji Algoritmaları

Bölüm 3'te sunulan Raspberry pi ve Arduino donanımları üzerinde hafif siklet kriptoloji algoritmalarının enerji tüketimi, şifreleme algoritmalarının mesaj gecikmelerine etkisi, bellek kullanımları gibi incelemeleri sonucu Camellia 128, RECTANGLE 128, Fantomas 64 ve AES 128 algoritmalarının güvenli ağ protokolünde kullanımı tercih edilmiştir. AES 128, Riverbed Modeler'da KAA için var olan tek şifreleme algoritmasıdır. Daha yüksek güvenlik, daha az karmaşıklık, daha az enerji tüketimi vb. gibi ayırıcı özellikler bir şifreleme algoritma seçiminde önde gelen kriterlerdir. Camellia, ISO/IEC organizasyonu tarafından onaylanmıştır ve AES ile karşılaştırılabilir işlem yeteneklerine ve güvenlik seviyelerine sahiptir. RECTANGLE

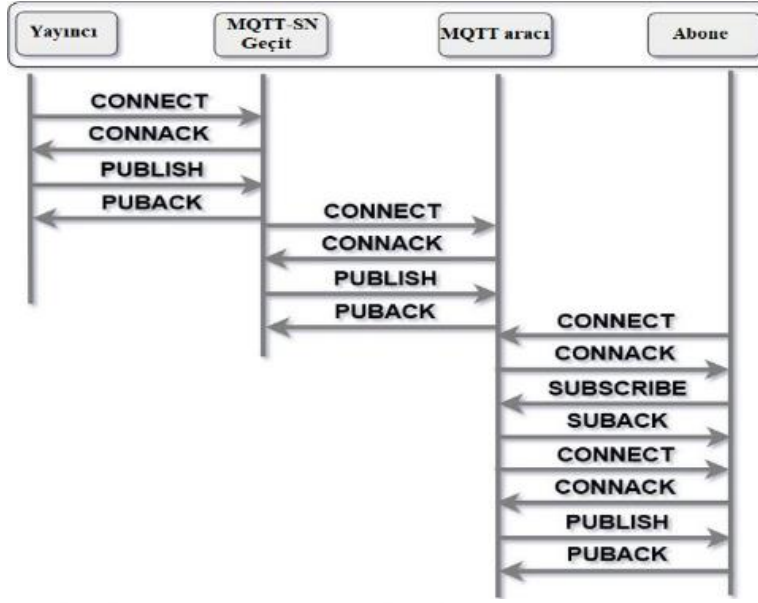
ve Fantomas, bitslice tekniğini kullanan yüksek hızlı yazılım uygulamasına sahiptir ve bu özellik onları çevrimiçi (Online) uygulamalar için iyi bir seçim haline getirmektedir. Adı geçen dört algoritma, enerji tüketimi, iş çıkarma oranı ve uçtan uca gecikme ölçümü kriterlerine göre karşılaştırılmıştır. Tablo 4.1 [13-14], seçilen algoritmaların kriptografik özelliklerini göstermektedir.

Tablo 4.1. Kullanılan hafif siklet şifreler karşılaştırması (Kriptografik özellikler).

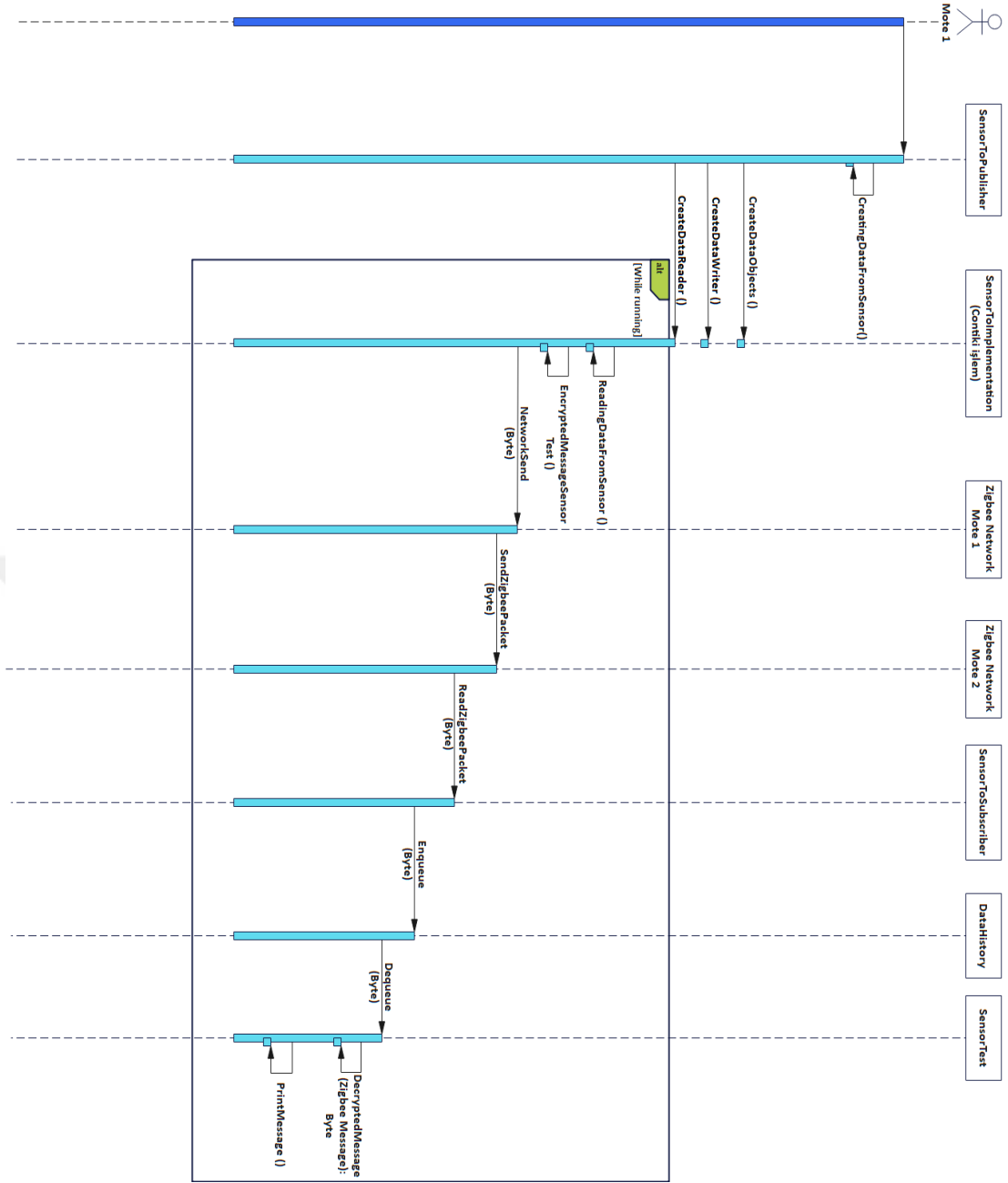
Algoritma	Blok Uzunluğu	Anahtar Uzunluğu	Round Sayısı	Yapı	Saldırıları
AES	128	128	10	SPN	square Partial sum Impossible Diff Boomerang
RECTANGLE	64	128	25	SPN	Differential
Fantomas	128	128	12	SPN	-
Camellia	128	128	18	Feistel	Square Truncated Diff Impossible Diff Cache timing

4.2. MQTT-SN

IoT'nin bir alt kümesi olan KAA'larda, MQTT-SN haberleşme protokolünde güvenli mesaj iletmek için uçtan uca şifreleme uygulanmıştır. Şifreleme işlemi Camellia 128, RECTANGLE 128, Fantomas 64 ve AES 128 algoritmaları uygulayarak gerçekleştirmiştir. MQTT-SN kontrol paketinde 3 rezerve biti kullanarak kullanıcı şifresiz (Düz metin) veya şifreli modlarında tanımlanan hafif siklet algoritmalarından birini seçebilir. Performans değerlendirmeleri, Zolertia Z1 ve Contiki (Cooja) simülör üzerinden çoklu senaryolar için bellek, iş çıkarma oranı, pil tüketimi, uçtan uca gecikme ve ortalama çalışma zamanı kriterlerine göre karşılaştırılmıştır. MQTT-SN, taşıma protokolü olarak UDP kullanılmaktadır. Tipik bir MQTT-SN mesaj alışveriş şeması Şekil 4.3'de gösterilmiştir. Örneğin, Bir MQTT-SN istemcisinden bir aracı veya ağ geçidine gönderilen ilk mesaj, bir CONNACK mesajı göndererek kabul edilir. Bir bağlantı kurulduktan sonra istemci abone olabilir ve mesajlar yayınlayabilir. Şekil 4.4 önerilen MQTT-SN güvenli mesaj gönderme sıralama diyagramı sunulmaktadır. Algılanan veri (veya mesaj) SensorToImplementation işlem kısmında şifrelenir ve Zigbee ağına gönderilir. Alıcıda, paketin kodu sırasına göre çözülür ve düz metin elde edilir.



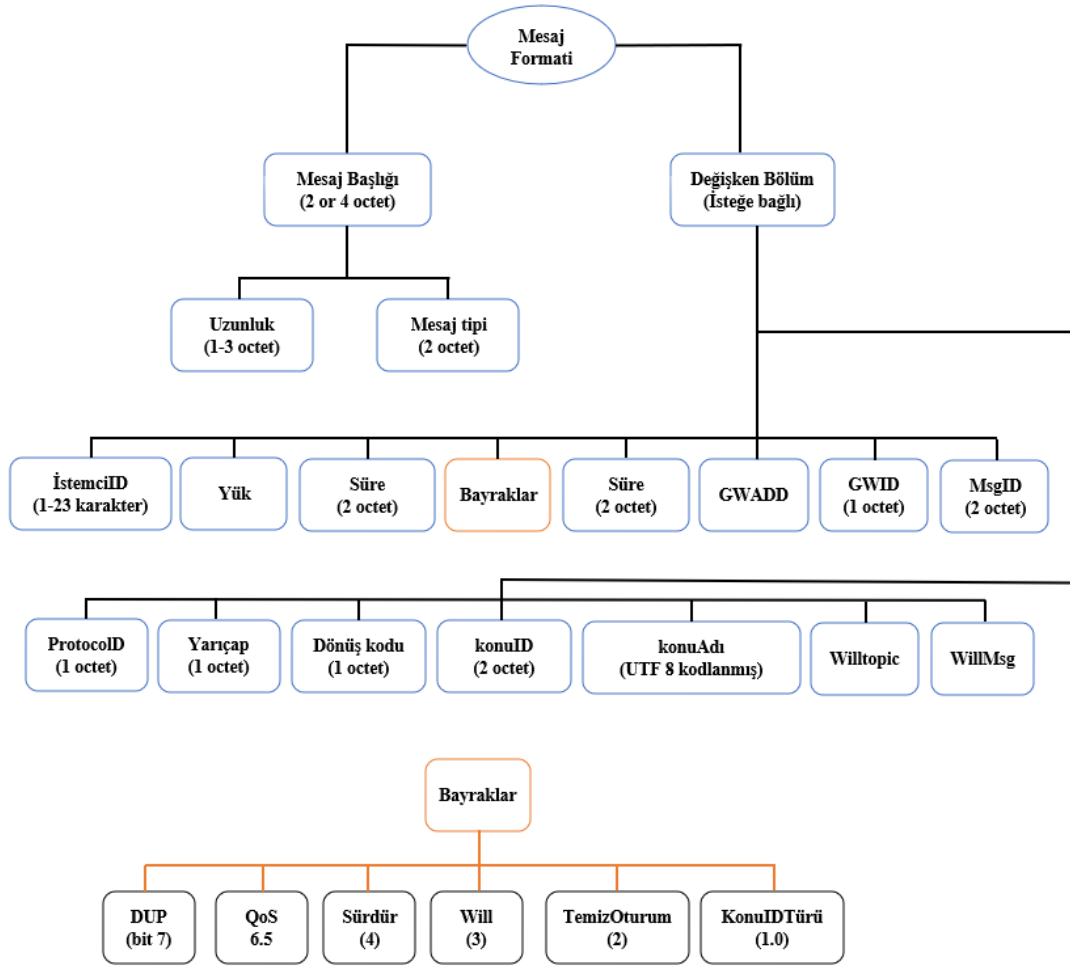
Şekil 4.3. MQTT-SN mesaj alışveriş şeması.



Şekil 4.4. Önerilen MQTT-SN güvenli mesaj gönderme sıralama diyagramı.

4.2.1. Mesaj Formatı

İstemci ve broker birbirleriyle iletişim kurarken, konu, yük vb. içinde nelerin yer alacağı gibi mesajdaki bazı kuralları takip ederler. Şekil 4.5, MQTT-SN mesaj formatını göstermektedir.



Şekil 4.5. MQTT-SN mesaj formatı.

4.2.1.1. Sabit Başlık

MQTT-SN sabit başlığın ilk baytı, mesajın uzunluğudur. İkinci bayt mesaj türüdür. Uzunluk 0 bayttan itibaren dikkate alınacaktır.

4.2.1.2. Değişken Başlık

DUP, Süredür (Retain), WILL, TemizOturum (CleanSession) alanları MQTT ile aynıdır. İhtiyaca göre 0 veya bir olarak ayarlanırlar. Flag'in 0 ve 1'inci biti, konu kimliği türünü belirtmek için kullanılacaktır.

- Normal konu için 0, 1 bit 00 olacaktır.
- Önceden kimliği 0 tanımlanmış olan konu için 1 bit 01 olacaktır
- Küçük konu (Adı 0 olan) için 1 bit 10 olacaktır

5, 6. bit, hizmet kalitesini belirtmek için kullanılacaktır.

- QoS 0 - 5, 6 bit 00 olmalıdır
- QoS 1 - 5, 6 bit 01 olmalıdır
- QoS 2 - 5, 6 bit 10 olmalıdır

KonuidTürü	TemizOturum	Will	Sürdür	QoS	DUP
0, 1	2	3	4	5, 6	7

Şekil 4.6. MQTT-SN değişken paket formatı.

4.2.2. MQTT-SN Kontrol Paket Tipleri

MQTT-SN kontrol paket tipleri, Tablo 4.2'de gösterilen değerlerden biri olan 1 bayt uzunluğundadır.

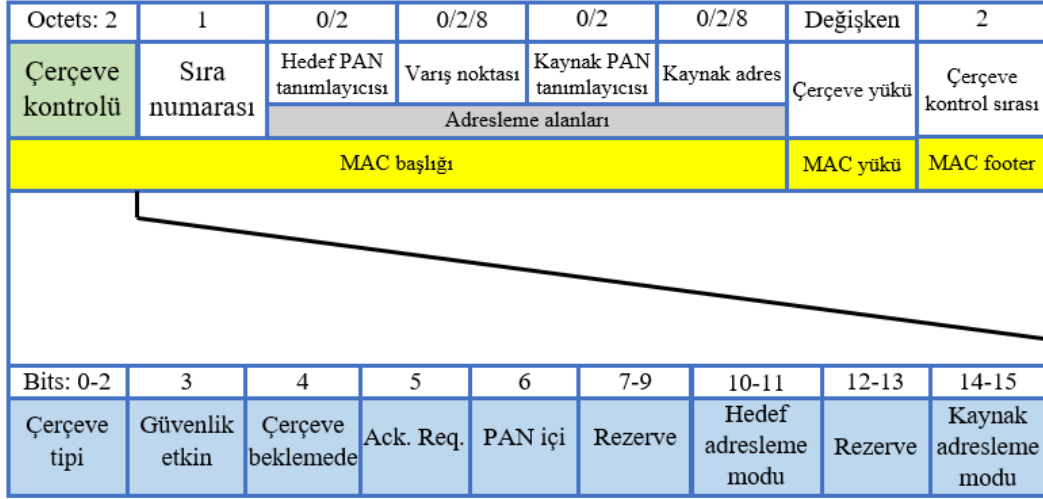
Tablo 4.2. MQTT-SN kontrol paket tipleri.

