

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

**MODBUS/TCP İÇİN HAFİF SIKLET ŞİFRELEMELİ
DONANIM MODÜLÜ İLE YENİ BİR GÜVENLİK YAKLAŞIMI**

DOKTORA TEZİ

Murat VAROL

Bilgisayar ve Bilişim Mühendisliği Anabilim Dalı

ARALIK 2025

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

**MODBUS/TCP İÇİN HAFİF SIKLET ŞİFRELEMELİ
DONANIM MODÜLÜ İLE YENİ BİR GÜVENLİK YAKLAŞIMI**

DOKTORA TEZİ

Murat VAROL

Bilgisayar ve Bilişim Mühendisliği Anabilim Dalı

Tez Danışmanı: Doç. Dr. Murat İSKEFİYELİ

ARALIK 2025

Murat VAROL tarafından hazırlanan “Modbus/TCP İçin Hafif Sıklet Şifrelemeli Donanım Modülü İle Yeni Bir Güvenlik Yaklaşımı” adlı tez çalışması 11.12.2025 tarihinde aşağıdaki jüri tarafından oy birliği ile Sakarya Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar ve Bilişim Mühendisliği Anabilim Dalı’nda Doktora tezi olarak kabul edilmiştir.

Tez Jürisi

Jüri Başkanı :

Jüri Üyesi :

Jüri Üyesi :

Jüri Üyesi :

Jüri Üyesi :



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 “MODBUS/TCP İÇİN HAFİF SIKLET ŞİFRELEMELİ DONANIM MODÜLÜ İLE YENİ BİR GÜVENLİK YAKLAŞIMI” 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.

(11/12/2025)

Murat VAROL





Hayat yolculuğumdakilere...



TEŞEKKÜR

Doktora süreci, sabır ve kararlılığın sınandığı uzun bir akademik yolculuktu. Doktora sürecimin her aşamasında değerli bilgi birikimi, yol göstericiliği ve desteği için danışmanım Sayın Doç. Dr. Murat İSKEFİYELİ hocama çok teşekkür ederim. Değerli öneri ve yorumlarıyla çalışmamı yönlendiren tez izleme komitesi ve tez savunma jüri üyelerine saygıyla teşekkür ederim.

Sevgi, sabır ve desteğiyle her zaman yanımda olan aileme teşekkürlerimi sunuyorum. Bu uzun akademik yolculukta bana eşlik eden dostlarıma ve hayatıma dokunmuş, akademik ve profesyonel gelişimime katkı sağlayan tüm herkese teşekkür ederim.

Ayrıca bu süreç boyunca beni destekleyen ve çalışmalarımı sürdürmemi sağlayan Zonguldak Bülent Ecevit Üniversitesi'ne de içtenlikle teşekkür ederim.

Bu çalışmanın gerçekleştirilmesinde altyapı ve teknik destek sunan Sakarya Üniversitesi Kritik Alt Yapılar Ulusal Test Yatağı Merkezi'ne teşekkür ederim.

Bu tez çalışması, Sakarya Üniversitesi Bilimsel Araştırma Projeleri Koordinatörlüğü tarafından 2023-19-44-46 nolu proje ile desteklenmiştir.

Murat VAROL



İÇİNDEKİLER

Sayfa

TEŞEKKÜR	ix
İÇİNDEKİLER	xi
KISALTMALAR	xiii
SİMGELER	xvii
TABLO LİSTESİ	xix
ŞEKİL LİSTESİ	xxi
ÖZET	xxiii
SUMMARY	xxvii
1. GİRİŞ	1
1.1. Tezin Kapsamı	4
1.2. Problem Tanımı ve Tezin Amacı	5
1.3. Literatür Araştırması	6
1.4. Hipotez	13
2. ENDÜSTRİYEL KONTROL SİSTEMLERİ	17
2.1. Endüstriyel Kontrol Sistemi	17
2.1.1. SCADA sistemi	20
2.1.2. Dağıtık kontrol sistemleri	22
2.1.3. Programlanabilir mantıksal denetleyici	23
2.2. Endüstriyel Haberleşme Protokolleri	24
2.3. EKS Güvenliği	25
3. KURAMSAL TEMELLER	29
3.1. Modbus/TCP Protokolü	29
3.1.1. Genel bakış	29
3.1.2. Modbus fonksiyonları ve kaydedicileri	31
3.1.3. Güvenlik açıkları	33
3.2. Hafif Sıklet Şifreleme Algoritmaları	34
3.2.1. Genel bakış	35
3.2.2. Present algoritması	36
3.2.3. XTEA algoritması	36
3.2.4. SKINNY algoritması	37
3.2.5. GRAIN algoritması	38
3.2.6. SIMON algoritması	39
3.2.7. ChaCha20 algoritması	40
3.2.8. Ascon algoritması	41
3.2.9. AES algoritması	42
4. ÖNERİLEN YÖNTEM	45
4.1. Şifreleme Modülü ve Bileşenleri	45
4.1.1. Raspberry Pi 4 tabanlı modül	46
4.1.2. Orange Pi R1 LTS plus tabanlı modül	48
4.2. Kullanılan Yazılım Bileşenleri	50
4.2.1. İşletim sistemi	50

4.2.2. Geliştirme ortamı.....	51
4.2.3. Benzetim programları.....	55
4.3. Test Ortamının Oluşturulması	59
4.3.1. Geliştirme test ortamı	59
4.3.2. Saldırı test ortamı	60
4.4. Şifreleme Algoritmalarının Gerçekleştirimi.....	61
5. UYGULAMA VE DEĞERLENDİRME	65
5.1. Modbus/TCP Paketinin İzlenmesi	65
5.2. Modbus/TCP Fonksiyonu Gerçekleştirilmesi	68
5.3. Modbus/TCP Paketinin Manipüle Edilmesi.....	70
5.4. Modbus/TCP Paketinin Şifrelenmesi	73
5.4.1. Modbus PDU şifrelenmesi	74
5.4.2. Modbus ADU şifrelenmesi.....	75
5.5. Şifreleme Sürelerinin Ölçülmesi	76
5.5.1. Raspberry Pi 4 tabanlı modül	77
5.5.2. Orange Pi R1 LTS Plus tabanlı modül	83
5.5.3. Genel karşılaştırma.....	89
5.6. Güvenlik Analizi	91
5.6.1. Modbus/TCP güvenlik analizi.....	91
5.6.2. Hafif sıklet şifreleme algoritmaları güvenlik analizi	93
5.7. Değerlendirme	94
6. SONUÇ VE ÖNERİLER.....	97
6.1. Önerilen Yöntemin Sınırlamaları ve Zorlukları	98
6.2. Gelecek Çalışmalar İçin Öneriler	98
KAYNAKLAR.....	101
ÖZGEÇMİŞ.....	113

KISALTMALAR

ABD	: Amerika Birleşik Devletleri
ADU	: Application Data Unit (Uygulama Veri Birimi)
AEAD	: Authenticated Encryption with Associated Data (İlişkili Verilerle Kimlik Doğrulanmış Şifreleme)
AES	: Advanced Encryption Standard (Gelişmiş Şifreleme Standardı)
ARM	: Advanced RISC Machine (Gelişmiş RISC Makinesi)
ARP	: Address Resolution Protocol (Adres Çözümleme Protokolü)
ASCII	: American Standard Code for Information Interchange (Amerikan Bilgi Alışverişi Standart Kodu)
ATHOM	: Akım Tahmin ve Havza Optimizasyon Modeli
C	: Ciphertext (Şifreli metin)
CIA	: Confidentiality, Integrity, and Availability (Gizlilik, Bütünlük ve Erişilebilirlik)
CIP	: Common Industrial Protocol (Ortak Endüstriyel Protokol)
CRC	: Cyclic Redundancy Check (Döngüsel Yedeklilik Kontrolü)
CTR	: Counter Mode (Sayaç Modu)
D	: Decryption (Şifre Çözme)
DC	: Direct Current (Doğru akım)
DDR	: Double Data Rate (Çift Veri Hızı)
DER	: Distributed Energy Resources (Dağıtılmış Enerji Kaynakları)
DERMS	: Distributed Energy Resources Management System (Dağıtılmış Enerji Kaynakları Yönetim Sistemi)
DKS	: Dağıtık Kontrol Sistemi (Distributed Control System)
DMZ	: Demilitarized Zone (Arındırılmış Bölge)
DoS	: Denial of Service (Hizmet Reddi)
DP	: Display Port (Ekran bağlantı noktası)
DSS	: Deep Security Scanner (Derin Güvenlik Tarayıcısı)
DT	: Decision Tree (Karar Ağacı)
DVI	: Digital Visual Interface (Dijital Görsel Arayüz)
E	: Encryption (Şifreleme)
ECB	: Electronic Code Book (Elektronik Kod Kitabı)

EKS	: Endüstriyel Kontrol Sistemi
EWS	: Engineering WorkStation (Mühendislik İş İstasyonu)
FDI	: False Data Injection (Yanlış Veri Enjeksiyonu)
GB	: Gigabayt
GCM	: Galois Counter Mode (Galois Sayaç Modu)
HDMI	: High-Definition Multimedia Interface (Yüksek Çözünürlüklü Multimedya Arayüzü)
HMAC	: Hashed Message Authentication Code (Karıştırılmış Mesaj Doğrulama Kodu)
HMI	: Human Machine Interface (İnsan Makine Arayüzü)
IACS	: Industrial Automation and Control Systems (Endüstriyel Otomasyon ve Kontrol Sistemleri)
ICS	: Industrial Control System (Endüstriyel Kontrol Sistemi)
IDPS	: Intrusion Detection and Prevention Systems (Saldırı Tespit ve Önleme Sistemleri)
IDS	: Intrusion Detection System (Saldırı Tespit Sistemi)
IED	: Intelligent Electronic Device (Akıllı Elektronik Cihaz)
IIoT	: Industrial Internet of Things (Endüstriyel Nesnelerin İnterneti)
IoT	: Internet of Things (Nesnelerin İnterneti)
IP	: Internet Protocol (İnternet Protokolü)
IT	: Information Technology (Bilgi Teknolojisi)
IV	: Initialization Vector (Başlatma Vektörü)
K	: Key (Anahtar)
LAN	: Local Area Network (Yerel Alan Ağı)
LR	: Logistic Regression (Logistik Regresyon)
LTS	: Long-Term Support (Uzun Süreli Destek)
MAC	: Message Authentication Code (Mesaj Doğrulama Kodu)
MBAP	: MODBUS Application Protocol (Modbus Uygulama Protokolü)
MiTM	: Man-in-The-Middle (Ortakdaki Adam)
MMBD	: Modbus-Markov Behavior Distinguisher (Modbus-Markov Davranış Ayırıcı)
MOSTO	: Modbus Simulator Tool (Modbus Simülasyon Aracı)
MTU	: Master Terminal Unit (Ana Terminal Ünitesi)
MUBUS	: Modified Ubus protocol (Değiştirilmiş Ubus protokolü)
NIST	: National Institute of Standards and Technology (Ulusal Standartlar ve Teknoloji Enstitüsü)

OPC UA	: Open Platform Communications Unified Architecture (Açık Platform İletişimi Birleşik Mimarisi)
OT	: Operational Technology (Operasyonel Teknoloji)
OTEC	: OT systems and Edge Computing (OT sistemleri ve Uç Bilişim)
OTP	: One Time Password (Tek Kullanımlık Şifre)
P	: Plaintext (Açık metin)
PDU	: Protocol Data Unit (Protokol Veri Ünitesi)
PERA	: Purdue Enterprise Reference Architecture (Purdue Kurumsal Referans Mimarisi)
PLC	: Programmable Logic Controller (Programlanabilir Mantık Denetleyici)
RAM	: Random Access Memory (Rastgele Erişimli Bellek)
RBAC	: Role Based Access Control (Rol Tabanlı Erişim Kontrolü)
RF	: Random Forest (Rastgele Orman)
RFID	: Radio Frequency IDentification (Radyo Frekansı Tanımlaması)
RISC	: Reduced Instruction Set Computer (Azaltılmış Komut Setli Bilgisayar)
RNN	: Recurrent Neural Network (Tekrarlayan Sinir Ağı)
RTU	: Remote Terminal Unit (Uzak Terminal Birimi)
SCADA	: Supervisory Control And Data Acquisition (Denetim Kontrolü ve Veri Toplama)
SHA	: Secure Hashing Algorithm (Güvenli Karma Algoritması)
SM3	: ShangMi 3
SM4	: ShangMi 4
SPN	: Substitution–Permutation Network (Değiştirme-Permütasyon Ağı)
SRMCP	: Secured Real-Time Machine Communication Protocol (Güvenli Gerçek Zamanlı Makine İletişim Protokolü)
SVM	: Support Vector Machine (Destek Vektör Makinesi)
SWaT	: Secure Water Treatment (Güvenli Su Arıtma)
TCP	: Transmission Control Protocol (İletim Kontrol Protokolü)
TLS	: Transport Layer Security (Taşıma Katmanı Güvenliği)
TSRR	: Test System Reception Rate (Test Sistemi Alım Oranı)
TÜBİTAK	: Türkiye Bilimsel ve Teknolojik Araştırma Kurumu
UART	: Universal Asynchronous Receiver-Transmitter (Evrensel Asenkron Alıcı-Verici)
USB	: Universal Serial Bus (Evrensel Seri Veri Yolu)
USB-C	: Universal Serial Bus Type-C (Evrensel Seri Veri Yolu Tip-C)

VGA	: Video Graphics Array (Video Grafik Dizisi)
VPN	: Virtual Private Network (Sanal Özel Ağ)
WAN	: Wide Area Network (Geniş Alan Ağı)
WGAN	: Wasserstein Generative Adversarial Network (Wasserstein Üretken Karşıt Ağ)
WWTP	: Waste Water Treatment Plant (Atık Su Arıtma Tesisi)
XOR	: eXclusive OR (Özel VEYA)
XTEA	: Extended Tiny Encryption Algorithm (Genişletilmiş Küçük Şifreleme Algoritması)



SİMGELER

$\gg$	: Bit sağa kaydırma
$\ll$	: Bit sola kaydırma
$\oplus$	: XOR işlemi
$\&$	: Mod alma işlemi (bit düzeyinde VE işlemi)
δ	: Sabit değer
$\wedge$	: Mantıksal VE işlemi



TABLO LİSTESİ

Sayfa

Tablo 2.1. EKS Tanımları.	19
Tablo 3.1. Temel Modbus Fonksiyonları.	32
Tablo 3.2. Modbus Kaydedici Referans Adresleri.	32
Tablo 3.3. Modbus Güvenlik Açıkları.....	34
Tablo 4.1. Kullanılan Raspberry Pi tabanlı modül özellikleri.....	48
Tablo 4.2. Kullanılan Orange Pi tabanlı modül özellikleri.	49
Tablo 5.1. Raspberry Pi 4 üzerinde Modbus PDU alanı işlem süreleri (s).	78
Tablo 5.2. Raspberry Pi 4 üzerinde Modbus ADU alanı işlem süreleri (s).....	79
Tablo 5.3. Orange Pi R1 LTS Plus üzerinde Modbus PDU alanı işlem süreleri (s).	84
Tablo 5.4. Orange Pi R1 LTS Plus üzerinde Modbus ADU işlem süreleri (s).....	84
Tablo 5.5. Şifreleme algoritmaları güvenlik problemleri.....	93



ŞEKİL LİSTESİ

Sayfa

Şekil 2.1. Basit bir EKS Çalışma Diyagramı.	20
Şekil 2.2. Örnek bir SCADA sistemi topolojisi.	21
Şekil 2.3. Örnek bir DKS topolojisi.	22
Şekil 2.4. Basit bir kontrol sistemi topolojisi.	23
Şekil 2.5. Purdue Kurumsal Referans Mimarisi (PERA)	27
Şekil 3.1. Modbus/TCP Paket Yapısı.	30
Şekil 3.2. Modbus/TCP Paket Kapsüllemesi.	31
Şekil 3.3. Present algoritması sözde kodu.	36
Şekil 3.4. XTEA algoritması sözde kodu.	37
Şekil 3.5. SKINNY algoritması sözde kodu.	38
Şekil 3.6. GRAIN algoritması sözde kodu.	39
Şekil 3.7. SIMON algoritması sözde kodu.	40
Şekil 3.8. ChaCha20 algoritması sözde kodu.	41
Şekil 3.9. Ascon algoritması sözde kodu.	42
Şekil 3.10. AES-128 algoritması sözde kodu.	43
Şekil 4.1. Önerilen Modüle Genel Bakış.	46
Şekil 4.2. Önerilen Modülün Çalışma Diyagramı.	46
Şekil 4.3. Raspberry Pi 4 Cihazı Genel Görünümü.	47
Şekil 4.4. USB 3.0 Ethernet Adaptörü Genel Görünümü.	47
Şekil 4.5. Orange Pi R1 LTS Plus Cihazı Görünümü.	49
Şekil 4.6. Ethernet arayüzleri köprüleme işlemi.	52
Şekil 4.7. IP checksum hesaplanırken kullanılan alanlar.	53
Şekil 4.8. TCP checksum hesaplanırken kullanılan alanlar.	54
Şekil 4.9. Proses Kullanımı.	54
Şekil 4.10. ModFlask Master Cihaz Ekran Görüntüsü.	56
Şekil 4.11. ModFlask Master Cihaz Web Arayüzü.	56
Şekil 4.12. ModFlask Slave Cihaz Çalıştırılması.	57
Şekil 4.13. ModFlask Modbus Fonksiyonları.	57
Şekil 4.14. Modbus Poll Master Cihaz Arayüzü.	58
Şekil 4.15. Modbus Slave Cihaz Görünümü.	58
Şekil 4.16. Geliştirme Test Ortamı.	59
Şekil 4.17. Saldırı test ortamı.	61
Şekil 4.18. Şifreleme Algoritmaları Sınıf Diyagramı Gösterimi.	63
Şekil 5.1. Wireshark ile Modbus/TCP paketlerinin filtrelenmesi.	66
Şekil 5.2. Wireshark ortamında yakalanmış bir Modbus/TCP paketi.	66
Şekil 5.3. Python ortamında Modbus/TCP paketi yakalanması.	67
Şekil 5.4. Python ortamında yakalanan Modbus/TCP paketi.	67
Şekil 5.5. Wireshark ortamında yakalanmış fonksiyon kodu 6 olan bir paket.	68
Şekil 5.6. Fonksiyon kodu 6 olan Modbus/TCP paketi tanımlanması.	68
Şekil 5.7. Fonksiyon 6 Modbus/TCP paketinin kapsüllemesi.	69
Şekil 5.8. Wireshark ortamında yakalanmış fonksiyon kodu 15 olan bir paket.	69

Şekil 5.9. Fonksiyon kodu 15 olan Modbus/TCP paketi tanımlanması.....	69
Şekil 5.10. Fonksiyon 15 Modbus/TCP paketinin kapsüllenmesi.	70
Şekil 5.11. Modbus master cihazından yollanan istek (a) ve slave taraftaki sonuç (b).	70
Şekil 5.12. Register değeri manipüle edilmiş Modbus/TCP paket yapısı.....	71
Şekil 5.13. Modbus master cihazından yollanan istek (a) ve slave taraftaki değiştirilmiş sonuç (b).	71
Şekil 5.14. Modbus master cihazından yollanan istek (a) ve slave taraftaki sonuç (b).	72
Şekil 5.15. Coil değerleri manipüle edilmiş Modbus/TCP paket yapısı.....	72
Şekil 5.16. Modbus master cihazından yollanan istek (a) ve slave taraftaki değiştirilmiş sonuç (b).	73
Şekil 5.17. Modbus veri alanının şifrelenmesi.....	74
Şekil 5.18. PDU alanı şifrelenmemiş Modbus/TCP paketi.....	74
Şekil 5.19. PDU alanı şifrelenmiş Modbus/TCP paketi.....	75
Şekil 5.20. Modbus başlık ve veri alanlarının şifrelenmesi.	75
Şekil 5.21. ADU alanı şifrelenmemiş Modbus/TCP paketi.	76
Şekil 5.22. ADU alanı şifrelenmiş Modbus/TCP paketi.	76
Şekil 5.23. 10 adet coil değeri içeren Modbus/TCP paketi.....	77
Şekil 5.24. 10 adet register değeri içeren Modbus/TCP paketi.....	77
Şekil 5.25. 100 adet register değeri içeren Modbus/TCP paketi.....	77
Şekil 5.26. Raspberry Pi 4 üzerinde Senaryo-1 için Modbus PDU işlem süreleri.....	80
Şekil 5.27. Raspberry Pi 4 üzerinde Senaryo-1 için Modbus ADU işlem süreleri. ...	80
Şekil 5.28. Raspberry Pi 4 üzerinde Senaryo-2 için Modbus PDU işlem süreleri.....	81
Şekil 5.29. Raspberry Pi 4 üzerinde Senaryo-2 için Modbus ADU işlem süreleri. ...	82
Şekil 5.30. Raspberry Pi 4 üzerinde Senaryo-3 için Modbus PDU işlem süreleri.....	82
Şekil 5.31. Raspberry Pi 4 üzerinde Senaryo-3 için Modbus ADU işlem süreleri. ...	83
Şekil 5.32. Orange Pi R1 LTS Plus üzerinde Senaryo-1 için Modbus PDU işlem süreleri.	85
Şekil 5.33. Orange Pi R1 LTS Plus üzerinde Senaryo-1 için Modbus ADU işlem süreleri.	86
Şekil 5.34. Orange Pi R1 LTS Plus üzerinde Senaryo-2 için Modbus PDU işlem süreleri.	86
Şekil 5.35. Orange Pi R1 LTS Plus üzerinde Senaryo-2 için Modbus ADU işlem süreleri.	87
Şekil 5.36. Orange Pi R1 LTS Plus üzerinde Senaryo-3 için Modbus PDU işlem süreleri.	88
Şekil 5.37. Orange Pi R1 LTS Plus üzerinde Senaryo-3 için Modbus ADU işlem süreleri.	89
Şekil 5.38. Orange Pi R1 LTS Plus üzerinde hesaplanan şifre çözme işlem süreleri.....	89
Şekil 5.39. Orange Pi R1 LTS Plus üzerinde hesaplanan şifre çözme işlem süreleri.....	90
Şekil 5.40. Raspberry Pi 4 üzerinde hesaplanan şifre çözme işlem süreleri.	90
Şekil 5.41. Raspberry Pi 4 üzerinde hesaplanan şifreleme işlem süreleri.....	91

MODBUS/TCP İÇİN HAFİF SIKLET ŞİFRELEMELİ DONANIM MODÜLÜ İLE YENİ BİR GÜVENLİK YAKLAŞIMI

ÖZET

Endüstriyel kontrol sistemleri, hayatımızı doğrudan etkileyen elektrik, su, enerji gibi kritik alt yapıları yönetmek için kullanılmaktadır. Bu sistemlerde kullanılan haberleşme protokolleri tasarlandıkları yıl itibarıyla güvenlik parametreleri göz önüne bulundurulmadan geliştirilmiştir. Teknolojinin sürekli gelişiminden ötürü bu sistemler artık internete bağlı hale gelmişlerdir. Dolayısıyla bu sistemlerin yapısı birçok güvenlik tehdidine karşı savunmasız durumda olabilmektedir. Bu tezde, endüstriyel kontrol sistemlerinde en çok kullanılan Modbus/TCP protokolünün güvenlik açıklarını kapatmaya yönelik çalışma yapılmıştır. Modbus/TCP protokolü yapısı gereği, veri bütünlüğü, veri gizliliği ve oturum açma gibi temel güvenlik kontrollerine sahip değildir. Bu eksiklikler Modbus/TCP protokolünü kullanan sistemleri; hizmet reddi, ortadaki adam, paket dinleme, paket manipüle etme gibi birçok saldırıya karşı açık hedef haline getirmektedir. Bu durum protokole bir güvenlik katmanı eklenmesi ihtiyacını doğurmuştur. Bu sebeple, protokolün güvenliğini artırmak için endüstriyel sistemin çalışmasını olumsuz yönde etkilemeyecek yaklaşımlar kritik önem arz etmektedir.

Bu problemleri çözmek için farklı tekniklerde fazla sayıda çalışma yapılmış ancak bu çalışmaların bir çoğu gerçek zamanlı sistem üzerinde uygulanabilir değildir. Yapılan çalışmalarda; makine öğrenmesi, derin öğrenme, kriptografik şifreleme gibi alanlarda bu sorunları çözmek için değerlendirmeler yapılmıştır. Makine öğrenmesi ve derin öğrenme tabanlı çözümlerde genellikle saldırı tespit veya engelleme sistemlerinin tasarlanması üzerine odaklanılmıştır. Kriptografik şifreleme tabanlı çözümlerin uygulanması esnasında şifreleme işlemlerinin hangi cihaz üzerinde yapılacağı tartışma konusu olmuştur. Endüstriyel cihazlar, düşük işlem kapasitesine sahip kaynak kısıtlı donanımlardır. Bu cihazlar üzerinde şifreleme yapıldığı düşünülürse, işlem gecikmesinden dolayı endüstriyel kontrol sisteminin işleyişi aksayabilir. Ayrıca, endüstriyel sistemlerde bulunan cihazların yazılımını güncellemek veya cihazların yenilenmesi büyük bir maliyet ve iş yükü oluşturmaktadır. Bu noktada, bu çalışmada şifreleme ve şifre çözme işlemleri için endüstriyel cihazların önüne bir modül yerleştirilmesi önerilmiştir. Böylelikle eski cihazlarla yaşanabilecek uyumluluk probleminin de önüne geçilmiştir. Önerilen modülü gerçekleştirmek için Raspberry Pi 4 ve Orange Pi R1 LTS Plus cihazları tercih edilmiştir. Orange Pi R1 LTS Plus cihazı kendi üzerinde çift ethernet portuna sahip olduğundan araya yerleştirme işlemine uygunken Raspberry Pi 4 cihazını araya yerleştirip bağlantı sağlamak için bir USB ethernet dönüştürücü kullanılmıştır. Modülü gerçekleştiren cihazlar üzerindeki ethernet arayüzleri köprü yapısı oluşturularak paketlerin arayüzler arasında yönlendirilmesi sağlanmıştır. Şifreleme işlemi aktif olmadığında gelen paketler direkt olarak hedef adrese doğru yönlendirilir. Şifreleme işlemi aktif olduğunda ise Modbus/TCP paketleri şifrelenmek üzere işleme alınmaktadır. Linux tabanlı işletim sistemi üzerinde Python programlama dili kullanılarak hafif sıklet şifreleme

algoritmaları ile Modbus/TCP paketleri iki farklı senaryoda şifrelenmektedir. İlk olarak, Modbus/TCP protokolünün PDU alanı; ikinci olarak ise Modbus/TCP ADU alanı şifrelenmektedir. Her iki senaryoda da IP ve TCP başlık alanları şifrelenmemektedir.

Tez çalışmasında, Python programlama dili ile aynı boyutta iki farklı fonksiyona ait Modbus/TCP paketleri oluşturulmuş, yedi adet hafif sıklet şifreleme algoritması ve şifreleme standardı olan AES algoritması ile oluşturulan bu paketler gönderen tarafta şifrelenmiş olup alıcı tarafta ise paketlerin şifresi çözülmüştür. Gönderilen Modbus/TCP paketleri Modbus fonksiyon kodu 15 olan Write Multiple Coils ve Modbus fonksiyon kodu 16 olan Write Multiple Register fonksiyonlarının verilerini içermektedir. Modbus/TCP verilerinin şifrelenmesi, gizlilik parametresinin gerçekleştirilmesini sağlamaktadır. Ancak, paketin değiştirilip değiştirilmediğinin ve tekrar gönderilip gönderilmediğinin bilgisine karşın bir çözüm sunmamaktadır. Bu hususta; bütünlük kontrolü için HMAC kodu ve tekrar gönderimin tespiti için nonce değeri önerilen modül üzerinde şifreleme esnasında Modbus/TCP paketine eklenmektedir. Bu eklenen iki özellikten alıcı taraftaki modül üzerinde öncelikle HMAC koduna bakılır, eğer hesaplanan değer ile uyuşmuyorsa paketin hedefe iletilmesi engellenir. Eğer HMAC kodu ve hesaplanan değer uyuşuyorsa bu sefer paketin şifresi çözülür ve nonce değeri kontrol edilir. Eğer nonce değeri daha önce gönderilen bir değer ise paketin hedefe ulaşması yine engellenir. Ancak daha önce gönderilmemiş bir değer ise paket başarıyla hedefe yönlendirilir.

