MQTT Protokolünde Şifreli Mesajların OTEL Tracing ve Zipkin Metrik Analizi: Algoritmaların Karşılaştırılması

Emre ERKAN
Bilgisayar Mühendisliği
Pamukkale Üniversitesi
Denizli, Türkiye
eerkan13@posta.pau.edu.tr

Özet—**Bu çalışma, MQTT protokolü üzerinde şifreli mesaj iletimini ele almakta ve bu şifreli mesajların OTEL protokolü ile izlenmesi ile Zipkin üzerinde çeşitli metriklerin analizini sunmaktadır. AES, Blowfish, Camellia, CAST5, IDEA, SEED, TripleDES gibi çeşitli şifreleme algoritmaları, 16, 32, 64, 128, 256, 512, 1024, 2048 uzunluğundaki veriler üzerinde 1000'er örnek üzerinde test edilmiştir. Elde edilen sonuçlar, her algoritmanın performansını, gecikme süresi, şifreleme/deşifreleme süreleri ve bellek tüketimi açısından ayrıntılı bir şekilde değerlendirmektedir. Bu analiz, güvenli ve etkin bir iletişim için algoritma seçiminde rehberlik sağlamayı amaçlamaktadır.**

Anahtar Kelimeler—mqtt, şifreli mesaj iletimi, otel protokolü, zipkin metrik analizi, şifreleme algoritmaları, iot

Abstract— **This study addresses encrypted message transmission on the MQTT protocol, presenting an analysis of various metrics on Zipkin through the monitoring of these encrypted messages using the OTEL protocol. Various encryption algorithms such as AES, Blowfish, Camellia, CAST5, IDEA, SEED, and TripleDES have been tested on data lengths of 16, 32, 64, 128, 256, 512, 1024, and 2048 in 1000 examples each. The results obtained comprehensively evaluate the performance of each algorithm in terms of latency, encryption/decryption times, and memory consumption. This analysis aims to provide guidance in selecting algorithms for secure and efficient communication.**

Anahtar Kelimeler— mqtt, encrypted message transmission, OTEL protocol, zipkin metric analysis, encryption algorithms, iot

I. GIRIŞ

Dijital iletişimde güvenlik, veri bütünlüğü ve gizliliği sağlamak son derece kritik bir konudur. Bu bağlamda, MQTT (Message Queuing Telemetry Transport) protokolü, hafif ve etkili iletişim sağlama yeteneği ile nesnelerin interneti (IoT) ve diğer uygulama alanlarında geniş bir kullanım bulmuştur. Bu protokolün kullanımında, özellikle hassas verilerin iletilmesi durumunda, şifreleme önemli bir rol oynamaktadır.

Bu çalışma, MQTT protokolünde şifreli mesajların iletimi üzerine odaklanmaktadır. Şifreli mesajların güvenli bir şekilde iletilmesi, veri bütünlüğünün sağlanması ve gizliliğin korunması, günümüzdeki birçok uygulama senaryosu için kritik öneme sahiptir. Bununla birlikte, şifreleme kullanımının getirdiği ek yükler ve performans etkileri, özellikle kaynak kısıtlı cihazlar ve ağlar için önemli bir konudur.

Bu bağlamda, MQTT protokolünde kullanılan şifreleme algoritmalarının performansını değerlendirmek amacıyla gerçekleştirilen bu çalışma, şifreli payload'ların OTEL (OpenTelemetry) protokolü ile izlenmesi ve Zipkin üzerinde çeşitli metriklerin analizini sunmaktadır. AES, Blowfish, Camellia, CAST5, IDEA, SEED, TripleDES gibi farklı şifreleme algoritmaları, değişen veri uzunlukları üzerinde test edilmiş ve bu algoritmaların performansı gecikme (latency), şifreleme/deşifreleme süreleri ve bellek tüketimi açısından detaylı bir şekilde incelenmiştir. Elde edilen sonuçlar, MQTT protokolünde şifreli iletişimin etkin bir şekilde yönetilebilmesi için önemli bir kılavuz sunmaktadır.

II. İLGILI ÇALIŞMALAR

Akıllı prizlerin kullanımında büyük ölçüde kablosuz teknolojiler tercih edilmektedir. En yaygın olarak kullanılan kablosuz iletişim protokolleri arasında Zigbee ve Wi-Fi bulunmaktadır. Bununla birlikte, Bluetooth gibi diğer teknolojilerle de bazı uygulamalar mevcuttur.