Alan değeri	Mesaj	Alan değeri	Mesaj
0x00	ADVERTISE	0x01	SEARCHGW
0x02	GWINFO	0x03	reserved
0x04	CONNECT	0x05	CONNACK
0x06	WILLTOPICREQ	0x07	WILLTOPIC
0x08	WILLMSGREQ	0x09	WILLMSG
0x0A	REGISTER	0x0B	REGACK
0x0C	PUBLISH	0x0D	PUBACK
0x0E	PUBCOMP	0x0F	PUBREC
0x10	PUBREL	0x11	reserved
0x12	SUBSCRIBE	0x13	SUBACK
0x14	UNSUBSCRIBE	0x15	UNSUBACK
0x16	PINGREQ	0x17	PINGRESP
0x18	DISCONNECT	0x19	reserved
0x1A	WILLTOPICUPD	0x1B	WILLTOPICRESP
0x1C	WILLMSGUPD	0x1D	WILLMSGRESP
0x1E-0xFD	reserved	0xFE	Encapsulated message
0xFF	reserved		

4.3. Önerilen Protokolün Benzetimi ve Performans Değerlendirmesi

Bu bölümde, Zigbee ağları için kullanıcı istekleri veya artan bit hata oranı gibi çeşitli uygulama güvenlik gereksinimlerine göre dört aşamalı bir güvenlik düzeyi (FS-SSL) önerilmiştir. MAC çerçeve kontrol alanının rezerve bitleri (9-12-13 bit) kullanarak, seçilen 4 algoritmanın (AES dahil) C kodu simülatöre eklenip, kullanıcıya beş moddan (Şifresiz veya diğer 4 algorithmadan arasından birini seçmek) birini seçmesine imkân

tanımlanmıştır. Kullanıcı seçimi durumunda beş mod seçilebilir (Tablo 4.3'de gösterilmiştir). Mod seçimleri iki şekilde yapılabilir: Kullanıcı tarafından Riverbed Modeler ağ simülatöründeki düğüm özellikler menüsü aracılığıyla (Gerekli C kodları simülatöre eklenmiştir) veya sistem tarafından dinamik olarak çapraz katmanlı mimari aracılığıyla bit hata oranının artışı durumunda uygulanacaktır. Şekil 4.7 Zigbee'nin genel MAC çerçeve formatı verilmektedir.



Şekil 4.7. Zigbee genel MAC çerçeve formatı.

Tablo 4.3. Kullanıcının MAC katmanında seçebileceği güvenlik seviyeleri.

Mod	Açıklama
0	Düz metin göndermek
1	AES
2	RECTANGLE
3	Fantomas
4	Camellia

Genellikle, uygulama katmanında verilerin şifrelenmesinden sonra, şifrelenen veriler ağ üzerinden gönderilmek üzere fiziksel katmana iletilir. Bu durumda, ağın BER [111] değerinin artması veya kullanıcı tarafından şifreleme algoritmasının değiştirilmesi talebi olması durumunda, şifrelenmiş veriyi uygulama katmanına tekrar şifrelemek için göndermek yerine, MAC katmanına yönlendirilir. BER'in azaltılması ağ ömrünü uzatır ve ayrıca düğüm enerji tüketimini azaltır [112]. Bir kullanıcının fiziksel katmanda başka bir şifreleme algoritması seçmesi için en olası senaryo ağ durumudur (Trafik yükü, güvenlik derecesi vb.). Zigbee MAC başlığındaki çerçeve kontrol alanının ayrılmış bitlerini kullanarak, kullanıcı veya sistem dinamik olarak (Fiziksel katmanda BER artış varsa, sistem daha düşük ağırlıklı bir algoritmaya tekrar şifreleyecektir), talep edilen güvenlik seviyesine göre hafif siklet şifreleme algoritmalarından birini seçebilir. Algoritmaların ağırlığı Tablo 4.4'de tanımlanmıştır.

MAC katmanındaki her bir algoritmanın ayrılmış bitler aracılığıyla ağırlığı, tez ‘de elde edilen simülasyon sonuçları, hesaplanan iş çıkarma oranı ve enerji tüketimi gibi metriklere dayalı olarak tanımlanmıştır. AES, seçilen algoritmalar arasında en yüksek güvenliğe sahiptir. Ancak daha fazla kaynak tüketmektedir. Güvenlik seviyesinin kullanıcı için daha az kritik olduğu zaman, ancak ağ ömrü ve pil kullanımı gibi sınırlı kaynakların verimli ve uzun ömürlü olması gerektiği durumlarda çözüm, hafif algoritmalarlardır. Tez çalışma kapsamında olan senaryoda kullanıcı, güvenlik gereksinimlerine göre uygulama katmanı üzerinden algoritmayı seçmektedir. Ağ aktifken, çapraz katman mekanizması tarafından kullanılan şifreleme algoritması, bit hata oranına (Fiziksel katmandan algılanan veriler) göre Tablo 4.4’de verilen algoritmalarından değiştirilebilir.

Tablo 4.4. Katmanlar arası güvenlik seviyesi seçmek.

Ağırlık	Açıklama
0	Düz metin göndermek
1	Camellia
2	Fantomas
3	RECTANGLE
4	AES



5. MQTT-SN TEMELLİ GÜVENLİ AĞ PROTOKOLÜNÜN BENZETİMİ

Tez çalışmasında önerilen MQTT-SN temelli güvenli ağ protokolünün benzetimi ve performans değerlendirmesi Riverbed Modeler ve Contiki (Cooja) simülatörleri kullanılarak gerçekleştirilmiştir. Riverbed Modeler simülatöründe, Zigbee ağındaki MICAz algılayıcılar üzerinde farklı performans değerleri (Dört hafif şifreleme algoritmaların C kodu MAC katmanına ilave edilmiştir) incelenmiştir. Bu simülatör, Windows ortamında çalıştırılmıştır.

Linux ortamında Contiki (Cooja) simülatörde seçilen dört hafif şifreleme algoritmasının C kodu Zolertia Z1 algılayıcılar içine gömülmüş ve performansı çeşitli kriterler açısından karşılaştırılmıştır. Bu bölümde, bir önceki bölümde algılayıcı ağlar için önerilen güvenlik çözümlerin uygulamaları ve simülasyonları sunulmaktadır.

5.1. Riverbed Modeler Uygulaması

Önerilen yaklaşımın performans değerlendirmeleri iki ayrı ortamda gerçekleştirilmiştir: MICAz algılayıcı üzerinde ve Riverbed Modeler 18.5.1 ağ simülatörü. Şifreleme ve şifre çözme çalıştırma süresi, enerji tüketimi ve iş çıkarma oranı gibi değerler Şekil 5.1’de görüldüğü gibi MICAz algılayıcı kullanarak hesaplanmıştır. Bellek kullanımını (RAM ve ROM) şifreleme ve şifre çözme modlarında hesaplamak için Microchip Studio’da dört algoritmanın C kodu Atmel ATmega 128L’e (MICAz mikro denetleyicisi) gömüldükten sonra bellek kullanım değerleri ölçülmüştür. Mesajları ve anahtar değişkenlerini tanımlayan ATmega 128L mikro denetleyicinin RAM’inde gerçekleştirilmiştir. Böylece, her blok şifrenin C kodu 8 bitlik ATmega 128L platformda en az RAM miktarını kullanacaktır. Her senaryo en az beş kere tekrarlanmıştır. Atmel ATmega 128L mikro denetleyici 128Kbyte flash belleğe, 4Kbyte EEPROM, 4Kbyte SRAM ve 16MHz’e kadar hıza sahip, düşük güç tüketimli bir denetleyicidir. Donanımsal özellikleri nedeniyle IoT çözümleri için tercih edilmektedir.

İkinci aşamada, ağ tarafı ölçümleri için, seçilen algoritmaların performansı Zigbee (802.15.4 standardı) ağ üzerinden iş çıkarma oranı, uçtan uca gecikme ve enerji

tüketimi açısından Windows 10 Pro'da, Riverbed Modeler 18.5.1 üzerinde gerçekleştirilmiştir. Senaryolar, farklı mesaj boyutları ve düğüm sayılarına göre tanımlanmıştır. Riverbed Modeler' de, kablosuz algılayıcı ağların enerji tüketimini ölçecek bir araç bulunmamaktadır. Bunun için Riverbed Modeler simülatöründe Open-ZB simülasyon modeli kullanılmıştır. Hafif siklet şifrelerin tüm kaynak kodları C dilinde optimize edilerek Open-ZB MAC katman düğüm modeline entegre edilmiştir. Bölüm 4'de belirtildiği gibi, kullanıcıya "Şifrenmemiş" modu veya dört algoritmadan birini seçme yeteneği vermek için çerçeve kontrol alanından (MAC katmanı üzerinden) beş rezerve bit kullanılmıştır. Tablo 5.1, Riverbed Modeler'daki simülasyon parametrelerini göstermektedir. Düğümler 100*100 metrekaare bir alanda rastgele dağılmıştır. Senaryolar, farklı mesaj boyutlarına 5, 10, 25 ve 50 düğüm üzerinde en az 5 defa çalıştırılmıştır.

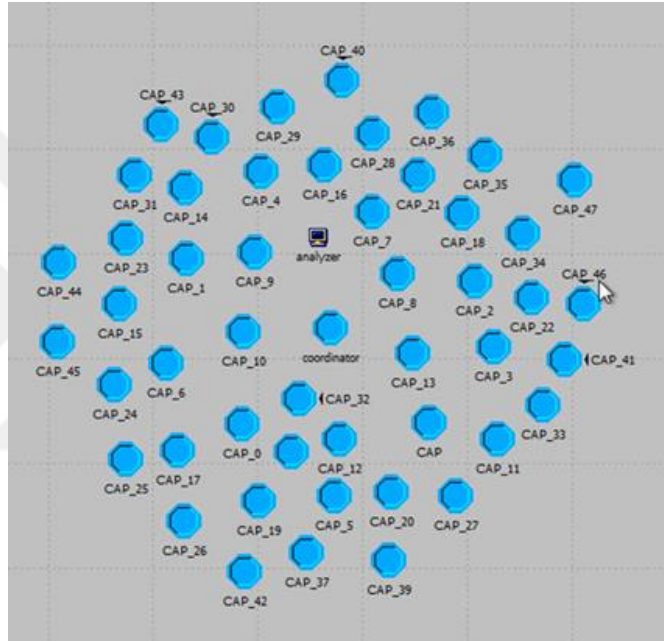
Tablo 5.1. Ağ simülasyon parametreleri.

Parametre	Değer
Düğüm sayısı	3, 5, 10, 25, 50
Algılayıcı tipi	MICAz
Algılayıcı iletim aralığı	100 feet
Veri hızı	256 kbit/s
Mesaj boyutu	16, 32, 64, 128, 256, 512, 1024 Bayt
Düğümlerin frekansı	2.4 GHz
Başlangıç enerjisi	2 AA (1.5 V, 1600mAh)
Alan boyutu	100*100 m
Simülasyon süresi	30 dakika
Seed	128
Yol kayıp modeli	Free Space

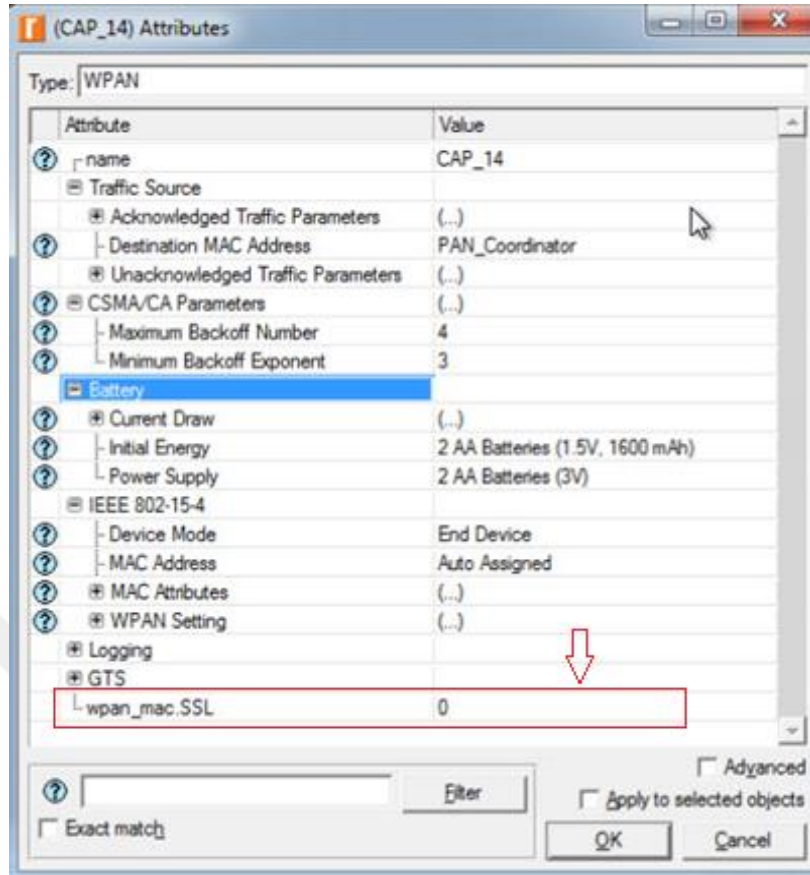
Şekil 5.1 ve Şekil 5.2 sırasıyla MICAz algılayıcı ve örnek Riverbed Modeler'de 50 düğümlük ağ yapısı simülasyonu sunmaktadır. Şekil 5.3, Riverbed Modeler simülatöründeki düğüm özellikleri (Attribute) menüsünün örnek bir görünümünü göstermektedir. Aşağıda gösterildiği gibi, kullanıcının tanımlı modlardan birini seçmesini sağlayan düğüm özellikleri menüsüne bir "wpan_mac.SSL" alanı eklenmiştir. Şekil 5.4 Open-ZB düğüm modelini göstermektedir.



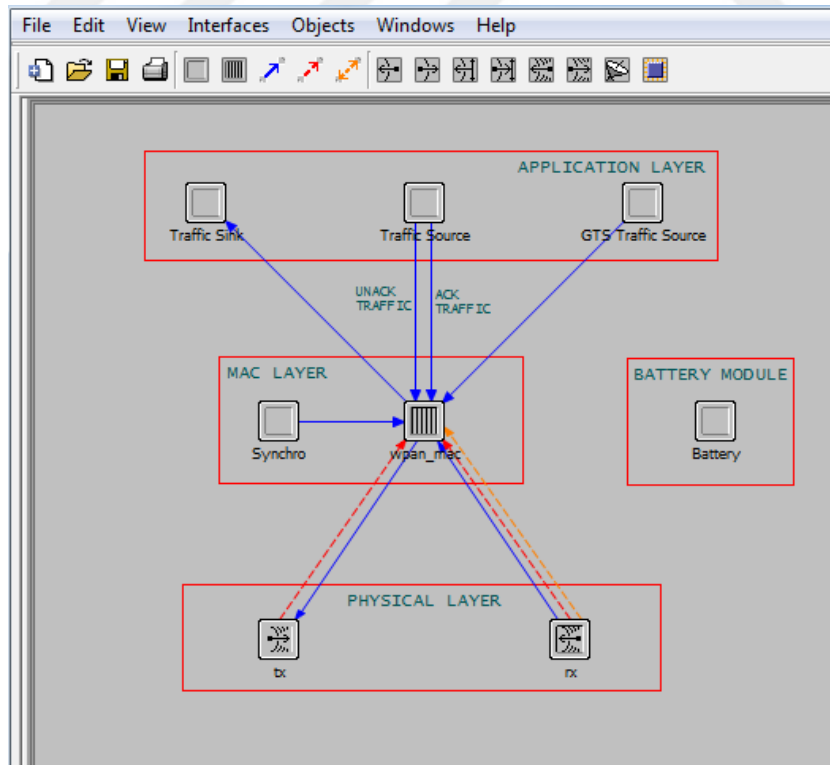
Şekil 5.1. MICAz algılayıcı.



Şekil 5.2. Riverbed Modeler’ de örnek 50 düğümlük ağ yapısı.

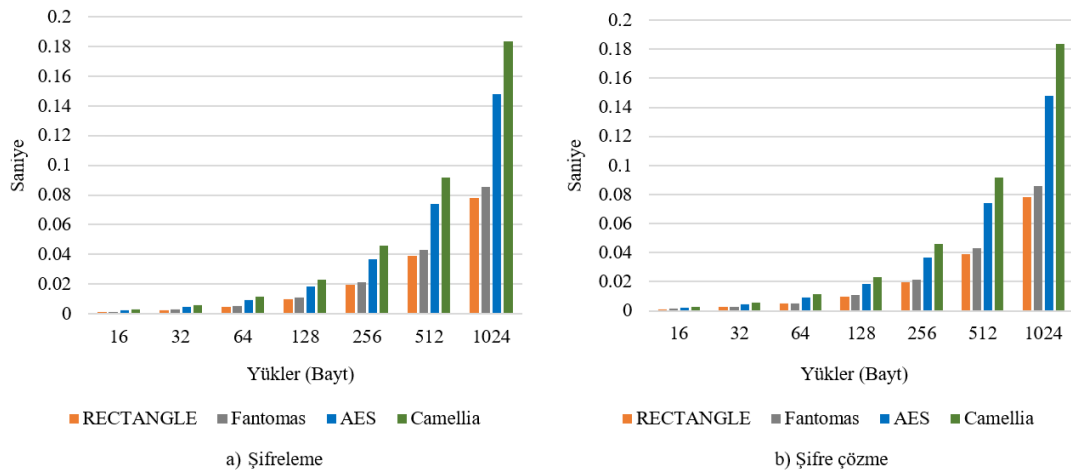


Şekil 5.3. Riverbed Modeler' da bir düğüm için örnek özellik menüsü.