Çalışmada, Modbus master ve Modbus slave cihazlarını temsil etmek için bilgisayarlar üzerinde benzetim programları ve iki adet önerilen modül kullanarak test ortamı oluşturulmuştur. Önerilen şifreleme modülleri, bilgisayarlar arasında yerleştirilerek şifreleme ve şifre çözme için uygun ortam hazırlanmıştır. Önerilen modül çalıştırıldığında kullanıcıdan şifreleme algoritmasını ve şifreleme modunu seçmesini istemektedir. Performans testleri yapılmak üzere, her bir şifreleme algoritması ve şifreleme modu Modbus master cihazından 100 adet aynı boyutlu Write Multiple Coils paketi ve 100 adet aynı boyutlu Write Multiple Register paketi slave cihaza doğru gönderilmiştir. Gönderilen paketler, gönderici taraftaki modül üzerinde şifrelenmiş ve alıcı taraftaki modül üzerinde şifresi çözülmüştür. Tüm bu işlemler yapılırken şifreleme ve şifre çözme işlem süreleri kayıt altına alınmış olup çalışmada detaylı olarak verilmiştir. Yapılan ölçümler neticesinde işlem sürelerinin farklılıkları değerlendirilmiş olup kullanılan donanım, şifreleme algoritması, şifreleme modu ve Modbus fonksiyonu seçimlerinin performansa etkisi karşılaştırılmıştır.

Ölçüm sonucunda elde edilen sürelerle bakıldığında; AES, Grain, Simon ve XTEA algoritmalarının diğer şifreleme algoritmalarına göre hız açısından öne çıktığı görülmektedir. AES algoritması hafif sıklet şifreleme algoritması olmamasına rağmen bazı hafif sıklet şifreleme algoritmalarından üstün performans göstermiştir. Bu durumun sebebi, AES algoritmasının donanım hızlandırması desteğine sahip olmasıdır. Ayrıca, Raspberry Pi 4 ve Orange Pi R1 LTS Plus cihazlarının performansları karşılaştırıldığında Raspberry Pi 4 cihazının öne çıktığı da görülmektedir. Raspberry Pi 4 cihazının daha üstün donanım özelliklerine sahip olduğu göz önüne alındığında bu durumun normal olduğu anlaşılmaktadır.

Elde edilen sonuçlar değerlendirildiğinde, hafif sıklet şifreleme algoritmalarının önerilen modüller kullanılarak Modbus/TCP paketleri üzerinde uygulanmasının performans bakımından kabul edilebilir bir işlem maliyeti oluşturduğu sonucuna varılmıştır. Şifreleme ve şifre çözme sürelerinin milisaniyeler boyutunda olması

önerilen modülün gerçek zamanlı çalışan endüstriyel kontrol sistemlerinde kullanılabilir olduğunu göstermektedir. Ayrıca, üzerinde işlem yapılan Modbus/TCP paketinin boyutu sabit olsa dahi şifreleme sürelerinin değişken olabileceği ve bunun sebebinin algoritmaların yapısı, kullanılan donanım, sistemin o anki iş yükü gibi etkenlerle bağlantılı olduğu da çalışmada gösterilmiştir.

Sonuç olarak bu tez çalışması, Modbus/TCP tabanlı endüstriyel ağ ortamında güvenliği artırmak için Raspberry Pi 4, Orange Pi R1 LTS Plus ve hafif sıklet şifreleme algoritmalarının kullanılabilir olduğunu deneysel çalışmalar ile ortaya koymaktadır. Ek olarak, Modbus/TCP protokolüne eklenen alanlar ile birlikte güvenlik katmanı oluşturulmakla birlikte ortadaki adam, yeniden gönderme, yetkisiz erişim, hizmet reddi gibi saldırılara karşı koruma sağladığı gösterilmektedir. Son olarak bu çalışma, EKS güvenliği konusunda literatüre katkı sunarak gelecekte yapılacak donanım tabanlı çalışmalara temel oluşturmaktadır.





A NEW SECURITY APPROACH WITH LIGHTWEIGHT ENCRYPTED HARDWARE MODULE FOR MODBUS/TCP

SUMMARY

Industrial control systems are used to manage critical infrastructures such as electricity, water, and energy, which directly impact our lives. The communication protocols used in these systems have been developed without considering security parameters since their conception. Due to the continuous advancement of technology, these systems are now connected to the Internet. Consequently, the structure of these systems can be vulnerable to numerous security threats. This thesis aims to address the security vulnerabilities of the Modbus/TCP protocol, one of the most commonly used in industrial control systems. The Modbus/TCP protocol, by its very nature, lacks fundamental security controls such as data integrity, data confidentiality, and login. These deficiencies make systems using the Modbus/TCP protocol vulnerable to numerous attacks, including denial of service, man-in-the-middle attacks, packet sniffing, and packet manipulation. This has led to the need to add a security layer to the protocol. Therefore, approaches that do not negatively impact the operation of industrial systems are critical for enhancing the security of protocol.

Numerous studies have been published using various techniques to address these problems, but many are not applicable to real-time systems. Studies have evaluated these issues in areas such as machine learning, deep learning, and cryptographic encryption. Machine learning and deep learning-based solutions generally focus on designing intrusion detection or prevention systems. When implementing cryptographic encryption-based solutions, the appropriate device for encryption operations has been a matter of debate. Industrial devices are resource-constrained hardware with limited processing capacity. Given that encryption is performed on these devices, processing delays can disrupt the operation of the industrial control system. Furthermore, updating or upgrading the software of devices in industrial systems creates significant costs and workloads. Therefore, this study proposes placing a module in front of industrial devices for encryption and decryption. This avoids potential compatibility issues with older devices. Raspberry Pi 4 and Orange Pi R1 LTS Plus were chosen to implement the proposed module. The Orange Pi R1 LTS Plus device has dual Ethernet ports, making it suitable for placement between devices. A USB-to-Ethernet converter was used to connect a Raspberry Pi 4 device to the placement between devices. The Ethernet interfaces on the devices implementing the module are bridged, allowing packets to be routed between them. When encryption is disabled, incoming packets are routed directly to the destination address. When encryption is enabled, Modbus/TCP packets are processed for encryption. On a Linux-based operating system, Modbus/TCP packets are encrypted using lightweight encryption algorithms using the Python programming language in two different scenarios. First, the PDU of the Modbus/TCP protocol is encrypted; second, the ADU of the Modbus/TCP protocol is encrypted. In neither scenario are the IP and TCP header fields encrypted.

In this thesis, Modbus/TCP packets of the same size were generated using the Python programming language for two different functions. These packets, generated using seven lightweight encryption algorithms and the AES encryption standard, were encrypted on the sender side and decrypted on the receiver side. The sent Modbus/TCP packets contained data have the Write Multiple Coils function (Modbus function code 15) and the Write Multiple Register function (Modbus function code 16). Encrypting Modbus/TCP data ensures the implementation of confidentiality parameter. However, no information whether the packet has been altered or retransmitted, this attempt does not provide a solution. In this regard, the HMAC code for integrity checking and the nonce value for retransmission detection are added to the Modbus/TCP packet during encryption in the proposed module. Of these two added features, the receiving module first examines the HMAC code; if it does not match the calculated value, the packet is prevented from being forwarded to the destination. If the HMAC code and the calculated value match, the packet is decrypted and the nonce value is checked. If the nonce value is a value that has been sent before, the packet is still blocked from reaching the destination. However, if it is a value that has not been sent before, the packet is successfully forwarded to the destination.

In this study, a test environment is created on two proposed modules and two computers that using simulation programs to represent Modbus master and Modbus slave devices. The proposed encryption modules were placed between the computers to provide a suitable environment for encryption and decryption. When the proposed module is run, it prompts the user to select the encryption algorithm and encryption mode. To conduct performance tests, 100 identically sized Write Multiple Coils packets and 100 identically sized Write Multiple Register packets were sent from the Modbus master device to the slave device for each encryption algorithm and encryption mode. The sent packets were encrypted on the sender module and decrypted on the receiver module. The encryption and decryption processing times for all these processes were recorded and detailed in the study. Differences in processing times were evaluated based on the measurements, and the performance effects of the hardware, encryption algorithm, encryption mode, and Modbus function selections were compared.

When looking at the measured times, the AES, Grain, Simon, and XTEA algorithms stand out in terms of speed compared to other encryption algorithms. Although the AES algorithm is not a lightweight encryption algorithm, it outperformed some lightweight encryption algorithms. This is due to the AES algorithm's hardware acceleration support. Furthermore, when comparing the performance of the Raspberry Pi 4 and the Orange Pi R1 LTS Plus devices, the Raspberry Pi 4 stands out. This is understandable given the Raspberry Pi 4's powerful hardware specifications.

The results indicate that implementing lightweight encryption algorithms on Modbus/TCP packets using the proposed modules results in an acceptable processing cost in terms of performance. Encryption and decryption times in milliseconds demonstrate the proposed module's suitability for use in real-time industrial control systems. Furthermore, the study demonstrates that even when the Modbus/TCP packet being processed is fixed in size, encryption times can vary, depending on factors such as the algorithm structure, the hardware used, and the system's current workload.

In conclusion, this thesis demonstrates through experimental studies that Raspberry Pi 4, Orange Pi R1 LTS Plus, and lightweight encryption algorithms can be used to enhance security in Modbus/TCP-based industrial networks. Furthermore, the addition

of fields to the Modbus/TCP protocol provides a security layer, demonstrating protection against attacks such as man-in-the-middle, replay, unauthorized access, and denial of service. Finally, this study contributes to the literature on ICS security and lays the groundwork for future hardware-based studies.





1. GİRİŞ

Kritik altyapılar, hayatın sürdürebilmesi için başta enerji üretim ve dağıtım sistemleri olmak üzere son derece önemli olan sistemlerdir. Gündelik yaşam ihtiyaçlarının sürdürülebilmesi, halkın sağlığının ve güvenliğinin sağlanması, ülkenin ekonomisinin korunması vb. için sürekli olarak çalışmasının gerekli olduğu ve büyük arz önem eden sistemlerin tamamı kritik altyapıları oluşturmaktadır. Bu sistemler; enerji, elektrik, su, doğalgaz, ulaşım, haberleşme, bankacılık ve finans, ulaştırma gibi alanlarda faaliyet göstermektedir. Kritik alt yapı sistemleri bilgi teknolojilerini içermekle birlikte operasyonel teknolojilerini kullanmaktadır. Operasyonel Teknoloji (OT), endüstriyel operasyonları yöneten bilgi işlem sistemlerini kapsamaktadır. Bu teknolojiler, petrol ve gaz enerjisinin, elektrik hizmet şebekesinin, üretim operasyonlarının ve birçok sistemin izlenmesini içermektedir. Basitçe örneklendirmek gerekirse OT, evdeki elektriğin açılması veya musluktan akan temiz su gibi ortak standartların sürdürülmesine izin veren ağları çalıştırmaktadır. Endüstriyel kontrol sistemi (EKS), endüstriyel işlemleri yönetmek veya otomatik hale getirmek için kullanılan çeşitli cihazlar, sistemler, ağlar ve kontrol fonksiyonlarını içeren ayrı tipteki sistemleri ve bileşenlerini tanımlamaktadır.

Endüstriyel Kontrol Sistemleri; SCADA sistemleri, dağıtılmış kontrol sistemleri, programlanabilir mantıksal denetleyiciler gibi diğer kontrol sistem yapılandırmalarında kullanılan bir terimdir. Bir EKS ağı, birçok kritik altyapı ve ham madde sistemini izlemeye olanak tanımaktadır. EKS ağları, görev açısından kritik dereceye sahiptir; anlık tepki ve yüksek kullanılabilirlik gerektirmektedir. Birçok bakımdan üzerinde durulan bu husus, IT ve OT arasındaki temel farkı ifade etmektedir. IT sistemleri için güvenlik parametresi; gizlilik, bütünlük ve erişilebilirlik (CIA) üçlüsü tarafından aynı yüksek öncelikte korunmaktadır. OT sistemlerinde ise güvenlik parametresi; hem bütünlük hem de gizlilik, erişilebilirlik unsurundan sonra gelmektedir. Başka bir deyişle bu sistemlerde yüksek öncelik olarak erişilebilirlik ele alınmaktadır. Çünkü bir OT sistemindeki bilginin ele geçirilmesinden daha çok o sistemin işleyişini sürdürmesi daha önemlidir. Sistemin gizliliğini sağlamak amacıyla işleyişin kesintiye uğratılması daha büyük kayıplara yol açabilir. SCADA, büyük ve

karmaşık süreçleri yönetmek için kullanılan bir sistem mimarisidir. SCADA sistemleri normalde, kontrol fonksiyonlarının geniş bir coğrafi alana dağıtıldığı, üretim ve dağıtım yapan hizmet sağlayıcılarda bulunmaktadır. SCADA sistemlerinin temel amacı veri toplamaktır. SCADA sistemi ağları; üst düzey kararlar almak amacıyla başvurabilecekleri merkezi komuta istasyonuna veri göndermek için kullanılan çoklu uzak terminal birimlerinden (RTU) oluşmaktadır. SCADA sistemlerinin temelinde bilgi teknolojileri (IT) yer almaktadır. SCADA sistemleri, WAN üzerinden uzaktan yönetim olanağı sunmaktadır. Dolayısıyla bu noktada sistemin güvenlik parametrelerinin kapsamlı bir şekilde ele alınması gerekmektedir. Aksi takdirde burada göz ardı edilecek bir unsur tüm sistemin işleyişini bozabilir ve büyük zararlara yol açabilir. Dağıtılmış Kontrol Sistemi (DKS), kontrolörleri, sensörleri, operatör terminallerini ve aktüatörleri birbirine bağlayan bir tür proses kontrol sistemidir. Bu sistemlerde, veri toplama ve kontrol işlevleri, verilerin toplandığı çevresel cihazların veya bunların yakınında bulunan dağıtılmış işlemciler tarafından gerçekleştirilmektedir.

Geçmişte EKS'ye yönelik çok sayıda siber saldırı gerçekleşmiştir. Bu saldırılardan en bilineni, İran'ın nükleer enerji santraline gerçekleştirilen ve varlığı 2010 yılında farkedilen Stuxnet saldırısıdır. Bu saldırı sonucunda İran 800 milyon dolarlık maddi zarara uğramıştır. Karabacak, kritik altyapıların korunması hakkında yaptığı çalışmada; ABD elektrik kesintisi saldırısı, Houston Limanı saldırısı, Davis-Besse nükleer santrali saldırısı, Batman Barajı saldırısı ve Atatürk Havalimanı saldırısı örneklerini anlatmıştır [1]. Bu saldırılara bakıldığında; 2003'teki ABD elektrik kesintisi saldırısı 50 milyon kişiyi etkilemiş ve 6 milyar dolar zarara sebep vermiş olup, saldırının temelinde elektrik dağıtım yazılımındaki hatadan kaynaklı olduğu belirtilmiştir. 2003 yılındaki Houston Limanı saldırısına bakıldığında, saldırının İngiltere'deki bir zombi bilgisayar tarafından gerçekleştirildiği tespit edilmiştir. Yine 2003 senesinde olan Davis-Besse nükleer santraline yönelik yapılan saldırıda, kurumun ağına Slammer solucanı bulaşmış ve sistemi aksatmıştır. 2009 yılında yayılan Conficker virüsü Atatürk Havalimanı'ndaki bilgisayarlara bulaşmış olup havalimanı içerisinde aksaklıklar yaşanmıştır. Parasının ödenmediği gerekçesiyle 2003 yılında Alman firması tarafından Batman Barajı'nın bilgisayarları kilitlenmiş ve sistem aksamıştır. Bu durum, yerli ve milli ürün geliştirmenin önemini vurgulamaktadır. TÜBİTAK tarafından barajlar için ATHOM adında yerli bir ürün

geliştirilmiştir. 2015 yılında Ukrayna’da gerçekleşen elektrik kesintisi saldırısının sebebi ortalama e-postaları olduğu anlaşılmıştır. 2020 yılında SolarWinds Orion ağ izleme platformunu hedef alan saldırıda birçok kurum etkilenmiş olup saldırının sebebi ise güncelleme yoluyla bilgisayarlara truva atı yüklenmesidir. 2021 yılında ABD koloni boru hattı saldırısında ise saldırının sebebi uzaktan yönetim için kullanılan VPN kullanıcı şifresinin güvenlik seviyesinin düşük olmasıdır.

EKS’de güvenlik sağlamak amacıyla yapılan çalışmaların ne kadar önem arz ettiği maddi ve manevi kayıpların sonuçlarından anlaşılmaktadır. Bu sistemlerde kullanılan birçok endüstriyel protokol bulunmaktadır. Modbus/TCP, DNP3, EtherNet/IP, Profinet, Profibus, Hart gibi protokoller kullanılmaktadır. Modbus/TCP, bu sistemlerde en çok kullanılan protokol olmakla birlikte SCADA sistemlerine yönelik geliştirilmiştir. Endüstriyel standart olarak kabul görülebilen bu protokol, farklı endüstriyel ağlardaki cihazlar arasında kullanılmakta ve istemci/sunucu iletişimi için TCP/IP mimarisi uygulama katmanında mesajlaşma görevini üstlenmektedir. Bu çalışma kapsamında yapılacak güvenlik çözümleri için Modbus/TCP protokolünü kullanan sistemler ele alınmıştır. Son zamanlarda endüstriyel sistem protokollerinin güvenliğini artırmak amacıyla birçok çalışma yapılmıştır. Bu çalışmaların çoğunda ele alınan protokol, Modbus/TCP protokolüdür.

Hafif sıklet şifreleme algoritmaları, kısıtlı donanım kaynaklarına sahip cihazlarda güvenlik sağlamak için geliştirilmiş ve hesaplama yükü düşük kriptografik algoritmalarlardır. Bu algoritmalar, geleneksel şifreleme algoritmalarına göre daha az enerji, işlem gücü ve bellek harcarlar. Bu sebepten ötürü IoT cihazları, RFID etikleri gibi kaynak kısıtlı cihazlarda tercih edilirler. Bu algoritmalar genellikle blok şifreleme yapısını kullanır. Enerji verimliliği, hız, güvenlik seviyesi, kaynak kullanımı gibi kriterler göz önünde bulundurularak bu algoritmalar değerlendirilir.

Endüstriyel Kontrol Sistemleri’nde en çok kullanılan protokol olan Modbus/TCP protokolü, dizayn edildiği dönemde sistemlerin internete bağlanması ve güvenlik mekanizmaları düşünülmeden tasarlanmıştır. Günümüzde EKS internete bağlı hale gelmiş ve güvenlik gereksinimleri artmıştır. Kimlik doğrulama, bütünlük kontrolü, şifreleme gibi mekanizmaların olmaması sistemin güvenliğini riske atmaktadır. Bu hususta hafif sıklet şifreleme algoritmaları kullanarak protokol paketlerinin şifrelenmesi etkili bir çözüm olarak değerlendirilmiştir. Geleneksel şifreleme yöntemlerinin aksine, hafif sıklet şifreleme algoritmaları düşük kaynak kullanımı ve

yüksek hesaplama hızı gibi avantajları sayesinde endüstriyel cihazların iletişimini güvenli hale getirecek şekilde bir modüle entegre edilebilir. Böylelikle, Modbus/TCP protokolü kullanan cihazların üzerinden iletilen paketler, sistemin performansını etkilemeyecek şekilde gizlilik ve bütünlük açısından güvenliği sağlanmış olur. Bunun yanı sıra modül üzerine eklenecek mekanizmalar ile diğer saldırı türlerine yönelik tedbirler alınabilir.

Bu çalışmada, endüstriyel cihazların haberleşmesinde kullanılan Modbus/TCP protokolünün güvenlik açıklarından bahsedilmiş ve bazı güvenlik açıkları kullanılarak sistem manipüle edilmiştir. Güvenlik açıklarına karşı önlem almak için hafif sıklet şifreleme algoritmalarının Modbus/TCP protokolü verileri üzerinde performansı araştırılmış ve uygulaması yapılmıştır. Bu doğrultuda, Python programlama dili kullanılarak Modbus/TCP paketinin şifrenmesi ve şifresinin çözülmesi işlemleri iki farklı modül üzerinde gerçekleştirilmiştir. Yapılan bu uygulamalardan elde edilen performans sonuçları ve güvenlik durumları analiz edilmiştir.

Bu bölümde tezin kapsamı, problem tanımı ve tezin amacı, literatür çalışmaları ve hipotez başlıkları bulunmaktadır. Bölüm 2’de Endüstriyel Kontrol Sistemleri hakkında genel bilgi, güvenlik ve protokoller hakkında bilgiler verilmiştir. Bölüm 3’de Modbus/TCP protokolü ve hafif sıklet şifreleme algoritmaları hakkında teorik bilgiler verilmiştir. Bölüm 4’de önerilen yöntem anlatılmış, kullanılan yazılım ve donanımlar hakkında bilgiler verilmiştir. Bölüm 5’de önerilen yöntemin uygulanması anlatılmış ve elde edilen bulgular verilmiştir. Bölüm 6’da ise çalışmanın sonucu, sınırlamaları ve gelecek çalışma önerilerinden bahsedilmiştir.

1.1. Tezin Kapsamı

Bu çalışma kapsamında Endüstriyel Kontrol Sistemleri’nde en çok kullanılan protokol olan Modbus/TCP protokolüne yönelik siber saldırılara çözüm üretilmesi amaçlanmıştır. Bu sistemler sürekli olarak hizmet verdiklerinden dolayı, işleyen sistem üzerinde akademik veya deneysel çalışmalar yapmak elverişli değildir. Çünkü bu sistemler, bilgi güvenliği unsurlarından erişilebilirlik ilkesi göz önünde bulundurularak çalışmaktadır. Dolayısıyla olası bir hata durumunda sistemin işleyişinin kesintiye uğraması kabul edilemez bir durum olarak değerlendirilmektedir. Çünkü sistemin kesintiye uğraması durumunda büyük ölçüde maddi ve manevi zararlar ortaya çıkabilir. Bu sebeple deneysel çalışmalar gerçek bir sistemden

bağımsız olarak test yatağı merkezinde yürütülmesi gerekir. Bu çalışmada benzetim programları kullanılarak Modbus iletişim ortamı oluşturulmuştur.

Bu çalışma, Modbus/TCP protokolü paketlerinin şifrenmesi ve şifresinin çözülmesi uygulamalarını içermektedir. Şifreleme işlemleri için hafif sıklet şifreleme algoritmaları tercih edilmiştir. Çalışmada, bu işlemleri gerçekleştirmek için Python programlama dili ve kütüphaneleri kullanılmıştır. Bu kapsamda, çalışma Orange Pi R1 LTS Plus ve Raspberry Pi 4 cihazlarında kurulu Ubuntu işletim sistemi üzerinde gerçekleştirilmiştir. Çalışmanın kapsamı sekiz farklı şifreleme algoritmasının şifreleme ve şifre çözme işlem sürelerinin incelenmesiyle sınırlandırılmıştır. Belirlenen bu kapsam Modbus/TCP protokolünün endüstriyel kontrol sistemlerinde güvenliğini artırılmış bir şekilde kullanımına yönelik bir temel oluşturmayı hedeflemektedir. Sonuç olarak, bu çalışma Modbus/TCP protokolüne ait paketlerin Python ortamında şifrenmesi ve şifresinin çözülmesi sürecini ve güvenlik açıklarına karşı alınan tedbirleri kapsamlı bir şekilde ele almaktadır.

1.2. Problem Tanımı ve Tezin Amacı

Bu çalışma, günümüz kritik alt yapı sistemlerinin en belirgin yapıtaşı olan endüstriyel kontrol sistemlerinin güvenliğini temel almıştır. Bu sistemlerde kullanılan haberleşme protokolleri genellikle güvenlik parametreleri düşünülmeden tasarlandığından protokol içerisindeki veriler açık metin (plaintext) olarak iletilmektedir. Bu sistemler günümüzde internete bağlı hale gelmiş olup uzaktan yönetilebilir durumdadır. Bu durum, avantaj gibi gözükse de sistemler dünya üzerindeki herhangi bir noktadan gelebilecek siber saldırıya açık konuma gelmiş olup ciddi bir güvenlik riski oluşturmaktadır. Dolayısıyla bu sistemlerle yapılacak olan iletişimin güvenilir olması büyük önem arz etmektedir.

Bu çalışmada, üzerinde araştırma yapılacak endüstriyel protokol olarak Modbus/TCP protokolü tercih edilmiştir. Modbus/TCP protokolünün güvenliğini sağlamak amacıyla bir şifreleme modülü önerilmiştir. Şifreleme işlem süresinin minimum olması için hafif sıklet şifreleme algoritmaları kullanılmıştır. Önerilen modül aracılığıyla Modbus/TCP iletişiminin şifrenmesi ve Modbus/TCP paketlerinin içeriğinin gizlenmesi amaçlanmıştır. Şifrelenen bu Modbus/TCP paketleri ele geçirilse bile şifreli olduğundan bir anlam ifade etmeyecektir. Paket üzerinde değişiklik yapılırsa tespit edilip geçersiz paket olarak değerlendirilecektir. Bilgi güvenliği

unsurlarından, OT için geri planda olan gizlilik ve bütünlük ilkesi de üst seviyeye taşınmış olacaktır.

Fidye yazılımı, hizmet reddi, sosyal mühendislik, yetkisiz erişim gibi saldırılar bu sistemlere karşı sıkça yapılan saldırı türleridir. Bu saldırılara bakıldığında temel sebeplerin; güvenlik parametresinin olmadığı ve eski donanımların kullanılması olduğu görülmektedir. Kritik alt yapıların önemi göz önüne alındığında günlük hayatı doğrudan etkileyen bu sistemlerin güvenliği en üst seviyede olmalıdır. Önerilen modül Modbus/TCP kullanan donanımlarla uyumlu olup, yaşanmış olan bu saldırıların benzerlerinin yaşanmaması adına büyük önem arz etmektedir. Böylelikle EKS sistemlerine olan tehditlerin önüne geçilmesi ve büyük bir arızanın veya felaketin engellenmesi hedeflenmektedir.

1.3. Literatür Araştırması

Bu bölümde, EKS’lerde kullanılan iletişim protokollerinin güvenliği, Modbus/TCP protokolüne yönelik çeşitli çalışmalar ve endüstriyel sistemlerin güvenlik uygulamalarını içeren literatür çalışmaları incelenmiştir. Günümüzde saldırıların hedefi haline gelen endüstriyel sistemler, protokollerin eskimiş yapısı, veri bütünlüğü, veri gizliliği açısından riskler taşımaktadır. Literatürdeki çalışmalar, kriptografik şifreleme algoritmalarının performansı, endüstriyel test ortamlarının oluşturulması ve değerlendirmesi, makine öğrenmesi ve derin öğrenme içeren yöntemler gibi alanlara odaklanmıştır. Önerilmiş çalışmaların değerlendirmelerinin incelenmesi, bu çalışmada önerilen yaklaşımın literatürdeki yerinin belirlenmesine yardımcı olmaktadır.

Xianghao ve arkadaşları, Modbus/TCP protokolünün güvenliğini sağlamak amacıyla kaynak kısıtlı cihazlar için AES ve ChaCha20 algoritmaları tabanlı çözümler önermişlerdir [2]. Önerdikleri çözümler Raspberry Pi üzerinde ve proxy sunucusu üzerinde çalıştırılmıştır. Yapılan değerlendirmeler sonucunda Raspberry Pi tabanlı çözüm proxy tabanlı çözüme kıyasla daha üstün performans sağlamıştır.

Eslam, Modbus protokolünü kullanan Endüstriyel Kontrol Sistemleri’ne yönelik saldırıları tespit etmek için bir çalışma yapmıştır [3]. Bu amaçla Modbus-Markov Davranış Ayırıcı (MMBD) yöntemi önermiş ve SVM, DT, RF, LR makine öğrenimi algoritmalarının derin öğrenme modellerinden daha iyi performans gösterdiğini elde etmiştir.

Divine ve arkadaşları, EKS güvenliğini doğrulamak için MITRE ATT&CK çerçevesini incelemişlerdir [4]. Önerdikleri çalışmada DoS ve ARP zehirlleme saldırıları gerçekleştirilmiş ve bu saldırılar açık kaynaklı yazılımlarla tespit edilmiştir.

Alaa ve Kumar, Modbus/TCP protokolünün ağ trafiği yapısına dayalı güvenlik açıklarını tespit eden ANDVI adlı bir yöntem önermişlerdir [5]. Önerdikleri bu yöntem Modbus trafiğinden parmak izi oluşturup çevrimiçi veritabanlarından sorgulama yapmaktadır.