Gunathilake ve diğerleri (Gunathilake, Buchanan ve Asif, 2019) hafif Şifreleme (LWC) algoritmalarının akıllı IoT cihazlarında uygulanması, özellikle de Long-Range Wide Area Network (LoRaWAN) teknolojisinin performansı ele alınmıştır. LoRaWAN, Low-Power Wide Area Network (LPWAN) teknolojisi için iletişim protokolünü tanımlayan bir açık standarttır [1].

Shah ve diğerleri (Shah, Arora ve Adhvaryu, 2020) gömülü sistemlerde kullanılan geleneksel kriptografik algoritmaların, fiziksel boyut, işleme gereksinimleri ve bellek kısıtlamaları gibi sorunlarla başa çıkma konusundaki zorlukları ele alan bir araştırma makalesi bulunmaktadır. Bu makale, AES (Gelişmiş Şifreleme Standardı), SHA-256 (Güvenli Karma Algoritması) ve RSA/Elliptic Curve gibi yaygın kullanılan algoritmaların, düşük güç tüketimli cihazlar, sensör ağları ve sağlık uygulamaları gibi kaynak kısıtlı ortamlarda etkisiz kalabileceğini vurgulamaktadır [2].

Gunathilake ve diğerleri (Gunathilake, Al-Dubai ve Buchana, 2020) hafif kriptografinin IoT platformlarına uygunluğunu vurgular. Geleneksel yöntemlerin ağırlıklarından kaçınarak yeni algoritmaların geliştirilmesi, gelişmiş IoT tehditlerine karşı direncin artırılmasını amaçlar. Çalışma, bu alanda literatür eksikliğini kapatmayı hedefler, gelişmeleri özetler ve gelecekteki geliştirmelere odaklanır [3].

Al-Odat ve diğerleri (Al-Odat, Al-Qtiemat ve Khan, 2020) büyük veri ve IoT uygulamaları için bir hafif kriptografi karma fonksiyonunu tanıtır. Önerilen tasarım, S-Box, lineer dönüşüm ve bit permütasyon işlevlerini kullanarak güvenli ve hızlı bir hafif kriptografi protokolü sağlar. Tasarım, hız, bellek ve güç tüketimi açısından diğer hafif protokollerden üstün performans sergilemektedir [4].

Bagla ve diğerleri (Bagla, Sharma, Mishra, Tripathi, Dumka ve Pandey, 2023) IoT ve bulut sistemlerinde güvenliği artırmak amacıyla örgü tabanlı kriptografiyi tanıtır. Örgü tabanlı şifreleme avantajları, geleneksel RSA ve eliptik eğri kriptografisiyle karşılaştırılarak incelenir. Ayrıca, IoT ve bulut sistemleri için özel olarak geliştirilen "Örgü Tabanlı Güvenli Veri Birleştirme" yaklaşımını sunar. Çalışma, veri şifreleme, birleştirme, deşifreleme ve sınırlı kaynaklı cihazlarda uygulama gibi çeşitli yönleri kapsar [5].

Madavi ve diğerleri (Madavi, Sowjanya ve Patwari, 2023) IoT cihazlarının ürettiği verilerin güvenliği için Hafif Kriptografi (LWC) tekniklerini önermektedir. Öne çıkan yöntem, sağlık giyilebilir IoT cihazlarından elde edilen verileri tek bir düz metin içinde birleştirip şifreleyerek gizlilik ve güvenlik sağlamayı amaçlamaktadır. Önerilen yöntem, şifreleme ve deşifre sürelerini azaltarak daha iyi bir performans sunmaktadır [6].

III. Geliştirilen Model

Bu bölümde, geliştirilen sistemin ayrıntılı bir teknik incelemesi sunulacaktır. Yazılım entegrasyonundan, iletişim protokollerine ve genel sistem mimarisine kadar her bir unsura geniş bir perspektiften yaklaşılacaktır.

A. Sistem Mimarisi

Geliştirilen modelin sistem mimarisi, şifreli mesaj üretiminden başlayarak çeşitli önemli bileşenleri içerir. İlk olarak, "Publisher" bileşeni, MQTT protokolü kullanarak şifrelenmiş mesajları üretir. Ardından, bu şifreli mesajlar "Gateway" bileşeni aracılığıyla iki ayrı MQTT brokerı arasında eşitlenir. Şifrelenmiş verilerin güvenli bir şekilde iletimi sağlanmış olur. Son aşamada ise, şifrelenmiş verileri alıp çözümleyen "Subscriber" bileşeni bulunmaktadır. Bu bileşen, orijinal veriyi elde ederek son kullanıcıya sunar. Bu süreç, güvenli ve etkili bir iletişimi temsil eder, bu da şifrelenmiş verilerin güvenli bir şekilde kaynak ve hedef arasında iletilmesini sağlar.