Şekil 5.4. Open-ZB düğüm modeli.

Şekil 5.5 (a) ve Şekil 5.5 (b), sırasıyla Atmel ATmega 128L üzerinden şifreleme ve şifre çözme işlemleri için dört algoritma arasındaki ortalama çalışma sürelerini karşılaştırır. Çalıştırma sürelerini ölçmek için, şifreleme ve şifre çözme anahtar zamanlaması, şifreleme ve şifre çözme fonksiyonları temel kriter olarak ele alınmış ve her senaryo MICAz (Atmel ATmega 128L) algılayıcı ve Microchip Studio üzerinden beş iterasyon için tekrarlanmıştır. İki grafik dahil, RECTANGLE, tüm mesaj boyutları için minimum ve Camellia maksimum çalışma sürelerine sahiptir. RECTANGLE, bit dilimleme tekniğini kullanır (Bu teknik yazılım uygulamaları için çok verimlidir), bu nedenle daha yüksek bir yazılım çalıştırma hızına sahiptir. Fantomas'ın S-Box algoritması aynı şekilde bit-dilimleme şeklinde uygulanmıştır. Şifreleme ve şifre çözme işlemleri için RECTANGLE ve Fantomas çalışma sürelerinin tüm mesaj boyutlarında yakın olduğu gözükmemektedir. Fantomasın dahili LS tasarımı [8], RECTANGLE ile karşılaştırıldığında, çalışma süreleri açısından ikinci sıraya yerleştirdi. AES, Camellia'nın aksine şifreleme ve şifre çözme modları için yaklaşık bir buçuk kat daha hızlı çalışma sürelerine sahiptir çünkü Camellia şifresi, her altı turdan sonra mantıksal fonksiyonlar ve anahtar beyazlatma işlemleri için ek bir tur kullanır.



Şekil 5.5. Ortalama çalıştırma süreleri.

ATmega 128L platformu ve ağ uygulamaları için Şekil 5.6'da farklı mesaj boyutları için şifreleme ve şifre çözme iş çıkarma oranı karşılaştırması sunulmaktadır. Grafikler, tüm yükler için en küçüğünden maksimum veriye kadar olan değerleri temsil etmektedir. İş çıkarma oranını hesaplamak için aşağıdaki denklemler kullanılmıştır:

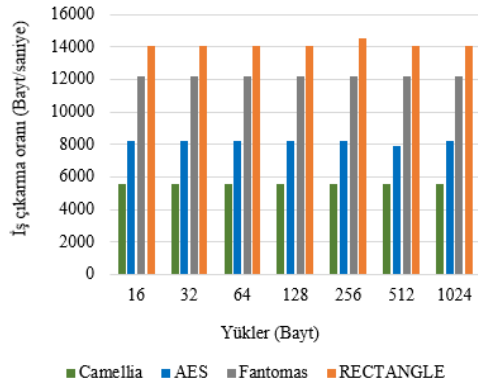
$$\text{Şifreleme iş çıkarma oranı (MICAz)} = \frac{N_B}{T_e} \quad (5.1)$$

$$\text{Şifre çözme iş çıkarma oranı (MICAz)} = \frac{N_B}{T_d} \quad (5.2)$$

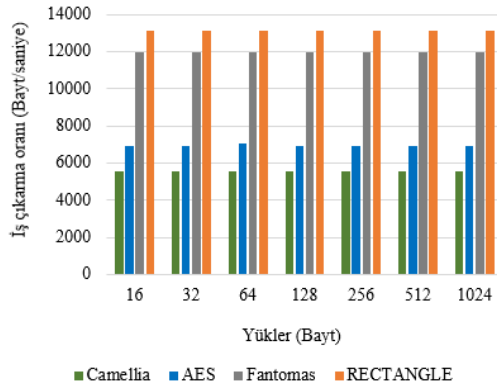
$$\text{İş çıkarma oranı (Ağ)} = \frac{N_D * P}{T} \quad (5.3)$$

Denklem 5.1 ve 5.2'de N_B , blok boyutu sayısıdır, T_e ve T_d sırasıyla şifreleme ve şifre çözme fonksiyonlar için çalıştırma süresini temsil etmektedir. Denklem 5.3'de (Riverbed Modeler), N_D teslim edilen paketlerin sayısıdır, P paket boyutu anlamına gelir ve T ise toplam simülasyon süresidir.

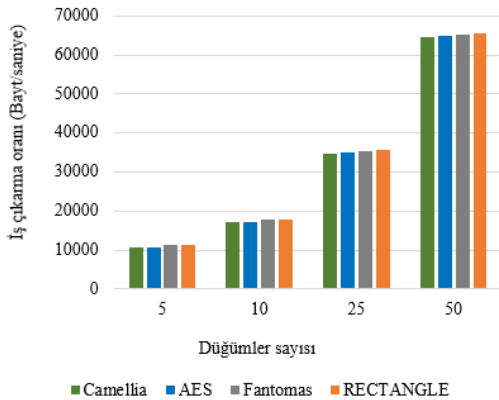
Hafif siklet şifreleri değerlendirmek için iş çıkarma oranı ve enerji tüketimi iki ana kriterdir. Şekil 5.6 (a) ve Şekil 5.6 (b), ATmega 128L (MICAZ algılayıcı) üzerindeki tüm yükler için en küçükten maksimum veriye kadar ortalama değerleri temsil eder. Bu kısmın çıktısını hesaplamak için (5.1) denklemini kullanılmıştır. Camellia algoritması, her iki grafikteki tüm mesaj boyutları için en düşük iş çıkarma oranına sahiptir. Camellia'dan sonra AES ve Fantomas, RECTANGLE (Tüm yükler için en hızlı çalışma süresine sahiptir) algoritmasını takip etmektedir.



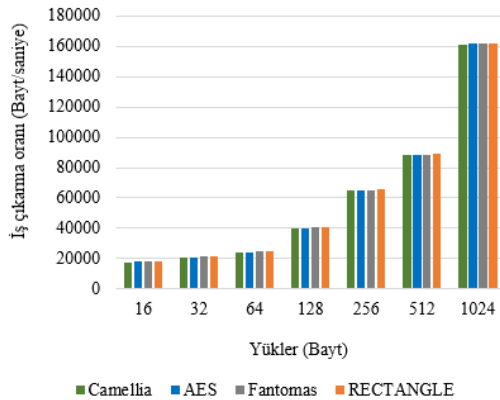
a) Atmel ATmega 128L ortalama iş çıkarma oranı (Şifreleme)



b) Atmel ATmega 128L ortalama iş çıkarma oranı (Şifre çözme)



c) 256 baytlık mesajın ortalama iş çıkarma oranı (Ağ simülasyonu)



d) 50 düğüm için ortalama iş çıkarma oranı (Ağ simülasyonu)

Şekil 5.6. Farklı senaryolar arasında iş çıkarma oranı karşılaştırması.

Şekil 5.6 (c) ve Şekil 5.6 (d), 256 bayt şifreli veri aktarımını ve 50 düğüm üzerindeki aktarım hızını göstermektedir. Senaryoları iki modda gerçekleştirmiştir: Güvensiz (Düz metin gönderme) ve güvenli iletim (Şifreli metin gönderme). Modelleyici bir metin dosyasından düz metin okur ve şifreli metin oluşturur.

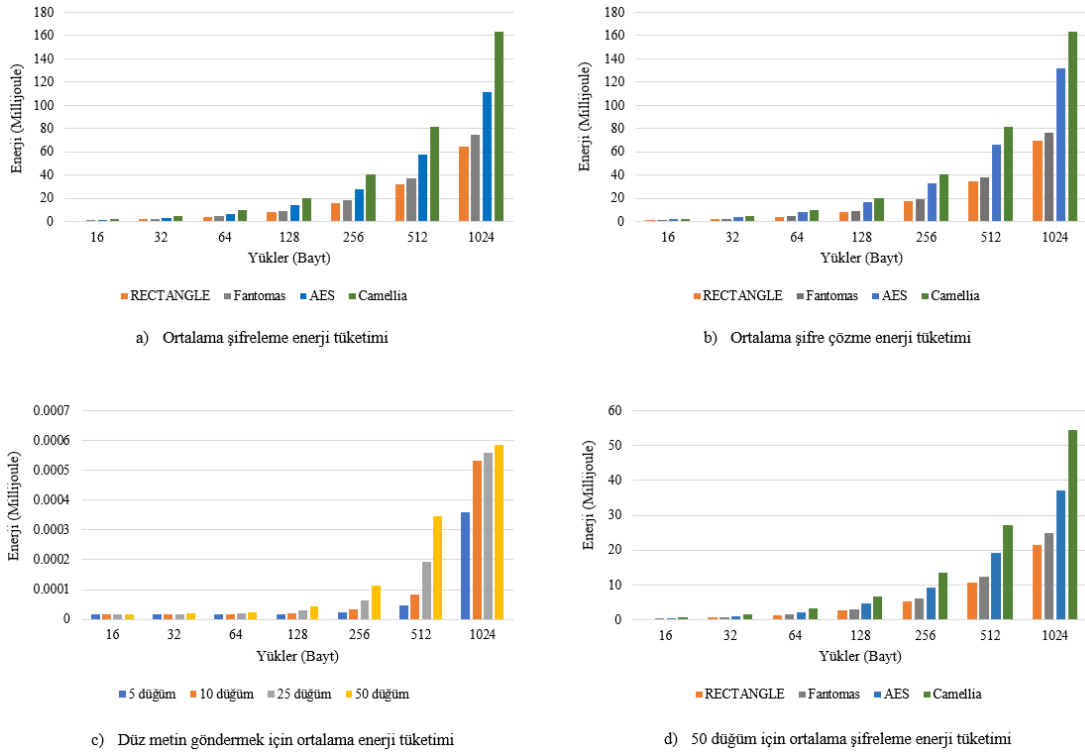
Her senaryo beş iterasyon için çalıştırılmış ve birden çok düğüm için ortalama iş çıkarma oranı hesaplanmıştır. Her iki grafik için beklendiği gibi, RECTANGLE diğer algoritmalar arasında en yüksek iş çıkarma oranına ve Camellia minimum değere sahiptir. 256 baytlık bir mesaj için, düğüm sayısındaki artış, özellikle Şekil 5.6 (c) beş düğümden sonra, iş çıkarma oranı değerini iki katına çıkarır. Camellia, tüm yükler için en düşük iş çıkarma oranı değerine sahiptir. Diğer yandan en yüksek ortalama iş çıkarma oranı RECTANGLE algoritmasına aittir. Sırasıyla tüm faydalı mesaj boyutları için en küçükten, Camellia, AES, Fantomas ve RECTANGLE olarak düzenlenebilir. En yüksek iş çıkarma oranı, daha verimli ağ anlamına gelir.

Ortalama enerji tüketimi Şekil 5.7'de görünmektedir. MICAz (İki AA pil ile çalışır) algılayıcı üzerinden enerji tüketimini hesaplamak için voltaj ve amper değerleri bir multimetre kullanarak ölçülmüştür. MICAz algılayıcı için tüketilen enerji aşağıdaki denklemler ile hesaplanmıştır.

$$I * t = Q \quad (5.4)$$

$$Q * V = E \quad (5.5)$$

Denklem 5.4'te, "I" mevcut akımdır, "t" zamanı ve "Q" yükü temsil eder. Denklem 5.5'te, "V" voltaj anlamına gelir ve "E", jul cinsinden enerji tüketimini gösterir. Daha önce de belirtildiği gibi, enerji tüketimi çalışma süresi ile yüksek oranda ilişkilidir.



Şekil 5.7. Ortalama enerji tüketimi karşılaştırması.

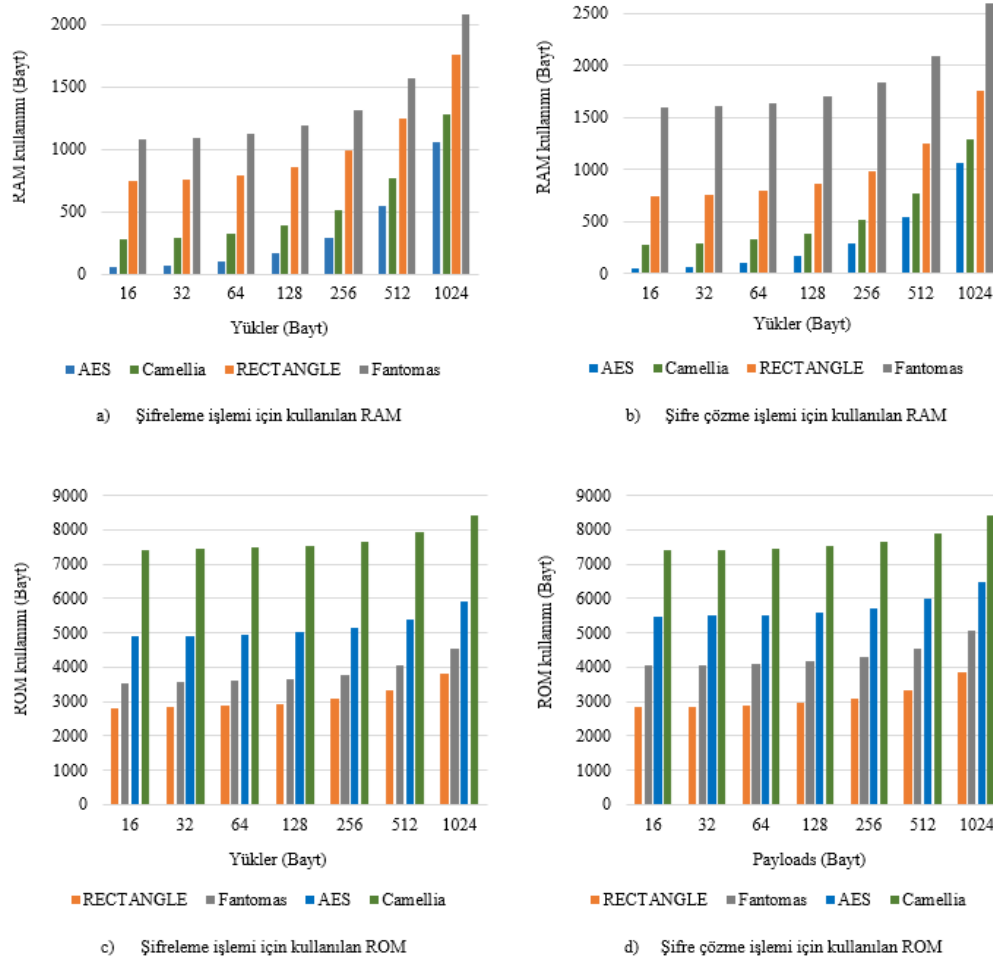
Ağ tarafında (Şekil 5.7 (c) ve Şekil 5.7 (d)), Open-ZB simülasyon modelinin pil modülü, düğümlerin tüketilen ve kalan enerji seviyelerini ölçebilir. Şekil 5.7 (a) ve 5.7 (b)'deki çubuk grafiklerin (MICAZ algılayıcı üzerinden) yüksekliklerindeki farklılıklara bakarak, RECTANGLE'ın tüm faydalı yükler için şifreleme ve şifre çözme işlemleri için diğer algoritmalarla kıyasla en düşük miktarda enerji tüketimi izlenmektedir. Çalışma süresi grafikleriyle karşılaştırıldığında, beklendiği gibi Camellia, özellikle 512 bayttan sonra daha fazla enerji tüketir ve 1024 bayt'ta maksimum değere ulaşmaktadır. Şekil 5.7 (d), ağ üzerinden düz metin göndermek için ortalama enerji tüketimini temsil eder. 256 bayta kadar enerji tüketiminin tüm düğümler için biraz arttığı görülmektedir. 512 bayt için artış hızı daha da yükselir, ancak 1024 bayt için enerji tüketimi çarpıcı biçimde artar. 50 düğümden oluşan Zigbee ağının enerji tüketimini Şekil 5.7 (d)'de temsil edilmiştir. RECTANGLE algoritması diğer blok şifrelere kıyasla minimum enerji tüketir (Şekil 5.7 (d)). Tüm taşıma yükleri için Camellia maksimum enerji tüketimine sahiptir.