Kurmi ve arkadaşları, Modbus protokolündeki güvenlik açıklarını çözmek için rol tabanlı erişim kontrol mimarisinin (RBAC) yeni bir versiyonunu önermişlerdir [6]. Önerdikleri çalışmada TLS protokolü, Modbus protokolüne entegre edilmiştir.

Tirthankar ve arkadaşları, endüstriyel kontrol sistemleri trafiğindeki anormallikleri tespit etmek için entropi ve sınıflandırmaya dayalı analiz kullanan istatistiksel bir yaklaşım sunmuşlardır [7]. Önerdikleri çalışmada J48 ve RF algoritmaları en iyi sonuçları vermiştir.

Ricardo ve arkadaşları, endüstriyel bir ortamda sızma testi gerçekleştirmek için MOSTO araç seti tarafından desteklenen pratik bir yöntem önermişlerdir [8]. MOSTO ile yapılan güvenlik değerlendirmesinde potansiyel güvenlik açıkları tespit edilmiştir.

Tripathy ve arkadaşları, Modbus ve Ethernet/IP protokollerinin güvenlik analizi için test yatağı geliştirmiş ve yüksek doğruluk oranına sahip bir saldırı tespit algoritması önermişlerdir [9]. Önerdikleri algoritma DoS ve FDI saldırılarını başarıyla tespit etmiştir.

Shampa ve arkadaşları, Modbus protokolündeki MiTM saldırısı gibi ağ açıklarını ele almak ve bunlara karşı tedbirleri uygulamak için bir siber güvenlik stratejisi önermişlerdir [10]. Yazarlar, iletişim ağını derinlemesine analiz etmek için test ortamında, sanal makineler ve ModbusPal yazılımı kullanmışlardır.

Nishiuchi ve arkadaşları, endüstriyel kontrol sistemlerinde kullanılan Modbus TCP iletişimde DoS ve yanıltma saldırılarının olasılığı ve göreceli entropisine dayalı saldırı tespit denemeleri yapmışlardır [11]. Önerdikleri çalışmada; işlem süresi, paket boyutu ve paketler arası gecikme kullanılarak saldırıların tespiti yapılabilmektedir.

Ochiai ve arkadaşları, Modbus/RS-485 protokolüne yönelik saldırı tespiti için bir yöntem önermiş ve denetimli öğrenme ile 0,9999 doğruluk, denetimsiz öğrenme ile

0,9992-0,9999 arası anormallik tespiti elde etmişlerdir [12]. Bu sonuçlar önerdikleri yöntemin, fieldbus saldırıları için yüksek bir tespit oranı elde edildiğini göstermektedir.

Wu ve arkadaşları; Edge Computing ve OT sistemlerini birleştiren bir OTEC güvenlik çerçevesi önermiş, %94 saldırı tespit oranı elde etmişlerdir [13]. Önerdikleri güvenlik çerçevesi ile IoT ortamları için kullanışlı olduğunu göstermişlerdir.

Liu ve arkadaşları, Modbus/TCP protokolünün güvenliğini sağlamak için SM3, SM4 yerel şifreleme algoritmaları kullanarak bir güvenlik şeması tasarlamışlardır [14]. Önerdikleri güvenlik şemasının minimum iş yükü ile Modbus/TCP protokolünün güvenlik problemini çözdüklerini göstermişlerdir.

Katulić ve arkadaşları, Modbus/TCP protokolü için MAC tabanlı bir kimlik doğrulama yöntemi önermiş, IACS'lerde performansı büyük ölçüde etkilemeden ağ saldırılarına karşı koruma sağladığını göstermişlerdir [15].

Zhang ve arkadaşları; ön işleme, yeniden kodlama ve sınıflandırma ile üç aşamadan oluşan MODLSTM adında bir sinir ağı mimarisi önermişlerdir [16]. Önerdikleri yöntem, diğer çalışmalarla yaptıkları kıyasalamada Modbus ile ilgili DoS saldırılarını tespit etmekte üstün performans elde etmişlerdir.

Alsabbagh ve arkadaşları, Modbus protokolünün güvenlik açıklarından faydalanarak SCADA operatöründen komut enjeksiyonunu gizleyen saldırı senaryosu gerçekleştirmiş ve böyle bir güvenlik tehdidini önlemek için bazı çözümler önermişlerdir [17].

Mohammed ve arkadaşları, SCADA sistemlerine yönelik yapılan DoS saldırısını test ortamında gerçekleştirmek için Field Flooding saldırı yöntemini kullanmış ve bu saldırıyı tespit etmek için XGBoost algoritmasına sahip denetimli makine öğrenimi kullanan bir mekanizma önermişlerdir [18].

Parul ve Faruk, Atık Su Arıtma Tesisi (WWTP) test yatağı kullanarak, EKS sistemi üzerindeki bileşenlerin güvenlik açıklarını ve bunları hedefleyen saldırı yollarını araştırmış, sızma testi için kullanılabilecek öneriler sunmuş ve ürün geliştiriciler için bazı yönergeler öne sürmüşlerdir [19].

Tiago ve Sergio, yeni Modbus/TCP Güvenlik protokolünü, yetkisiz erişime karşı güvenliği garanti eden kimlik doğrulama ve yetkilendirme işlevleriyle geliştirmeyi

önermiş ve yapılan testlerde önerilen yöntemin potansiyellerini ve uygulanabilirliklerini kanıtlamışlardır [20].

Esteban ve arkadaşları, kritik altyapılardaki siber saldırılara karşı güvenlik önlemlerini değerlendirmek için bir test ortamı sunmuş ve MITM saldırılarını tespit etmek için DER sistemleri üzerindeki SunSpec protokolüne dayalı bir kullanım senaryosunda doğrulamışlardır [21].

Luo ve Li çalışmalarında simetrik anahtar ve digital imza teknolojisi kullanan Modbus-E adında bir protokol önermiş ve Modbus/TCP iletişimini kapsamlı bir şekilde iyileştirmişlerdir [22].

Ametov ve arkadaşları, oturum bazlı olmayan Modbus/TCP protokolü için kimlik doğrulama mekanizması önermiş ve siber tehditlere karşı dayanıklılığı artırdığını öne sürmüşlerdir [23].

Chang ve arkadaşları, bir EKS'de Modbus protokolünü kullanan bir sistemin güvenliğini analiz etmiş ve hizmet reddi, komut enjeksiyonu, yanıt enjeksiyonu gibi farklı saldırı türlerine karşı gizlilik, bütünlük ve erişebilirlik parametreleri bazında güvenlik test sistemi kurmuşlardır [24].

Panagiotis ve arkadaşları, Smod sızma testi aracı tarafından sağlanan siber saldırıları incelemiş ve Modbus TCP ile ilgili siber saldırıları %81 doğrulukla tespit edebilen anormallik tabanlı bir saldırı tespit sistemi sunmuşlardır [25].

Luswata ve arkadaşları, Modbus TCP protokolünü kullanan sistemlerde güvenlik açığı çalışması yapmış, SCADA sistemlerine özgü mevcut güvenlik önemlerini inceleyip test etmiş ve iyileştirmeler önermişlerdir [26].

Cristina ve Sherali yaptıkları çalışmada kritik altyapıların güvenlik açıklarını ve tehditlerini araştırmış, endüstriyel kontrol sistemlerine odaklanarak incelemekte ve koruma tedbirlerini açıklamışlardır [27].

Farsi ve arkadaşları, yaptıkları çalışmada endüstriyel kontrol ağlarını korumak için düşük hesap süresine sahip anormallik tespiti yapan bir yöntem önermiş ve deneysel sonuçlar önerdikleri yöntemin düşük yanlış-pozitif sonucunu elde etmiştir [28].

Ghaleb ve arkadaşları yaptıkları çalışmada bir test yatağında PLC cihazların kurulumu, yapılandırması ve iletişimi üzerine tekrarlama, ortadaki adam ve komut modifikasyon olmak üzere farklı saldırı türlerini gerçekleştirmiş ve çözümler önermişlerdir [29].

Mcginthy ve Michaels yaptıkları çalışmada Endüstriyel Nesnelerin İnterneti (IIoT) ve kritik altyapı entegrasyonunu açıklamış; bellek koruması, şifreleme ve mesaj doğrulama gücü, fiziksel katman güvenliği ve güvenlik parametrelerini içeren seviye temelli bir güvenlik aday mimarisi oluşturmuşlardır [30].

Erdal ve İsmail yaptıkları çalışmada Modbus TCP protokolünün kimlik doğrulama eksikliği istismar edilmiş, bu güvenlik sorununa yönelik çözüm olarak Python dilinde bir yöntem sunmuşlar ve yapılan testlerde saldırıların engellenebildiğini göstermişlerdir [31].

Lai ve arkadaşları, Modbus TCP protokolü güvenlik açıklarını tespit etmek için zafiyet madenciliği üzerine RNN kullanarak çalışma yapmış ve başarılı zafiyetler ile üç farklı Modbus/TCP güvenlik açığını elde etmişlerdir [32].

Chen ve arkadaşları akıllı güç şebekelerinde Modbus/TCP protokolüne yönelik saldırılar gerçekleştirilmiş, hizmet reddi ve ortadaki adam saldırılarının tespiti ve engelleme için test ortamında iyileştirmeler önermişlerdir [33].

Nyasore ve arkadaşları; Modbus/TCP tabanlı ağ mimarilerine yönelik yaygın saldırılar için IDPS imzaları geliştirmiş, saldırıları tespit etmek ve önlemek için kullanılan IDS yazılımlarını test etmiş ve bu yazılımların saldırıları engellemede yetersiz olduğunu vurgulamışlardır [34].

Yi ve arkadaşları, Modbus TCP protokolünün güvenlik sorununu ele almış, güvenliğin artırılması için HMAC ve SM4 algoritmalarını kullanarak IP adreslerine ve fonksiyon kodlarına dayalı bir güvenlik çözüm mekanizması önermişlerdir [35].

Lupton ve arkadaşları; akıllı şehirlerde kullanılan IoT cihazlarının güvenlik açıklarını ele almış, PLC ve SCADA sistemlerinin bu açıkları içerdiği ve bu açıkların saldırı tespit sistemi ve Port Aynalama ile çözülebileceğini göstermişlerdir [36].

Mahendra ve arkadaşları; Modbus-TCP iletişim tabanlı endüstriyel kontrol sistemine uyarlanmış, bu ağlara yapılan saldırıları algılayan bir derin güvenlik tarayıcısı (DSS) önermiş ve yapılan testlerde saldırıların pasif bir şekilde tespit ettiğini göstermişlerdir [37].

Chan ve arkadaşları; akıllı şebeke kavramı, geleneksel şebeke, DER'in (Dağıtılmış Enerji Kaynakları) arka planını, bazı DER mimarilerini, DERMS'in uluslararası

uygulamalarını, DER iletişim protokollerini, DER'e yönelik olası saldırıları ve bazı azaltma stratejilerini incelemişlerdir [38].

Tarek ve arkadaşları; veri akışı, OTP ve Yensi mekanizması kullanan Modbus tabanlı yeni güvenli MUBus protokolünü önermiş ve gizlilik ve bütünlük sağlayarak önerdikleri protokolün etkinliğini göstermişlerdir [39].

Holasova ve arkadaşları; Modbus TCP protokol paketlerini ayrıştırıp bir Modbus Güvenlik Modülü oluşturmuş, bu paketlerden elde edilen bilgileri yapay sinir ağı kullanarak saldırıları %99,52 doğruluk oranıyla tespit etmişlerdir [40].

Özçelik ve arkadaşları tarafından; test, eğitim, güvenlik araştırması ve saldırı/savunma analizi için kritik altyapılar ulusal test yatağı merkezi CENTER adıyla Sakarya Üniversitesi'nde DNP3, S7, EtherCAT, Modbus gibi çoklu EKS protokollerini destekleyen bir test yatağı merkezi kurulmuştur [41].

Ren ve arkadaşları, yeni bir endüstriyel iletişim protokolü olan SRMCP'yi önermiş olup bu protokol, mevcut ve gelecekteki kriptografik tehditlere karşı koruma sağlamak için post-kuantum kriptografi teknikleri ve Port Hopping mekanizması kullanmaktadır [42]. Önerdikleri bu yöntem Modbus ve OPC UA protokollerle karşılaştırmalı olarak analiz edilmiştir.

Lazaridis ve arkadaşları, Modbus TCP protokolünü kullanan siber fiziksel sistemlerdeki güvenlik açıklarının kritik altyapılar üzerindeki potansiyel etkilerini incelemiş ve bu tehditlere yönelik etkili önlemler geliştirilmesi için tahminler sunmaktadır [43].

Almendli ve Mohasefi, bir siber fiziksel sistem uygulaması olan SWaT tesisine odaklanmış, sensör ve ağ verilerini analiz edip bir saldırı tespit sistemi önermişlerdir [44]. Önerdikleri yöntem sensör verisinde %99.9, Modbus verisinde %95 doğruluk oranı elde etmiştir.

Altschaffel ve arkadaşları, endüstriyel sistemlerdeki gizli iletişim saldırılarının tespiti için istatistiksel yöntemlere dayalı bir yaklaşım önermiş ve Modbus/TCP protokolü üzerinde yapılan değerlendirme ile bir temel oluşturmayı hedeflemişlerdir [45].

Zhang ve arkadaşları, yaptıkları çalışmada Modbus/TCP protokolünün güvenliğini artırmak için SM3 ve ZUC kriptografik algoritmalarını kullanan ve kimlik doğrulama,

veri gizliliği, bütünlük doğrulaması, tekraralama saldırısı koruma özelliklerini içeren bir çözüm önermişlerdir [46].

Le ve arkadaşları, Ascon şifreleme algoritmasını kullanarak Modbus/TCP güvenlik açıklarını gidermeye yönelik bir yöntem önermiş ve önerdikleri yöntem minimum düzeyde hesaplama yükü getirerek sistemin dayanıklılığını artırmaya yönelik bir yol göstermektedir [47].

Rashid ve arkadaşları, yaptıkları çalışmada saldırganların kolaylıkla Modbus/TCP protokolüne yönelik hizmet reddi saldırısı gerçekleştirebileceğini göstermiş ve protokolün güvenlik mekanizmaların iyileştirilmesine yönelik kritik gereksinime dikkat çekmeye çalışmışlardır [48].

Tawfik ve Badawy, yaklaşık 610 bin Modbus paketinden oluşan bir veri seti üzerinde WGAN modeli eğitmiş ve eğitim sonucunda elde edilen paketler %93.92 TSRR oranına ulaşarak uygulamanın geçerli olduğunu göstermişlerdir [49].

Trungadi ve arkadaşları, eski endüstriyel cihazlar için kriptografik işlemleri yöneten ve güvenli iletişimi sağlamak için proxy kullanan bir güvenlik çerçevesi sunmuş ve kabul edilebilir bir operasyonel etki ile sistemi güvence altına aldıklarını belirtmektedirler [50].

Dao ve arkadaşları, Modbus/TCP tabanlı endüstriyel ağlardaki ARP zehirlleme saldırılarını tespit etmek bir cihaz önermiş, test ortamında elde edilen sonuçlar önerdikleri yöntemin saldırıları etkili bir şekilde tespit edebildiğini göstermişlerdir [51].

Yapılan incelemeler sonucunda, literatürde yer alan ve Modbus/TCP protokolünün güvenliğine yönelik olan çalışmaların; güvenlik açıklarının tespit edilmesi, saldırı tespitlerinin doğruluğunun vurgulanması, kriptografik algoritmaların uygulanması konularına odaklandığı görülmektedir. Bu çalışmada literatürdeki çalışmalardan esinlenerek, hafif sıklet şifreleme algoritmaları kullanarak Modbus/TCP protokolünün güvenliğini saha cihazından bağımsız olarak sağlamak ve Python ortamında performans karşılaştırması yapılmıştır. Böylelikle önerilen çalışma, literatürdeki mevcut çalışmalara ilave olarak donanım tabanlı bir uygulama sunmakta olup EKS güvenliğinde farklı bir bakış açısı kazandırmaktadır.

1.4. Hipotez

Bu çalışmanın odak noktası, Modbus/TCP protokolünün güvenliğini artırmak için endüstriyel cihazdan bağımsız olarak bir donanım üzerinde hafif sıklet şifreleme algoritmaları kullanılarak endüstriyel iletişimin güvenli bir şekilde sağlanmasıdır. Bu hipotezin uygulanabilirliği göstermek için Raspberry Pi 4 ve Orange Pi R1 LTS Plus cihazları üzerinde Python programlama dili kullanılarak hafif sıklet şifreleme algoritmaları ile Modbus/TCP paketleri şifrelenmiş olup elde edilen şifreleme ve şifre çözme süreleri değerlendirilmiştir.

Endüstriyel kontrol sistemlerinde güvenliği sağlamak için son yıllarda birçok farklı açıdan çözümler önerilmiştir. Bu çalışmalar incelediğinde; ilk olarak protokolün yapısında değişiklik yapılarak çözümler önerilmiştir. Bu çözümlere örnek olarak; Luo ve Li'nin önerdikleri Modbus-E protokolü [22], Ametov ve arkadaşlarının kimlik doğrulama içeren protokolü [23], Schneider Electric'in geliştirdiği Modbus Secure protokolü , Yi ve Zhang tarafından önerilen güvenliği artırılmış Modbus protokolü [35], Cherifi ve arkadaşlarının önerdiği MUBus protokolü [39], Martins ve Oliveira tarafından geliştirilen Modbus/TCP Güvenlik protokolü [20] verilebilir. Bu çözümlerin gerçekleştirilmesi için EKS ekosisteminde bulunan ve Modbus/TCP protokolünün kullanan tüm cihazların yazılımının güncellenmesi veya cihazların yeni sürümleri ile değiştirilmesi gerekmektedir. Tüm cihazların değiştirilmesi ise bir maliyet gerektirir ve bu durum ölçeklenebilir değildir. Bu cihazlardan bazıları kaynak kısıtlı cihazlar olabildikleri için yazılım güncellemesi yapılarak önerilen çözümlerden biri gerçekleştirilmek istenirse bile şifreleme esnasında gecikme yaşanabilir. Burada şifreleme algoritmasının karmaşıklığı da göz önünde bulundurularak işlem süresinin en aza indirilmesi problemi ortaya çıkmaktadır. Bu durumda önerilmiş çözümlerde kullanılan geleneksel şifreleme algoritmaları yerine kaynak kısıtlı cihazlar için tasarlanmış hafif sıklet şifreleme algoritmaları tercih edilmiştir. Bu açıdan önerilmiş çözümlerde iki farklı problem ortaya çıkmaktadır; ölçeklenebilirlik ve işlem yükü. Bu çalışmada önerilen çözüm saha cihazlarından bağımsız olarak başka bir modül üzerinde gerçekleştirileceği için cihazlar üzerinde herhangi bir yazılım güncellemesi olmayacaktır ve cihazlar üzerinde ekstra bir işlem yükü getirmeyecektir. SCADA iletişimi için kullanılacak bu modül sayesinde dışardan gelen veya giden tüm paketler şifreli olarak iletilecektir.

Diğer güvenlik bakış açısı ise protokolün yapısında bulunan güvenlik açıklarını tespit edip bunlara yönelik çözümler üretmektir. Bu çalışmalara örnek olarak; Luswata ve arkadaşlarının önerdiği çözüm [26], Chang ve arkadaşlarının önerdiği yöntem [24], Panagiotis ve arkadaşlarının önerdiği çözüm [25], Alsabbagh ve arkadaşlarının izledikleri senaryo [17], Mlot ve arkadaşlarının uyguladıkları saldırılar ve çözümler [21] verilebilir. Bu çalışmalarda, tespit edilebilen veya daha önce bilinen güvenlik açıklarına yönelik çözüm önerileri mevcuttur. Dolayısıyla ortaya çıkabilecek herhangi bir yeni güvenlik açığında başarısız olabilir. Bu durum sistemin işleyişini bozabilir ve büyük maliyetlerle sonuçlanabilir. Burada yeni bir problem ortaya çıkıyor; geçerlilik. Önerilmiş çözüm tedbirlerini kapsamayan bir güvenlik açığı keşfedildiğinde bu yöntemler sistemin ele geçirilmesini engellemede yetersiz kalabilir. Bu çalışmada önerilen çözüm sayesinde yeni bir güvenlik açığı keşfedilse bile iletişim şifreli olacağından dolayı paketler geçersiz olarak görüleceği için işleme alınmayacaktır. Dolayısıyla şifreleme işlemi sayesinde yeni bir güvenlik açığına karşı korunma sağlanabilecektir.

Bir diğer güvenlik bakış açısı ise makine öğrenmesi veya derin öğrenme algoritmalarını içeren yöntemlerin kullanılmasıdır. Bu çalışmalara örnek olarak; Eslam'ın önerdiği Modbus-Markov Davranış Ayırıcı [3], Tirthankar ve arkadaşlarının önerdikleri anormallik tespiti yöntemi [7], Ochiai ve arkadaşlarının önerdikleri denetimli ve denetimsiz öğrenme tabanlı yöntem [12], Zhang ve arkadaşlarının önerdiği MODLSTM yöntemi [16], Holasova ve arkadaşlarının önerdikleri çalışma [40] ile Tawfik ve Badawy'nin önerdiği WGAN modeli [49] verilebilir. Bu çözümlerin uygulanabilmesi için önceden belirlenmiş saldırı paketleri ve normal paketlerin olduğu veri seti gerekmektedir. Veri setinde olmayan bir saldırı türü, tespit sistemleri tarafından tanımlanamayabilir. Bu durum, belirtilen çözümleri yeni saldırı yöntemlerine karşı savunmasız bırakabilir. Bu çalışmada önerilen yöntem için bir veri setine ihtiyaç yoktur. Tak-çalıştır mantığıyla önerilen çözüm kolaylıkla uygulanabilir.

Bu çalışmada önerilen güvenlik modülü gerçek zamanlı olarak bahsedilen problemlere yanıt verecek olup böylelikle diğer güvenlik bakış açılarına göre farklı olup ön plana çıktığı görülmektedir. Önerilen modülün saha cihazlarından bağımsız olması, şirketleri önceki yöntemlerin getirdiği tüm cihazların yenilenmesi maliyetinden kurtarmaktadır. Diğer önerilmiş yöntemlerin yetersiz kaldığı durumlarda ise şifreleme sayesinde yeni oluşabilecek tehditlere karşı koruma sağlanacaktır. Böylelikle modül

geçerliliğini koruyacak şekilde gerekirse güncelleme yapılarak ilgili sistemlerin güvenliğini en üst seviyede tutmak için gerçek zamanlı olarak çalışmaya devam edebilecektir.





2. ENDÜSTRİYEL KONTROL SİSTEMLERİ

Günümüzde endüstri dünyasında üretim ve dağıtım süreçlerini sürdürebilir, güvenli ve verimli bir şekilde yönetmek için otomasyon sistemlerinin kullanımı gerekmektedir. Bu sistemler güncel teknolojilerle uyumlu olmalıdır. Endüstriyel Kontrol Sistemleri; üretim ve dağıtım hatlarının kontrol edilmesi, izlenmesi ve sistemleri en uygun hale getirmek için kullanılan teknolojilerin bütünüdür. Endüstriyel Kontrol Sistemleri temel bileşenleri olan SCADA, DKS, PLC gibi alt sistemler endüstriyel tesislerin bilgi teknolojileriyle olan bütünleşmesinde önemli rol oynamaktadır.

SCADA sistemleri, farklı coğrafi alanlardaki tesislerin bir merkezden izlenmesi ve yönetimini imkan sunarken; DKS sistemleri ise tek bir alanda bulunan ve üretim yapan tesislerde dağıtık yapıya sahiptir. PLC'ler ise programlanabilir yapıya sahip olmakla birlikte lojik seviyede kontrol görevlerini yerine getirmektedir. Bu sistemlerin kendi aralarında ve diğer endüstriyel cihazlarla iletişim kurabilmesi için endüstriyel protokoller kullanılır.

Eski zamanlarda internete bağlı olmayan Endüstriyel Kontrol Sistemleri, artık günümüzde internete bağlı hale gelmiştir. Bu durum, kötücül yazılımlara, yetkisiz erişim, verilerin değiştirilmesi gibi güvenlik risklerini beraberinde getirmiştir. Bu nedenle EKS güvenliği yalnızca fiziksel güvenliği değil, siber güvenliği de kapsamaktadır.

Endüstriyel Kontrol Sistemleri, sadece üretim odaklı değil, enerji yönetimi, uzaktan izleme, arıza tespiti, bakım gibi kuruluşlara ek özellikler sunmaktadır. Bu bölümde; EKS'nin temel bileşenleri olan SCADA, DKS, PLC sistemleri ele alınmış ve kullanılan endüstriyel protokoller ile bu sistemlerin güvenliği anlatılmıştır.

2.1. Endüstriyel Kontrol Sistemi

Endüstriyel Kontrol Sistemi (EKS), sanayi tesislerinde ve kritik altyapılarda endüstriyel süreçleri kontrol etmek, izlemek ve otomatikleştirmek amacıyla kullanılan yazılım ve donanım entegrasyonunu ifade etmektedir. Bu sistemler; üretim, enerji, gaz, petrol, su yönetimi, elektrik şebekesi, telekomünikasyon, otomotiv, ilaç gibi

sektörlerin işleyişini sağlayan temel bileşenlerdir. Bu sistemler; güvenliği sağlamak, verimliliği artırmak ve insan hatasını en aza indirmek amaçlarıyla kullanılmaktadır. Her bir EKS, kendisine verilen görevleri otomatik olarak yürütmek ve yönetmek için oluşturulmuştur. EKS'nin işlevi; konfigürasyonuna, amacına ve endüstriyel uygulama alanına bağlı olarak değişebilmektedir.

Endüstriyel Kontrol Sistemi (EKS), genel olarak kontrol döngülerini, insan-makine arayüzlerini (HMI) ve uzaktan yönetim araçlarını içermektedir. EKS bileşenleri birbirleri arasında ağ üzerinden iletişim kurmak istediğinde karşımıza farklı protokoller çıkmaktadır. Endüstriyel Kontrol Sistemleri; Dağıtılmış Kontrol Sistemleri (DKS), SCADA ve Programlanabilir Mantık Denetleyicileri (PLC) gibi farklı alt sistemleri içerisinde bulunduran bir yapıya sahiptir. EKS, tanımlanmış bir üretim sürecinde çıktı temin etmeye odaklanırken; bu çıktının nasıl olacağı sistemin kontrol bölümünde tanımlanır.

Endüstriyel kontrol sistemlerinde kontrol işlemi otomatik şekilde gerçekleştirilebilir veya insan müdahalesiyle birlikte de çalışabilir. Kontrol işleminin üç temel çalışma moduna vardır; manuel, açık döngü ve kapalı döngü. Manuel modda kontrol işlemi, operatörler tarafından (insan eliyle) doğrudan yapılır. Açık döngü kontrol sistemlerinde ise sistem, önceden tanımlanmış parametrelere göre işlem yapar ve bu işlem sonucunda elde edilen çıktılar, girişi etkileyecek şekilde tekrar kullanılmak üzere parametre hesaplamasına dahil edilmez. Kapalı döngü sistemlerde ise işlem sonucunda elde edilen çıktılar kullanılarak sistemin giriş parametreleri hesaplamasına dahil edilerek hedeflenen çıktıya erişilmesi temin edilir. Kontrolör bileşeni, sistemin çalışma düzenine uygun olup olmadığını denetleme görevini yerine getirmektedir. Endüstriyel Kontrol Sistemleri, ulaşım, üretim, enerji, petrol ve dağıtım olmak üzere birçok farklı endüstriyel sektörde ve kritik altyapılarda kullanılmaktadır.

Endüstriyel kontrol sisteminin çalışması için parametre ayarlamalarının yapılması gerekmektedir. Bu parametreler sistemin girişleri olarak tanımlanmaktadır. Sistemin çalıştıktan sonra elde edilen çıktılar ise değişkenler olarak tanımlanmıştır. Bir EKS'nin tanımları Tablo 2.1'de verilmiştir.

Tablo 2.1. EKS Tanımları.