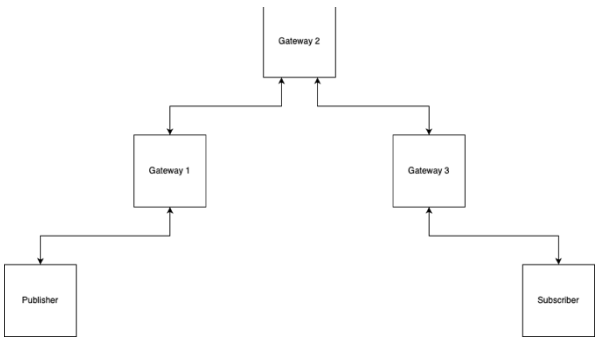

Fig. 1: Sistem mimarisi

B. MQTT Protokolü

MQTT (Message Queuing Telemetry Transport), hafif, açık kaynaklı bir mesajlaşma protokolüdür ve özellikle düşük bant genişliğine sahip ve bağlantı kaynakları sınırlı cihazlar arasında güvenilir mesaj iletişimi sağlamak için tasarlanmıştır. Bu protokol, yayın/abone modelini temel alır ve genellikle IoT cihazları, sensör ağları ve dağıtık sistemlerde kullanılır.

Geliştirilen projede, MQTT protokolü şifreli mesaj iletimi için temel iletişim protokolü olarak kullanılmaktadır. "Publisher" bileşeni, MQTT protokolünü kullanarak şifrelenmiş mesajları üretir ve bu mesajları belirli bir MQTT topic'ine gönderir. Ardından, bu şifreli mesajlar, "Gateway" bileşeni aracılığıyla farklı MQTT brokerları arasında eşitlenir. Bu sayede, güvenli bir şekilde şifrelenmiş veri, farklı brokerlar arasında güvenli bir iletim sağlanır. "Subscriber" bileşeni ise şifreli veriyi alır, çözümleme işlemi gerçekleştirir ve orijinal veriyi elde eder. MQTT protokolü, projede dağıtık ve güvenli iletişimi mümkün kılarak sistemin sağlıklı çalışmasını destekler.

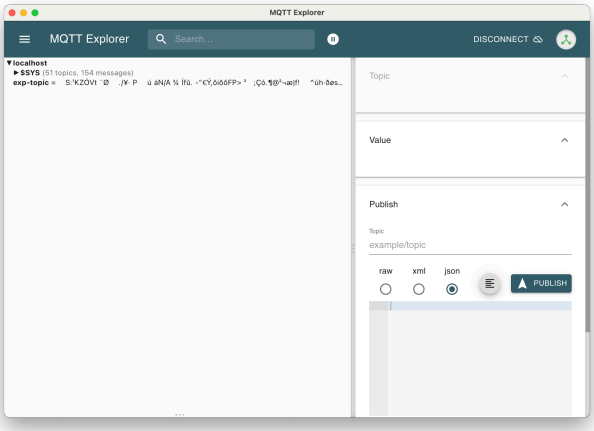

Fig. 2: MQTT broker'da bulunan şifrelenmiş mesaj

C. Zipkin ve OTEL

Zipkin, mikroservis mimarisine sahip sistemlerdeki dağıtık uygulamalardan elde edilen verilerin izlenmesini ve analizini sağlayan bir açık kaynaklı bir izleme sistemidir. Zipkin, uygulamadaki farklı mikroservisler arasındaki bağlantıları ve veri akışını görselleştirmek için kullanılır. Bu sayede, uygulama içindeki performans sorunlarını tespit etmek ve iyileştirmeler yapmak daha kolay hale gelir.

OpenTelemetry (OTEL), çeşitli programlama dilleri ve çerçeveler arasında uygulama performansını izlemek ve analiz etmek için kullanılan bir standarttır. OTEL, dağıtık uygulamalardan gelen verilerin toplanması, izlenmesi ve analiz edilmesi için genel bir çerçeve sağlar. Bu standart, farklı dil ve teknolojileri kullanan sistemler arasında tutarlı izleme ve analiz olanağı sağlar.