RAM ve ROM kullanım miktarı, Microchip Studio programında ölçülmüştür. RAM kullanım miktarı, başlatılmış veri (Veri segmenti) artı başlatılmamış verinin (BSS segmenti) toplamıdır. Blok şifrelerin RAM boyutu, Round anahtarın boyutundan dolayı değişmektedir. RAM kullanımını optimize etmek ve azaltmak için cihazın

(Atmel ATmega 128L) RAM bölmesinde, başlatma (Initialize) vektörü, Round anahtarı, anahtar ve düz metin tanımlanmıştır.

Şekil 5.8 (a) ve Şekil 5.8 (b), şifreleme ve şifre çözme işlemleri için RAM kullanımlarını göstermektedir. AES, şifreleme ve şifre çözme grafiklerinde en iyi RAM kullanımına sahiptir ve 11 Round anahtara sahiptir (Her biri 128 bit uzunluğunda). Kullanıcı tarafından temin edilen şifreleme anahtarı, AES algoritması için yuvarlak anahtarları sağlamaktadır. AES'den sonra Camellia ikinci sırada yer alıyor. Camellia-128 algoritmasında ana anahtar ve 26 adet yuvarlak anahtar (Her biri 64-bit) mevcuttur. Round anahtarları, anahtar zamanlamasını kullanarak elde edilir. RECTANGLE üçüncü sırada yer alır ve 128 bit ana anahtara ve 64 bit round anahtara sahiptir. Fantomas, her iki grafik için diğer algoritmalar arasında en yüksek RAM kullanımına sahiptir. Fantomas bir başlatma vektörü ve ana anahtara sahiptir ve herhangi bir anahtar zamanlayıcısı tasarımında yer almıyor.

Şekil 5.8 (c) ve Şekil. 5.8 (d), Atmel ATmega 128L üzerinden seçilen algoritmaların ROM (Kod boyutu) kullanımlarını temsil etmektedir. Kod boyutu, sabitler (L, S, P- kutuları, round), yazılım kodu ve arama tablolarını depolamak için kullanılan alandır. Bir blok şifrenin kod boyutu, temel olarak sabitlerin boyutuna ve sayısına göre hesaplanır. Her iki çubuk grafik benzer bir algoritma sıralaması düzenine sahiptir. RECTANGLE en düşük ROM'u kullanır ve Camellia diğer blok şifreler arasında en yüksek ROM kullanımına sahiptir. RECTANGLE, tek bir küçük 4-bite 4-bit S-Box ve 5-bitlik round sabiti kullanır. Fantomas, RECTANGLE'den sonra ikinci sırada yer alıyor ve 8x8 S-Box ve 16*16 bit L-Box'a sahiptir. Fantomas, 3 adet yuvarlak 3/5 bit S-Box uygular ve işlemleri korumak için yuvarlak sabit uygulamaktadır. Fantomas'tan sonra AES üçüncü sırada yer alıyor. AES, tüm turlar (Tüm mesajlarda) için tek bir S-Box kullanır ve S-Box'ı 16'ya 16 eşleme Tablosu olarak görülebilir. 2round_number modulo Galois Alan 28, AES'in round sabitlerinin değerleridir. Camellia, tüm mesaj uzunluklarında maksimum ROM kullanımına sahiptir. Camellia, dört adet 8*8 bit S-Box kullanır.

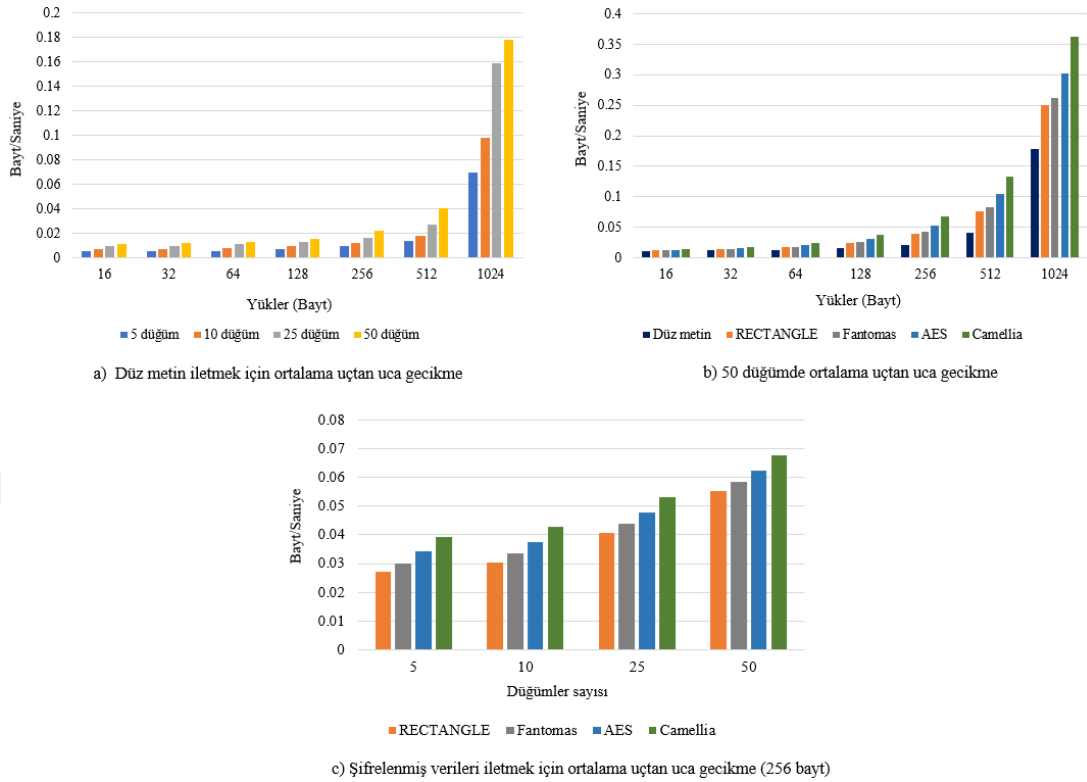


Şekil 5.8. Bellek (RAM ve ROM) kullanımları.

Şekil 5.9, ağ üzerinden güvensiz (Düz metin gönderme) ve güvenli (Şifreli metin gönderme) veri aktarımı için seçilen düğümler arasında uçtan uca gecikme senaryolarını göstermektedir. Uçtan uca gecikme, bir paketin ağ üzerinden kaynaktan hedefe aktarılması için geçen süredir. Uçtan uca gecikme, verilerin boyutundan ciddi bir şekilde etkilenir, bu nedenle mikro denetleyicinin algoritmayı çalıştırmak için ihtiyaç duyduğu süreyi azaltmak çok önemlidir. Şekil 5.9 (a), düz metinleri birden çok düğüm üzerinden iletmek için ortalama uçtan uca gecikmeyi temsil eder. Beklendiği gibi, 256 bayta kadar, grafik, tüm yükler için kademeli olarak yukarı hareketi gösterir. 50 düğüm için uçtan uca gecikme değeri, 1024 bayt sütünü için en üst noktaya ulaşır (Şekil 5.9 (b)). Genel olarak, düğümlerin artması, uçtan uca gecikme miktarında bir artışa yol açar.

RECTANGLE tüm yükler için minimum uçtan uca gecikmeye sahiptir (Şekil 5.9 (b) ve Şekil 5.9 (c)). Fantomas, AES ve Camellia sırasıyla RECTANGLE'i takip ediyor.

Dört algoritmanın uçtan uca gecikme açısından düzenlenmesi, çalışma süresi grafikleriyle eşleşmektedir.



Şekil 5.9. Ortalama uçtan uca gecikme karşılaştırması.

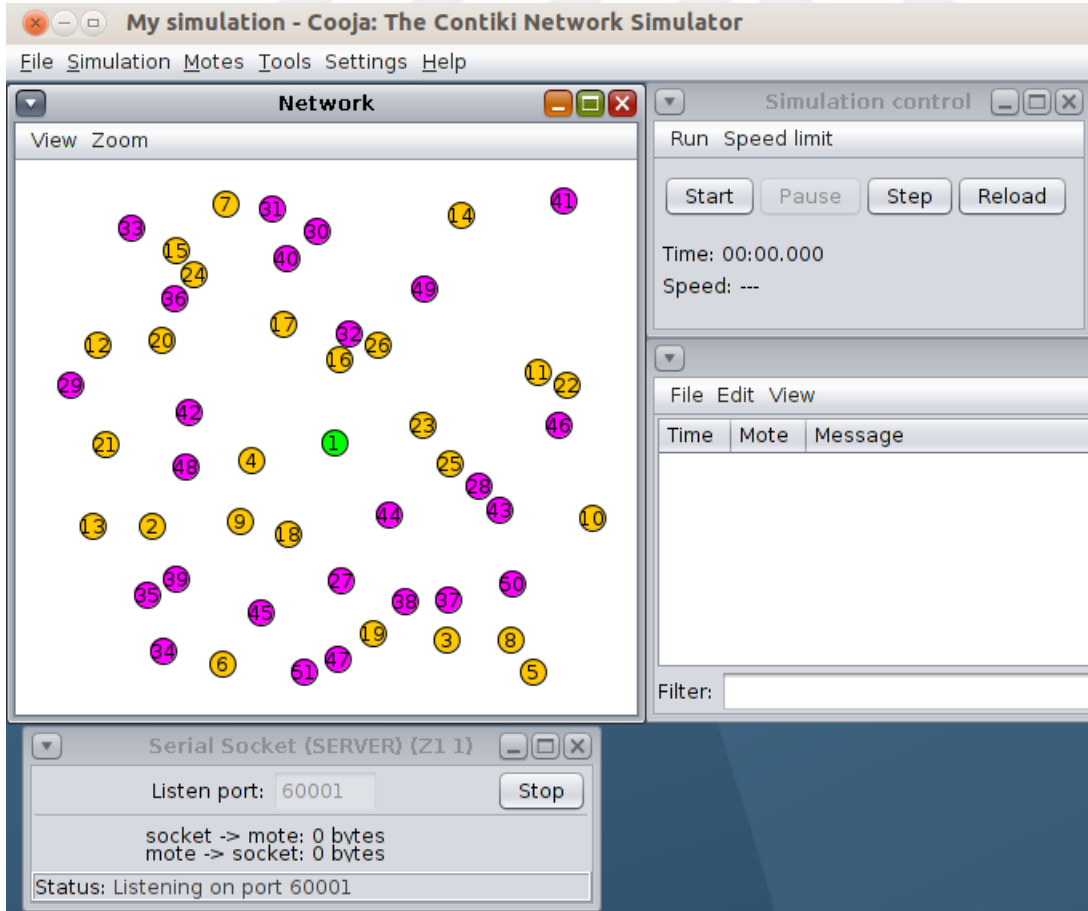
5.2. Contiki-Cooja Linux Uygulaması

Contiki, düşük güçlü kablosuz Nesnelerin İnterneti cihazlarına odaklanan, ağa bağlı, bellek kısıtlamalı sistemler için bir işletim sistemidir. Cooja, kablosuz algılayıcı ağlar için özel olarak tasarlanmış bir ağ simülatörüdür. MQTT-SN haberleşme standardı yaygın olarak kablosuz algılayıcı ağlarında kullanılmaktadır. Bir önceki bölümde MQTT-SN haberleşme protokollüne önerilen uçtan uca şifreleme yöntemi, Linux Ubuntu 14.04 ortamında, Contiki 3.0, Cooja simülatörü kullanarak gerçekleştirilmiştir. Testler Zolertia Z1 algılayıcı üzerinden gerçekleştirilmiştir. Z1, MSP430F2617 ultra düşük güç 16-bit MCU 16 MHz mikro denetleyici ile donatılmıştır. 2.4 GHz IEEE 802.15.4 (Zigbee) çalışıp, 6LowPAN uyumludur. Pille çalışabilir: 2xAA veya AAA pil. Z1, 8KB RAM ve 92KB Flash belleğe sahiptir. Dört (Camellia, AES, RECTANGLE, Fantomas) hafif siklet şifrelerin C kodları mikro denetleyiciye entegre edildi. Böylece, MQTT-SN mesajlaşma protokolde güvenli mod seçildiğinde, uçtan uca şifrelenmiş gerçekleşecektir. Tablo 5.2, simülasyon parametrelerini

göstermektedir. Düğümler 100*100 metrekaire bir alanda rastgele dağılmıştır. Senaryolar, farklı mesaj boyutlarına 5, 10, 25 ve 50 düğüm üzerinde en az 5 defa çalıştırılmıştır. Kablosuz kanal modeli, Unit Disk Graph Medium (UDGM) modelidir. UDGM'de ağ topolojisi, düğüm konumlarını, aktarım aralığını, girişim aralığını ve aktarım oranını (Tx ve Rx) içerir. Cooja Simulator UDGM, yalnızca homojen bir ağ oluşturulduğunda kullanılır.

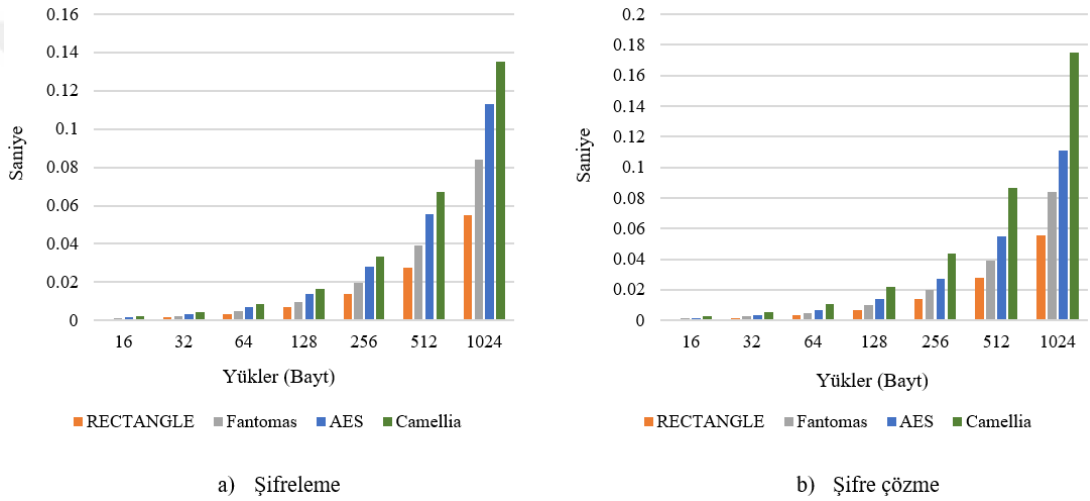
Tablo 5.2. Cooja simülasyon parametreleri.

Parametre	Değer
Kablosuz kanal modeli	UDGM
MAC katmanı	CSMA + ContikiMAC
Protokol	6LoWPAN/RPL
Taşıma protokolü	UDP
Düğüm sayısı	5, 10, 25, 50
Algılayıcı tipi	Zolertia Z1
İletim Aralığı	50 metre
Veri hızı	250 kbit/s
Mesaj boyutu	16, 32, 64, 128, 256, 512, 1024 Bayt
Düğümün frekansı	2.4 GHz
Başlangıç enerjisi	2 AA (1.5 V, 1600mAh)
Simülasyon süresi	10 dakika



Şekil 5.10. Örnek 50 düğüm Cooja simülasyonu.

Şekil 5.11 (a) ve Şekil 5.11 (b), sırasıyla Zolertia Z1 üzerinden şifreleme ve şifre çözme işlemleri için dört algoritma arasındaki ortalama çalıştırma sürelerini karşılaştırır. Çalıştırma sürelerini ölçmek için, Cooja simülatörde algılayıcılar menüsünden Z1 seçildi. Dört hafif şifrenin C kodları Z1 algılayıcının üzerinde derlenmiştir. Böylece tek bir Z1 algılayıcı üzerinde şifreleme ve şifre çözme fonksiyonlarının çalışma süresi elde edilmiştir. Deneyler beş iterasyon için tekrarlanmıştır. Şifreleme ve şifre çözme gafları (Şekil 5.11) dahil, RECTANGLE, tüm mesaj boyutları için minimum ve Camellia maksimum çalışma sürelerine sahiptir. Sırasıyla, RECTANGLE, Fantomas, AES ve Camellia algoritmaların şifreleme ve şifre çözme fonksiyonları çalıştırma süreleri en azdan en yükseğe sıralanmışlar.