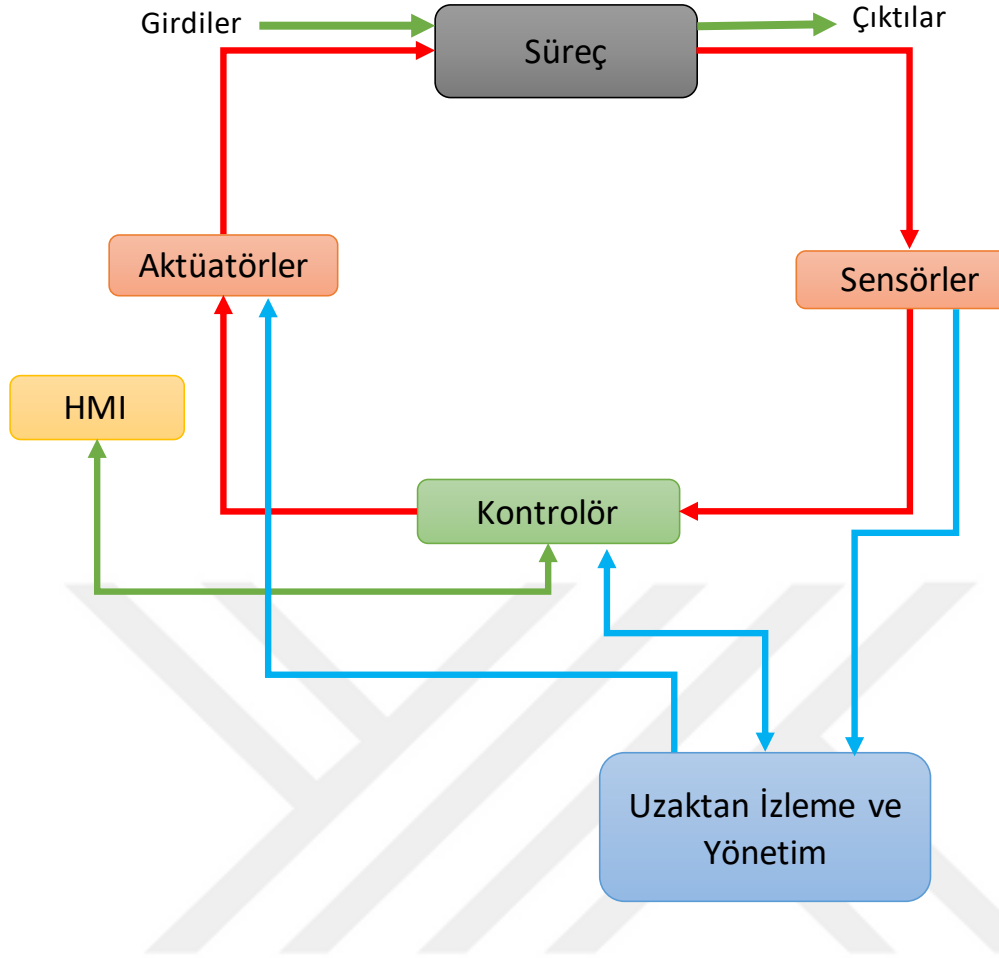
Tanım	Açıklama
Değişkenler	Sistemin Çıktıları
Parametreler	Sistemin Girdileri

Sistemin girdileri ve çıktıları, sürekli ve ayrık olmak üzere iki farklı veri tipine sahip olabilir. Sürekli değişken veya parametreler, analog olarak da adlandırılmakta olup proses boyunca kesintisiz değerlere sahiptir. Sürekli değişken veya parametrelere örnek olarak akış hızı, kuvvet, hız, sıcaklık vb. verilebilir. Ayrık değişken veya parametreler ise bir aralıkta yalnızca belirli bir değere sahiptir. En sık kullanılan ayrık veri tipleri şunlardır; ikili, ayrık. İkili veri tipi; açık/kapalı, aktif/pasif, mevcut/mevcut değil gibi değerleri alabilir. Ayrık veri tipi ise ikiden fazla değer alabilir ancak alabileceği değerler sınırlıdır. Örnek olarak haftalık üretilen araç sayısı verilebilir.

Genellik bir EKS, kontrol döngüsü, insan makine arayüzü (HMI) ve uzaktan izleme ve yönetim birimlerinden oluşmaktadır. sensör ve aktüatörlerden oluşmaktadır. Uzaktan izleme ve yönetim işlemi, endüstriyel ağ protokolleri aracılığıyla yapılmaktadır. Operatörler, HMI cihazı ile kontrolör aracılığıyla kontrol döngüsüne müdahale edebilir. Kontrol döngüsü içerisinde kontrolör, sensörler ve aktüatörler yer almaktadır. Sensörler ölçüm yapar ve yaptığı bu ölçümleri kontrolöre gönderir. Kontrolör aldığı bu ölçüm sonuçları ile değişkenleri, belirlenen hedeflere göre manipüle eder ve bu yeni değişkenleri aktüatörlere gönderir. Aktüatörler ise aldıkları değişkenlere göre direkt olarak fiziksel işlemleri (vana açma/kapama, motor açma/kapama vb.) gerçekleştirir.

Operatörler, belirlenen hedeflere göre giriş parametlerini HMI aracılığıyla sisteme tanımlayabilmektedir. Uzaktan izleme ve yönetim birimi ile bu parametreler, sensör verileri ve aktüatör verileri izlenebilmektedir. HMI aracılığıyla işlem yapabilmek için fiziksel olarak sistemin yanında bulunmak gerekmektedir. Ancak uzaktan izleme ve yönetim araçları ile internet aracılığıyla sistemin yanında olmadan herhangi bir lokasyondan ilgili işlemler gerçekleştirilebilir.

Basit bir EKS çalışma diyagramı Şekil 2.1’de verilmiştir.



Şekil 2.1. Basit bir EKS Çalışma Diyagramı.

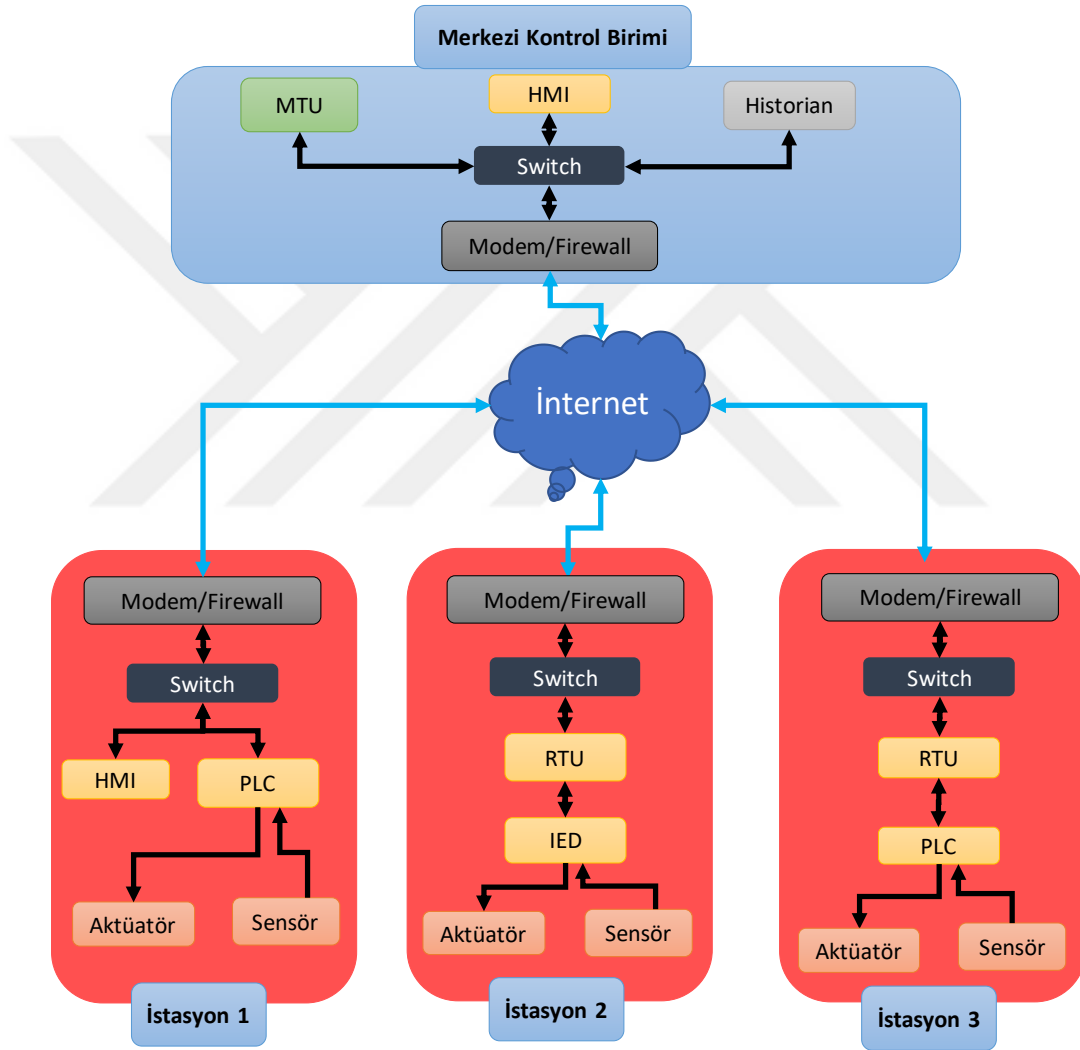
Günümüzde EKS’yi tasarlayıp simüle etmek ve güvenliğini sağlamak için literatürdeki çalışmaların sayısı hızla artmaktadır [52-63]. Sistemin güvenlik analizini yapmak, saldırıları karşı koruma sağlamak, ağ üzerinden gelebilecek siber saldırıları tespit etmek amacıyla birçok araştırma yapılmaktadır. Bir EKS’nin tasarımına karar verilirken işlevine uygun şekilde topolojiye sahip olması gerekmektedir. Bu noktada SCADA, PLC, DKS gibi alt sistemler akla gelmektedir. Bu terimlere ait detaylı bilgiler alt başlıklarda verilmiştir.

2.1.1. SCADA sistemi

SCADA, uzak lokasyonlarda bulunan fabrika veya tesislerdeki süreçleri denetleme, veri toplama ve kontrol etme işlemlerini merkezi bir bilgisayar sistemi aracılığıyla yöneten sistemi ifade etmektedir. SCADA sistemi; merkezi bilgisayar sistemi, insan-makine arayüzü, PLC, RTU, endüstriyel iletişim ağı gibi bileşenleri içermektedir. Su dağıtım ve atıksu toplama sistemleri, elektrik iletim ve dağıtım sistemleri, petrol ve doğalgaz boru hatları gibi dağıtım sistemlerinde tercih edilen mimari SCADA

sistemleridir. SCADA sistemleri, farklı sahalardaki bilgileri toplamak ve bu bilgileri merkezi bilgisayar sistemine aktarmak ve raporlamak amacıyla düşünülmüştür. Operatör, fiziksel olarak sahada bulunmadan uzaktaki merkezi bir konumdan sistemi izleyip kontrol edebilir. Uzaktan yapılan bu işlemlerin alt yapısında endüstriyel iletişim protokolleri yer almaktadır. İlerleyen başlıklarda endüstriyel protokoller hakkında detaylı bilgi verilmiştir.

Şekil 2.2’de basit bir SCADA sistemi topolojisi verilmiştir.

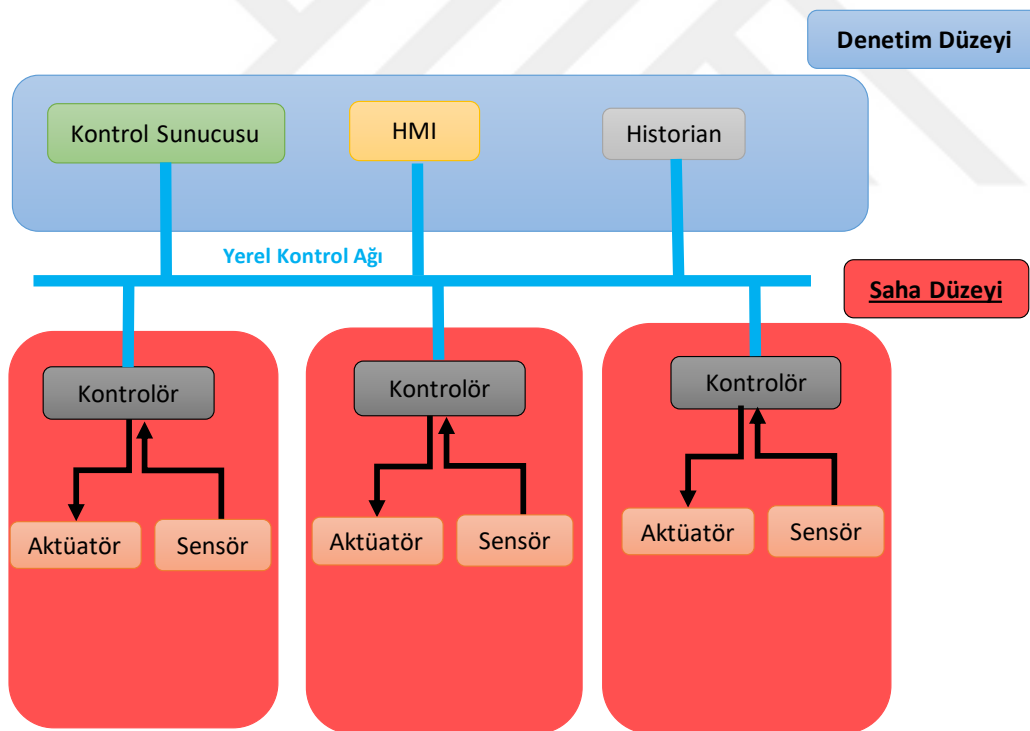


Şekil 2.2. Örnek bir SCADA sistemi topolojisi.

Verilen bu topolojide Merkezi kontrol birimi ve bu merkeze bağlı üç farklı istasyon yer almaktadır. Merkezi kontrol biriminde MTU, HMI, Historian ve Modem bileşenleri bulunmaktadır. MTU cihazı, kontrol işlemlerinin yapıldığı merkez sunucudur. HMI cihazı uzak istasyonlardaki bilgileri ekranda göstermektedir.

Uzak istasyonlar ile merkez arasındaki bağlantı internet aracılığıyla sağlanmaktadır. Uzak istasyonların her birinde birer modem bulunmaktadır. PLC cihazı direkt olarak modeme bağlanabilir veya RTU üzerinden dolaylı olarak da bağlanabilir. Uzak istasyonlardaki bileşenler arasındaki bağlantı da LAN aracılığıyla yapılabilmektedir. Uzak istasyonlarda HMI cihazı olmayabilir. Bu durumda müdahale merkez istasyondan yapılmaktadır. IED cihazı elektrik şebekesi sistemlerinde kullanılmaktadır. IED cihazı, PLC yerine kullanılıp aktüatör ve sensörlerle çalışabilmektedir. Genel endüstriyel işlemlerde PLC cihazı tercih edilir.

Şekil 2.3’de basit bir Dağıtık Kontrol Sistemi (DKS) topolojisi verilmiştir.



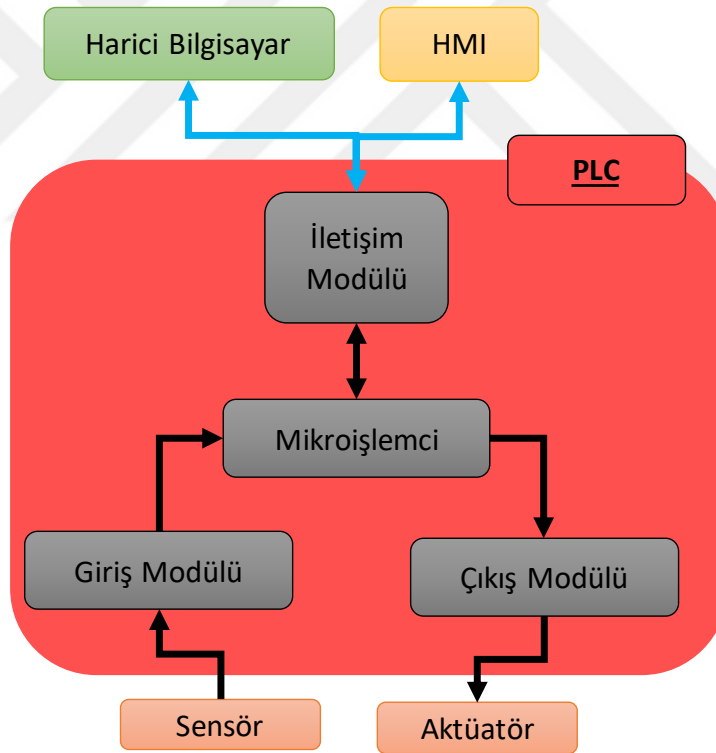
Dağıtık Kontrol Sistemleri, SCADA sistemlerinin aksine aynı coğrafi konumdaki üretim sistemlerini kontrol etmek için kullanılmaktadır. Bu sistemler genellikle ayrı parça kontrol sistemleridir. Bu sistemlerde kontrol işlemleri merkezi bir istasyondan yönetilmez. Bunun yerine birlikte çalışan kontrolörler arasında görev dağılımı yapılır.

Bu görev dağılımındaki alt sistemleri denetleyen bir kontrol mimarisi DKS'ye entegre edilmiştir. Bu kontrol mimarisi, merkezi bir kontrol döngüsü içermektedir.

Şekil 2.3'deki topolojide saha düzeyi ve denetim düzeyi olmak üzere iki seviye gösterilmektedir. Saha düzeyinde farklı süreçleri yürüten kontrolörler yerel kontrol ağı ile denetim düzeyine bağlıdır. Denetim düzeyinde bulunan HMI ile operatörler tüm sistemi monitor edebilmektedir. Kontrol sunucusu ise tüm kontrolörlere komut gönderebilir. Kontrolörler ise kontrol sunucusundan gelen komutlara ve sensör verilerine göre aktüatörleri kontrol etmektedir. PLC cihazları da birer kontrolör olduğundan dolay bu sistemlerde sıklıkla tercih edilmektedir.

2.1.3. Programlanabilir mantıksal denetleyici

Şekil 2.4'de PLC kullanılarak tasarlanmış basit bir kontrol sistemi topolojisi verilmiştir.



Şekil 2.4. Basit bir kontrol sistemi topolojisi.

PLC cihazı, üretim süreçlerini kontrol etmek için kullanılan ve girişlerinden aldığı bilgileri kullanarak çıkışlarına komut gönderen endüstriyel bir bilgisayardır. PLC cihazları kullanıcı tarafından programlanabilir bir belleğe sahiptir. PLC cihazları, zamanlama, mantıksal, sayıcı ve aritmetik vb. fonksiyonlara sahiptir. Hem DKS hem

de SCADA sistemlerinde kontrol bileşeni olarak kullanılmaktadır. Ek olarak PLC cihazları daha küçük kontrol sistemi yapılandırmasında ana kontrolör olarak kullanılmaktadır.

Şekil 2.4'deki topolojide direkt olarak PLC cihazına bağlı bir bilgisayar ve HMI cihazı bulunmaktadır. Operatör, sensörlerden gelen veriye göre aktüatörlerin istenen şekilde çalışması için PLC cihazını bilgisayar aracılığıyla programlama yapar. HMI üzerinden süreci görsel olarak takip edebilir.

2.2. Endüstriyel Haberleşme Protokolleri

Endüstriyel sistemlerdeki cihazlar arasında veri iletişimini sağlayan protokoller endüstriyel haberleşme protokolleri olarak adlandırılmaktadır. Bu protokoller aracılığıyla HMI, PLC, RTU, aktüatör, sensör gibi endüstriyel cihazlar arasında hızlı bir iletişim sağlanmaktadır. Bu sayede cihazlar birlikte çalışabilmekte ve gerçek zamanlı olarak veri aktarımı yapabilmektedir. Endüstriyel cihazlar ile kontrol sistemleri arasındaki iletişimin güvenliği için temel oluşturmaktadır.

Birçok endüstriyel haberleşme protokolü mevcuttur. Bir zamanlar endüstriyel sistemlerde seri ağlar kullanılmaktaydı ancak günümüz şartlarında daha yüksek hızlı ve daha verimli olan Ethernet tabanlı iletişim protokolleri kullanılmaktadır. Seri ağ yapısını kullanan protokollere; Modbus-RTU, Profibus, CAN, DeviceNet vb. örnekler verilebilir. Bu protokoller Fieldbus sistemini kullanmaktadır. Ethernet tabanlı iletişim protokollere; Modbus/TCP, DNP3, EtherCAT, Profinet, EtherNet/IP vb. protokoller örnek verilebilir. Kablosuz ağ tabanlı protokollere ise WirelessHART örnek verilebilir. Her endüstriyel protokol her endüstriyel cihaz ile uyumlu olmayabilir. Üretici firma maliyet göz önünde bulundurarak tüm protokollerle çalışacak şekilde endüstriyel cihaz üretmeyebilir.

Endüstriyel protokollerde iletişim türü Master/Slave ya da Client/Server yapısı olarak tanımlanmaktadır. Master ve Client türündeki cihazlar istek yapar veya komut gönderir. Slave ve Server türündeki cihazlar ise bu isteklere veya komutlara karşılık verir. Her bir endüstriyel cihazın sayısal bir adresi vardır. Bu adresler üzerinden cihazlar birbiriyle direkt olarak haberleşebilir.

EtherNet/IP, açık kaynaklı TCP/IP üzerine kurulu ve Ortak Endüstriyel Protokol (CIP) kullanan uygulama katmanı protokolüdür. EtherNet/IP, CIP sayesinde diğer

protokoller ile iletişim kurabilmektedir. PROFINET, Siemens gibi büyük üreticiler tarafından kullanılan ve üç adet sınıfı (A, B, C) olan bir protokoldür. EtherCAT, Ethernet çerçevelerini kullanan ve donanım seviyesinde hızlı işlem yapan bir protokoldür. DNP3, uzaktan izleme ve yönetim için tasarlanmış bir protokoldür. Modbus/TCP ise Ethernet üzerinden Modbus seri mesajlaşmasını kullanan protokoldür.

Eski fabrikalar ve üretim tesisleri Ethernet tabanlı olmayan protokoller kullanılarak tasarlanmıştır. Zamanla Ethernet tabanlı protokoller yaygın olarak kullanılmaya başlanmıştır. İletişim ağının, sorunsuz bir şekilde tüm endüstriyel cihazları birbirine bağlaması önem arz etmektedir. Eski tesislerin alt yapı yenilemelerinde Ethernet tabanlı protokollerin tercih edileceği öngörülmektedir. Bu noktada Ethernet tabanlı protokollerin seçiminde güvenlik, gerçek zamanlılık, kullanılabilirlik gibi parametrelerin göz önünde bulundurulması gerekmektedir.

2.3. EKS Güvenliği

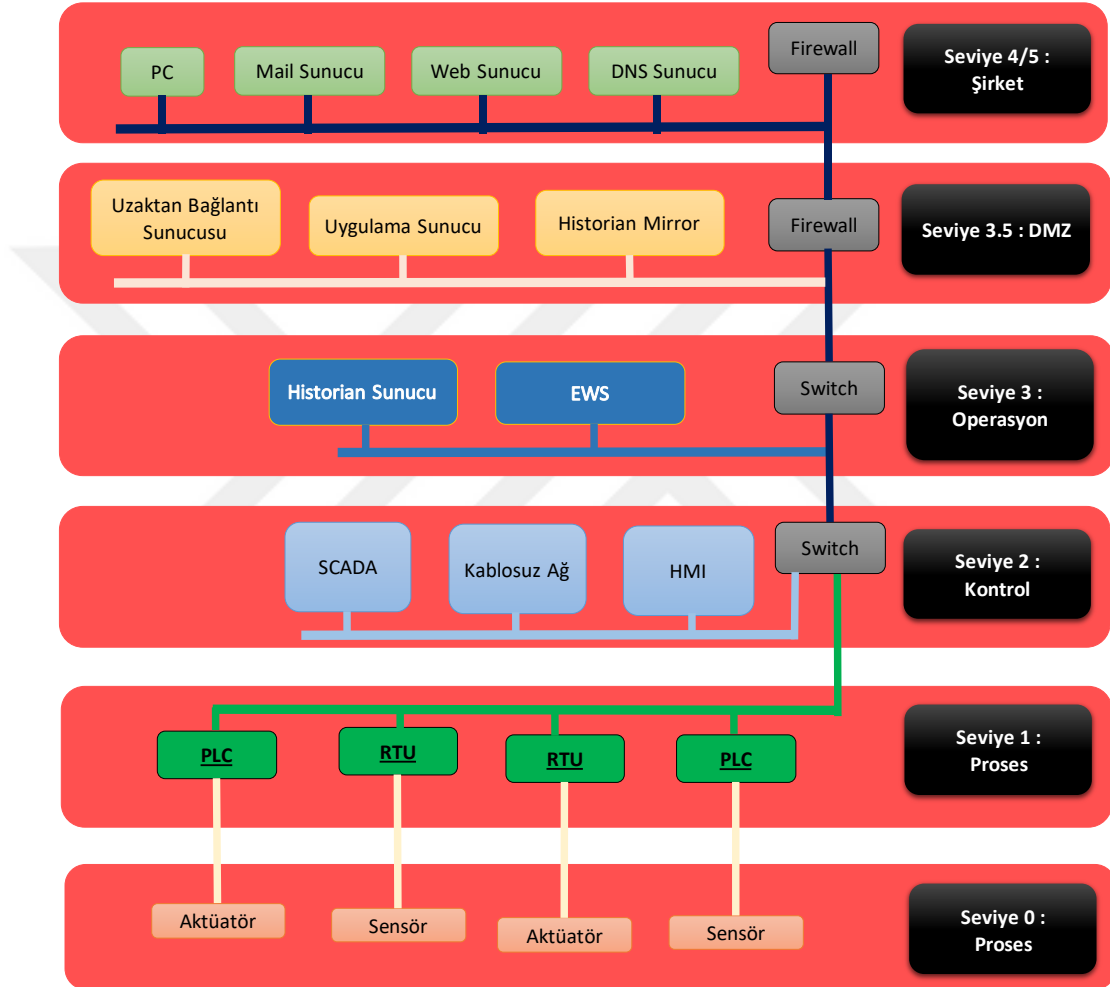
Tarihsel olarak, enerji santrallerinin, su yönetimi tesislerinin, elektrik şebekelerinin ve diğer kritik altyapıların üretiminde ve işletilmesinde kullanılan mühendislik bileşenlerinin çoğu protokol bazlı akıllı özelliklere sahip değildi. Bilgisayar sistemine bağlı olan bileşenler özel endüstriyel protokoller kullanıyordu. Bu protokollerin tasarımında kullanılabilirlik parametresi ön plana çıkmış ve gizlilik parametresi göz ardı edilmiştir. Endüstriyel sistemlerdeki bileşenlerin bağlı olduğu ağlar, internet ortamına bağlı değildi. EKS, artık uzaktan yönetebilir ve izlenebilir duruma gelmiştir. Günümüzdeki EKS bileşenleri genellikle internete bağlı durumdadır. Böylelikle yerel ağda olmayan kötü amaçlı bir bilgisayar kullanıcısı, uzaktan bu bileşenlerle iletişim kurabilme şansı elde etmiştir. Bu noktada gizlilik önem arz etmekte olup güvenlik endişeleri öne çıkmaktadır.

Endüstriyel sistemlere, bilgi teknolojileri özelliklerinin entegre edilmesi büyük bir yenilik olarak gözüktse de güvenlik tehditlerini akla getirmektedir. Endüstriyel cihazlarda kullanılan sistem yazılımları, endüstriyel haberleşme protokolü, veri ihlalleri gibi konular büyük risk taşımaktadır. Sistemi tasarlayan mühendislerin göz önünde bulundurması gereken bir çok risk faktörü bulunmaktadır. Siber saldırılara açık bir sistemin işleyişi her an kesintiye uğrayabilir veya hatalı işlem yapabilir. Bunun sonucunda toplumu etkileyen felaketler meydana gelebilir.