Geliştirilen modelde, Zipkin ve OTEL protokolleri, MQTT üzerinden iletilen şifreli mesajların takibini ve analizini sağlamak amacıyla entegre edilmiştir. Zipkin, şifrelenmiş verilerin kaynağından hedefine kadar olan yolunu görselleştirmekte ve uygulamadaki her bir mikroservisin performansını izlemektedir. OTEL ise çeşitli dil ve teknolojiler kullanılarak geliştirilen bileşenlerden elde edilen metrik verilerin toplanmasını sağlar. Bu sayede, geliştirilen modelin performansını değerlendirmek ve olası iyileştirmeleri belirlemek için kapsamlı bir izleme ve analiz yapılmaktadır.

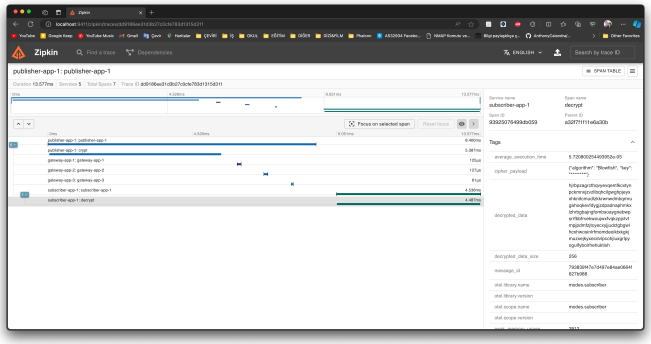

Fig. 3: İzlenilen mesajın zipkin görünümü

D. Şifreleme Algoritmaları

AES, Rijndael algoritmasına dayanır ve 128, 192 veya 256 bit uzunluğunda anahtarlar kullanır. Blok başına 128 bit veri işler. SubBytes, ShiftRows, MixColumns ve AddRoundKey adlı dört temel işlem içerir. Bu sayede saldırılara karşı dirençli ve hızlı bir şifreleme sağlar.

Blowfish, Feistel ağı yapısına dayanır ve değişken uzunluktaki anahtarları destekler. Bir turdaki işlemler arasında sırasıyla XOR, S-Box uygulaması, Permutasyon ve Substitution işlemleri bulunur. Esnek anahtar uzunlukları ve hızlı işleme yetenekleri nedeniyle tercih edilir.

Camellia, Feistel ağı yapısına dayalı simetrik bir blok şifreleme algoritmasıdır. S-Box, Permutasyon ve anahtar eklemesi gibi işlemler içerir. Japon ve Güney Kore kriptografi standartları için kullanılır ve AES'e benzer güvenlik düzeyleri sağlar.

CAST5, Carlisle Adams ve Stafford Tavares tarafından geliştirilen bir simetrik blok şifreleme algoritmasıdır. Feistel şifreleme yapısını kullanır ve anahtar uzunluğu değiştirilebilir. Veri bloklarını şifrelemek için S-Box ve anahtar eklemesi gibi işlemleri içerir.

IDEA, 64-bit blokları şifrelemek ve deşifre etmek için kullanılan bir simetrik anahtarlı şifreleme algoritmasıdır. 128 bit uzunluğundaki anahtarları destekler. Permutasyon, XOR ve modüler aritmetik gibi işlemler içerir. Güvenilir ve hızlı bir şifreleme sağlar.

SEED, Güney Kore'nin geliştirdiği bir blok şifreleme algoritmasıdır. 128 bit uzunluğundaki anahtarları kullanır. Feistel ağı yapısını benimser ve XOR, Permutasyon ve Substitution gibi işlemleri içerir. Hızlı ve güvenilir bir şifreleme sağlar.

TripleDES, DES algoritmasının üç kere uygulanmasıyla oluşan bir şifreleme yöntemidir. Anahtar uzunluğu 168 bit olabilir. DES'e dayalı olması, daha güvenilir ve dayanıklı bir şifreleme sağlar. Feistel ağı yapısını üç katmanlı olarak kullanır.