Şekil 5.11. Ortalama çalıştırma süreleri.

Zolertia Z1 platformu ve Cooja ağ uygulamaları için Şekil 5.12'de farklı mesaj boyutlarında şifreleme ve şifre çözme iş çıkarma oranlarını karşılaştırılmıştır. Grafikler, tüm yükler için en küçüğünden maksimum veriye kadar olan değerleri göstermektedir. İş çıkarma oranını hesaplamak için aşağıdaki denklemler uygulanmıştır:

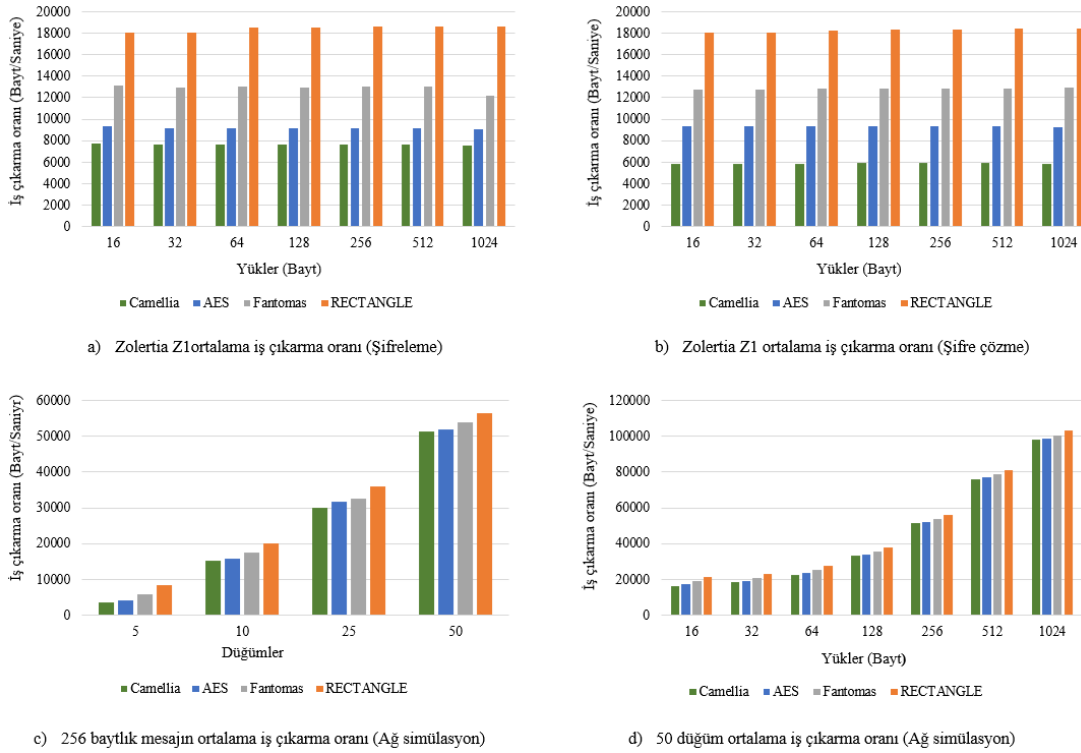
$$\text{Şifreleme iş çıkarma oranı (Zolertia Z1)} = \frac{N_B}{T_e} \quad (5.6)$$

$$\text{Şifre çözme iş çıkarma oranı (Zolertia Z1)} = \frac{N_B}{T_d} \quad (5.7)$$

$$\text{İş çıkarma oranı (Cooja ağ)} = \frac{N_D * P * 8}{T} \quad (5.8)$$

Denklem 5.6 ve 5.7'de N_B , blok boyutudur, T_e ve T_d sırasıyla şifreleme ve şifre çözme fonksiyonlar için çalıştırma süresini temsil etmektedir. Denklem 5.8'de (Cooja

simülasyon), N_D teslim edilen paketlerin sayısıdır, P paket boyutu anlamına gelir ve T ise toplam simülasyon süresidir. Cooja ağ ortamında iş çıkarma oranını hesaplamak için gerekli C kodları komut dosyası düzenleyicisi (Script editor) menüsüne eklenmiştir. RECTANGLE algoritması, her iki grafikteki (5.12 (a) ve (b)) tüm mesaj uzunluklarına en yüksek iş çıkarma oranına sahiptir. RECTANGLE'dan sonra Fantomas, AES ve Camellia yer almaktalar. Her iki grafik için beklendiği gibi, Camellia minimum değere sahiptir. Beklendiği gibi tüm mesaj uzunlukları için Camellia en düşük iş çıkarma oranı değerine sahiptir. Sırasıyla tüm mesaj boyutları için en küçükten, Camellia, AES, Fantomas ve RECTANGLE olarak düzenlenebilir. En yüksek iş çıkarma oranı, daha verimli ağ anlamına gelir.



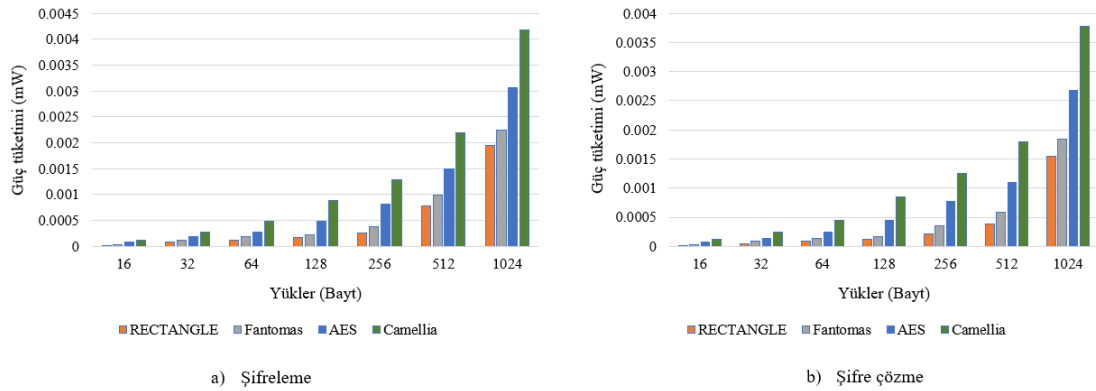
Şekil 5.12. Farklı senaryolar arasında iş çıkarma oranı karşılaştırması.

Ortalama enerji tüketimi Şekil 5.13'de görünmektedir. Cooja simülatöründeki enerji kullanımı ölçümü dört parametreye dayanıyor: İşlemci gücü, düşük güç modu (LPM), dinleme gücü ve iletim gücü. İşlemci gücü, düğüm tarafından gereken hesaplama enerjisini ifade eder. LPM güç, algılayıcı düğümün boş durumdayken kullanılan enerjiyi ifade eder. Dinleme gücü, algılayıcı düğümün komşu düğümlerinden veri paketini almaya hazır olduğu zaman gerekli olan enerjiyi ifade eder. İletim gücü, algılayıcı düğümün veri paketini komşularına iletmek için ihtiyaç duyduğu enerjiyi ifade eder. Enerji tüketimini hesaplamak için, ilk önce Z1 algılayıcısının saniyedeki tik

sayısı kontrol edilmiştir (RTIMER_SECOND = 32768). Coojanın Powertrace uygulaması projenin Makefile dosyasına “APP += powertrace” komutu ile eklenmiştir. Kaynak dosyaya #include "powertrace.h" komutu eklendi. Güç tüketim profili her 10 saniyede bir “powertrace_start (CLOCK_SECOND * 10);” komutu ile ekranda yazılıp ve .log dosyası olarak kaydedilmiştir. Enerji tüketimi hesaplamak için aşağıdaki denklem kullanılmıştır:

$$Z1 \text{ enerji tüketimi} = \frac{\text{Energes_değeri} * \text{Akım} * \text{Voltaj}}{\text{RTIMER_SECOND} * \text{Simülasyon zamanı}} \quad (5.9)$$

Denklem 5.9’da “Energest” değeri elde edilen 10 saniyelik profil değerlerden, işlemci, LPM, TX ve RX değerleri için ayrı hesaplanmaktadır. Voltaj değeri 3 volt, TX akımı 17.4 mA, RX akımı 18.8 mA olarak Zolertia Z1 veri sayfasından referans alınmıştır. Şekil 5.13 (A ve b)’de RECTANGLE tüm mesaj uzunluklar için şifreleme ve şifre çözme işlemlerinde diğer algoritmalara kıyasla en düşük miktarda enerji tüketmektedir. Beklendiği gibi, Camellia, özellikle 512 bayttan sonra daha fazla enerji tüketir ve 1024 baytta maksimum değere ulaşır. 256 bayta kadar enerji tüketiminin tüm düğümler için arttığı görülmektedir ancak 512 bayttan sonra artış hızı daha da yükselir. Şekil 5.13 şifreleme ve şifre çözme işlemlerinde, sırasıyla RECTANGLE, Fantomas, AES ve Camellia güç tüketim değerleri en düşükten en yüksek miktara kadar görülmektedir.

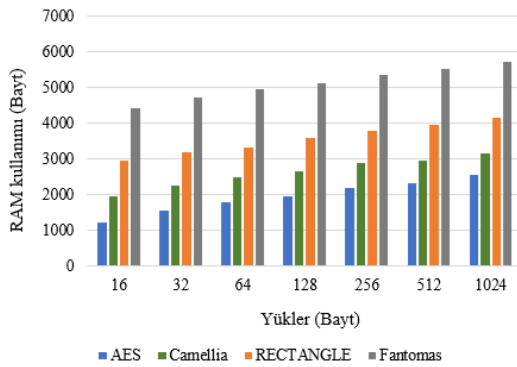


Şekil 5.13. Ortalama güç tüketim karşılaştırması.

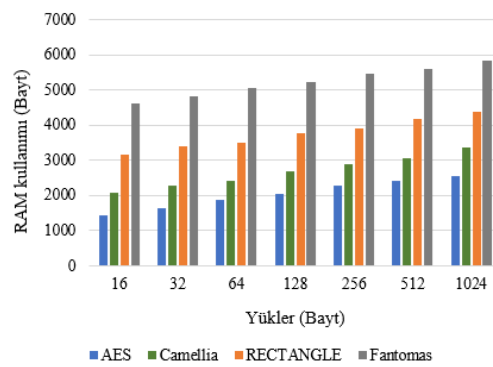
Şekil 5.14 (a) ve Şekil 5.14 (b), şifreleme ve şifre çözme işlemleri için Cooja simülörde RAM kullanımlarını temsil etmektedir. Bellek kullanımını ölçmek için msp430-size komutu kullanılmıştır. RAM kullanım açısından AES algoritması şifreleme ve şifre çözme grafiklerde en az tüketim miktarına sahiptir. Şifreleme ve şifre çözme işlemleri dahil, Fantomas maksimum RAM kullanımına sahiptir. Şekil

5.14 (a) ve 5.14 (b) için AES algoritmasından sonra, Camellia, RECTANGLE, Fantomas yer almaktadır.

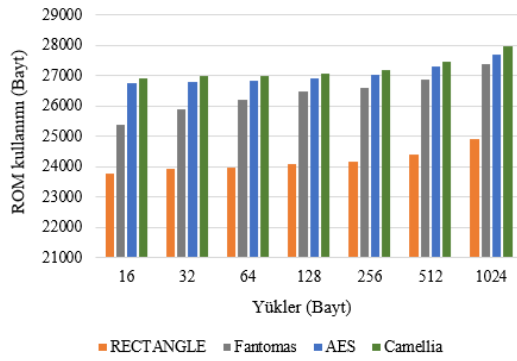
Şekil 5.14 (c) ve Şekil. 5.14 (d), Zolertia Z1 algılayıcının ROM (Kod boyutu) şifreleme ve şifre çözme kullanımlarını dört şifreleme algoritmasını uyguladıktan sonra göstermektedir. Şifreleme ve şifre çözme modlar dahil, RECTANGLE en düşük ROM'u kullanır ve Camellia diğer blok şifreler arasında en yüksek ROM kullanımına sahiptir. Fantomas, RECTANGLE'den sonra ikinci sırada yer almaktadır. En yüksek ROM kullanım şifreleme ve şifre çözme grafiklerinde Camellia algoritmasına aittir. Camellia'dan sonra AES ikinci sırada yer almaktadır.



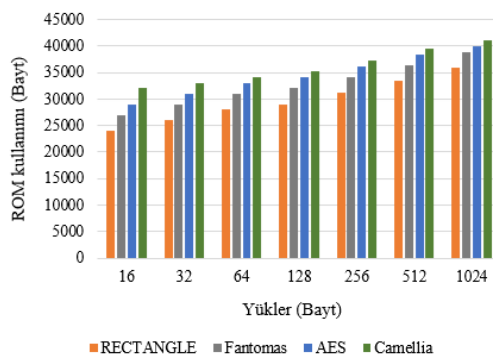
a) Şifreleme işlemi için kullanılan RAM



b) Şifre çözme işlemi için kullanılan RAM



c) Şifreleme işlemi için kullanılan ROM

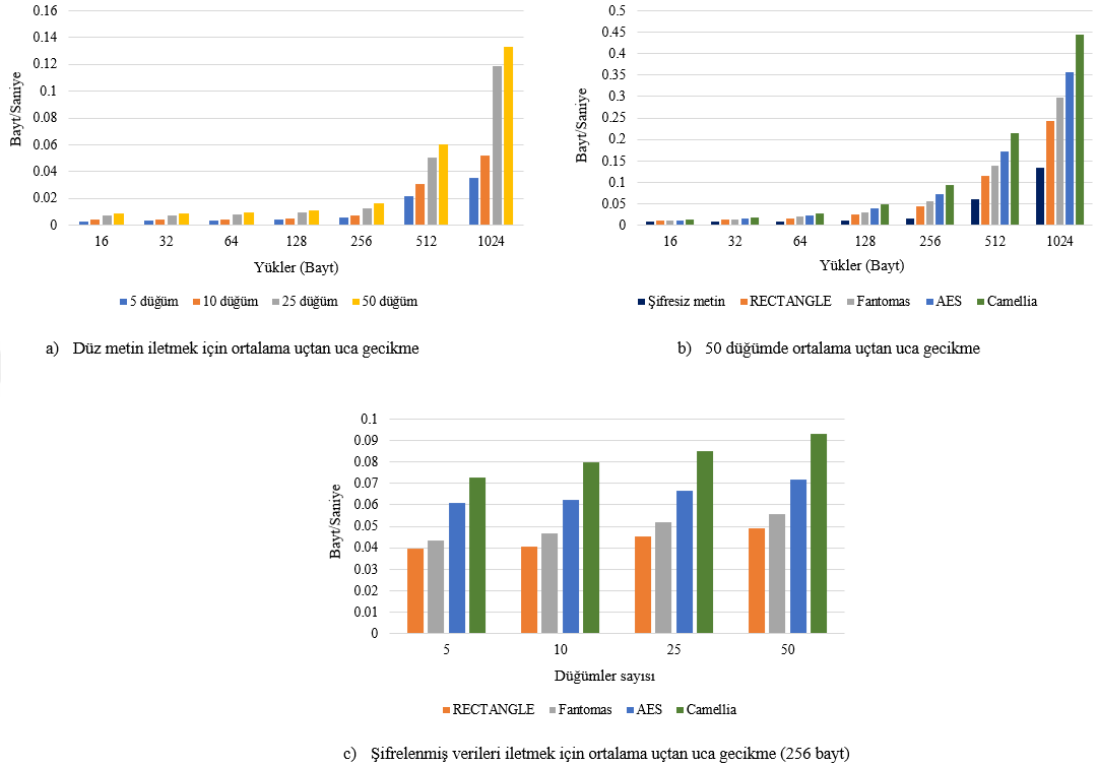


d) Şifre çözme işlemi için kullanılan ROM

Şekil 5.14. Bellek (RAM ve ROM) kullanımı.

Şekil 5.15, Cooja ağ üzerinden güvensiz (Düz metin gönderme) ve güvenli (Şifreli metin gönderme) veri aktarımı için seçilen düğümler arasında uçtan uca gecikme senaryolarını göstermektedir. Şekil 5.15 (a), beklendiği gibi, 256 bayta kadar, grafik, tüm yükler için kademeli olarak yukarı hareketi gösterir. 50 düğüm için uçtan uca gecikme değeri, 1024 bayt sütunu için en üst noktaya ulaşır (Şekil 5.15 (b)). Düğümlerin artması, uçtan uca gecikme miktarında bir artışa yol açmaktadır.