EKS güvenliği; mühendisler, siber güvenlik uzmanları ve risk yönetimi liderleri tarafından ele alınması gerekir. Mühendisler, sistemi tasarlama aşamasında internete ve bilgi teknolojileri alt yapısına bağlanmasını düşünerek yeni oluşabilecek siber güvenlik açıklarını ele alması gerekmektedir. Ayrıca eski endüstriyel cihazların güvenliklerinin nasıl sağlanacağı konusu da mühendisler tarafından ele alınmalıdır. Siber güvenlik uzmanlarının görevi ise EKS cihazlarının bağlı olduğu güvenilir ağ tasarımı yapmaktır. EKS sistemlerinde kullanılabilirlik parametresi, gizlilik parametresinden daha önemlidir. Siber güvenlik uzmanları bu durumu gözeterek operasyonel işlemleri aksaklık olmayacak şekilde ağ alt yapısını kurması gerekir. Risk yönetimi liderleri, sistemlere yapılacak saldırıların potansiyel etkilerine odaklanmalıdır. Endüstriyel sistemlerin aksaklık yaşaması halinde meydana gelebilecek etkilerin anlaşılması gerekmektedir. Bu durumu tetikleyebilecek tüm risklerin hesaplanması, risk yönetimi liderlerinin temel görevidir. Ayrıca göz önünde bulundurulması gereken önemli bir nokta şudur; bilgi teknolojilerinde uygulanan siber güvenlik çözümleri direkt olarak endüstriyel sistemlere uygulanamaz. Bunun sebebi, endüstriyel cihazlar kaynak kısıtlı, düşük hızlı işlem kapasitesine sahiptir. Buna karşın bilgi teknolojileri sistemleri güçlü bilgisayar yapısına sahiptir. BT sistemlerinde uygulanan bir güvenlik çözümü endüstriyel sistemlerde, sistemin işleyişini yavaşlatabilir veya cihazlarla uyumlu olmayabilir. Bu aşamada uygulanacak siber güvenlik çözümleri, endüstriyel cihazlara uygun olacak şekilde yapılandırılmalıdır.

Purdue Kurumsal Referans Mimarisi (PERA), EKS güvenliği için tasarlanmış bir yapısal modeldir. 1990'lı yıllarda J. Williams ve Purdue Üniversitesi Konsorsiyumu üyeleri tarafından geliştirilmiştir. Bu mimari, EKS alt yapısında bulunan cihazların farklı düzeylerde nasıl güvenlik önlemi alınacağı konusunda rehberlik etmektedir. Seviye 0, sistemin çalışmasını doğrudan etkileyen kritik bileşenlerin bağlı olduğu cihazları (sensör, aktüatör) içermektedir. Seviye 1, akıllı kontrol cihazlarının (PLC, RTU) bulunduğu seviyedir. Bu cihazlar doğrudan internete bağlanmamalıdır. Seviye 2, sistemin genel süreçlerini kontrol eden cihazlar (SCADA, HMI) bu seviyede yer almaktadır. Operatörler sürecin izlenmesini ve yönetmesini sağlar. Seviye 3, tesis genelindeki süreçler burada yönetilir. Şirketin verilerinin depolandığı Historian Sunucu ve yapılandırma işlemlerinin yapıldığı Engineering Workstation (EWS) bu seviyede yer almaktadır. Seviye 3.5 olarak da adlandırılan DMZ alanı, operasyonel teknolojiler ve bilgi teknolojileri katmanları arasında ayrılmasını sağlamaktadır.

Burada, Historian sunucusundaki verilerin kopyalandığı bir sunucu, uzaktan bağlantı için bir sunucu ve uygulamalar için bir sunucu bulunmaktadır. Seviye 4 ve 5, iç içe kullanılabilmektedir. Seviye 4, üretim planlama ve operasyonel yönetim için Seviye 5 ise kurumsal yönetim ve iş yönetimi için kullanılmaktadır. Şekil 2.5'te Purdue Kurumsal Referans Mimarisi gösterilmektedir. Bu gösterimde Seviye 4/5, internete açık olup tüm iş süreçlerini kapsamaktadır.



Şekil 2.5. Purdue Kurumsal Referans Mimarisi (PERA)



3. KURAMSAL TEMELLER

Endüstri 4.0 ile birlikte endüstriyel sistemler akıllı özelliklere sahip olmuş, internet bağlantısı vasıtasıyla kolaylıkla erişilebilir hale gelmiştir. Endüstriyel Kontrol Sistemleri'nde en fazla kullanılan Modbus/TCP protokolü, endüstriyel cihazlar arasında iletişimi Ethernet üzerinden sağlayan bir protokoldür. Endüstriyel Kontrol Sistemi'ndeki bileşenler (SCADA, DKS, PLC, sensör, aktüatör) bu protokol aracılığıyla kontrol merkezi sistemleriyle haberleşmektedir. Modbus/TCP protokolünün bu kadar yaygın kullanılması, güvenlik risklerini de beraberinde getirmektedir. İnternete bağlı olan kaynak kısıtlı olan endüstriyel cihazların sayıları artmıştır. Bu cihazların güvenliğini sağlamak için kullanılan geleneksel şifreleme algoritmaları bu cihazlar üzerinde yoğun işlem yükü getirmekte olup verimliliği azaltmaktadır. Bu noktada, hafif sıklet şifreleme algoritmaları, düşük işlem maliyeti ve enerji kullanımı ile güvenliğini sağlamak ön plana çıkmaktadır. Bu bölümde, Modbus/TCP protokolünün genel yapısı, fonksiyonları, güvenlik açıkları ile hafif sıklet şifreleme algoritmaları hakkında temel bilgiler verilecektir.

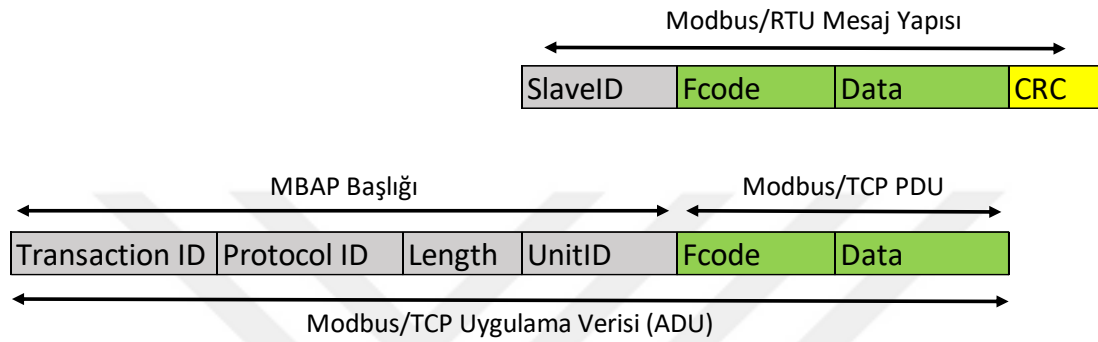
3.1. Modbus/TCP Protokolü

Modbus/TCP protokolü endüstriyel sistemlerde en yaygın kullanılan, basit yapısı ve sadeliği ile ön planda olan bir endüstriyel iletişim protokolüdür. Bu protokol, endüstriyel cihazlar arasında hızlı bir şekilde veri alışverişi yapmak üzere sıklıkla kullanılmaktadır. Modbus/TCP protokolü istemci-sunucu mantığıyla çalışmaktadır. Modbus/TCP protokolü TCP/IP mimarisini kullanır ve varsayılan olarak 502 port numarasına sahiptir. Modbus/TCP basit bir protokol olduğundan dolayı farklı markaya ait cihazların birlikte çalışabilmesine imkan tanımaktadır. Bu alt bölümde Modbus protokolüne yönelik tanımlayıcı bilgiler, fonksiyonlar ve güvenlik hususları hakkında bilgiler verilmektedir.

3.1.1. Genel bakış

Modbus protokolü, DKS ve SCADA sistemlerine yönelik olarak 1979 yılında geliştirilmiş bir endüstriyel haberleşme protokolüdür. Modbus protokolü farklı iletişim ortamında çalışabilmektedir. Seri haberleşme ortamında çalışan versiyonu

Modbus/RTU ve Ethernet ortamında çalışan versiyonu Modbus/TCP olarak adlandırılmaktadır. Bu tez kapsamında yapılan çalışmalar Modbus/TCP protokölüne yöneliktir. Modbus protokolü, endüstriyel ağ ortamına bağlı cihazlar arasında master/slave yapısı kullanarak haberleşme sağlamaktadır. Modbus, kaydedici değerlerinin okunması ve veri yazılması gibi işlemler için tanımlanmış fonksiyon kodları kullanarak çeşitli endüstriyel otomasyon süreçlerini yönetmektedir. Şekil 3.1’de Modbus protokolünün paket yapısı gösterilmiştir.

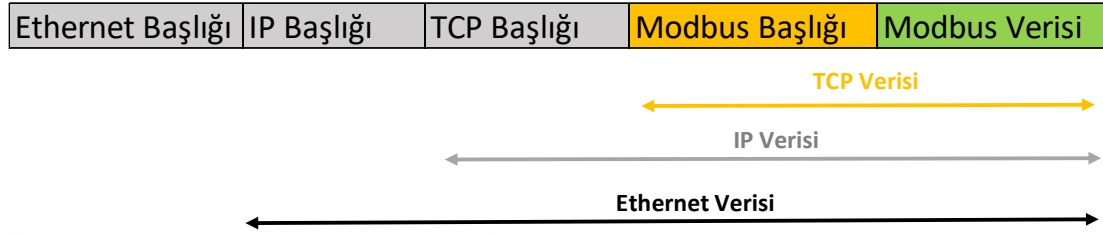


Şekil 3.1. Modbus/TCP Paket Yapısı.

Modbus/RTU mesaj yapısında SlaveID, Fcode, Data ve CRC alanları bulunmaktadır. SlaveID alanı, 1 bayt olup her bir cihazın adresini tanımlamaktadır. Fcode alanı, Modbus fonksiyon kodlarını saklamaktadır. Data alanı, Modbus protokolü ile iletilmek istenen veriyi ifade etmektedir. CRC alanı ise 2 bayt uzunluğa sahip olup Modbus mesajının bütünlüğünün kontrolü için kullanılmaktadır. Modbus protokolü Ethernet ortamına taşınırken Modbus/RTU mesaj çerçevesi korunmuştur. CRC alanı çerçeveden çıkarılmıştır çünkü Ethernet çerçevesinde zaten hata kontrolü için mekanizmalar mevcuttur.

Modbus/TCP çerçeve yapısında; temelde başlık ve protokol veri alanları mevcuttur. Modbus/TCP paketi; MBAP başlığı, fonksiyon kodu ve veri alanlarından oluşmaktadır. MBAP başlığı 7 bayt, fonksiyon kodu 1 bayt, veri alanı ise değişken bayt uzunluğuna sahiptir. MBAP başlığı içerisinde UnitID, Length, ProtocolID ve TransactionID alanları vardır. UnitID alanı, Modbus/RTU çerçeve yapısındaki SlaveID alanına eşdeğerdir. Length alanı 2 bayt uzunluğunda olup Modbus mesajının uzunluğunu belirtir. ProtocolID alanı 2 bayt uzunluğunda olup Modbus protokolü için 0 varsayılan değerine sahiptir. TransactionID alanı ise TCP üzerindeki istek/cevap mesajlarının sıra beklenmeden işlenmesi amacıyla işlem eşleştirmesi için kullanılır ve

2 bayt uzunluğa sahiptir. Şekil 3.2’de gösterildiği gibi Modbus/TCP uygulama verisi (ADU), TCP segmentinin veri alanına eklenir. Katmanlı mimariye göre TCP başlık ve TCP veri alanı, IP verisini oluşturmaktadır. IP başlık ve IP veri alanı da Ethernet verisini oluşturmaktadır. Ethernet verisi ile Ethernet başlık alanı birleştiğinde artık Modbus/TCP iletişimi için mesaj hazır hale gelmiştir.



Şekil 3.2. Modbus/TCP Paket Kapsüllemesi.

3.1.2. Modbus fonksiyonları ve kaydedicileri

Modbus haberleşmesi fonksiyon kodlarıyla sağlanır. Modbus master cihazı yapmak istediği işlemi Modbus Slave cihazına fonksiyon kodlarıyla iletir. Fonksiyon kodu 1, 2, 5 ve 15 olan fonksiyonlar dijital olup coil bazlı yani tek bit ile işlem yaparlar. Coil terimi, açma/kapama görevini yerine getiren değişkeni ifade etmekte olup 0 ya da 1 değerini alır. Fonksiyon kodu 3, 4, 6 ve 16 olan fonksiyonlar analog olup kaydedici bazlı yani 16 bit üzerinden işlem yaparlar. Bu fonksiyonların dışında arıza tespitleri için kullanılan fonksiyonlar da vardır. Fonksiyon kodu 7, 8, 11, 12, 17 ve 21 olan fonksiyonlar arıza tespitinde kullanılmaktadır.

Modbus protokolüne sahip endüstriyel cihazlarda veriler, kaydedicilerde (registers) saklanır. 0xxxx referans adresine sahip olan kaydediciler çıkış verilerini dijital çıkış kanalına iletmek için kullanılmaktadır. Bu tür kaydediciye bağlı bir cihaz, Modbus protokolü ile açma/kapama işlemi yapılabilir. 1xxxx referans adresinin durumu, ilgili dijital giriş kanalına bağlı cihaz tarafından yönetilir. Bu tür kaydedici üzerinden bir cihazın açık olup olmadığı bilgisine protokol ile erişilebilir. 3xxxx referans adresleri, dış bir kaynaktan alınan analog bir verinin 16 bitlik sayısal verisini içerir. 4xxxx referans adresleri ise 16 bitlik sayısal verileri bir çıkışa göndermek veya depolamak için kullanılmaktadır.

Modbus protokolünde kullanılan en sık fonksiyon tipleri Tablo 3.1’de belirtilmiştir.

Tablo 3.1. Temel Modbus Fonksiyonları.

Fonksiyon Kodu	Fonksiyon Kodu (hex)	Fonksiyon Adı	Açıklama
01	0x01	Read Coils	Dijital çıkışları oku
02	0x02	Read Discrete Inputs	Dijital girişleri oku
03	0x03	Read Holding Registers	Analog çıkış kaydedicilerini oku
04	0x04	Read Input Registers	Analog giriş kaydedicilerini oku
05	0x05	Write Single Coil	Sadece bir dijital çıkış yaz
06	0x06	Write Single Register	Sadece bir analog çıkış kaydediciye yaz
15	0x0F	Write Multiple Coils	Birden fazla dijital çıkış yaz
16	0x10	Write Multiple Registers	Birden fazla analog çıkış kaydediciye yaz

Modbus kaydedici adresleri ve türleri Tablo 3.2’de verilmiştir.

Tablo 3.2. Modbus Kaydedici Referans Adresleri.

Adres	Özellik	İsim	Açıklama
0xxxx	Okuma/Yazma	Coils	Dijital çıkışlar
1xxxx	Okuma	Discrete Inputs	Dijital giriş
3xxxx	Okuma	Input Registers	Analog girişler
4xxxx	Okuma/Yazma	Holding Registers	Analog çıkışlar

Örnek olarak bir endüstriyel otomasyon sisteminde bir motoru çalıştırmak istediğimizi düşünelim. Motorun anahtarının bağlı olduğu PLC cihazındaki coil adresi “00055” olsun. Sıcaklık verisinin bağlı olduğu register adresi “30001” olsun. Motorun dönme hızının bağlı olduğu çıkış “40001” olsun. Motoru çalıştırmak istediğimizde kullanacağımız fonksiyon kodu 5 yani Write Single Coil adlı fonksiyondur. Bu fonksiyon ile “00055” adresine 1 verisi yazılır ve motor çalışır. Eğer motorun çalışıp çalışmadığı kontrol edilmek istenirse kullanılacak fonksiyon kodu 1 yani Read Coils fonksiyonudur. Bu fonksiyon ile “00055” adresini gönderilir ve gelen cevap 1 ise motor çalışıyordur, 0 ise motor çalışır durumda değildir. Sıcaklık bilgisini okumak için fonksiyon kodu 4 yani Read Input Register fonksiyonu kullanılır. Okunan sıcaklık bilgisine göre motorun dönme hızı kontrol edilmek istenildiğinde, bir program veya operatör aracılığıyla bu işlem yapılabilir. Bu durumda kullanılacak fonksiyon kodu 6 yani Write Single Register fonksiyonudur. Operatör bu fonksiyonu kullanarak motorun dönme hızını ayarlayabilir.

3.1.3. Güvenlik açıkları

Modbus/TCP protokolüne yönelik saldırılar, protokolün kendisini ve ağ alt yapısını hedef alabilmektedir. Modbus protokolünde kimlik doğrulama, mesaj şifreleme, erişim kontrolü, komut güvenliği yoktur. IP ağına bağlı olduğu için DoS ve Replay saldırısı gibi bilinen ağ saldırısı türlerine de açıktır.

Modbus/TCP uygulama verisi, şifreleme kullanılmadan açık metin (plaintext) olarak iletilmektedir. Dolayısıyla ağ ortamına erişimi olan bir saldırgan bu paketi ele geçirebilir ve iletilen bilgileri okuyabilir. Ayrıca elde ettiği bilgileri kullanarak kötü amaçlı komut içeren bir Modbus paketi oluşturabilir ve bunu hedefe gönderebilir. Kimlik doğrulama mekanizması olmadığından dolayı paketi alan cihaz gönderici kimliğine bakılmadan direkt komutu çalıştırır. Bu durum protokol için önemli bir güvenlik sorunu oluşturmaktadır.

Tablo 3.3’de Modbus protokolüne ait temel güvenlik açıkları belirtilmiştir.

Tablo 3.3. Modbus Güvenlik Açıkları.

Güvenlik Açığı	Açıklama
Kimlik Doğrulama Eksikliği	Kimlik doğrulama olmadığından yetkisiz kişiler tarafından erişilebilir.
Şifreleme Eksikliği	Veriler şifresiz olduğundan kolaylıkla ele geçirilebilir.
Yetkilendirme Eksikliği	Hassas verilerin erişimine yönelik detaylı yetkilendirme seviyeleri yoktur.
DoS Saldırısı	Saldırgan hedef cihaz aşırı istek yaparak süreçleri aksatabilir.
Ortakdaki Adam Saldırısı	Saldırganlar, cihazlar arasındaki iletişimi manipüle edebilir.
Tekrarlama Saldırısı	Saldırganlar, ağ trafiğini yakalayıp kullanıcıların paketlerini taklit ederek sistem işleyişini bozabilir.
Bütünlük doğrulama eksikliği	Verilerin bütünlüğünü doğrulamak için kriptografik veya checksum gibi bir kontrol mekanizması yoktur.
Zafiyetli Firmware	Güncel olmayan ve güvenlik açığı bulunan yazılımların kullanılması risk oluşturur.

3.2. Hafif Sıklet Şifreleme Algoritmaları

Şifreleme, bir verinin kodlanması ve yetkili kişiler tarafından tekrar orijinal durumuna geri döndürülme sürecini kapsayan bir işlemdir. Şifreleme sayesinde gizlilik, erişim kontrolü, bütünlük, kimlik doğrulama vb. işlemler gerçekleştirilebilmektedir. Geleneksel şifreleme algoritmaları işlem yükü ve bellek gereksinimi kısıtlamaları düşünülmeyen tasarlanmıştır. IoT ve IIoT cihazları, kaynak kısıtlı cihazlardır.

Dolayısıyla bu sistemler üzerinde çalışacak şifreleme algoritmalarının yüksek kaynak talep etmesi sistemi gecikmeye uğratacaktır. Bu sebeple, kaynak kısıtlı cihazlar için hafif sıklet (lightweight) şifreleme algoritmaları geliştirilmiştir. Bu algoritmalar, düşük işlem maliyetiyle yüksek güvenlik sağlamayı amaçlamaktadır. Bunun yanı sıra, enerji kullanımı, hız, bellek kullanımı gibi performans ölçütlerini de gözetmektedirler. Bu alt bölümde yedi farklı hafif sıklet şifreleme algoritması ve karşılaştırma yapmak için şifreleme standartı olan AES algoritmasından bahsedilmiştir.

3.2.1. Genel bakış

Hafif sıklet şifreleme algoritmaları kaynak kısıtlı cihazların iletişiminin güvenliğinin sağlanması için tasarlanmış işlem yükü düşük şifreleme algoritmalarıdır. Nesnelerin İnterneti ve Endüstriyel Nesnelerin İnterneti ile birlikte internete bağlı cihaz sayısı oldukça artmıştır. Bu cihazların işlem gücü ve bellek kapasitesi düşüktür. Dolayısıyla cihazlar üzerinde çalışacak algoritmaların tasarımında güvenlik seviyesinden ödün verilmeden bu kriterler göz önüne alınmalıdır. Geleneksel şifreleme algoritmaları bu kriterler düşünülmeden güçlü bilgisayarlar için tasarlanmıştır. Hafif sıklet şifreleme algoritmaları kaynak kısıtlı cihazlar ve gömülü sistemler üzerinde şifreleme işlemlerini etkili şekilde gerçekleştirmeyi amaçlamaktadır. Son yıllarda, hafif sıklet şifreleme algoritmaları ve IoT cihazlarında kullanımı üzerinde yapılan çalışmalar önem kazanmaktadır [64-74]. NIST, 2018 yılında hafif sıklet kriptografi yarışması başlatmıştır. Bu yarışmanın amacı hafif sıklet kriptografide standartizasyon sağlamak için bir algoritma seçilmesidir. Yarışmayı 2023 yılında Ascon algoritması kazanmıştır.

Denklem 3.1 şifrelemeyi ve Denklem 3.2 şifre çözme göstermektedir. Bu denklemlerde C şifrelenmiş veriyi, P şifrelenmemiş veriyi, E şifreleme işlemini, K anahtarı ve D ise şifre çözme işlemini ifade etmektedir. Hafif sıklet algoritmaları olan Present, Ascon, Grain, Skinny, XTEA, Simon, ChaCha20 ve şifreleme standartı olan AES algoritması ilerleyen alt başlıklarda kısaca anlatılmıştır.

$$C = E_K(P) \quad (3.1)$$

$$P = D_K(C) \quad (3.2)$$

3.2.2. Present algoritması

Present algoritması 2007 yılında geliştirilmiştir [75]. Blok şifreleme türünde ve SPN mimarisinde çalışan bir algoritmadır. Blok boyutu 64 bit, tur sayısı 31 olup anahtar boyutu 80 veya 128 bit olarak seçilebilir. Şifreleme işlemi 3 adımdan oluşmaktadır; anahtar ile XOR işlemi, Substitution işlemi ve Permütasyon işlemi. Her bir turda (döngüde) farklı anahtar kullanılmak üzere anahtar üretimi yapılır. Substitution işleminde 4-bit Sbox kutuları kullanılır. Permütasyon işleminde bitlerin bulunduğu pozisyonlar belirlenen kurala göre yer değiştirir. Son olarak, döngü tamamlandığında 32 numaralı anahtar ile tekrar XOR işlemi yapılır ve şifreli metin elde edilir.

Denklem 3.3’de Present algoritmasının şifreleme işlemi verilmiştir. Bu denklemde C şifrelenmiş veriyi, P şifrelenmemiş veriyi, E şifreleme işlemini, K anahtarı, R ise ilgili turdaki SPN işlemini ifade etmektedir.

$$C = E_K(P) = R_{31}(R_{30}(\dots R_1(P \oplus K_0) \dots)) \quad (3.3)$$

Şekil 3.3’de Present algoritmasının şifreleme işleminin sözde kodu verilmiştir.

```
GİRDİ
    açık_metin (64 bit)
    anahtar (80 veya 128 bit)

ÇIKTI
    şifreli_metin (64 bit)

Present_Sifrele(açık_metin, anahtar)
    state = açık_metin
    turAnahtarları = TurAnahtarlarıÜret(anahtar)

    for i=1:31
        state = state XOR turAnahtarları[i]
        state = SBoxKatmanı(state)
        state = PKatmanı(state)
    end

    state = state XOR turAnahtarları[32]

    return state
```

Şekil 3.3. Present algoritması sözde kodu.

3.2.3. XTEA algoritması

XTEA, kendinden önceki TEA algoritmasının eksiklerinin tamamlamak için tasarlanmış bir blok şifreleme algoritmasıdır [76]. Fiestel yapısını kullanmaktadır. Blok boyutu 64 bit, anahtar boyutu 128 bit, tur sayısı 32’dir. Fiestel mimarisini

kullandığı için giriş metni ikiye bölünür ve 32 bitlik bloklar halinde işleme alınır. 128 bitlik anahtar ise yine 32 bit halinde 4 parça olarak işleme alınır. İkiye bölünen giriş metinleri; sola ve sağa bit kaydırma, XOR, MOD işlemleri olan hesaplamalara tabi tutulur. Bu işlemler 32 kez tekrarlanır ve sonucunda şifreli metin elde edilir.

Denklem 3.4’de tekrarlanan bu işlemler verilmiştir. Burada v_0 , v_1 giriş verisinin ikiye ayrılmış 32 bitlik bloklarını, sum sayaç değerini, δ sabit değeri, $\oplus$ XOR işlemini, $\ll$ ve $\gg$ bit kaydırma işlemini, $K[0..3]$ ise 128 bitlik anahtarın 4 parçaya bölünmüş halini temsil etmektedir.

$$v_0 += ((v_1 \ll 4 \oplus v_1 \gg 5) + v_1) \oplus (sum + K[sum \& 3])$$

$$sum += \delta \quad (3.4)$$

$$v_1 += ((v_0 \ll 4 \oplus v_0 \gg 5) + v_0) \oplus (sum + K[sum \gg 11 \& 3])$$

Şekil 3.4’de XTEA algoritmasının şifreleme işleminin sözde kodu verilmiştir.

```

GİRİDİ
    açık_metin (64 bit)
    anahtar (128 bit)

ÇIKTI
    şifreli_metin (64 bit)

XTEA_Sifrele(açık_metin, anahtar)
    v0 = açık_metin[1:32]
    v1 = açık_metin[33:64]
    key [0,1,2,3] = AnahtarBöl(anahtar)
    toplam = delta x 32

    for i=1:32
        v1 = v1 - (((v0 << 4) XOR (v0 >> 5)) + v0) XOR (toplam + key[(toplam >> 11) MOD 4])
        toplam = toplam - delta
        v0 = v0 - (((v1 << 4) XOR (v1 >> 5)) + v1) XOR (toplam + key[toplam MOD 4])
    end

    return [v0,v1]

```

Şekil 3.4. XTEA algoritması sözde kodu.

3.2.4. SKINNY algoritması

Skinny algoritması 64 bit veya 128 bit bloklar üzerinde çalışan, ayarlanabilir bir blok şifreleme algoritmasıdır [77]. Bu algoritma SPN mimarisine sahiptir. Anahtar boyutu 128, 192 ya da 256 bit olabilir. Anahtar türüne göre 32 veya 56 döngüye sahiptir.

Döngü içerisinde tur anahtar ekleme, hücrelere bölme, satırları kaydırma, sütunları karıştırma ve anahtar güncelleme işlemleri yapılmaktadır.

Denklem 3.5’de her turda yapılan bu işlemler verilmiştir. Burada S ilgili turdaki durum, M sütun karıştırma işlemini, R satır kaydırma işlemini, T tur anahtar ekleme işlemini, RC_i tur sabitini ve TK_i ilgili turdaki anahtar bileşenini ifade etmektedir.

$$S_{i+1} = M(R(T(SBox(S_i) \oplus RC_i \oplus TK_i))) \quad (3.5)$$

Şekil 3.5’de Skinny-64-128 varyantına ait bir sözde kod verilmiştir.

```

GİRİDİ
    açık_metin (64 bit)
    tweakey (128 bit)

ÇIKTI
    şifreli_metin (64 bit)

SKINNY_Sifrele(açık_metin, tweakey)
    state = açık_metin

    for i=1:32
        state = TurAnahtarıEkle(state, tweakey[i])
        state = HücrelereBöl(state)
        state = SatırlarıKaydır(state)
        state = SütünlarıKarıştır(state)
        tweakey = AnahtarGüncelle(tweakey)

    end

    return state

```

Şekil 3.5. SKINNY algoritması sözde kodu.

3.2.5. GRAIN algoritması

GRAIN algoritması kısıtlı donanıma sahip cihazlar için saldırılara dirençli olacak şekilde tasarlanmış bir akış şifreleme algoritmasıdır [78]. Anahtar boyutu 128 bit, başlatma vektörü ise 96 bittir. AEAD desteği sayesinde kimlik doğrulama ve şifreleme işlemlerini aynı anda yapabilmektedir. Algoritma blok tabanlı değildir, bit düzeyinde işlemler yapılır.