E. Algoritmaların Karşılaştırılması

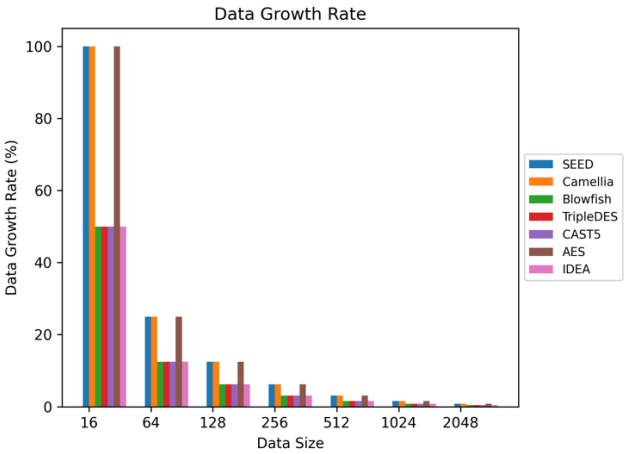

Fig. 4: Şifrelendikten sonra verinin büyüme oranı

TABLE I. VERI BÜYÜME ORANI

Veri Boyutu	*Algoritmalar sırasıyla en düşükten en yükseğe doğru*
16	Blowfish, TripleDES, CAST5, IDEA, SEED, Camellia, AES
64	Blowfish, TripleDES, CAST5, IDEA, SEED, Camellia, AES
128	Blowfish, TripleDES, CAST5, IDEA, SEED, Camellia, AES
256	Blowfish, TripleDES, CAST5, IDEA, SEED, Camellia, AES
512	Blowfish, TripleDES, CAST5, IDEA, SEED, Camellia, AES
1024	Blowfish, TripleDES, CAST5, IDEA, SEED, Camellia, AES
2048	Blowfish, TripleDES, CAST5, IDEA, SEED, Camellia, AES

Küçük veri boyutlarında, Blowfish, TripleDES ve CAST5 algoritmaları, şifreleme işlemi sonucunda veri boyutunu daha etkin bir şekilde küçük tutmaktadır. Bu algoritmalar, düşük veri büyüme oranlarıyla öne çıkar. Bu özellikleri, veri güvenliğini korurken aynı zamanda kaynakları daha verimli bant genişliği kullanma avantajı sunmaktadır.

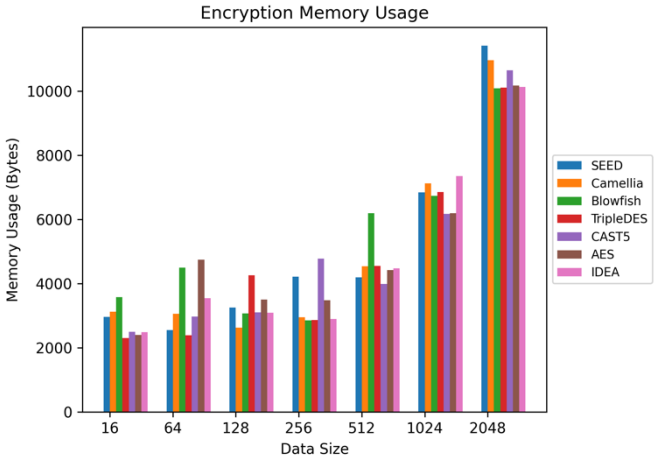

Fig. 5: Şifreleme bellek kullanımı

TABLE II. ŞIFRELEME BELLEK KULLANIMI

Veri Boyutu	*Algoritmalar sırasıyla en düşükten en yükseğe doğru*
16	TripleDES, AES, IDEA, CAST5, SEED, Camellia, Blowfish
64	TripleDES, SEED, CAST5, Camellia, IDEA, Blowfish, AES
128	Camellia, Blowfish, IDEA, CAST5, SEED, AES, TripleDES
256	Blowfish, TripleDES, IDEA, Camellia, AES, SEED, CAST5
512	CAST5, SEED, AES, IDEA, Camellia, TripleDES, Blowfish
1024	CAST5, AES, Blowfish, SEED, TripleDES, Camellia, IDEA
2048	Blowfish, TripleDES, IDEA, AES, CAST5, Camellia, SEED

Bellek tüketiminde verinin boyutuna göre TripleDES, CAST5 ve Blowfish algoritmaları öne çıkmaktadır.