RECTANGLE tüm yükler için minimum uçtan uca gecikmeye sahiptir (Şekil 5.15 (b) ve Şekil 5.15 (c)). RECTANGLE' den sonra Fantomas, AES ve Camellia yer almaktadır. Dört algoritmanın uçtan uca gecikme açısından düzenlenmesi, çalışma süresi grafikleriyle eşleşmektedir.



Şekil 5.15. Ortalama uçtan uca gecikme karşılaştırması.

5.3. Bölüm Sonuçları

Bölüm 4’de önerilen çözümler, bu bölümde Windows ve Linux ortamlarında iki farklı simülasyon uygulamasında, iki ayrı algılayıcı (MICAz ve Zolertia Z1 algılayıcılar) üzerinde (Kablosuz algılayıcı ağlar ve MQTT-SN mesajlaşma protokolü) gerçekleştirilmiştir. Hafif şifreleme algoritmaları adı geçen algılayıcılarda uygulanarak uçtan uca şifreleme sağlanmıştır. Şifreleme ve şifre çözme gafları dahil, RECTANGLE, tüm mesaj boyutları için minimum ve Camellia maksimum çalışma sürelerine sahiptir. Camellia en düşük iş çıkarma oranı değerine sahiptir. Sırasıyla tüm mesaj boyutları için en küçükten, Camellia, AES, Fantomas ve RECTANGLE olarak düzenlenebilir. Şifreleme ve şifre çözme işlemlerinde, sırasıyla RECTANGLE, Fantomas, AES ve Camellia güç tüketim değerleri en düşükten en yüksek miktara

kadar görülmektedir. Şifreleme ve şifre çözme işlemleri dahil, Fantomas maksimum RAM kullanımına sahiptir. RAM kullanım açısından, AES algoritmasından sonra, Camellia, RECTANGLE, Fantomas yer almaktadır. En yüksek ROM kullanım şifreleme ve şifre çözme grafiklerinde Camellia algoritmasına aittir. Camellia'dan sonra AES ikinci sırada yer almaktadır. RECTANGLE en düşük ROM'u kullanıyor. Uçtan uca gecikme açısından, RECTANGLE' den sonra Fantomas, AES ve Camellia en yüksek değerlere sahipler.



6. SONUÇLAR VE ANALİZLER

Bu doktora tezinde, IoT'nin alt kümesi olarak kabul edilen kablosuz algılayıcı ağları için hafif siklet şifreleme algoritmaları kullanarak, güvenli veri iletimi sağlanmıştır. Düşük güçlü IoT cihazlarda geleneksel şifreleme yöntemler ve algoritmaları doğrudan uygulamak zordur. Şifrelemek ve kodunu çözmek için çok sayıda tur gerektirir ve cihazlarda bulunan sınırlı enerjiyi boşa harcar. Hafif şifreleme algoritmalar, düşük maliyetli, saldırılara karşı dayanıklılığı, daha az enerji tüketen, daha küçük paket boyutu olan ve daha az hesaplama yüküne sahiptir.

Hafif algoritmalar hızlı bir şekilde uygulanabilir ve kaynak kısıtlamalı algılayıcı düğümleri için uygundur. AES 128 algoritması, bazı saldırılara karşı zayıflıkları olan Kablosuz Algılayıcı Ağlar (KAA) için kullanılabilen tek şifreleme algoritmasıdır. Farklı güvenlik taleplerine göre, hafif siklet şifreleme algoritmalar, uçtan uca şifreleme için önemli ölçüde tercih edilen çıkar yoludur.

Tezin amacı, IoT nesnelerin çalışması dikkate alınarak, yeni ve özgün haberleşme teknolojilerin geliştirmesi içermektedir. Bu doğrultuda, MQTT-SN haberleşme protokolü Kontrol Paketinde rezerve olan bitleri kullanarak, kullanıcı şifresiz veya şifreli modlarda tanımlanan hafif siklet algoritmalarından birini seçebilir. MQTT-SN mesaj paketi formatında 3 rezerve MsgType bit alanını kullanarak adı geçen 3 algoritma MQTT-SN mesaj paketinde tanımlanmıştır. Mevcut hafif siklet algoritmalar arasında üç algoritma (RECTANGLE, Fantomas, Camellia) seçildi ve performans değerleri AES 128-bit algoritmasına karşı Linux (Cooja simülatör) ve Windows (Riverbed (OPNET) Modeler), iki farklı algılayıcı üzerinde (MICAz ve Zolertia Z1) ölçülmüştür. Windows'da her blok şifre tarafından kullanılan bellek (RAM & ROM) miktarını analiz etmek için Microchip Studio 7 kullanılmıştır. Linux ortamında bellek kullanım ölçümü msp430-size komutu ile gerçekleştirilmiştir.

Performans değerlendirmeler, farklı senaryolar için algılayıcılar üzerinde ve ağ ortamında bellek kullanımı, iş çıkama oranı, pil tüketimi ve uçtan uca gecikmeye dayalı olarak yapıldı ve karşılaştırmıştır. Daha yüksek güvenlik, daha az karmaşıklık, daha az enerji tüketimi vb. gibi ayırıcı özellikler bir şifreleme algoritma seçiminde önde

gelen kriterlerdir. AES, seçilen algoritmalar arasında en yüksek güvenliğe sahiptir. Ancak daha fazla kaynak tüketmektedir. Güvenlik seviyesinin kullanıcı için daha az kritik olduğu zaman, ancak ağ ömrü ve pil kullanımı gibi sınırlı kaynakların verimli ve uzun ömürlü olması gerektiği durumlarda çözüm, hafif algoritmalarlardır. Camellia, ISO/IEC organizasyonu tarafından onaylanmıştır ve AES ile karşılaştırılabilir işlem yeteneklerine ve güvenlik seviyelerine sahiptir. RECTANGLE ve Fantomas, bitslice tekniğini kullanan yüksek hızlı yazılım uygulamasına sahiptir ve bu özellik onları çevrimiçi (Online) uygulamalar için iyi bir seçim haline getirmektedir.

Riverbed Modeller ve Cooja ortamlarında yapılan simülasyonlar sonucunda, ortalama çalıştırma süresini karşılaştırmada en düşükten en yükseğe algoritmalar RECTANGLE, Fantomas, AES ve Camellia olarak sıralanmaktadır. RECTANGLE, MICAz ve Z1 algılayıcılarda en hızlı şekilde şifrele ve şifre çözme fonksiyonları çalıştırır. Camellia listenin sonunda yer almaktadır.

Cooja ve Riverbed simülatörlerde MICAz ve Z1 algılayıcılar üzerinde yapılan simülasyon sonuçlarına göre, ortalama iş çıkarma oranı (Şifreleme ve şifre çözme işlemleri dahil) dört algoritma Camellia, AES, Fantomas ve RECTANGLE olarak sıralanmaktadır. RECTANGLE her iki cihazda listenin sonunda yer almaktadır. Camellia algoritmalar arasında en iyi iş çıkarma oranına sahiptir.

Şifreleme ve şifre çözme işlemlerinde iki algılayıcı için, ortalama güç tüketim sıralaması RECTANGLE, Fantomas, AES, Camellia şeklindedir. Beklendiği gibi RECTANGLE en az enerji tüketmektedir ve Camellia mevcut algoritmalar arasında en yüksek güç tüketimine sahiptir.

Dört algoritmanın RAM kullanımını açısından Windows ve Linux ortamlarında yapılan ölçmeler sonucunda algoritmalar AES, Camellia, RECTANGLE ve Fantomas olarak sıralamıştır. Şifreleme ve şifre çözme işlemleri dahil, AES minimum ve Fantomas maksimum RAM kullanımına sahipler.

Linux ve Windows'da elde edilen ROM kullanım değerlere göre RECTANGLE en az ve Camellia en çok ROM kullanıyor. MICAz ve Z1 algılayıcılar üzerinde yapılan ölçümlerde dört algoritma sırasıyla en azdan çoğa doğru RECTANGLE, Fantomas, AES ve Camellia olarak görülmektedir.

MICAz ve Z1 algılayıcıları kullanarak mesaj uzunluğuna göre ortalama uçtan uca gecikme sıralamasını RECTANGLE, Fantomas, AES ve Camellia olarak

görülmektedir. RECTANGLE algoritması için uçtan uca gecikme değerleri minimum ve Camellia en yüksek uçtan uca gecikme değerlere sahipler.

Tez çalışmasında çeşitli hafif siklet blok şifreler incelenip, IoT uygulama geliştiriciler için donanımlara uygun algoritma seçimi için yol gösterilmiştir. Ayrıca, Önerilen şema, IoT'nin bir alt kümesi kabul edilen KAA'lar için yazılım tabanlı bir güvenlik çözümü olarak adil bir ölçüm ve karşılaştırma yapmaktadır. Kullanıcı istekleri veya artırılmış BER gibi çeşitli uygulama güvenlik gereksinimlerine göre önerilen şema uçtan uca şifreleme yöntemi ile hassas verilerin gizliliğini korunmaktadır.





KAYNAKLAR

- [1] Doç. Dr. Cüneyt BAYILMIŞ ve Doç. Dr. Kerem KÜÇÜK- Nesnelerin İnternet'i: Teori ve Uygulamaları- Papatya Bilim- pp.240 – 2019.
- [2] Vaidya, B.; Mouftah, H.T.: IoT applications and services for connected and autonomous electric vehicles. Arab. J. Sci. Eng. 45, 2559–2569 (2020).
- [3] Goyal, Krishan Kumar, Amit Garg, Ankur Rastogi, and Saurabh Singhal. "A literature survey on Internet of Things (IoT)." International Journal of Advanced Networking and Applications 9, no. 6 (2018): 3663-3668.
- [4] <https://www.owasp.org/images/1/1c/OWASP-IoT-Top-10-2018-final.pdf>
- [5] Babar, Sachin, Parikshit Mahalle, Antonietta Stango, Neeli Prasad, and Ramjee Prasad. "Proposed security model and threat taxonomy for the Internet of Things (IoT)." In International Conference on Network Security and Applications, pp. 420-429. Springer, Berlin, Heidelberg, 2010.
- [6] Souissi, I., Azzouna, N. B., & Said, L. B. (2019). A multi-level study of information trust models in WSN-assisted IoT. Computer Networks, 151, 12-30
- [7] Burhanuddin, M. A., Ali Abdul-Jabbar Mohammed, Ronizam Ismail, Mustafa Emad Hameed, Ali Noori Kareem, and Halizah Basiron. "A review on security challenges and features in wireless sensor networks: IoT perspective." Journal of Telecommunication, Electronic and Computer Engineering (JTEC) 10, no. 1-7 (2018): 17-21.
- [8] Grosso V., Leurent G., Standaert FX., Varıcı K. (2015) LS-Designs: Bitslice Encryption for Efficient Masked Software Implementations. In: Cid C., Rechberger C. (eds) Fast Software Encryption. FSE 2014. Lecture Notes in Computer Science, vol 8540. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-46706-0_2
- [9] Zhang, W., Bao, Z., Lin, D., Rijmen, V., Yang, B., & Verbauwhede, I. (2015). RECTANGLE: a bit-slice lightweight block cipher suitable for multiple platforms. Science China Information Sciences, 58(12), 1-15.
- [10] Aoki K, Ichikawa T, Kanda M, Matsui M, Moriai S, Nakajima J, Tokita T. Camellia: A 128-bit block cipher suitable for multiple platforms—design and analysis. In International workshop on selected areas in cryptography 2000 Aug 14 (pp. 39-56). Springer, Berlin, Heidelberg.
- [11] Isobe T., Shibutani K. (2012) Security Analysis of the Lightweight Block Ciphers XTEA, LED and Piccolo. In: Susilo W., Mu Y., Seberry J. (eds) Information Security and Privacy. ACISP 2012. Lecture Notes in Computer Science, vol 7372. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-31448-3_6
- [12] M. Cazorla, K. Marquet and M. Minier, "Survey and benchmark of lightweight block ciphers for wireless sensor networks," 2013 International Conference on Security and Cryptography (SECRYPT), Reykjavik, Iceland, 2013, pp. 1-6.

- [13]Panahi, P., Bayılmış, C., Çavuşoğlu, U. et al. Performance Evaluation of Lightweight Encryption Algorithms for IoT-Based Applications. Arab J Sci Eng (2021). <https://doi.org/10.1007/s13369-021-05358-4>
- [14]Deebak, B. D., & Al-Turjman, F. (2020). A hybrid secure routing and monitoring mechanism in IoT-based wireless sensor networks. Ad Hoc Networks, 97, 102022.
- [15]Gopalakrishnan, K. (2020). Security Vulnerabilities and Issues of Traditional Wireless Sensors Networks in IoT. In Principles of Internet of Things (IoT) Ecosystem: Insight Paradigm (pp. 519-549). Springer, Cham.
- [16]Pawar, R., & Kalbande, D. R. (2019, October). Elliptical Curve Cryptography Based Access Control Solution for IoT Based WSN. In International Conference on Innovative Data Communication Technologies and Application (pp. 742-749). Springer, Cham.
- [17]Maw, Htoo Aung, Hannan Xiao, Bruce Christianson, and James A. Malcolm. "A survey of access control models in wireless sensor networks." Journal of Sensor and Actuator Networks 3, no. 2 (2014): 150-180.
- [18]Tabassum, A., Sadaf, S., Sinha, D., & Das, A. K. (2020). Secure Anti-Void Energy-Efficient Routing (SAVEER) Protocol for WSN-Based IoT Network. In Advances in Computational Intelligence (pp. 129-142). Springer, Singapore.
- [19]Poongavanam, N., Lathaparthiban, P., Vadivelu, K., Sathiyamoorthi, H., & Balaji, N. (2018). A SURVEY ON ATTACKS AND SECURITY GOALS IN WSN. International Journal of Pure and Applied Mathematics, 119(14), 1641-1646.
- [20]Hassan, Nabaa A., and Alaa K. Farhan. "Security improve in ZigBee protocol based on RSA public algorithm in WSN." Engineering and Technology Journal 37, no. 3B (2019): 67-73.
- [21]Ghani, A., Mansoor, K., Mehmood, S., Chaudhry, S. A., Rahman, A. U., & Najmus Saqib, M. (2019). Security and key management in IoT-based wireless sensor networks: An authentication protocol using symmetric key. International Journal of Communication Systems, 32(16), e4139.
- [22]Li, X., Niu, J., Kumari, S., Wu, F., Sangaiah, A. K., & Choo, K. K. R. (2018). A three-factor anonymous authentication scheme for wireless sensor networks in internet of things environments. Journal of Network and Computer Applications, 103, 194-204.
- [23]Abdulkader, O., Bamhdi, A. M., Thayananthan, V., Jambi, K., & Alrasheedi, M. (2018, February). A novel and secure smart parking management system (SPMS) based on integration of WSN, RFID, and IoT. In 2018 15th Learning and Technology Conference (L&T) (pp. 102-106). IEEE.
- [24]Othman, A., & Maga, D. (2018, October). Relation between security and energy consumption in wireless sensor network (WSN). In 2018 New Trends in Signal Processing (NTSP) (pp. 1-8). IEEE.
- [25]Zirem, A., & Senouci, M. R. (2018, October). Efficient lightweight chaotic secure communication system for WSNs and IoT. In 2018 International Conference on Smart Communications in Network Technologies (SaCoNeT) (pp. 43-48). IEEE.