Denklem 3.6’da Grain algoritmasının şifrelemesi basitçe gösterilmiştir. Burada C_i ilgili şifreli biti, P_i ilgili şifrelenmemiş biti, z_i ise anahtar akışı bitini ifade etmektedir.

$$C_i = P_i \oplus z_i \quad (3.6)$$

Grain algoritması, Doğrusal Geri Beslemeli Kaydırma Yazmacı ve Doğrusal Olmayan Geri Beslemeli Kaydırma Yazmacı yapısından oluşmaktadır. Bu iki yapı sayesinde anahtar akışı üretilmektedir. Anahtar akışından elde edilen veri ile şifrelenecek açık metin verisi XOR işlemine tabi tutularak şifreli metin elde edilmektedir. Ayrıca şifrelenecek metin içerisindeki belirli alanların değiştirilip değiştirilmediğini tespit etmek için doğrulama etiketi verisi de üretilebilmektedir. Şekil 3.6’da basitçe GRAIN-128 algoritmasının sözde kodu verilmiştir.

```

GİRİDİ
    açık_metin
    başlatma_vektörü (96 bit)
    anahtar (128 bit)

ÇIKTI
    şifreli_metin

GRAIN_Sifrele(açık_metin, anahtar, başlatma_vektörü)
    LFSR = başlatma_vektörü[0:95] + '1' + 32 x '0'
    NFSR = anahtar

    for i=1:255
        b = BitOluştur(LFSR, NFSR)
        LFSR = LFSRGüncelle(LFSR, NFSR)
        NFSR = NFSRGüncelle(NFSR, LFSR)
    end

    foreach bit in açık_metin
        b = BitOluştur(LFSR, NFSR)
        sb = bit XOR b
        şifreli_metin.Ekle(sb)
        LFSR = LFSRGüncelle(LFSR, NFSR)
        NFSR = NFSRGüncelle(NFSR, LFSR)
    end

    return şifreli_metin

```

Şekil 3.6. GRAIN algoritması sözde kodu.

3.2.6. SIMON algoritması

SIMON algoritması donanım için optimize edilmiş bir hafif sıklet şifreleme algoritmasıdır [79]. Blok boyutu 32, 48, 64, 96 veya 128 bit olabilir. Anahtar boyutu ise blok boyutuna bağlı olarak 64 ile 256 bit arasında değişiklik göstermektedir. Döngü sayısı yine blok boyutuna bağlı olarak 32 ile 72 arasında değişmektedir. Simon algoritması Fiestel mimarisine sahip bir blok şifreleme algoritmasıdır. Bundan dolayı şifrelenecek açık metin ikiye bölünür. Ardından tur anahtarları üretilir. Sonrasında bölünmüş veri parçaları XOR, AND ve bit sola döndürme fonksiyonlarını içeren işleme tabi tutulur. Bu işlemler 44 kez tekrarlanır. Son olarak döngüden elde edilen iki parçalı metin birleştirilerek şifreli metin oluşturulur.

Denklem 3.7’de her döngü yapılan işlemler gösterilmiştir. Burada R_i ve L_i sağ ve sol yarım blokları, K_i ilgili turdaki tur anahtarını, $\oplus$ XOR işlemini, $\wedge$ AND işlemini, $S^k(x)$ ise x ’in k bit döngüsel sola kaydırma işlemini temsil etmektedir.

$$L_{i+1} = R_i \quad (3.7)$$

$$R_{i+1} = L_i \oplus ((S^1(L_i) \wedge S^8(L_i)) \oplus S^2(L_i)) \oplus K_i$$

Şekil 3.7’de blok boyutu 64 bit anahtar boyutu 128 bit olan SIMON-64/128 varyantına bir sözde kod verilmiştir.

```

GİRİŞ
    açık_metin (64 bit)
    anahtar (128 bit)

ÇIKTI
    şifreli_metin (64 bit)

SIMON_Sifrele(açık_metin, anahtar)
    v0 = açık_metin[1:32]
    v1 = açık_metin[33:64]
    turAnahtarları = TurAnahtarlarıÜret(anahtar)

    for i=1:44
        temp = v0
        v0 = v1 XOR ( (SolaDöndür(v0,1) AND SolaDöndür(v0,8) XOR SolaDöndür(v0,2))
        v0 = v0 XOR turAnahtarları[i]
        v1 = temp
    end

    return [v0,v1]

```

Şekil 3.7. SIMON algoritması sözde kodu.

3.2.7. ChaCha20 algoritması

ChaCha20 algoritması yazılım performansı yüksek bir akış şifreleme algoritmasıdır [80]. Blok boyutu 512 bit, anahtar boyutu 256 bit, başlatma vektörü 96 bittir. Tur sayısı ise 20’dir. Açık metin verisi 512 bit bloklar halinde işleme alınır. Anahtar, sayaç ve başlatma vektörü parametreleri ile anahtar akışı üretilir ve açık metin blokları ile XOR işlemi yapılır. Elde edilen şifreli metin bloğu döngü tamamlanana kadar birbirine eklenir ve şifreli metin elde edilir.

Denklem 3.8’de ChaCha20 algoritmasının şifrelemesi basitçe gösterilmiştir.

$$C_i = P_i \oplus \text{ChaCha20_Blok}(K, \text{nonce}, \text{counter}) \quad (3.8)$$

Burada C_i şifreli metin bloğunu, P_i açık metin bloğunu, K anahtarı, nonce 96 bitlik tekil sayıyı, counter ise 32 bitlik blok sayacını, $\oplus$ ise XOR işlemini ifade etmektedir.

ChaCha20-Poly1305 varyantı AEAD özelliğini desteklemektedir [81]. Şekil 3.8’de algoritmaya ait sözde kod verilmiştir.

```

GİRİDİ
    açık_metin (512 bit)
    başlatma_vektörü (96 bit)
    anahtar (256 bit)

ÇIKTI
    şifreli_metin

ChaCha20_Sifrele(açık_metin, anahtar, başlatma_vektörü)

    foreach açıkmetin_blok in açık_metin
        blok = ChaCha20_Blok(anahtar, sayaç, başlatma_vektörü)
        anahtarAkışı = blok.to_bytes()
        şifreliMetinBlok = açıkmetin_blok XOR anahtarAkışı
        şifreli_metin.Ekle(şifreliMetinBlok)
        sayaç++
    end

    return şifreli_metin

```

Şekil 3.8. ChaCha20 algoritması sözde kodu.

3.2.8. Ascon algoritması

Ascon algoritması 2023 yılında hafif sıklet kriptografi alanında standart olarak seçilmiştir [82]. Blok boyutu 64 bit, anahtar boyutu 128 bit, başlatma vektörü 128 bit ve doğrulama etiketi boyutu 128 bittir. Ascon algoritması AEAD özelliği sayesinde hem şifreleme hem de kimlik doğrulama işlevlerini yerine getirmektedir.

$$C = P \oplus f(S) \quad (3.9)$$

$$T = g(S \oplus K)$$

Denklem 3.9’da Ascon algoritmasının şifreleme işlemi basitçe gösterilmiştir. Burada C şifreli metin, P şifrelenmemiş metin, S iç durum bloğu, $f(S)$ sünger fonksiyonu ile

üretilecek anahtar akışını, K anahtarı, T ise doğrulama etiketini ifade etmektedir.

Şekil 3.9’da Ascon-128 algoritmasının basitçe sözde kodu verilmiştir. Giriş parametresi olarak şifrelenecek metin, anahtar, başlatma vektörü ve doğrulama etiketi üretilecek veri alınmaktadır. Başlangıç durumunda anahtar, başlangıç vektörü ve sabit veri ile 320 bitlik veri elde edilir. Ardından permütasyon ve XOR fonksiyonlarını içeren işlemler dizisi uygulanır. Açık metin ve ilişki veri, state değişkeninin farklı bölgelerinde tutularak ilgili işlemler uygulanır. Şifreli metin ve doğrulama etiketi bilgisi de state değişkeninin içerisinde alınarak elde edilir ve geri döndürülür.

```
GİRİDİ
    açık_metin (64 bit)
    başlatma_vektörü (128 bit)
    anahtar (128 bit)

ÇIKTI
    şifreli_metin (64 bit)

Ascon_Sifre(açık_metin, anahtar, başlatma_vektörü)

    state = [anahtar || başlatma_vektörü || sabit_veri]
    state = Permütasyon(state)
    state = state XOR anahtar

    state = state XOR ilişkili_Veri
    state = Permütasyon(state)
    state = state XOR 0x8000000000000000

    şifreli_metin = açık_metin XOR state[0:64]
    state[0:64] = şifreli_metin
    state = Permütasyon(state)

    state = state XOR anahtar
    state = Permütasyon(state)
    state = state XOR anahtar

    doğrulama_etiketi = state[128 bitlik kısım]

return şifreli_metin, doğrulama_etiketi
```

Şekil 3.9. Ascon algoritması sözde kodu.

3.2.9. AES algoritması

AES günümüzde en yaygın ve standart olarak kullanılan blok şifreleme algoritmasıdır [83]. Hafif sıklet şifreleme algoritmaları ile kıyaslamak için bu çalışmaya dahil edilmiştir. Blok boyutu 128 bit olarak sabit, anahtar boyutları 128, 192 ve 256 bit olarak seçilebilen, tur sayısı ise anahtar boyutuna bağlı olarak sırasıyla 10, 12, 14 olarak değişebilmektedir. AES algoritması SPN mimarisine sahiptir.

Denklem 3.10’da AES algoritmasının şifrelemesi gösterilmiştir. Burada S_i ilgili turdaki durumu, Sub S-box dönüşümünü, Shift satırları sola kaydırma işlemini, M sütunlarını karıştırma işlemini, K_i tur anahtarını ve $\oplus$ ise XOR işlemini ifade etmektedir.

$$S_{i+1} = M(Shift(Sub(S_i))) \oplus K_i \quad (3.10)$$

AES algoritmasında şifreleme işleminde her turda farklı anahtar kullanılmaktadır. Sbox kutuları ile yer değiştirme, sütunları karıştırma, satırları kaydırma ve tur anahtarı ekleme işlemleri döngü içerisinde yapılır. Ancak son turda sütunları karıştırma işlemi yapılmaz. Bu işlemlerin sonucunda şifreli metin elde edilir. Şekil 3.10’da AES-128 algoritmasının basitçe sözde kodu verilmiştir.

```

GİRDİ
    açık_metin (128 bit)
    anahtar (128 bit)

ÇIKTI
    şifreli_metin (128 bit)

AES_Sifrele(açık_metin, anahtar)

    turAnahtarları = TurAnahtarlarıÜret(anahtar)
    state = açık_metin
    state = TurAnahtarıEkle(state, turAnahtarları[0])

    for i=1:9
        state = SBoxKatmanı(state)
        state = SatırlarıKaydır(state)
        state = SütunlarıKarıştır(state)
        state = TurAnahtarıEkle(state, turAnahtarları[i])
    end

    state = SBoxKatmanı(state)
    state = SatırlarıKaydır(state)
    state = TurAnahtarıEkle(state, turAnahtarları[10])

    return şifreli_metin, doğrulama_etiketi

```

Şekil 3.10. AES-128 algoritması sözde kodu.



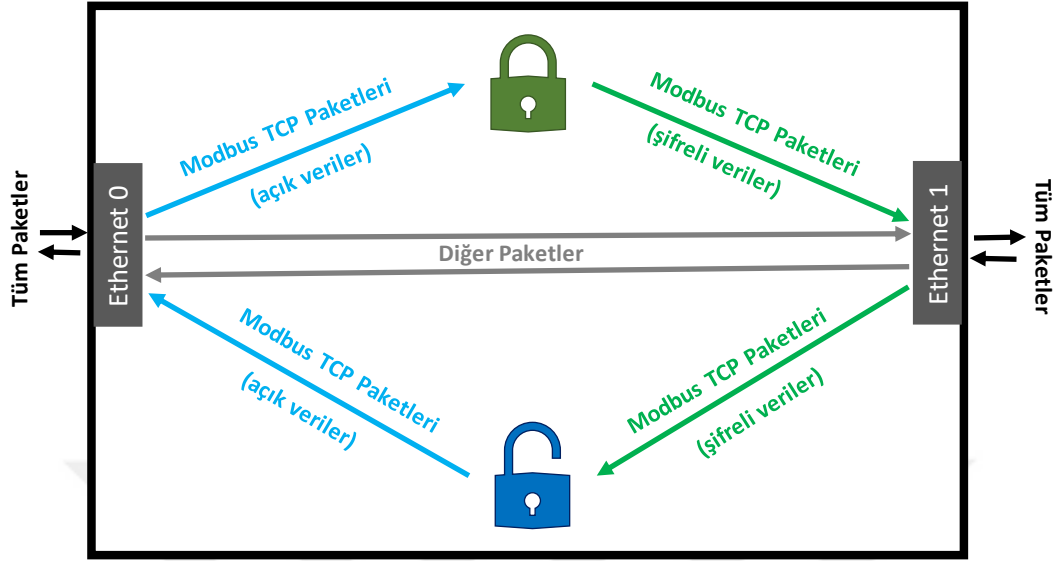
4. ÖNERİLEN YÖNTEM

Bu bölümde, Modbus/TCP protokolü kullanan endüstriyel sistemlerde güvenliği artırmak amacıyla Raspberry Pi 4 ve Orange Pi R1 Plus LTS tabanlı önerilen modülün tasarımı, kullanılan donanım, kullanılan yazılım kütüphaneleri ve kurulumları açıklanmaktadır. Bu çalışmada, geleneksel şifreleme algoritmalarının performans yükü göz önüne alınarak bunların yerine düşük işlem maliyetine sahip hafif sıklet şifreleme algoritmaları tercih edilmiştir. Önerilen yöntem kapsamında, basit bir endüstriyel kontrol sistemi topolojisi tasarlanmıştır. Bu topolojide endüstriyel cihazları temsil etmek için benzetim programları kullanılmış ve önerilen modüller cihazlar arasında yerleştirilmiştir. Python programlama dili ve kütüphaneleri kullanılarak Modbus/TCP paketleri üzerinde işlemler yapılmıştır. Ayrıca şifreleme algoritmalarını tasarlama aşamasında bir soyut sınıf oluşturulmuş, her bir algoritma bu soyut sınıftan kalıtım almıştır. Böylelikle şifreleme algoritmalarının gerçekleştirilmesinde bir standart sağlanmıştır. Alt bölümlerde, önerilen şifreleme modülü ve bileşenleri hakkında detaylı bilgiler verilecektir.

4.1. Şifreleme Modülü ve Bileşenleri

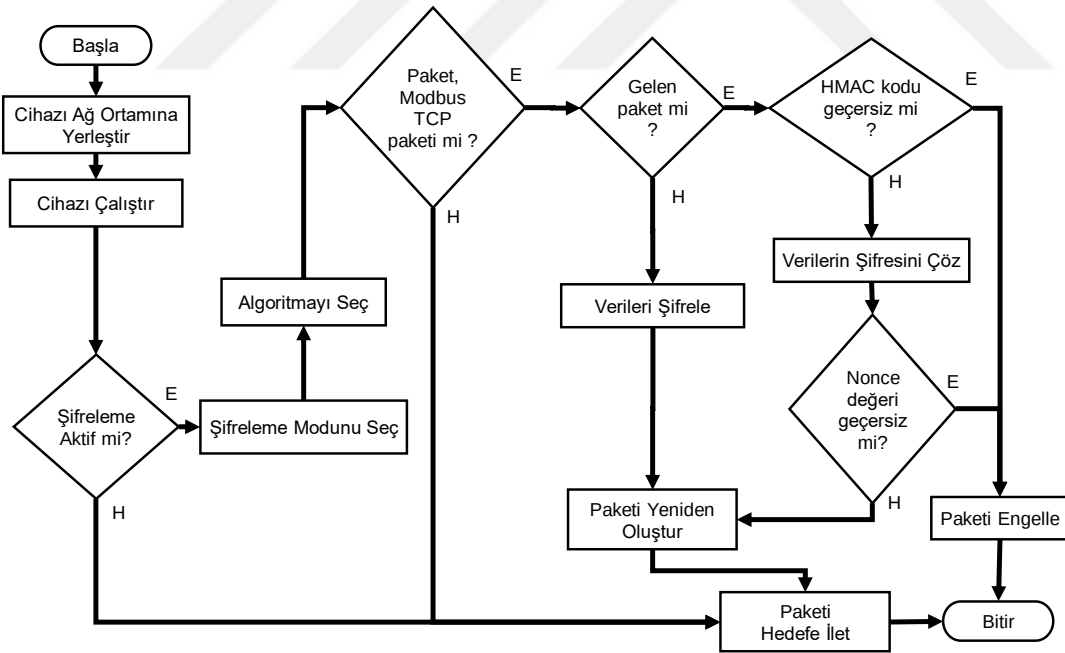
Önerilen şifreleme modülünde iki adet ethernet arayüzü vardır. Her iki Ethernet arayüzünde hem giriş hem çıkış yani çift yönlü ağ trafiği iletişim halindedir. Önerilen modül Ethernet0'dan aldığı paketteki verileri şifreleyerek Ethernet1'e iletecek, Ethernet1'den aldığı şifreli paket verilerini de şifresini çözüp Ethernet0'a iletecektir. Ethernet1 arayüzündeki giden/gelen veriler şifreli, Ethernet0 arayüzündeki giden/gelen veriler şifresiz olarak iletişim halindedir. Bu işlemleri gerçekleştirmek için iki adet Ethernet arayüzüne sahip bir cihaz tercih edilmesi gereklidir. İlk tercih olarak Raspberry Pi 4 cihazı ve USB Ethernet adaptörü seçilmiştir. Raspberry Pi 4 üzerinde mevcut bir adet Gigabit Ethernet portu bulunmaktadır. İkinci Ethernet portu ise USB Ethernet adaptörü ile Raspberry Pi 4 cihazına bağlanarak elde edilir. İkinci tercih olarak üzerinde tümleşik olarak iki adet Ethernet portu bulunan Orange Pi R1 LTS Plus cihazı seçilmiştir. İkinci tercihde USB Ethernet adaptörüne ihtiyaç

duyulmamıştır. Şekil 4.1’de önerilen şifreleme modülünün temsili gösterimi verilmiştir.



Şekil 4.1. Önerilen Modüle Genel Bakış.

Şekil 4.2’de modülün çalışmasını gösteren akış diyagramı verilmiştir.



Şekil 4.2. Önerilen Modülün Çalışma Diyagramı.

4.1.1. Raspberry Pi 4 tabanlı modül

Önerilen şifreleme modülü hem şifreleme hem de şifre çözme işlemlerini gerçekleştirecektir. İletişimin çift yönlü olarak sağlanması için modül, giriş ve çıkış

portu olmak üzere iki adet Ethernet arayüzüne sahip olmalıdır. Bu işlemler için Şekil 4.3’de gösterilen Raspberry Pi 4 cihazı tercih edilmiştir [84].



Şekil 4.3. Raspberry Pi 4 Cihazı Genel Görünümü.

Raspberry Pi 4 cihazı üzerinde tümleşik olarak bir adet Gigabit Ethernet arayüzü bulunmaktadır. İkinci bir Ethernet arayüzü kazandırmak için Şekil 4.4’de gösterilen bir USB 3.0 Gigabit Ethernet adaptörü kullanılacaktır.



Şekil 4.4. USB 3.0 Ethernet Adaptörü Genel Görünümü.

USB Ethernet adaptörü 1000 Mbit/s desteklemekte olup Raspberry Pi 4 üzerinde bulunan 2 adet USB 3.0 portundan herhangi birine bağlanabilir. Bağlantı yapılacak USB 3.0 Ethernet adaptörünün USB 2.0 portlarına bağlanması durumunda ise hız düşüklüğü olacaktır. Bunun sebebi, USB 2.0 portunun desteklediği en yüksek bağlantı hızının 480 Mbps olmasıdır. Kullanılan USB 3.0 Ethernet adaptörünün çipseti Realtek 8153'dür. Tablo 4.1'de modülün temel özellikleri verilmiştir. Bu modüle kurulacak işletim sisteminin grafik arayüze sahip olması durumunda micro-HDMI portu üzerinden görüntü alınabilir. Böylelikle yapılandırma ayarları direkt olarak cihazın kendi üzerinde yapılır. Modülün işlemcisi ARM tabanlı olduğu için kurulacak işletim sisteminin de ARM tabanlı sistemler için üretilmiş olması gerekmektedir. 4 GB ve 8 GB belleğe sahip olan Raspberry Pi 4 cihazları tercih edilmiştir. Ethernet arayüzleri birbirine köprülenmiştir.

Tablo 4.1. Kullanılan Raspberry Pi tabanlı modül özellikleri.

Özellik	Açıklama
İşlemci	Dört çekirdekli Cortex-A72 (ARM tabanlı)
Bellek	4 GB veya 8 GB DDR4 RAM
Ağ Kartı	1 adet Gigabit Ethernet
Kablosuz	802.11ac Wireles, Bluetooth 5.0
USB portu	2 adet USB 2.0, 2 adet USB 3.0
Görüntü çıkışı	2 adet micro-HDMI
Depolama	Micro-SD kart yuvası
Güç girişi	5V DC USB-C konektör

4.1.2. Orange Pi R1 LTS plus tabanlı modül

Raspberry Pi 4 tabanlı modülden farklı olarak Orange Pi R1 LTS Plus tabanlı modülde tümleşik iki adet gigabit ethernet portu bulunmaktadır [85]. Böylelikle ek bir USB ethernet adaptörüne ihtiyaç duyulmadan çift yönlü iletişim sağlanabilmektedir.

Şekil 4.5’de Orange Pi tabanlı modülün görseli verilmiştir. Raspberry Pi cihazına göre daha ufak boyutları vardır.



Şekil 4.5. Orange Pi R1 LTS Plus Cihazı Görünümü.

Tablo 4.2’de modülün temel özellikleri verilmiştir. Bu modülde, görüntü çıkışı ve kablosuz ağ bağdaştırıcısı yoktur. Dolayısıyla yapılandırma işlemleri uzaktan bağlantı veya UART modülü üzerinden bağlantı kurularak yapılabilmektedir. Bu modüle kurulacak işletim sisteminin grafik arayüze sahip olmasına gerek yoktur. Çünkü modül üzerinde direkt olarak herhangi bir görüntü çıkışı (VGA, DVI, HDMI, DP) bulunmamaktadır.

Tablo 4.2. Kullanılan Orange Pi tabanlı modül özellikleri.

Özellik	Açıklama
İşlemci	Dört çekirdekli Cortex-A53 (ARM tabanlı)
Bellek	1 GB DDR3 RAM
Ağ Kartı	2 adet Gigabit Ethernet
USB portu	1 adet USB 2.0
Depolama	Micro-SD kart yuvası
Güç girişi	5V DC USB-C konnektör

Bu modülün işlemcisi ARM tabanlı olduğu için kurulacak işletim sisteminin de ARM tabanlı sistemler için üretilmiş olması gerekmektedir. Orange Pi R1 LTS Plus cihazı üzerindeki ethernet kartlarının çipsetleri YT8531C ve Realtek 8153B'dir. Ethernet arayüzleri birbirine köprülenmiştir.

4.2. Kullanılan Yazılım Bileşenleri

Bu alt bölümde, çalışmada kullanılan işletim sistemi, yazılım kütüphaneleri ve kurulumları açıklanmaktadır. Çalışmanın temel uygulaması Python programlama dili kullanılarak geliştirilmiştir. Python programlama dilinin geniş kütüphane desteği, Modbus/TCP protokol verilerinin şifrelenmesi ve şifresinin çözülmesi için uygun bir ortam sağlamıştır.

4.2.1. İşletim sistemi

Modül üzerinde işlem yapabilmek için öncelikle işletim sisteminin yüklenmesi gerekmektedir. Bunun için ARM tabanlı olan işletim sistemleri tercih edilmelidir. Raspberry Pi 4 için kullanılabilir işletim sistemlerinin bazıları aşağıda verilmiştir:

- Raspberry Pi OS (Raspbian)
- Ubuntu Desktop
- Ubuntu Server
- DietPi
- Retropie
- Armbian

Orange Pi R1 LTS Plus için kullanılabilir işletim sistemlerinin bazıları aşağıda verilmiştir:

- Ubuntu
- Debian
- Armbian
- DietPi

Her iki cihaz için de işletim sisteminin yükleme işlemi Micro-SD kart üzerinden yapılmaktadır. Kart okucuya boş bir Micro-SD kart takılır. Herhangi bir imaj yazdırma

programı ile işletim sisteminin imajı Micro-SD kart üzerine yazdırılır. Bu işlemi yapan bazı programlar aşağıda verilmiştir:

- Raspberry Pi Imager
- BalenaEtcher
- Win32 Disk Imager
- Rufus

İşletim sistemi yüklendikten sonra, Micro-SD kart modül üzerindeki yuvaya takılır ve cihazın güç adaptörü bağlanır. Raspberry Pi 4 cihazı monitör vasıtasıyla kullanılabildiğinden direkt olarak yapılandırma ayarları yapılır. Ancak Orange Pi R1 LTS Plus cihazının yapılandırılabilmesi için uzaktan bağlantı veya UART üzerinden bağlantı gerekmektedir.

4.2.2. Geliştirme ortamı

Modül üzerinde yapılacak işlemlerde yazılım dili olarak Python 3 tercih edilmiştir. Linux tabanlı işletim sistemlerinde genellikle Python 3 kütüphanesi yüklü olarak gelmektedir. Python 3, yüklü olmayan sistemlerde kurulum yapmak için aşağıdaki komut kullanılır:

- `sudo apt-get install python3`

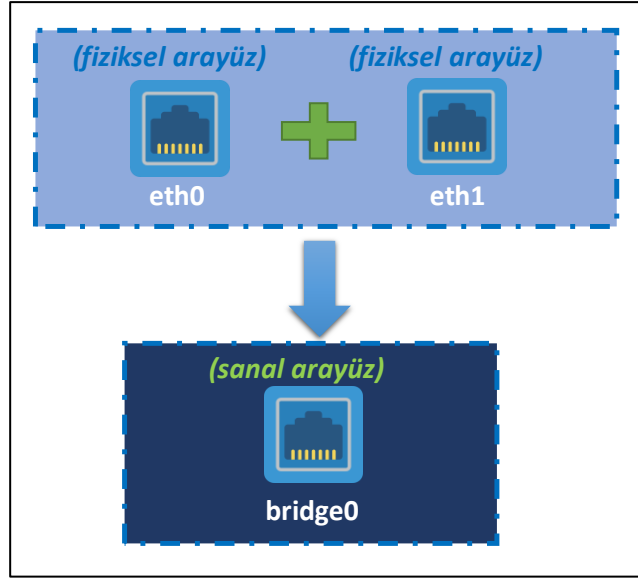
Uygulamada Python tabanlı bazı yazılım paketleri (ModFlask, scapy, netfilterqueue vb.) kullanılmıştır. Bu paketlerin yüklenmesi için Python PIP aracı kullanılmaktadır. PIP aracını yüklemek için aşağıdaki komutlar kullanılmaktadır:

- `curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py`
- `python get-pip.py`

Yapılan yükleme işlemini doğrulamak için aşağıdaki komut kullanılır:

- `pip help`

Her iki cihaz üzerinde ethernet arayüzleri birbirleri arasında paketleri yönlendirmek üzere köprülenmiştir.



Şekil 4.6. Ethernet arayüzleri köprüleme işlemi.

Şekil 4.6’da köprüleme (bridge) yapısı gösterilmektedir. Köprüleme işleminin Linux ortamında yapılması için farklı yöntemler mevcuttur. Bu çalışmada kullanılan köprü yapısını gerçekleştiren komutlar aşağıda verilmiştir:

- `sudo apt install bridge-utils`
- `sudo apt install net-tools`
- `ip link set dev enx0067 down`
- `ip link set dev enx0067 name eth1`
- `ip link set eth1 up`
- `ip addr flush dev eth0`
- `ip addr flush dev eth1`
- `brctl addbr bridge0`
- `brctl addif bridge0 eth0 eth1`
- `ip link set dev bridge0 up`

Öncelikle komutları çalıştırmak için gerekli olan ağ kütüphaneleri yüklenir. Ardından USB ethernet arayüz adı eth1 olarak değiştirilir. Eth0 ve eth1 arayüzlerinin ip adresleri kaldırılır ve bridge0 adında bir köprü arayüz oluşturulur. Bu köprü arayüzüne fiziksel arayüzler olan eth0 ve eth1 eklenerek, köprü arayüz aktif duruma getirilir. Modül bu şekilde EKS’ye bağlandığında şifreleme olmaksızın çalışabilir.