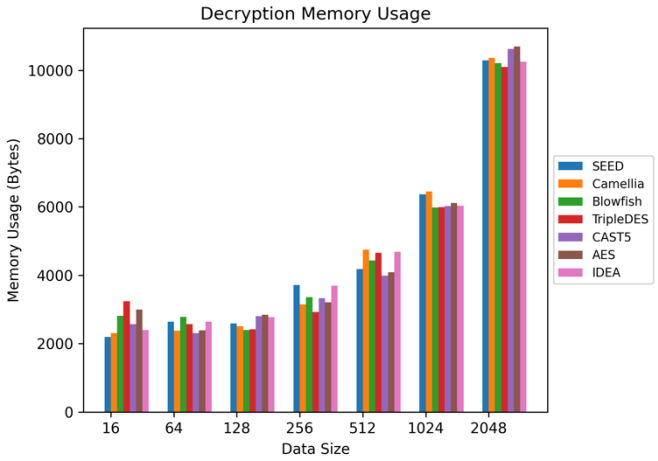

Fig. 6: Deşifreleme bellek kullanımı

TABLE III. DEŞIFRELEME BELLEK KULLANIMI

Veri Boyutu	*Algoritmalar sırasıyla en düşükten en yükseğe doğru*
16	SEED, Camellia, IDEA, CAST5, Blowfish, AES, TripleDES
64	CAST5, Camellia, AES, TripleDES, SEED, IDEA, Blowfish
128	Blowfish, TripleDES, Camellia, SEED, IDEA, CAST5, AES
256	TripleDES, Camellia, AES, CAST5, Blowfish, IDEA, SEED
512	CAST5, AES, SEED, Blowfish, TripleDES, IDEA, Camellia
1024	Blowfish, TripleDES, CAST5, IDEA, AES, SEED, Camellia
2048	TripleDES, Blowfish, IDEA, SEED, Camellia, CAST5, AES

Deşifreleme sürecinde, düşük veri boyutlarında SEED, Camellia ve CAST5 algoritmaları belirgin bir performans sergilemektedir. Veri boyutları arttıkça TripleDES algoritması daha yüksek performans sağlamaktadır. Bu durum, farklı algoritmaların veri boyutlarına bağlı olarak deşifreleme sürelerindeki etkinlik farklılıklarını vurgular. Optimal performans için veri boyutuyla uyumlu şifreleme algoritmalarının seçilmesi önemlidir.

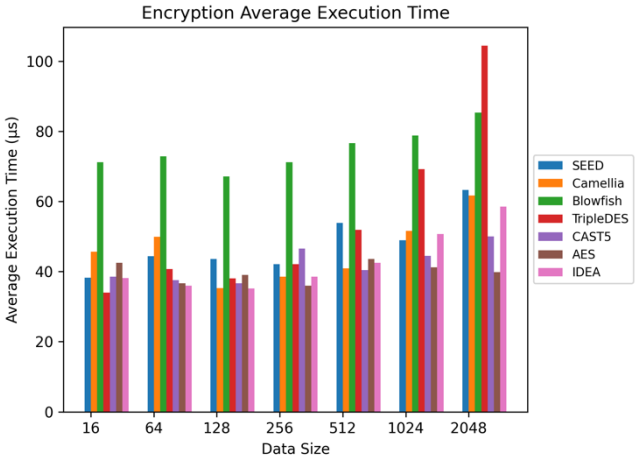

Fig. 7: Şifreleme süresi

TABLE IV. ŞIFRELEME SÜRESI

Veri Boyutu	*Algoritmalar sırasıyla en düşükten en yükseğe doğru*
16	TripleDES, IDEA, SEED, CAST5, AES, Camellia, Blowfish
64	IDEA, AES, CAST5, TripleDES, SEED, Camellia, Blowfish
128	IDEA, Camellia, CAST5, TripleDES, AES, SEED, Blowfish
256	AES, IDEA, Camellia, SEED, TripleDES, CAST5, Blowfish
512	CAST5, Camellia, IDEA, AES, TripleDES, SEED, Blowfish
1024	AES, CAST5, SEED, IDEA, Camellia, TripleDES, Blowfish
2048	AES, CAST5, IDEA, Camellia, SEED, Blowfish, TripleDES

Şifreleme süresinde büyük veri boyutlarında AES algoritması ön plana çıkmaktadır. Çoğu durumda Blowfish algoritması düşük performans sağlamıştır.