- [26] Abbou, A. N., Baddi, Y., & Hasbi, A. (2018, October). Software defined networks in internet of things integration security: Challenges and solutions. In 2018 6th International Conference on Wireless Networks and Mobile Communications (WINCOM) (pp. 1-6). IEEE.
- [27] Dwivedi, R. K., Saran, M., & Kumar, R. (2019, January). A survey on security over sensor-cloud. In 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence) (pp. 31-37). IEEE.
- [28] Teguig, E. H., & Touati, Y. (2018, November). Security in Wireless Sensor Network and IoT: An Elliptic Curves Cryptosystem based Approach. In 2018 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON) (pp. 526-530). IEEE.
- [29] Smys, S. (2019). Energy-aware security routing protocol for WSN in big-data applications. *Journal of ISMAC*, 1(01), 38-55.
- [30] Abidin, S. (2018, August). Enhancing Security in WSN by Artificial Intelligence. In *International Conference on Intelligent Data Communication Technologies and Internet of Things* (pp. 814-821). Springer, Cham.
- [31] Abirami, S. (2019). A complete study on the security aspects of wireless sensor networks. In *International Conference on Innovative Computing and Communications* (pp. 223-230). Springer, Singapore.
- [32] Dewal, P., Narula, G. S., Jain, V., & Baliyan, A. (2018). Security Attacks in Wireless Sensor Networks: A Survey. In *Cyber Security* (pp. 47-58). Springer, Singapore.
- [33] Mugheri, A. A., Siddiqui, M. A., & Khoso, M. (2018). Analysis on Security Methods of Wireless Sensor Network (WSN). *Sukkur IBA Journal of Computing and Mathematical Sciences*, 2(1), 52-60.
- [34] Patil, S., & Lilhare, P. (2019). Security Issues and Challenges in Wireless Sensor Networks. *Proceedings of Recent Advances in Interdisciplinary Trends in Engineering & Applications (RAITEA)*.
- [35] Tsai, K. L., Leu, F. Y., Su, T. H., & Chang, Y. C. (2017, November). A light weight data encryption method for WSN communication. In *International Conference on Broadband and Wireless Computing, Communication and Applications* (pp. 788-795). Springer, Cham.
- [36] Sumalatha, M. S., & Nandalal, V. (2020). An intelligent cross layer security based fuzzy trust calculation mechanism (CLS-FTCM) for securing wireless sensor network (WSN). *Journal of Ambient Intelligence and Humanized Computing*, 1-15.
- [37] Haseeb, K., Almogren, A., Islam, N., Ud Din, I., & Jan, Z. (2019). An energy-efficient and secure routing protocol for intrusion avoidance in IoT-based WSN. *Energies*, 12(21), 4174.
- [38] Sharma, Kalpana, and M. K. Ghose. "Cross layer security framework for wireless sensor networks." *International Journal of Security and Its Applications* 5, no. 1 (2011): 39-52.
- [39] Gawdan, Idrees Sarhan, Chee-Onn Chow, Tanveer A. Zia, and I. Gawdan. "Cross-layer based security solutions for wireless sensor networks." *International Journal of Physical Sciences* 6, no. 17 (2011): 4245-4254.

- [40] Mendes, Lucas DP, and Joel JPC Rodrigues. "A survey on cross-layer solutions for wireless sensor networks." *Journal of Network and Computer Applications* 34, no. 2 (2011): 523-534.
- [41] Devi, C. Dhivya, and K. Vidya. "A survey on cross-layer design approach for secure wireless sensor networks." In *International conference on innovative computing and communications*, pp. 43-59. Springer, Singapore, 2019.
- [42] Boubiche, D.E., Athmani, S., Boubiche, S. et al. Cybersecurity Issues in Wireless Sensor Networks: Current Challenges and Solutions. *Wireless Pers Commun* 117, 177–213 (2021). <https://doi.org/10.1007/s11277-020-07213-5>
- [43] Khashan, Osama A., Rami Ahmad, and Nour M. Khafajah. "An automated lightweight encryption scheme for secure and energy-efficient communication in wireless sensor networks." *Ad Hoc Networks* 115 (2021): 102448.
- [44] Kardi A., Zagrouba R. (2021) Hybrid Cryptography Algorithm for Secure Data Communication in WSNs: DECRSA. In: Sharma H., Saraswat M., Yadav A., Kim J.H., Bansal J.C. (eds) *Congress on Intelligent Systems. CIS 2020. Advances in Intelligent Systems and Computing*, vol 1334. Springer.
- [45] Wang J, Gao Y, Yin X, Li F, Kim HJ. An enhanced PEGASIS algorithm with mobile sink support for wireless sensor networks. *Wireless Communications and Mobile Computing*. 2018 Dec 2;2018.
- [46] Tabatabaei S. A Novel Fault Tolerance Energy-Aware Clustering Method via Social Spider Optimization (SSO) and Fuzzy Logic and Mobile Sink in Wireless Sensor Networks (WSNs). *Comput. Syst. Sci. Eng.*. 2020;35(6):477-94.
- [47] Wang J, Gao Y, Zhou C, Sherratt S, Wang L. Optimal coverage multi-path scheduling scheme with multiple mobile sinks for WSNs. *Computers, Materials & Continua*. 2020 Jan 10;62(2):695-711.
- [48] Vijayalakshmi K, Anandan P. Global levy flight of cuckoo search with particle swarm optimization for effective cluster head selection in wireless sensor network. *Intelligent Automation and Soft Computing*. 2020 Jan 1;26(2):303-11.
- [49] Radosavljević N, Babić D. Power Consumption Analysis Model in Wireless Sensor Network for Different Topology Protocols and Lightweight Cryptographic Algorithms. *Journal of Internet Technology*. 2021 Jan 1;22(1):71-80.
- [50] Lara-Nino CA, Diaz-Perez A, Morales-Sandoval M. Energy and area costs of lightweight cryptographic algorithms for authenticated encryption in WSN. *Security and Communication Networks*. 2018 Sep 4;2018.
- [51] Anisha Mahato, Dr. M. Pushpalatha "A Lightweight Cryptosystem for Wireless Sensor Networks using ECC" *International Journal of Engineering Trends and Technology* 68.4(2020):7-12.
- [52] Kalyane SJ, Patil NB. Lightweight & Energy Efficient Secure Data Transmission in WSN. *International Journal of Advanced Networking and Applications*. 2019 Sep 1;11(2):4205-12.
- [53] Dutta, I. K., Ghosh, B., Bayoumi, M. 2019. "Lightweight Cryptography for Internet of Insecure Things: A Survey", *IEEE 9th Annual Computing and Communication Workshop and Conference*, 475-481.

- [54] Malina, L., Hajny, J., Fujdiak, R., Hosek, J. 2016. "On perspective of security and privacy-preserving solutions in the internet of things", *Computer Networks*, 102, 83-95.
- [55] Sciancalepore, S., Piro, G., Vogli, E., Boggia, G., Grieco, L.A., Cavone, G. 2016. "LICITUS: A lightweight and standard compatible framework for securing layer-2 communications in the IoT", *Computer Networks*, 108, 66-77.
- [56] Raza, S., Shafagh, H., Hewage, K., Hummen, R., Voigt, T. 2013. "Lithe: Lightweight Secure CoAP for the Internet of Things", *IEEE Sensor Journal*, 13(10), 3711-3720.
- [57] Singh, M., MA, R., VL, S., Balamuralidhar, P. 2015. "Secure MQTT for Internet of Things (IoT), 2015 Fifth International Conference on Communications Systems and Network Technologies, 746-751.
- [58] Mektoubi, A., Hassani, H. L., Belhadaoui, H., Rifi, M., Zakari, A. 2016. "New approach for securing communication over MQTT protocol A comparison between RSA and Elliptic Curve", *Third International Conference on System of Collaboration (SysCo)*, 1-6.
- [59] Panahi, P., Bayılmış, C., Çavuşoğlu, U., Kaçar, S. 2019. "Comparing PRESENT and LBlock block ciphers over IoT Platform", *12th International Conference on Information Security and Cryptology*, 66-69.
- [60] Panahi, P., Bayılmış, C., Çavuşoğlu, U., Kaçar, S. 2018 "Performance Evaluation of L-Block Algorithm for IoT Applications", *3. Uluslararası Bilgisayar Bilimleri ve Mühendisliği Konferansı (UBMK2018)*, 609-612.
- [61] Bayilmis, C., Cavusoglu, U., Akgul, A., Kacar, S., Sevin, A. 2017. "Enhanced Secure Data Transfer for WSN Using Chaotic-Based Encryption", *Technical Gazette*, 24(4), 1065-1069.
- [62] Bayilmis, C., Cavusoglu, U., Akgul, A., Sevin, A., Kacar, S. 2014. "Employing Chaotic Encryption for IEEE 802.15.4-based LR-WPANs", *International Conference on Computer Science and Information Systems (ICSIS'2014)*, 89-92.
- [63] Çakıroğlu, M., Bayılmış, C., Özcerit, A.T., Çetin, Ö. 2010. "Performance evaluation of scalable encryption algorithm for wireless sensor networks", *Scientific Research and Essays*, 5(9), 856-861.
- [64] Bayılmış, C., Çakıroğlu, M. 2008. "SEA Şifreleme Algoritması Kullanarak Güvenli Kablosuz Algılayıcı Ağ Haberleşmesinin Gerçekleştirilmesi", *3. Uluslararası Katılımlı Bilgi Güvenliği ve Kriptoloji Konferansı*, 173-177.
- [65] Bandırmalı, N., Ertürk, İ., Çeken, C., Bayılmış, C. 2008. "Skipjack Şifreleme Algoritması Kullanarak Gecikme Duyarlı ve Enerji Etkin Kablosuz Algılayıcı Ağ Güvenlik Hizmeti", *Elektrik-Elektronik-Bilgisayar Mühendisliği Sempozyumu ELECO 2008*, 152-157.
- [66] Uras Panahi, Cüneyt Bayılmış, "Enabling secure data transmission for wireless sensor networks based IoT applications", *Ain Shams Engineering Journal*, 2022, 101866, ISSN 2090-4479, <https://doi.org/10.1016/j.asej.2022.101866>.
- [67] Hassan, Syed Ali, Sidra Shaheen Syed, and Fatima Hussain. "Communication technologies in IoT networks." In *Internet of Things*, pp. 13-26. Springer, Cham, 2017.

- [68] Soni, Dipa, and Ashwin Makwana. "A survey on mqtt: a protocol of internet of things (iot)." In International conference on telecommunication, power analysis and computing techniques (ICTPACT-2017), vol. 20, pp. 173-177. 2017.
- [69] Park, Chang-Seop, and Hye-Min Nam. "Security architecture and protocols for secure MQTT-SN." IEEE Access 8 (2020): 226422-226436.
- [70] Vinoski, Steve. "Advanced message queuing protocol." IEEE Internet Computing 10, no. 6 (2006): 87-89.
- [71] Luzuriaga, Jorge E., Miguel Perez, Pablo Boronat, Juan Carlos Cano, Carlos Calafate, and Pietro Manzoni. "A comparative evaluation of AMQP and MQTT protocols over unstable and mobile networks." In 2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC), pp. 931-936. IEEE, 2015.
- [72] Al-Masri, Eyhab, Karan Raj Kalyanam, John Batts, Jonathan Kim, Sharanjit Singh, Tammy Vo, and Charlotte Yan. "Investigating messaging protocols for the Internet of Things (IoT)." IEEE Access 8 (2020): 94880-94911.
- [73] Bendel, Sven, Thomas Springer, Daniel Schuster, Alexander Schill, Ralf Ackermann, and Michael Ameling. "A service infrastructure for the Internet of Things based on XMPP." In 2013 IEEE international conference on pervasive computing and communications workshops (PERCOM Workshops), pp. 385-388. IEEE, 2013.
- [74] Rahman, Reem Abdul, and Babar Shah. "Security analysis of IoT protocols: A focus in CoAP." In 2016 3rd MEC international conference on big data and smart city (ICBDSC), pp. 1-7. IEEE, 2016.
- [75] <https://stomp.github.io/>
- [76] Mesnil, Jeff. Mobile and Web Messaging: Messaging Protocols for Web and Mobile Devices. " O'Reilly Media, Inc.", 2014.
- [77] Setiawan, Herri, Andi Abdul Adhiem Sumitro, and Rendra Gustriansyah. "The change data capture and the web application messaging protocol on the real time dashboard." International Journal of Engineering and Advanced Technology 8, no. 4 (2019).
- [78] Lehnhoff, Sebastian, Sebastian Rohjans, Mathias Uslar, and Wolfgang Mahnke. "OPC unified architecture: A service-oriented architecture for smart grids." In 2012 First International Workshop on Software Engineering Challenges for the Smart Grid (SE-SmartGrids), pp. 1-7. IEEE, 2012.
- [79] Rao, Suhas, Devaiah Chendanda, Chetan Deshpande, and Vishwas Lakkundi. "Implementing LWM2M in constrained IoT devices." In 2015 IEEE Conference on Wireless Sensors (ICWiSe), pp. 52-57. IEEE, 2015.
- [80] Borda, Monica. "Cryptography Basics." In Fundamentals in Information Theory and Coding, pp. 121-208. Springer, Berlin, Heidelberg, 2011.
- [81] Daemen, Joan, and Vincent Rijmen. "AES proposal: Rijndael." (1999).

- [82] Bogdanov, Andrey, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew JB Robshaw, Yannick Seurin, and Charlotte Vikkelse. "PRESENT: An ultra-lightweight block cipher." In International workshop on cryptographic hardware and embedded systems, pp. 450-466. Springer, Berlin, Heidelberg, 2007.
- [83] Borghoff, Julia, Anne Canteaut, Tim Güneysu, Elif Bilge Kavun, Miroslav Knezevic, Lars R. Knudsen, Gregor Leander et al. "PRINCE—a low-latency block cipher for pervasive computing applications." In International conference on the theory and application of cryptology and information security, pp. 208-225. Springer, Berlin, Heidelberg, 2012.
- [84] Beaulieu, Ray, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. "The SIMON and SPECK lightweight block ciphers." In Proceedings of the 52nd annual design automation conference, pp. 1-6. 2015.
- [85] Hong, Deukjo, Jaechul Sung, Seokhie Hong, Jongin Lim, Sangjin Lee, Bon-Seok Koo, Changhoon Lee et al. "HIGHT: A new block cipher suitable for low-resource device." In International workshop on cryptographic hardware and embedded systems, pp. 46-59. Springer, Berlin, Heidelberg, 2006.
- [86] Wu, Wenling, and Lei Zhang. "LBlock: a lightweight block cipher." In International conference on applied cryptography and network security, pp. 327-344. Springer, Berlin, Heidelberg, 2011.
- [87] Cruz, Rafael J., Antonio Guimarães, and Diego F. Aranha. "Efficient and secure software implementations of Fantomas." *Journal of Cryptographic Engineering* 10, no. 3 (2020): 211-228.
- [88] Bansod, G., Pisharoty, N. and Patil, A., 2017. BORON: an ultra-lightweight and low power encryption design for pervasive computing. *Frontiers of Information Technology & Electronic Engineering*, 18(3), pp.317-331.
- [89] Salunke R., Bansod G., Naidu P. (2019) Design and Implementation of a Lightweight Encryption Scheme for Wireless Sensor Nodes. In: Arai K., Bhatia R., Kapoor S. (eds) *Intelligent Computing. CompCom 2019. Advances in Intelligent Systems and Computing*, vol 998.
- [90] Hosseinzadeh, Jaber, and Abbas Ghaemi Bafghi. "Software Implementation and Evaluation of Lightweight Symmetric Block Ciphers of the Energy Perspectives and Memory." *arXiv preprint arXiv:1706.03909* (2017).
- [91] Singh, Praneet, and Kedar Deshpande. "Performance Evaluation of Cryptographic Ciphers on IoT Devices." *arXiv preprint arXiv:1812.02220* (2018).
- [92] Appel, Michael, et al. "Block ciphers for the IoT—SIMON, SPECK, KATAN, LED, TEA, PRESENT, and SEA compared." (2016).
- [93] Grossschadl, Johann, Stefan Tillich, Christian Rechberger, Michael Hofmann, and Marcel Medwed. "Energy evaluation of software implementations of block ciphers under memory constraints." In 2007 Design, Automation & Test in Europe Conference & Exhibition, pp. 1-6. IEEE, 2007.
- [94] Botta, Miroslav, Milan Simek, and Nathalie Mitton. "Comparison of hardware and software based encryption for secure communication in wireless sensor networks." 2013 36th International Conference on Telecommunications and Signal Processing (TSP). IEEE, 2013.