Önerilen yöntemde yalnızca Modbus/TCP paketleri şifrelenecektir. Modbus/TCP protokolü 502 nolu portu kullanmaktadır. Dolayısıyla bu paketleri filtrelemek

amacıyla iptables paketi ve NetFilterQueue paketi kullanılacaktır. Bu paketleri yüklemek için aşağıdaki komutlar kullanılır:

- sudo apt-get install iptables
- pip install NetFilterQueue

502 port numarasına sahip paketler iptables komutu ile filtrelenir ve üzerinde işlem yapılmak üzere netfilterqueue ile oluşturulan kuyruğa yönlendirilir. Yönlendirilen bu paketler üzerinde işlem yapmak için scapy paketi kullanılır. Bu paketi yüklemek için aşağıdaki komut kullanılır:

- pip install scapy

Scapy paketi, IP paketi üzerinde kolaylıkla işlem yapmayı sağlayan bir araçtır. Paketler üzerinde değişiklik yapmak için kullanılmaktadır. Paketin Modbus başlık ve veri alanına karşılık gelen kısımları ayrıştırılır. İki farklı şekilde şifreleme işlemi gerçekleştirilir; Modbus PDU (veri alanı) ve Modbus ADU (başlık ve veri alanı). Şifrelenecek kısımlar seçilen şifreleme algoritması ile şifrelenir. Şifrelendikten sonra paketin ilgili kısmına yerleştirilir. Ancak paket üzerindeki veriler değiştiği için bazı kontrol alanı verilerinin (IP checksum ve TCP checksum) yeniden hesaplanması gerekmektedir. Veri şifrelendiği için IP başlığındaki toplam uzunluk (total length) alanının değeri değişebilir. Bu alanın verisi değişirse IP checksum değeri de değişmektedir. IP checksum değeri, Şekil 4.7’de gösterildiği gibi IP başlığındaki tüm alanların ikili toplam değeriyle hesaplanır.

IP Başlığı	Sürüm (Version)	Başlık Uzunluğu (IHL)	Servis Tipi (Type of Service)	Toplam Uzunluk (Total Length)	
	Tanıtıcı (Identification)			Bayraklar (Flags)	Parça No (Fragment offset)
	Yaşam Süresi (TTL)		Protokol	IP Checksum	
	Kaynak IP Adresi (Source IP Address)				
	Hedef IP Adresi (Destination IP Address)				
	Seçenekler (Options)				

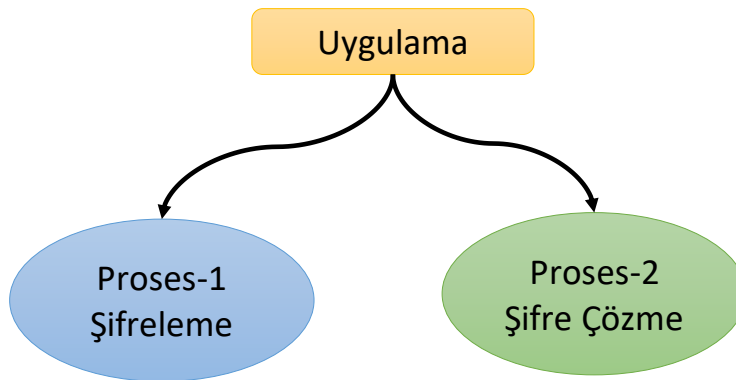
Şekil 4.7. IP checksum hesaplanırken kullanılan alanlar.

Bu işleme IP paketinin veri kısmı dahil değildir. Şekil 4.8’de gösterildiği gibi TCP checksum hesaplanırken; kaynak IP adresi, hedef IP adresi, ayrılmış alan (reserved), protokol alanı, TCP segment uzunluğu verileri içeren sözde IP başlığı, TCP başlığı ve TCP veri alanları işleme alınır. Checksum bilgileri yeniden hesaplandıktan sonra şifrelenmiş paket hedefe yönlendirilir.

Sözde IP Başlığı	Kaynak IP Adresi (Source IP Address)			
	Hedef IP Adresi (Destination IP Address)			
	Reserved	Protokol	TCP segment uzunluğu (TCP segment length)	
TCP Segmenti	Kaynak Port (source port)		Hedef Port (destination port)	
	Dizi Numarası (Sequence Number)			
	Onay Numarası (Acknowledgement Number)			
	Veri No (Data offset)	Ayrılmış alan (Reserved)	Bayraklar (Flags)	Pencere (window)
	TCP Checksum			Acil işareti (Urgent Pointer)
	Seçenekler (Options)			
	Data (Veri)			

Şekil 4.8. TCP checksum hesaplanırken kullanılan alanlar.

Bu işlemler aynı şekilde paket şifre çözülmesi adımı da yapılmaktadır. Sistemin sürdürülebilir olması amacıyla şifreleme ve şifre çözme işlemlerinin eş zamanlı olarak yapılabilmesi için Şekil 4.9’da gösterildiği gibi iki farklı proses tanımlanmıştır. Proses 1, şifreleme işlemi için Proses 2 ise şifre çözme işlemi için yapılandırılmıştır.



Şekil 4.9. Proses Kullanımı.

Şifreleme ve şifre çözme işlemlerin yapılmasını sağlayan kod parçacıkları aşağıda verilmiştir:

- p1 = multiprocessing.Process(target=Şifrele)
- p2 = multiprocessing.Process(target=ŞifreÇöz)
- p1.start()
- p2.start()

Proses-1’de paket şifrlenirken içerisine doğrulama kodu ve nonce değeri eklenmektedir. Doğrulama kodu (HMAC) hesaplanırken, şifrelenecek veri ile bir anahtar hash işlevine tabi tutulur. Proses-2 kendisine ulaşan paketin öncelikle doğrulama kodunu test eder. Eğer doğrulama kodu geçersizse bir manipülasyon olduğu anlaşılır ve paketin hedefe gitmesi engellenir. Doğrulama kodu geçerli ise paketin nonce değerine bakılır. Proses-1, paketin yalnızca bir kere gönderilmesini garanti etmek için paketin içine şifrelenmiş bir nonce değeri eklemektedir. Proses-2 kendisine gelen bu nonce değerini, kendisine daha önce ulaşmış olan nonce değerleri ile karşılaştırır. Eğer daha önce aynı değer gelmişse paketin bir tekrarlama saldırısı olduğu anlaşılır ve hedef yönlendirilmesi engellenir. Eğer daha önce böyle bir nonce değeri gelmemişse paket başarıyla hedefe yönlendirilir.

4.2.3. Benzetim programları

Endüstriyel Kontrol Sistemleri, sürekli çalışan sistemlerdir. Bu sistemlerin kesintiye uğraması şirkete önemli ölçüde zarar verebilmektedir. Dolayısıyla bu sistemler üzerinde test çalışmaları yapmak sistemi aksatabilmektedir. Bunun yerine aynı işlevleri gerçekleştiren programları kullanarak endüstriyel ortamın simüle edilmesi tercih edilir bir çözüm olacaktır. Modbus/TCP protokolü ile çalışan cihazların işleyişini birebir benzetmek için kullanılan programlar vardır. Bu programlar, bir bilgisayara yüklendiğinde o bilgisayarın sanki gerçek bir PLC veya RTU cihazıymış gibi davranmasını sağlamaktadır. Böylelikle gerçek bir endüstriyel cihaz ortamına ihtiyaç duyulmadan sistem üzerinde çalışma imkanı olmaktadır. Birçok benzetim programı olmakla birlikte bu çalışmada kullanılan benzetim programları aşağıdaki gibidir:

- Modbus Poll [86]
- ModFlask [87]

Bu programlar ile bilgisayarlar Modbus Master ve Slave cihazı olarak yapılandırılabilir. ModFlask programı github deposundan indirilebilir. ModFlask programı Python ortamında geliştirilmiştir. Şekil 4.10'da ModFlask programının master cihaz olarak çalıştırıldığı görsel verilmiştir.

```
ModFlask>python main.py
* Serving Flask app 'configurations' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:8001/ (Press CTRL+C to quit)
127.0.0.1 - - [12/Dec/2021 14:55:29] "GET /api/clientstatus HTTP/1.1" 200 -
b'{"open_client":true,"ip_address":"192.168.1.36","port":502}'
b'{"open_client":false,"ip_address":"192.168.1.36","port":502}'
127.0.0.1 - - [12/Dec/2021 14:55:29] "POST /api/clientstatus HTTP/1.1" 200 -
127.0.0.1 - - [12/Dec/2021 14:55:29] "POST /api/clientstatus HTTP/1.1" 200 -
127.0.0.1 - - [12/Dec/2021 14:55:31] "GET /api/clientstatus HTTP/1.1" 200 -
127.0.0.1 - - [12/Dec/2021 14:55:34] "GET /api/clientstatus HTTP/1.1" 200 -
127.0.0.1 - - [12/Dec/2021 14:55:37] "GET /api/clientstatus HTTP/1.1" 200 -
127.0.0.1 - - [12/Dec/2021 14:55:38] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [12/Dec/2021 14:55:38] "GET /static/css/bootstrap.min.css HTTP/1.1" 200 -
127.0.0.1 - - [12/Dec/2021 14:55:38] "GET /static/css/style.css HTTP/1.1" 200 -
127.0.0.1 - - [12/Dec/2021 14:55:38] "GET /static/css/bootstrap-toggle.css HTTP/1.1" 200 -
127.0.0.1 - - [12/Dec/2021 14:55:38] "GET /static/js/bootstrap-toggle.js HTTP/1.1" 304 -
127.0.0.1 - - [12/Dec/2021 14:55:38] "GET /static/js/jquery.min.js HTTP/1.1" 304 -
127.0.0.1 - - [12/Dec/2021 14:55:38] "GET /static/js/bootstrap.min.js HTTP/1.1" 304 -
127.0.0.1 - - [12/Dec/2021 14:55:38] "GET /static/js/script.js HTTP/1.1" 304 -
127.0.0.1 - - [12/Dec/2021 14:55:38] "GET /static/images/loading32x32.gif HTTP/1.1" 304 -
127.0.0.1 - - [12/Dec/2021 14:55:38] "GET /static/images/loading16x16.gif HTTP/1.1" 304 -
127.0.0.1 - - [12/Dec/2021 14:55:38] "GET /api/clientstatus HTTP/1.1" 200 -
```

Şekil 4.10. ModFlask Master Cihaz Ekran Görüntüsü.

Program çalıştıktan sonra artık web arayüzü ile Modbus slave cihaz ile bağlantı gerçekleştirebilir. Bu arayüze erişmek için tarayıcının adres çubuğuna <http://127.0.0.1:8001/> adresi yazılır. Ardından Şekil 4.11'deki gibi ekrana gelen arayüzde slave cihazın IP adresi ve port numarasının girilmesi gerekir. Bu bilgiler girildikten sonra bağlantı sağlanabilmektedir. Slave cihazın IP adresi 192.168.5.4 olarak atanmıştır. Modbus/TCP port numarası olarak, varsayılan port numarası olan 502 kullanılmıştır.

Modbus Web Client

IP Address

19216854

Port Number

502

Connection

Off

Şekil 4.11. ModFlask Master Cihaz Web Arayüzü.

Master cihazda bu işlemler yapılırken slave cihaz tarafında da ModFlask programının çalıştırılması gerekmektedir. Aşağıdaki komut ile slave cihaz üzerinde program Şekil 4.12’de gösterildiği gibi çalıştırılmıştır.

- python ModbusTcpServerSYN.py --host 0.0.0.0

```

ModFlask\Modbus TCP Server>python ModbusTcpServerSYN.py --host 0.0.0.0
ModbusTcpServerSYN.py:4: SyntaxWarning: "is not" with a literal. Did you mean "!="?
  if sys.version_info.major is not 3:
2021-12-13 00:58:45,546 MainThread DEBUG context :90 setValues[3] 54:1
2021-12-13 00:58:45,546 MainThread DEBUG context :90 setValues[3] 33:1
2021-12-13 00:58:45,546 MainThread DEBUG context :90 setValues[3] 34:1
2021-12-13 00:58:45,546 MainThread DEBUG context :90 setValues[3] 62:1
2021-12-13 00:58:45,562 MainThread DEBUG context :90 setValues[3] 27:1
2021-12-13 00:58:45,562 MainThread DEBUG context :90 setValues[3] 90:1
2021-12-13 00:58:45,562 MainThread DEBUG context :90 setValues[3] 91:1
2021-12-13 00:58:45,562 MainThread DEBUG context :90 setValues[3] 3:1
2021-12-13 00:58:45,562 MainThread DEBUG context :90 setValues[3] 11:1
2021-12-13 00:58:45,562 MainThread DEBUG context :90 setValues[3] 53:1
2021-12-13 00:58:45,562 MainThread DEBUG context :90 setValues[3] 33:1

```

Şekil 4.12. ModFlask Slave Cihaz Çalıştırılması.

Master ve slave cihaz tarafında gerekli programlar çalıştırıldıktan sonra artık Modbus fonksiyonları gerçekleştirilebilir. Şekil 4.13’de Master cihazdaki Modbus işlevleri görülmektedir. Bu arayüzden Modbus okuma/yazma işlemlerini gerçekleştiren fonksiyonlar kullanılabilir.

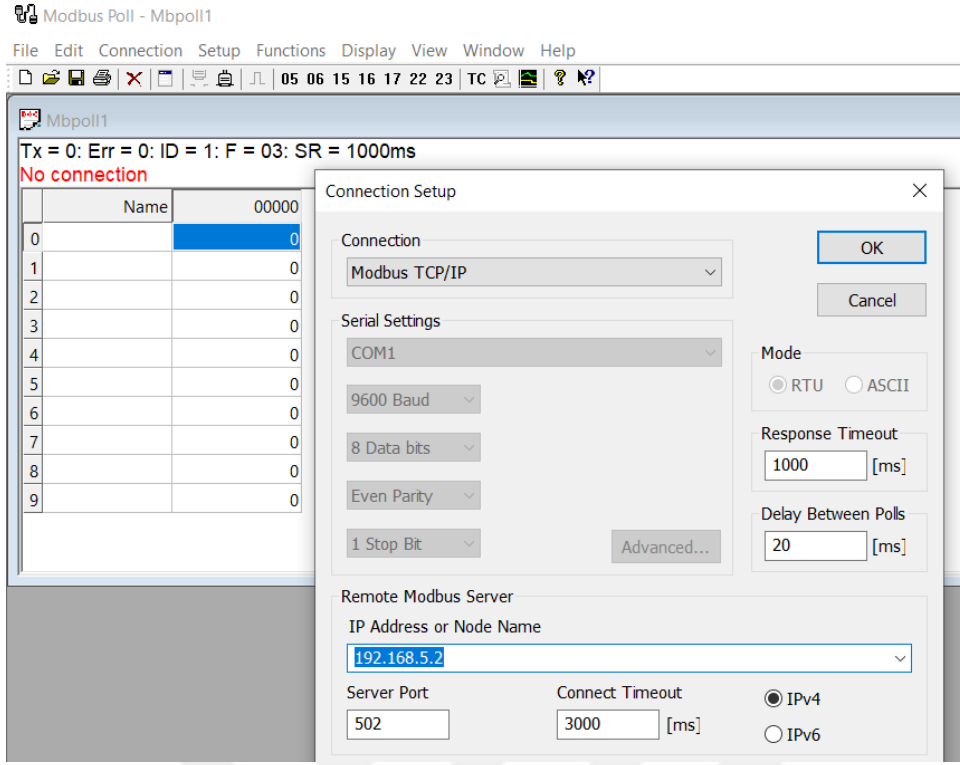
The screenshot displays the ModFlask Modbus Functions interface, which is organized into several sections for different Modbus operations:

- Operations:**
 - Write Single Coil (Discrete Outputs):** Includes a 'Coil Address' field (0) and a status indicator (1 green, 0 red).
 - Write Multiple Coils (Discrete Outputs):** Includes 'Coil Start Address' (0), 'Quantity of Coils' (10), and a 'Bits Array' (1111100000) with a 'Send' button.
- Read Coils (Discrete Outputs):** Includes 'Coil Start Address' (0), 'Quantity of Coils' (10), and a 'Receive' button.
- Write Single Register (Holding Registers):** Includes 'Register Address' (0) and 'Register Value' (65535) with a 'Send' button.
- Write Multiple Registers (Holding Registers):** Includes 'Register Start Address' (0), 'Quantity of Registers' (10), and a 'Registers Array' (0,0,0,0,0,0,0,0) with a 'Send' button.
- Read Registers (Holding Registers):** Includes 'Register Start Address' (0), 'Quantity of Registers' (10), and a 'Receive' button.
- Read Contacts (Discrete Inputs):** Includes 'Contact Start Address' (0), 'Quantity of Contacts' (10), and a 'Receive' button.
- Read Input Registers:** Includes 'Register Start Address' (0), 'Quantity of Registers' (10), and a 'Receive' button.

Şekil 4.13. ModFlask Modbus Fonksiyonları.

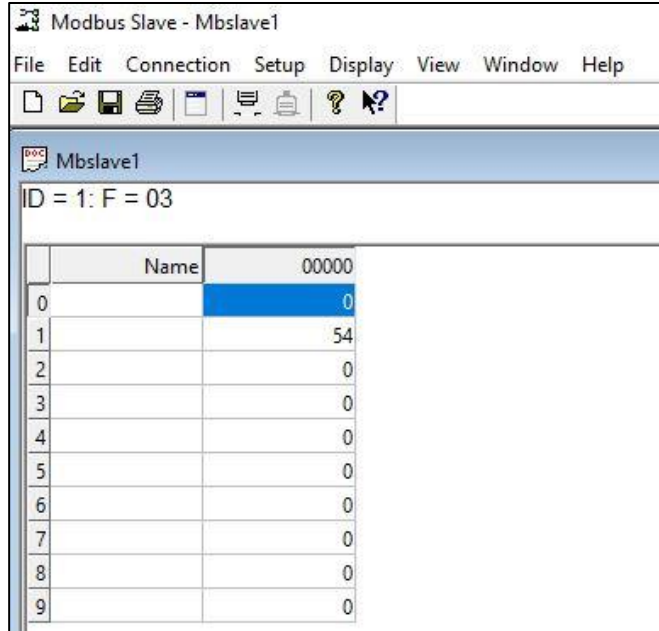
ModbusPoll programı ise grafik arayüzü olan masaüstü bir programdır. Master ve slave cihaz için farklı sürümleri vardır. Master cihaz için olan sürümün adı Modbus Poll’dür.

Şekil 4.14’de Modbus Poll programı üzerinden slave cihaza bağlantı ayarları gösterilmektedir. Burada yine slave cihazın IP adresi gerekmektedir.



Şekil 4.14. Modbus Poll Master Cihaz Arayüzü.

Slave cihaz üzerinde Modbus Slave adlı program çalıştırılır. Şekil 4.15’de slave cihazın ekran görüntüsü verilmiştir.



Şekil 4.15. Modbus Slave Cihaz Görünümü.

Master cihaz üzerinden slave cihaza bağlantı kurulduğunda Şekil 4.14’de gösterilen “No connection” ifadesi ortadan kalkar. Bağlantı kurulduktan sonra test amaçlı olarak

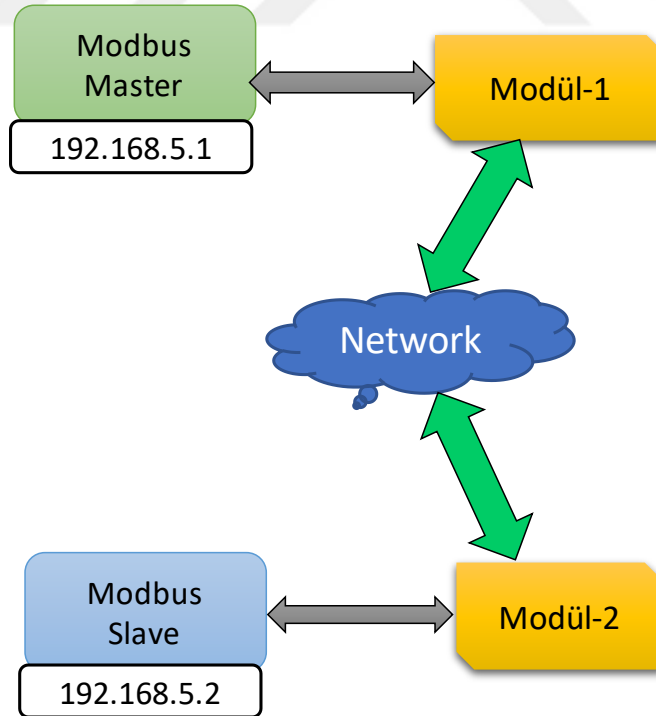
slave cihaza veri gönderimi yapılmıştır. Bunun için fonksiyon kodu 6 olan Write Single Register fonksiyonu kullanılmıştır. Yazmaç (register) adresi 1, değer olarak 54 girilmiştir.

4.3. Test Ortamının Oluşturulması

Modbus/TCP tabanlı bir endüstriyel sistemi simüle etmek için Modbus master ve Modbus slave cihazları gerekmektedir. Bu cihazların yerine benzetim programları kullanılarak aynı işlevsellik bilgisayarlara kazandırılmıştır. İstemci/sunucu mimarisiyle bilgisayarlar endüstriyel mesajlar ile haberleşmektedir. Modülün geliştirme aşamasında basit bir ağ topolojisi üzerine iki adet modül eklenerek çalışma yapılmıştır. Şifreleme aktif olmadığında paketler üzerinde herhangi bir işlem yapılmamıştır. Şifreleme aktif olduğunda ise paketler bir modülde şifrelenmiş diğer modülde ise şifresi çözülmüştür.

4.3.1. Geliştirme test ortamı

Şekil 4.16’da önerilen modül üzerinde geliştirme yapmak için kullanılan test ortamı verilmiştir.



Şekil 4.16. Geliştirme Test Ortamı.

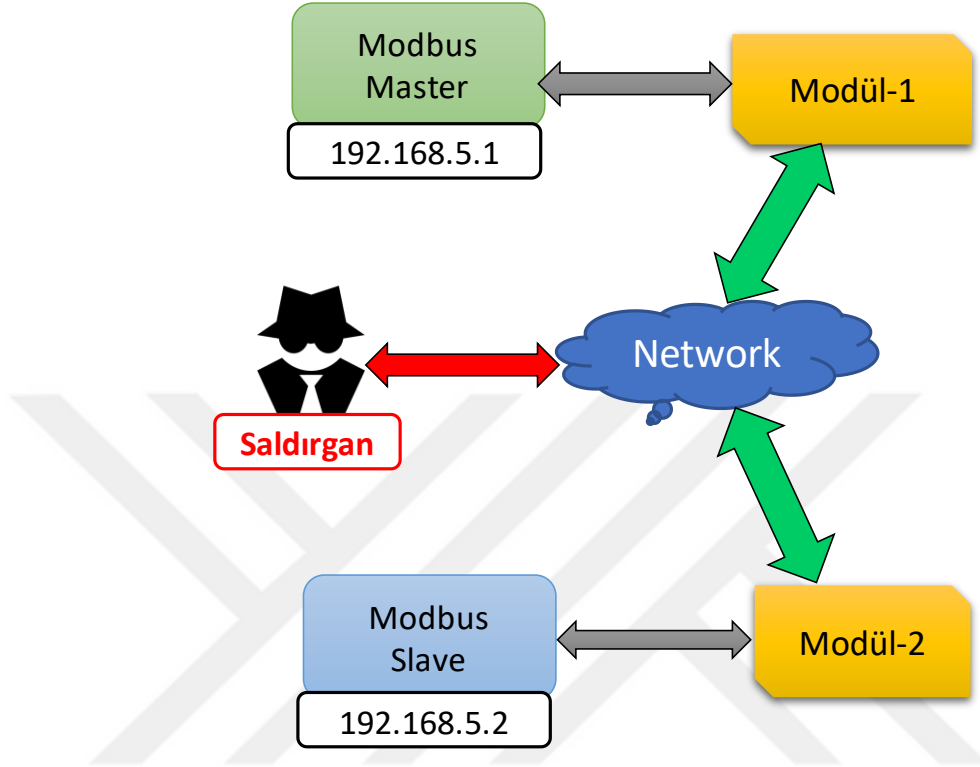
Modbus master ve slave cihazlarının arasındaki trafik üzerinde manipüle, şifreleme yapmak için önerilen modül cihazların önüne yerleştirilir. Modbus/TCP ağ trafiğini en basit şekilde ele almak için Şekil 4.16'daki gibi basit bir topoloji tasarlanmıştır. İki adet modbus yazılımı yüklü bilgisayar ve iki adet şifreleme modülünden oluşmaktadır. Burada amaç verilerin doğru bir şekilde şifrelenip şifrelenmediğini test etmektir. Cihazlar doğrudan birbirine bağlanabildiği gibi ağ cihazları üzerinden birbirine bağlanmıştır. Modbus Master cihazı olarak yapılandırılan bilgisayar 192.168.5.1 IP adresine sahiptir. Master cihazı ethernet portu üzerinden 1 numaralı modüle bağlanmıştır. Modül 1 ve modül 2 arasındaki bağlantı ağ cihazı (switch, bridge) üzerinden kurulmuştur. Modbus slave cihazı olarak yapılandırılan bilgisayar 192.168.5.2 IP adresine sahiptir. Slave cihazı ethernet arayüzü üzerinden 2 numaralı modüle bağlanmıştır.

Şekil 4.16'da yeşil renkli gösterilen ağ trafiği şifreleme modu aktif olduğunda, şifreli olarak iletilecek şekilde yapılandırma sağlanmıştır. Şifreleme modu aktif olmadığında bu aralıktaki ağ trafiği şifresiz iletilecektir. Gri renkli gösterilen ağ trafiği direkt olarak şifreleme modülüne bağlıdır ve şifresiz Modbus/TCP trafiğini temsil etmektedir. Modbus Master cihazından çıkan bir Modbus/TPC paketi modül-1'de şifrelenecek, şifreli olarak modül-2'ye gelecek ve burada şifresi çözülerek slave cihaza ulaşacaktır. Modbus Slave cihazından çıkan bir Modbus/TPC paketi ise modül-2'de şifrelenecek, şifreli olarak modül-1'ye gelecek ve burada şifresi çözülerek master cihaza ulaşacaktır. Bu şekilde çift yönlü olarak ağ trafiği şifreli olacak iletilecektir.

4.3.2. Saldırı test ortamı

Modbus/TCP paketi IP ağına bağlı olduğundan saldırılara açık durumdadır. Ağ ortamına içerden veya dışardan bağlantısı olan kötü niyetli bir kişi veya saldırgan, ağ ortamında bir güvenlik önlemi alınmamışsa ağ trafiğini dinleyebilir. Ayrıca Modbus/TCP paketleri şifresiz olarak iletildiğinden, iletilen veriler bu kötü niyetli kişiler tarafından açıkça görülebilmektedir. Önerilen şifreleme modülü sayesinde Modbus/TCP paketleri ele geçirilse bile şifreli olacağından dolayı veriler gizli kalmış olacaktır. Bu durumu test etmek amacıyla, geliştirme test ortamına bir saldırgan bilgisayar eklenmiştir. Saldırgan, ağa erişimi olan bir kötü niyetli operatör olabilir ya da uzaktan bağlantı yoluyla operatör bilgisayarına erişmiş birisi de olabilir. Kısacası ağa erişimi olan her bilgisayar üzerinden kötü niyetli bir saldırı gerçekleştirilebilir.

Şekil 4.17’de saldırı test ortamı topolojisi verilmiştir. Saldırgan ağ üzerindeki Modbus/TCP paketlerine ulaşabilmektedir.



Şekil 4.17. Saldırı test ortamı.

4.4. Şifreleme Algoritmalarının Gerçekleştirimi

Şifreleme algoritmalarının modül üzerinde uygulanabilmesi için nesne yönelimli programlama ilkelerine uygun olarak sınıf yapısı kullanılmıştır. Her bir algoritma için ayrı bir sınıf tasarlanmıştır. Ayrıca paketler üzerinde işlem yapabilmek için de paket sınıfı tasarlanmıştır. Tüm şifreleme algoritması sınıfları IAlgoritma soyut sınıfından kalıtım almaktadır. IAlgoritma soyut sınıfı, şifreleme ve şifre çözme işlemleri için birer methoda sahiptir. Bu metodlar parametre olarak bayt türünde veri alır ve bayt türünde veri döndürür. Bu metodlar, kalıtım alan sınıfta implement edilmek zorundadır.