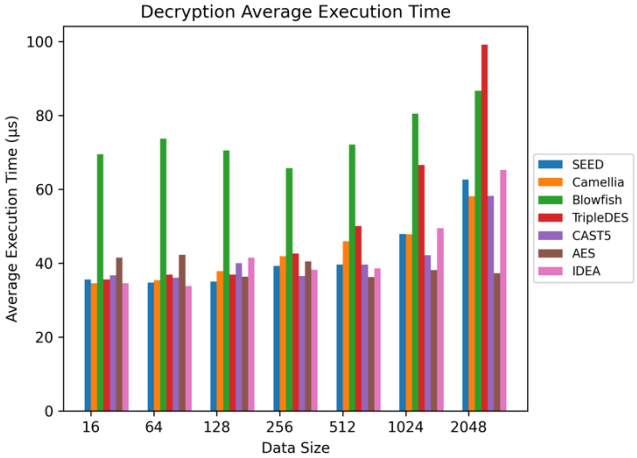

Fig. 8: Deşifreleme süresi

TABLE V. DEŞIFRELEME SÜRESI

Veri Boyutu	*Algoritmalar sırasıyla en düşükten en yükseğe doğru*
16	IDEA, Camellia, TripleDES, SEED, CAST5, AES, Blowfish
64	IDEA, SEED, Camellia, CAST5, TripleDES, AES, Blowfish

Veri Boyutu	*Algoritmalar sırasıyla en düşükten en yükseğe doğru*
128	SEED, AES, TripleDES, Camellia, CAST5, IDEA, Blowfish
256	CAST5, IDEA, SEED, AES, Camellia, TripleDES, Blowfish
512	AES, IDEA, CAST5, SEED, Camellia, TripleDES, Blowfish
1024	AES, CAST5, Camellia, SEED, IDEA, TripleDES, Blowfish
2048	AES, Camellia, CAST5, SEED, IDEA, Blowfish, TripleDES

Deşifreleme süresi incelendiğinde düşük veri boyutlarında IDEA algoritması ön plana çıkmaktadır. Daha yüksek veri boyutlarında AES algoritması daha iyi performans sergilemiştir.

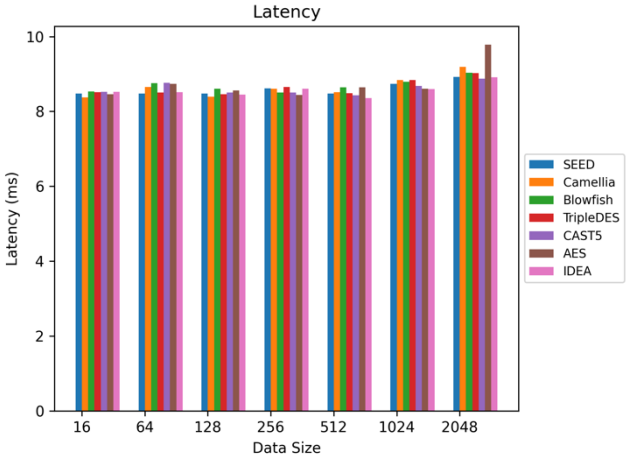

Fig. 8: Gecikme süresi

TABLE VI. GECIKME SÜRESI

Veri Boyutu	*Algoritmalar sırasıyla en düşükten en yükseğe doğru*
16	Camellia, AES, SEED, TripleDES, IDEA, CAST5, Blowfish
64	SEED, TripleDES, IDEA, Camellia, AES, Blowfish, CAST5
128	Camellia, IDEA, TripleDES, SEED, CAST5, AES, Blowfish
256	AES, Blowfish, CAST5, Camellia, IDEA, SEED, TripleDES
512	IDEA, CAST5, SEED, TripleDES, Camellia, Blowfish, AES
1024	IDEA, AES, CAST5, SEED, Blowfish, TripleDES, Camellia
2048	CAST5, IDEA, SEED, TripleDES, Blowfish, Camellia, AES

Geliştirdiğimiz sistemde, gözlemlenen ortalama 8-9 ms arası gecikme süresi, algoritmanın etkisinin minimal olduğunu göstermektedir. Bu gecikme süresindeki artışın temel nedeni, MQTT broker ve Gateway programının çalışma yapısındadır. Algoritmanın hızlı ve etkili bir şekilde çalışmasına rağmen, bütün sistemin genel performansını etkileyen faktörler arasında MQTT broker ve Gateway programının belirgin bir rolü olduğu görülmektedir. Bu durum, sistem performansını daha fazla optimize etmek için bu bileşenlere odaklanmayı gerektirebilir.

IV. UYGULAMALAR

Geliştirilen sistem, özellikle şifreleme odaklı özellikleri sayesinde çeşitli uygulama alanlarında güvenli iletişim ihtiyaçlarına hitap etmektedir. Bu sistem, Docker teknolojisi ve Python programlama dilinin entegrasyonu ile şifreli iletişim sağlamak amacıyla tasarlanmıştır.