- [95] C. A. Lara-Niño, M. Morales-Sandoval and A. Díaz-Pérez, "An evaluation of AES and present ciphers for lightweight cryptography on smartphones," 2016 International Conference on Electronics, Communications and Computers (CONIELECOMP), Cholula, 2016, pp. 87-93.
- [96] Kotel, Sonia, Fatma Sbiaa, Medien Zeghid, Mohsen Machhout, Adel Baganne, and Rached Tourki. "Performance evaluation and design considerations of lightweight block cipher for low-cost embedded devices." In 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA), pp. 1-7. IEEE, 2016.
- [97] Diehl, William, Farnoud Farahmand, Panasayya Yalla, Jens-Peter Kaps, and Kris Gaj. "Comparison of hardware and software implementations of selected lightweight block ciphers." In 2017 27th International Conference on Field Programmable Logic and Applications (FPL), pp. 1-4. IEEE, 2017.
- [98] Shah, Ankit, and Margi Engineer. "A survey of lightweight cryptographic algorithms for iot-based applications." Smart Innovations in Communication and Computational Sciences. Springer, Singapore, 2019. 283-293.
- [99] Sehrawat, Deepti, and Nasib Singh Gill. "Performance Evaluation of Newly Proposed Lightweight Cipher, BRIGHT.", International Journal of Engineering and Advanced Technology (IJEAT), ISSN: 2249-8958, Volume-8 Issue-5, June 2019.
- [100] Doomun, M. Razvi, and K. M. S. Soyjaudah. "Analytical Comparison of Cryptographic Techniques for Resource-constrained Wireless Security." IJ Network Security 9, no. 1 (2009): 82-94.
- [101] Batina, Lejla, Amitabh Das, Barış Ege, Elif Bilge Kavun, Nele Mentens, Christof Paar, Ingrid Verbauwhede, and Tolga Yalçın. "Dietary recommendations for lightweight block ciphers: power, energy and area analysis of recently developed architectures." In International Workshop on Radio Frequency Identification: Security and Privacy Issues, pp. 103-112. Springer, 2013.
- [102] Lee, W. K., Raphael C-W. Phan, and B. M. Goi. "Fast and Energy-Efficient Block Ciphers Implementations in ARM Processors and Mali GPU." IETE Journal of Research (2020): pp.1-8.
- [103] Hatzivasilis, George, Konstantinos Fysarakis, Ioannis Papaefstathiou, and Charalampos Manifavas. "A review of lightweight block ciphers." Journal of Cryptographic Engineering 8, no. 2 (2018): 141-184.
- [104] Malina, Lukas, Vlastimil Clupek, Zdenek Martinasek, Jan Hajny, Kimio Oguchi, and Vaclav Zeman. "Evaluation of software-oriented block ciphers on smartphones." In International Symposium on Foundations and Practice of Security, pp. 353-368. Springer, Cham, 2013.
- [105] Omrani, Tasnime, Rhouma Rhouma, and Layth Sliman. "Lightweight cryptography for resource-constrained devices: a comparative study and rectangle cryptanalysis." In International Conference on Digital Economy, pp. 107-118. Springer, Cham, 2018.

- [106] Ertaul, Levent, and Sachin Katteppura Rajegowda. "Performance Analysis of CLEFIA, PICCOLO, TWINE Lightweight Block Ciphers in IoT Environment." In Proceedings of the International Conference on Security and Management (SAM), The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), pp. 25-31. 2017.
- [107] Alizadeh, Mojtaba, Mazleena Salleh, Mazdak Zamani, Jafar Shayan, and Sasan Karamizadeh. "Security and performance evaluation of lightweight cryptographic algorithms in RFID." Kos Island, Greece, pp.45-50. (2012).
- [108] Çakiroglu, M., 2010. Software implementation and performance comparison of popular block ciphers on 8-bit low-cost microcontroller. Int. J. Phys. Sci, 5(9), pp.1338-1343.2010.
- [109] Murvay, Pal-Stefan, et al. "Development of an autosar compliant cryptographic library on state-of-the-art automotive grade controllers." 2016 11th International Conference on Availability, Reliability and Security (ARES). IEEE, 2016.
- [110] <https://www.microchip.com/mplab/avr-support/atmel-studio-7>
- [111] Nithya, V., Ramachandran, B. & Bhaskar, V. BER Evaluation of IEEE 802.15.4 Compliant Wireless Sensor Networks Under Various Fading Channels. Wireless Pers Commun 77, 3105–3124 (2014).
- [112] Dias GM, Bellalta B, Oechsner S. Reducing the energy consumption in WSNs: A data scientific mechanism. arXiv preprint arXiv:1509.08778. 2015 Sep.



EKLER

EK A. Tablolar



EK A

Tablo A.3. Hafif siklet şifreler karşılaştırması.

Algoritma	Blok uzunluğu (Bit)	Anahtar uzunluğu (Bit)	Tekrar sayısı	Yapı	Uygulanan ortam
AES	128	128, 192, 256	10, 12, 14	SPN	Donanım + Yazılım
KLEIN	64	64, 80, 96	12, 16, 20	SPN	Donanım + Yazılım
LED	64	64, 128	32, 48	SPN	Donanım + Yazılım
SKINNY	64, 128	64, 128, 192 128, 256, 384	32, 36, 40 40, 48, 56	SPN	Donanım + Yazılım
Mysterion	128, 256	128, 256	12, 16	SPN	Donanım + Yazılım
RECTANGLE	64	80, 128	25	SPN	Donanım + Yazılım
NOEKEON	128	128	16	SPN	Donanım + Yazılım
PICO	64	128	32	SPN	Donanım + Yazılım
GIFT	64/128	128	28,40	SPN	Donanım + Yazılım
QARMA	64/128	128/256	16/24	SPN	Donanım + Yazılım
Loong	64	64, 80, 128	16, 20, 32	SPN	Donanım + Yazılım
NVLC	64	80, 128	20	SPN	Donanım + Yazılım
BORON	64	80, 128	25	SPN	Donanım + Yazılım
Midori	64, 128	128	16, 20	SPN	Donanım
Zorro	128	128	24	SPN	Donanım
Fantomas/Robin	128	128	12/16	SPN	Yazılım
PRIDE	64	128	20	SPN	Donanım
MANTIS	64	128	14	SPN	Donanım
mCrypton	64	64, 96, 128	12	SPN	Donanım
PRESENT	64	80, 128	31	SPN	Donanım
PRINCE	64	128	12	SPN	Donanım
OLBCA	64	80	22	SPN	Donanım
KHAZAD	64	128	8	SPN	Donanım
PRINTCIPHER	80/160	48/96	48/96	SPN	Donanım
SPNRX & LAX	64	128	24	SPN	Yazılım
	128	128, 256	32/40	(ARX)	
Chaskey Cipher	128	128	8	ARX	Yazılım
	32	64	22		
	48	72/96	22/23		
SPECK	64	96/128	26/27	ARX	Yazılım
	96	96/144	28/29		
	128	128/192/256	32/33/34		
CHAM	64, 128	128, 256	16, 32	ARX	Donanım + Yazılım
HIGHT	64	128	32	Feistel (ARX)	Donanım ve Yazılım

Tablo A.3. (Devamı) Hafif siklet şifreler karşılaştırması.

Algoritma	Blok uzunluğu (Bit)	Anahtar uzunluğu (Bit)	Tekrar sayısı	Yapı	Uygulanan ortam
RC2	64	8-1024	2, 16	Feistel	Re2 (Yazılım),
RC5	32, 64, 128	0...2040	12	ARX	RC5 RC6 (Donanım
RC6	128	128, 192, 256	20	Feistel (II)	veya Yazılım)
	64				
Cast 5	128, 160,	40-128	12-16	Feistel	Yazılım
Cast 6	192, 224,	128	48	GFN	
	256				
	64	80, 96, 128	32, 33, 34	GFN	Yazılım
BRIGHT	128	128, 192, 256	35, 36, 37		
Khudra	64	80	18	GFN	Donanım
QTL	64	64, 128	25, 31	GFN	Donanım
Piccolo	64	80, 128	25, 31	GFN	Donanım
CLEFIA	128	128, 192, 256	18, 22, 26	GFN	Donanım + Yazılım
LEA	128	128/192/256	24/28/32	GFN	Donanım + Yazılım
TWINE	64	80, 128	36	GFN	Donanım + Yazılım
Camellia	128	128, 192, 256	24	Feistel	Donanım + Yazılım
SIMECK	32, 48, 64	64, 96, 128	32, 36, 44	Feistel	Donanım + Yazılım
LiCi	64	128	31	Feistel	Donanım + Yazılım
KASUMI/MIST					
Y1	64	128	8	Feistel	Donanım + Yazılım
LBlock	64	80	32	Feistel	Donanım + Yazılım
SEA	96	96	93	Feistel	Donanım + Yazılım
ANU	64	80, 128	25	Feistel	Donanım + Yazılım
BlowFish	64, 128	Up to 448	16	Feistel	Yazılım
Twofish	128	128, 192, 256	16		
Seed	128	128	16	Feistel	Yazılım
TEA					
XTEA	64	128	64	Feistel	Yazılım
LILIPUT	64	80	30	Feistel	Yazılım
NASE	Değişken	Değişken	Değişken	Feistel	Yazılım
ITUBEE	80	80	20	Feistel	Yazılım
RoadRunner	64	80, 128	10, 12	Feistel	Yazılım
DES	64	56	16		
G-DES	Değişken	Değişken	Değişken (Çift)		
3DES	64	168, 112, 56	48	Feistel	Donanım
DESL	64	56	16		
DESX	64	184	16		
DESLX	64	184	16		
KAMAR	128	128, 192, 256	16, 20, 32	Feistel	Donanım

Tablo A.3. (Devamı) Hafif siklet şifreler karşılaştırması.

Algoritma	Blok uzunluğu (Bit)	Anahtar uzunluğu (Bit)	Tekrar sayısı	Yapı	Uygulanan ortam
Skipjack	64	80	32	Feistel	Donanım
HISEC	64	80	15	Feistel	Donanım
GOST	64	256	32	Feistel	Donanım
DLBCA	32	80	15	Feistel	Donanım
	32	64	32		
	48	72/96	36		
SIMON	64	96/128	42/44	Feistel	Donanım
	96	96/144	52/54		
	128	128/192/256	68/69/72		
SFN	64	96	32	Feistel + SPN	Donanım + Yazılım
SIT	64	64	5	Feistel + SPN	Donanım + Yazılım
Iceberg	64	128	16	Devrimsel mimari	Donanım
PUFFIN	64	128	32	Devrimsel	Donanım
PUFFIN2	64	80	34	SPN	Donanım
FeW	64	80/128	32	Balanced GFN + SPN	Yazılım
μ^2	64	80	15	GFN + SPN	Donanım + Yazılım
LRBC	16	16	24	Feistel + SPN	Donanım
FLY	64	128	20	Bitsliced şifre	Donanım + Yazılım
KATAN/KTAN TAN	32, 48, 64	80	254	Bivium	Donanım
TREYFER	64	64	32	Basit	Donanım
BKSQ	96, 144, 192	96	10, 14, 18	SQUARE şifre	Donanım

ÖZGEÇMİŞ

Ad-Soyad : Uras PANAHI

ÖĞRENİM DURUMU:

- **Lisans** : 2005, SHIRAZ Shahid Bahonar No.1 Teknik Koleji (İran), Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü
- **Yükseklisans** : 2008, İslam Azad Üniversitesi (İran), IT ve Bilgisayar Mühendisliği, IT Mühendisliği (Bilgisayar Ağlarında Uzman)

MESLEKİ DENEYİM VE ÖDÜLLER:

- 2017-2021 yılları arasında Bolu Abant İzzet Baysal Üniversitesinde, Bilgisayar Mühendislik Bölümünde, tam zamanlı öğretim görevlisi.
- 2011- 2016 yılları Ankara Üniversitesi, Bilgisayar Mühendislik Bölümünde, tam zamanlı öğretim görevlisi.

TEZDEN TÜRETİLEN ESERLER:

- Uras Panahi, Cüneyt Bayılmış, “Enabling secure data transmission for wireless sensor networks based IoT applications”, Ain Shams Engineering Journal, Volume 14, Issue 2, 2023, 101866, ISSN 2090-4479, <https://doi.org/10.1016/j.asej.2022.101866>, SCIE, Q1. (SCIE Dergi Örneği)
- Panahi, P., Bayılmış, C., Çavuşoğlu, U. et al. “Performance Evaluation of Lightweight Encryption Algorithms for IoT-Based Applications”, Arabian Journal for Science and Engineering, 46, 4015–4037 (2021), Springer. <https://doi.org/10.1007/s13369-021-05358-4>, SCIE, Q2. (SCIE Dergi Örneği)
- Pejman Panahi, Cüneyt Bayılmış, Unal Çavuşoğlu, Sezgin Kaçar, “Comparing PRESENT and LBlock block ciphers over IoT Platform”, 12th International Conference on Information Security and Cryptology (ISC 2019), 16-17 Oct, pp:66-69. (Konferans Makale Örneği)
- Pejman Panahi, Cüneyt Bayılmış, Unal Çavuşoğlu, Sezgin Kaçar, “Performance Evaluation of L-Block Algorithm for IoT Applications”, 3rd International Conference on Computer Science and Engineering (UBMK 2018), 20-23 Sept 2018, Bosnia & Herzegovina, Sarajevo, pp: 609-612. (Konferans Makale Örneği)

DİĞER ESERLER (Makale, bildiri):

- 1- Baki Koyuncu, Pejman Panahi, Sefika Varlioglu, "*Comparative Indoor Localization by using Landmarc and Cricket systems*", International Journal of Emerging Technology and Advanced Engineering, ISSN 2250-2459, ISO 9001:2008, certified Journal, volume 5, issue 6, June 2015, pages: 453-456.
- 2- Baki Koyuncu, Erkan Meral and Pejman Panahi, "*Real time Geolocation tracking by using GPS+GPRS and Arduino based SIM908*", IFRSA INTERNATIONAL JOURNAL OF ELECTRONICS CIRCUITS AND SYSTEMS (IJECS), vol 4, issue 2, July 2015, pages: 148-150.
- 3- Baki Koyuncu and Pejman Panahi, "*Kalman Filtering of Link Quality Indicator Values for Position Detection by Using WSNS*", International Journal of Computing, Communications & Instrumentation Eng. (IJCCIE) vol. 1, issue 1 (2014), ISSN 2349-1469 EISSN 2349-1477, pages: 129-133.
- 4- Baki Koyuncu, B.Uğur, Pejman Panahi, "*Indoor Location Determination by Using RFIDs*", International Journal of Mobile and Adhoc Network (IJMAN), vol 3, issue 1, Feb 2013, pages: 7-11.
- 5- Pejman Panahi, Michelle Freund, "*Safety Application Schema for Vehicular Virtual Ad Hoc Grid Networks*", International Journal of Academic Research (IJAR), vol 3, no 2, March 2011, pages: 26-29.
- 6- Pejman Panahi, "*The Feedback Based Mechanism for Video Streaming Over Multipath Ad Hoc Networks*", Journal of Sciences, Islamic Republic of Iran; ISSN: 1016-1104, May 2010, pages: 169-179.
- 7- Pejman Panahi, Cüneyt Bayılmış, "*Car Indoor GAS Detection System*", 2nd International Conference on Computer Science and Engineering (UBMK 2017), 5-8 October 2017, Antalya, Turkey, IEEE indexed, pp: 957-960.
- 8- Baki Koyuncu, Alper Gokce and Pejman Panahi, "*Introduction of the Unity Game Engine in Reconstruction of an Archeological Site*", 19th Symposium on Mediterranean Archaeology, 12-14 November 2015 – Kemer, Turkey, pp: 95-103.
- 9- Baki Koyuncu, Alper Gokce and Pejman Panahi, "*Reconstruction of an Archeological site in real time domain by using software techniques*", The Fifth International Conference on Communication Systems and Network Technologies (CSNT-2015), 04-06 April 2015, Gwalior, India, IEEE Proceedings, pp: 1350-1354.
- 10- Pejman Panahi, F.Borna, "*Distance Learning, Challenges, New Solution*", 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO 2014), 26-30 May 2014, Opatija, Croatia, IEEE Proceedings, pp: 771-774.
- 11- Pejman Panahi, "*Providing Consistent Global Sharing Service Over VANET Using New Plan*", CSICC 2009, 14th annual International CSI Computer Conference, 20-21 October 2009, Tehran, Iran, IEEE Proceedings, pp: 213-218.

- 12- Pejman Panahi, "*New Plan for Hardware Resource Utilization in Multimedia Applications Over Multi Processor Based System*", 32nd International Convention MIPRO 2009, Conference on GRID AND VISUALIZATION SYSTEMS (GVS), 25-29 May 2009, Opatija, Croatia, pp: 256-260.
- 13- Pejman Panahi, "*Multipath Local Error Management Technique Over Ad Hoc Networks*", AXMEDIS 2008, 4th International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution, 17-19 November 2008, Florence, Italy, IEEE Proceedings, pp:187-194.
- 14- Pejman Panahi, H. K. Maragheh, M. Abdolzadeh and M. Sharifi, "A Novel Schema for Multipath Video Transferring over Ad Hoc Networks," 2008 The Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, Valencia, 2008, pp. 77-82, doi: 10.1109/UBICOMM.2008.60.
- 15- Pejman Panahi, Mehdi Dehghan, "*Multipath Video Transmission over Ad hoc Networks using Layer Coding and Video Caches*", ICEE2008, 16th Conference on Electrical Engineering, 13th – 17th May 2008, pp: 50-55.