Öncelikle, Internet of Things (IoT) projelerinde bu sistem, IoT cihazları arasında güvenli ve şifrelenmiş veri iletimini destekleyerek endüstriyel IoT uygulamalarında, akıllı ev sistemlerinde ve medikal cihazlarda güvenlik sağlar. Hassas verilerin güvenli bir şekilde iletilmesi, bu tip projelerde kritik bir gerekliliktir.

Merkezi olmayan veri senkronizasyonu, farklı MQTT broker'ları arasında şifreli veri eşitleme ihtiyacını karşılamaktadır. Bu, finansal projelerden enerji sektörüne kadar farklı alanlarda şifrelenmiş veri akışını yönetme gereksinimini karşılar.

Docker konteyner entegrasyonu, şifreleme özelliklerini taşıyan uygulamaların hızlı bir şekilde dağıtılmasını ve ölçeklenmesini sağlar. Özellikle mikro hizmet mimarisi tabanlı projelerde, Docker'ın taşınabilirlik avantajları ile birleşerek projelerin daha hızlı ve etkili bir şekilde geliştirilmesine olanak tanır.

Bu şifreleme odaklı sistem, güvenli iletişim ve veri bütünlüğüne vurgu yaparak IoT projelerinden finansal uygulamalara kadar geniş bir uygulama yelpazesinde kullanılabilir.

V. SONUÇLAR

Bu çalışma, Docker teknolojisi ve Python programlama dili kullanılarak geliştirilen bir MQTT tabanlı sistemle ilgili önemli uygulama alanlarına odaklanmaktadır. Sistem, özellikle Internet of Things (IoT) cihazları arasında güvenli iletişimi sağlamak üzere tasarlanmıştır. Şifreleme özellikleri ve güvenli veri iletimi, IoT cihazlarının hassas verilerini korumak adına etkili bir çözüm sunmaktadır.

Ayrıca, sistem farklı broker'lar üzerinde çalışan MQTT cihazları arasında merkezi olmayan veri senkronizasyonunu desteklemektedir. Bu sayede, veri bütünlüğü ve güvenliği sağlanırken, Docker konteyner teknolojisinin hızlı dağıtım ve ölçeklenme avantajlarından faydalanılmaktadır.

Geliştirilen model, iki ayrı MQTT broker kullanılabilirliği ile farklı şebeke altyapılarına uyum sağlamakta ve gateway bileşenleri aracılığıyla veri akışını güvenli bir şekilde yönetmektedir. Sistem, bu sayede çeşitli sektörlerdeki güvenli iletişim ihtiyaçlarına etkili bir çözüm sunmaktadır.

Sonuç olarak, bu çalışmanın elde ettiği bulgular, geliştirilen sistem ile güvenli IoT iletişimi, veri senkronizasyonu ve esnek broker yapıları gibi önemli konularda çeşitli uygulama alanlarına hitap ettiğini göstermektedir. Bu çalışma, gelecekteki güvenli iletişim sistemlerinin tasarımında ve uygulamasında rehberlik sağlayacak temel bir adım olarak değerlendirilebilir.

REFERANSLAR

[1] Nilupulee A. Gunathilake, William J. Buchanan, Rameez Asif, "Next Generation Lightweight Cryptography for Smart IoT Devices: Implementation, Challenges and Applications," presented at the 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), 2019.

[2] Pooja Shah, Mukesh Arora, Kinjal Adhvaryu, "Lightweight Cryptography Algorithms in IoT - A Study" presented at the 2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2020.

[3] Nilupulee A. Gunathilake, Ahmed Al-Dubai, William J. Buchanan, "Recent Advances and Trends in Lightweight Cryptography for IoT Security," presented at the 2020 16th International Conference on Network and Service Management (CNSM), 2020.

[4] Zeyad A. Al-Odat, Eman M. Al-Qtiemat, Samee U. Khan, "An Efficient Lightweight Cryptography Hash Function for Big Data and IoT Applications," presented at the 2020 IEEE Cloud Summit, 2020.

[5] Piyush Bagla, Ravi Sharma, Amit Kumar Mishra, Neha Tripathi, Ankur Dumka, Neeraj Kumar Pandey, "An Efficient Security Solution for IoT and Cloud Security Using Lattice-Based Cryptography," presented at the 2023 International Conference on Emerging Trends in Networks and Computer Communications (ETNCC), 2023, IEEE.

[6] Bindu Madavi K P, Krishna Sowjanya K, Neha Patwari, "Embedded Light-Weight Cryptography Technique to Preserve Privacy of Healthcare Wearable IoT Device Data," presented at the 2023 International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE), 2023.