Present Algoritması 64 bit blok boyutuna sahiptir ve bu uygulamada 80 bitlik anahtar seçilmiştir. Ascon algoritması için 64 bit blok boyutuna sahiptir, 128 bit nonce değeri ve 128 bit anahtar seçilmiştir. XTEA algoritması 64 bit blok boyutuna sahiptir ve 128 bit anahtar seçilmiştir. Skinny algoritması için 128 bit blok boyutu ve 128 bit anahtar

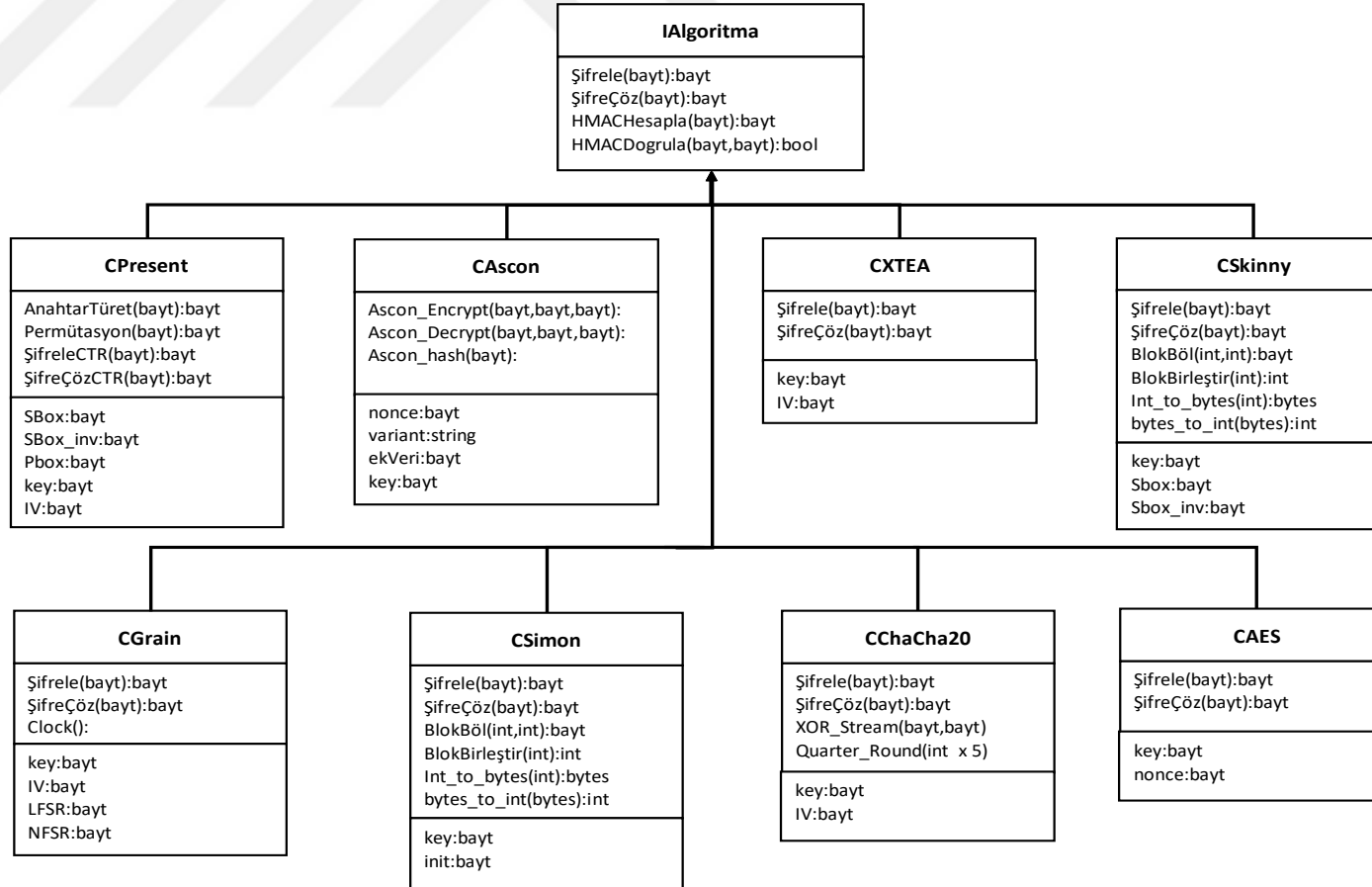
tercih edilmiştir. Grain algoritması için 128 bit anahtar ve 96 bit başlatma vektörü seçilmiştir. Grain algoritması akış şifrelemeye sahiptir ve blok şifreleme olmadığından dolayı blok boyutu bilgisi verilmemiştir. Simon algoritması için 64 bit blok boyutu ve 128 bit anahtar seçilmiştir. ChaCha20 algoritması için 256 bit anahtar ve 64 bit başlatma vektörü tercih edilmiştir. ChaCha20 algoritması da bir akış şifreleme algoritmasıdır ve blok boyutu bilgisi verilmemiştir. AES algoritması bir hafif sıklet şifreleme algoritması değildir ve karşılaştırma için bu çalışmaya eklenmiştir. AES algoritması için 128 bit nonce değeri, 256 bit anahtar ve 128 bit blok boyutuna sahiptir.

ChaCha20 ve Grain algoritmaları akış şifreleme olduğundan dolayı şifreleme kipiyle çalışmamaktadır. Bu çalışmada kullanılan diğer tüm blok şifreleme algoritmaları Counter Mode (CTR) şifreleme kipinde çalışmaktadır. CTR şifreleme kipi, blok şifreleme algoritmalarını akış şifreleme algoritmasıymış gibi çalıştırmaktadır. Böylelikle, performans ve uyumluluk açısından avantajlar sunmaktadır. CTR modunda, her blok bağımsız olarak şifrelenir. Bu durum, algoritmanın hızlı şekilde çalışmasını sağlamaktadır. Bir nonce değeri kullanılarak, her oturumda farklı bir başlangıç değeri seçilir ve bu durum güvenlik açısından önemlidir. CTR kipi doğrulama içermemekte olup yalnızca veri gizliliği sağlamaktadır.

Paketlerin tekrar gönderilip gönderilmediğini tespit etmek için şifrelenen her pakete algorithmadan bağımsız 8 baytlık bir nonce değeri eklenmiştir. Alıcı tarafta paketin şifresi çözülürken nonce değeri kontrol edilir, eğer daha önce bu değer gönderilmemişse paket kabul edilir ve hedefe gönderilir. Ancak, nonce değeri daha önce gönderilmişse bu bir saldırı olarak algılanır ve paket engellenir.

Veri bütünlüğünü doğrulamak için IAlgoritma sınıfında 8 baytlık bir HMAC kodu oluşturan fonksiyon tanımlanmıştır. Bir anahtar yardımıyla şifrelenen verinin tamamının doğrulamak için SHA256 algoritmasından yararlanılmıştır. Elde edilen HMAC kodu, ilgili Modbus/TCP paketinin sonuna eklenmiştir.

Şekil 4.18’de şifreleme algoritmaları sınıf diyagramı ile gösterilmiştir.



Şekil 4.18. Şifreleme Algoritmaları Sınıf Diyagramı Gösterimi.



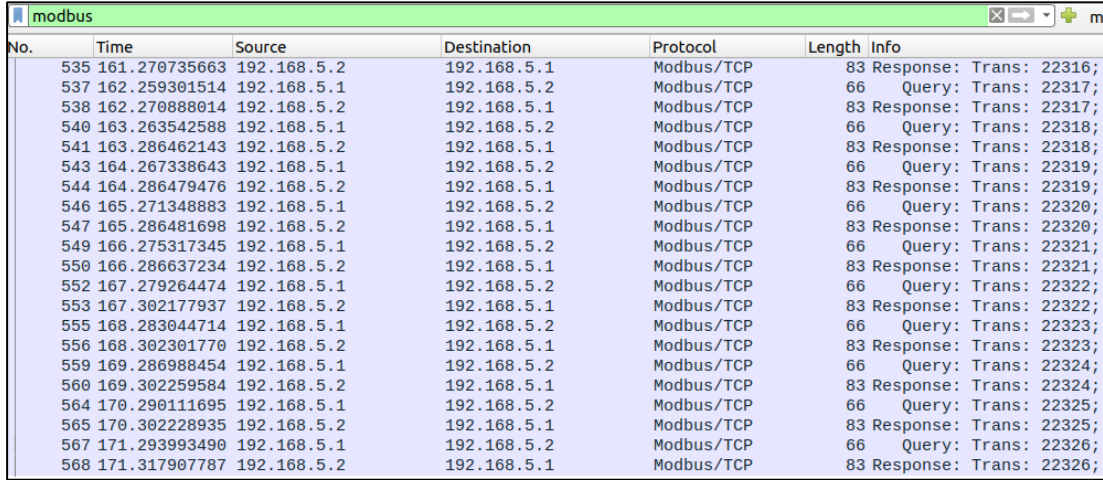
5. UYGULAMA VE DEĞERLENDİRME

Bu bölümde, önerilen yöntemin çalıştırılması ve buradan elde edilen sonuçların değerlendirilmesi anlatılmaktadır. Alt bölümlerde, modül üzerinde önerilen yöntemin çalıştırılması, Modbus/TCP paketlerinin izlenmesi, oluşturulması, değiştirilmesi, şifrelenmesi, şifreleme sürelerinin ölçülmesi, güvenlik analizi ve genel değerlendirme ayrıntılı olarak açıklanmıştır. Python programlama dili ile hafif sıklet şifreleme algoritmaları kullanılarak şifreleme ve şifre çözme işlemleri yapılmıştır. Her bir algoritmanın iki farklı senaryoda şifreleme ve şifre çözme işlem süreleri hesaplanmıştır. Hesaplanan süreler, tablo ve şekiller ile gösterilmiştir. Elde edilen bulgular, önerilen yöntemin güvenli endüstriyel veri haberleşmesi için uygulanabilir olduğunu göstermektedir.

5.1. Modbus/TCP Paketinin İzlenmesi

Modbus/TCP protokolü, en basit topolojide master ve slave cihaz arasında çalışabilmektedir. Gerçek bir kontrol sisteminde protokolü kullanan cihazların sayısı fazla olabilmektedir. Bu çalışmada, master ve slave cihaz arasında yerleştirilmiş olan Raspberry Pi 4 tabanlı modül ile Modbus/TCP paketleri tespit edilmiştir. Bu işlem için Wireshark gibi ağ izleme aracı kullanılabildiği gibi Python ortamında yazılan basit bir program da kullanılabilmektedir. Wireshark ortamında Modbus/TCP olan paketleri filtrelemek için “modbus” anahtar ifadesi yazılır. Raspberry Pi 4 tabanlı modül, bu durumda ağ TAP cihazı görevi görmektedir [88]. Ağ TAP cihazları, ağ üzerindeki paketleri izlemeye olanak sağlamaktadır.

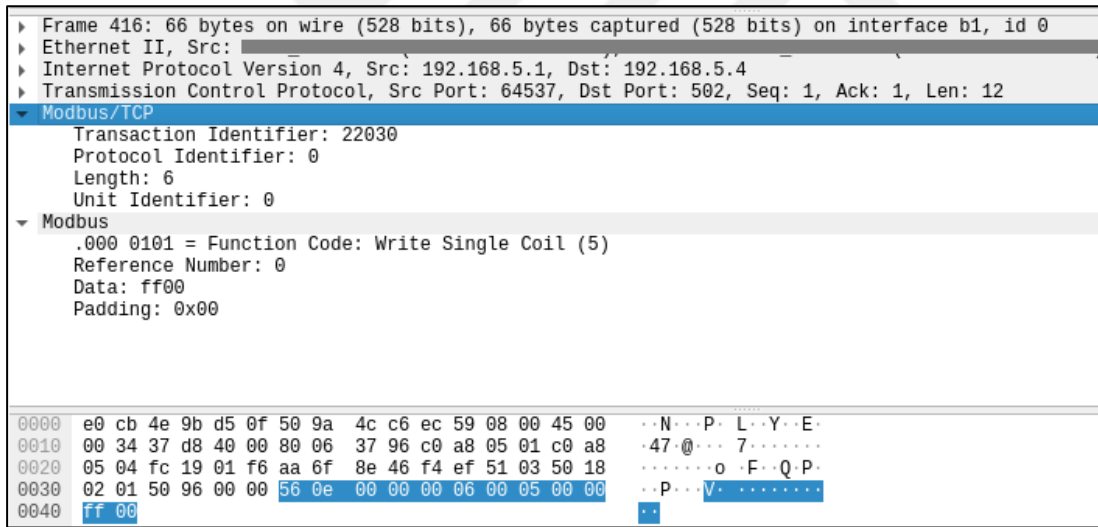
Şekil 5.1’de Wireshark ortamında filtrelenen Modbus/TCP paketleri gösterilmektedir.



No.	Time	Source	Destination	Protocol	Length	Info
535	161.270735663	192.168.5.2	192.168.5.1	Modbus/TCP	83	Response: Trans: 22316;
537	162.259301514	192.168.5.1	192.168.5.2	Modbus/TCP	66	Query: Trans: 22317;
538	162.270888014	192.168.5.2	192.168.5.1	Modbus/TCP	83	Response: Trans: 22317;
540	163.263542588	192.168.5.1	192.168.5.2	Modbus/TCP	66	Query: Trans: 22318;
541	163.286462143	192.168.5.2	192.168.5.1	Modbus/TCP	83	Response: Trans: 22318;
543	164.267338643	192.168.5.1	192.168.5.2	Modbus/TCP	66	Query: Trans: 22319;
544	164.286479476	192.168.5.2	192.168.5.1	Modbus/TCP	83	Response: Trans: 22319;
546	165.271348883	192.168.5.1	192.168.5.2	Modbus/TCP	66	Query: Trans: 22320;
547	165.286481698	192.168.5.2	192.168.5.1	Modbus/TCP	83	Response: Trans: 22320;
549	166.275317345	192.168.5.1	192.168.5.2	Modbus/TCP	66	Query: Trans: 22321;
550	166.286637234	192.168.5.2	192.168.5.1	Modbus/TCP	83	Response: Trans: 22321;
552	167.279264474	192.168.5.1	192.168.5.2	Modbus/TCP	66	Query: Trans: 22322;
553	167.302177937	192.168.5.2	192.168.5.1	Modbus/TCP	83	Response: Trans: 22322;
555	168.283044714	192.168.5.1	192.168.5.2	Modbus/TCP	66	Query: Trans: 22323;
556	168.302301770	192.168.5.2	192.168.5.1	Modbus/TCP	83	Response: Trans: 22323;
559	169.286988454	192.168.5.1	192.168.5.2	Modbus/TCP	66	Query: Trans: 22324;
560	169.302259584	192.168.5.2	192.168.5.1	Modbus/TCP	83	Response: Trans: 22324;
564	170.290111695	192.168.5.1	192.168.5.2	Modbus/TCP	66	Query: Trans: 22325;
565	170.302228935	192.168.5.2	192.168.5.1	Modbus/TCP	83	Response: Trans: 22325;
567	171.293993490	192.168.5.1	192.168.5.2	Modbus/TCP	66	Query: Trans: 22326;
568	171.317907787	192.168.5.2	192.168.5.1	Modbus/TCP	83	Response: Trans: 22326;

Şekil 5.1. Wireshark ile Modbus/TCP paketlerinin filtrelenmesi.

Şekil 5.2’de Wireshark ortamında yakalanmış bir Modbus/TCP paketi gösterilmektedir.



Frame 416: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface b1, id 0			
Ethernet II, Src: [redacted], Dst: [redacted]			
Internet Protocol Version 4, Src: 192.168.5.1, Dst: 192.168.5.2			
Transmission Control Protocol, Src Port: 64537, Dst Port: 502, Seq: 1, Ack: 1, Len: 12			
Modbus/TCP			
Transaction Identifier: 22030			
Protocol Identifier: 0			
Length: 6			
Unit Identifier: 0			
Modbus			
.000 0101 = Function Code: Write Single Coil (5)			
Reference Number: 0			
Data: ff00			
Padding: 0x00			
0000 e0 cb 4e 9b d5 0f 50 9a 4c c6 ec 59 08 00 45 00 ..N...P. L..Y..E.			
0010 00 34 37 d8 40 00 80 06 37 96 c0 a8 05 01 c0 a8 .47.@...7.....			
0020 05 04 fc 19 01 f6 aa 6f 8e 46 f4 ef 51 03 50 18 o.F..Q.P.			
0030 02 01 50 96 00 00 56 0e 00 00 00 06 00 05 00 00 ..P...V.....			
0040 ff 00 ..			

Şekil 5.2. Wireshark ortamında yakalanmış bir Modbus/TCP paketi.

Wireshark ortamında izlenen bu paketler daha sonradan analiz vb. işlemler için kullanılmak üzere kaydedilebilir. Şekil 5.2’de görüldüğü üzere yakalanan bu paket içerisindeki veriler açıkça okunabilmektedir. Modbus alanında bulunan bilgilerin fonksiyon kodu 5 ve verisi FF00 olarak gözükmemektedir. Bu bilgilerin kötü amaçlı kişilerin eline geçmesi durumunda sistemin işleyişini öğrenip, sisteme müdahale edebilir. Veriler açık halde iletiliğinden bu durum bir güvenlik riski oluşturmaktadır. Bu çalışmada ise belirtilen güvenlik riskine karşın verilerin şifreli olarak iletilmesi

gerçekleştirilmiştir. Paket yakalama işlemi Python ortamında da yapılmıştır. Şekil 5.3’de verilen kod parçasında Python kütüphaneleri olarak struct ve socket kullanılmaktadır.

```
import struct, socket

s = socket.socket(socket.PF_PACKET, socket.SOCK_RAW, socket.ntohs(3))

while True:
    try:
        pkt = s.recvfrom(2048)
        (src_port, dest_port) = struct.unpack('! H H', pkt[0][34:38])
        kaynak = format(src_port)
        hedef = format(dest_port)

        if kaynak == "502" or hedef == "502":
            print('Kaynak Port: {}, Hedef Port: {}'.format(src_port, dest_port),
                  <<<< Bir modbus paketi tespit edildi !")
            mb_fonksiyonKodu = struct.unpack('! B', pkt[0][61:62])
            kalan = pkt[0][62:]
            fmt = '!' + 'B'*len(kalan)
            mb_fonksiyonVeri = struct.unpack(fmt, kalan)
            print("Modbus Fonksiyon Kodu :", mb_fonksiyonKodu)
            print("Modbus Fonksiyon Verisi :", mb_fonksiyonVeri)
        else:
            print('Kaynak Port: {}, Hedef Port: {}'.format(src_port, dest_port))
```

Şekil 5.3. Python ortamında Modbus/TCP paketi yakalanması.

Python ortamında bir socket nesnesi oluşturularak, paketlerin kaynak ve hedef port numaraları tespit edilmiştir. Eğer bu port numaralarından herhangi bir tanesi 502 ise ilgili paketin bir Modbus/TCP paketi olduğu tespit edilmekte olup ekrana kaynak ve hedef port numaralarıyla birlikte Modbus paketi olduğu ve fonksiyonun bilgisi yazılmaktadır. Eğer port numaralarından herhangi birisi 502 değilse de yalnızca kaynak ve hedef port numaraları ekrana yazdırılmaktadır. Program çalıştırıldığında Şekil 5.4’deki gibi bir çıktı elde edilmektedir.

```
Kaynak Port: 64057, Hedef Port: 502 <<<< Bir modbus paketi tespit edildi !
Modbus Fonksiyon Kodu : (3,)
Modbus Fonksiyon Verisi : (0, 0, 0, 10)

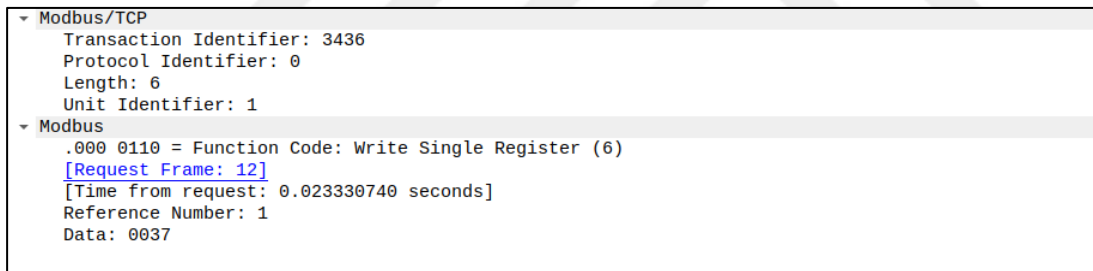
Kaynak Port: 502, Hedef Port: 64057 <<<< Bir modbus paketi tespit edildi !
Modbus Fonksiyon Kodu : (3,)
Modbus Fonksiyon Verisi : (20, 52, 159, 58, 220, 223, 142, 127, 2, 116, 30, 255,
197, 77, 167, 177, 199, 110, 167, 66, 127)
```

Şekil 5.4. Python ortamında yakalanan Modbus/TCP paketi.

5.2. Modbus/TCP Fonksiyonu Gerçekleştirilmesi

Modbus/TCP protokolünde fonksiyonlar; okuma ve yazma fonksiyonları olarak iki başlık altında öne çıkmaktadır. Modbus master cihazı, sürekli olarak slave cihazdan bilgi alışverişi yapmak için okuma fonksiyonlarını kullanır. Çalışmanın bu aşamasında, Modbus fonksiyon kodu 6 olan Write Single Register ve fonksiyon kodu 15 olan Write Multiple Coils fonksiyonları gerçekleştirilmiştir. Modbus/TCP fonksiyonları içeren paketin oluşturulabilmesi için IP Başlığı, TCP başlığı, Modbus başlığı ve Modbus veri alanlarının bilgileri gerekmektedir. IP başlığı, TCP başlığı ve Modbus başlığı içerisindeki bilgiler akış halinde olan bir Modbus/TCP trafiğinden elde edilebilir. Ancak Modbus veri alanındaki bilgilerin fonksiyon koduna göre oluşturulması gerekmektedir. Şekil 5.5'te görüldüğü gibi fonksiyon kodu 6 için gerekli olan bilgiler aşağıdaki gibidir.

- Function code
- Reference Number
- Data



Şekil 5.5. Wireshark ortamında yakalanmış fonksiyon kodu 6 olan bir paket.

Fonksiyon kodu 6 olan bir Modbus/TCP paketi oluşturmak için Python ortamında scapy kütüphanesi kullanılarak tanımlanan sınıf Şekil 5.6'da gösterilmiştir:

```
class Modbus(Packet):
    name = 'Modbus'
    fields_desc = [
        ShortField("transaction", int('0e6b'),16),
        ShortField("protocol", int('0000'),16),
        ShortField("len", int('6'),16),
        XByteField("unit", int('1'),16),
        XByteField("func", int('6'),16),
        ShortField("refNumber", int('1'),16),
        ShortField("Data", int('0011'),16) ]
```

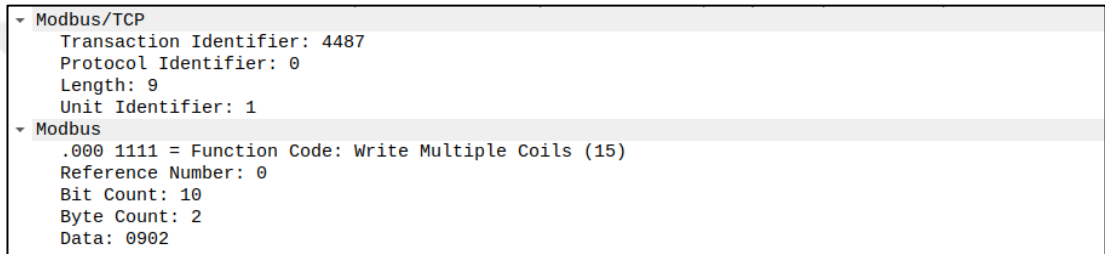
Şekil 5.6. Fonksiyon kodu 6 olan Modbus/TCP paketi tanımlanması.

Yapılan tanımlamada ShortField terimi 2 bayt, XByteField terimi ise 1 baytlık veriyi ifade etmektedir. Paketin tamamını oluşturmak için scapy kütüphanesinden yararlanılarak Şekil 5.7’deki gibi tanımlama yapılır:

```
paket_f6 = Ether(src_mac, dst_mac) \ IP(src_ip, dst_ip, len, id, flags, ttl, frag) \
TCP(sport, dport, seq, ack, flags, window, urgptr) \ Modbus()
```

Şekil 5.7. Fonksiyon 6 Modbus/TCP paketinin kapsüllenmesi.

Paketi oluştururken gereken diğer başlık alanlarının verileri iletişim halindeki paketlerden okunarak elde edilebilir. Fonksiyon kodu 15 olan Write Multiple Coils fonksiyonu gerçekleştirmek için gereken bilgiler Şekil 5.8’de gösterilmektedir.



Şekil 5.8. Wireshark ortamında yakalanmış fonksiyon kodu 15 olan bir paket.

- Function Code
- Bit Count
- Byte Count
- Data

Fonksiyon kodu 15 olan bir Modbus/TCP paketi oluşturmak için Python ortamında scapy kütüphanesi kullanılarak tanımlanan sınıf Şekil 5.9’da gösterilmiştir.

```
class Modbus(Packet):
    name = 'Modbus'
    fields_desc = [
        ShortField("transaction", int('4487'),16),
        ShortField("protocol", int('0000'),16),
        ShortField("len", int('9'),16),
        XByteField("unit", int('1'),16),
        XByteField("func", int('15'),16),
        ShortField("refNumber", int('0'),16),
        ShortField("bitCount", int('10'),16),
        ByteField("byteCount", int('2'),16),
        ShortField("Data", int('0EC0'),16)]
```

Şekil 5.9. Fonksiyon kodu 15 olan Modbus/TCP paketi tanımlanması.

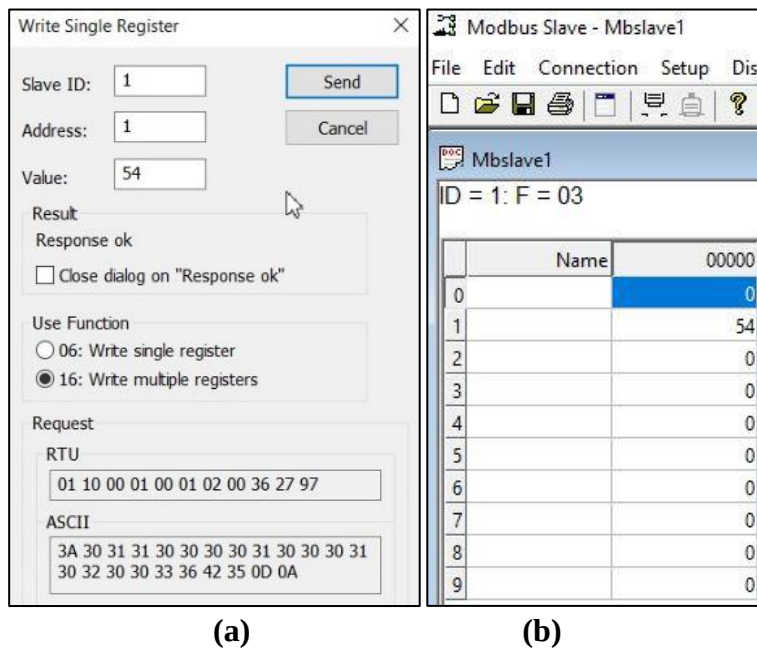
Yapılan tanımlamada ByteField terimi ise 1 baytlık veriyi ifade etmektedir. Paketin tamamını oluşturmak için scapy kütüphanesinden yararlanılarak Şekil 5.10'daki gibi tanımlama yapılır. Paket oluşturulduktan sonra ilgili ethernet arayüzünden gönderilir.

```
paket_f15 = Ether(src_mac, dst_mac) \ IP(src_ip, dst_ip, len, id, flags, ttl, frag) \
TCP(sport, dport, seq, ack, flags, window, urgptr) \ Modbus()
```

Şekil 5.10. Fonksiyon 15 Modbus/TCP paketinin kapsüllemesi.

5.3. Modbus/TCP Paketinin Manipüle Edilmesi

Çalışmanın bu aşamasında Modbus master ve slave cihazların arasındaki iletişime müdahale edilmesi anlatılmıştır. Modbus fonksiyon kodları 6 ve 15 olan fonksiyonlar üzerinde değişiklik yapılmıştır. Öncelikle Modbus master cihazından yollanan istek slave cihaz tarafından doğru alınıp alınmadığı test edilmiştir. Sonrasında ise paket yollanmak istediğinde aradaki Raspberry Pi 4 tabanlı modül ile iletilen paketin bilgileri değiştirilip hedefe yönlendirilmiştir. Ancak hedefe ulaşan değerlerin, master cihaz tarafından yollanan değerler değil, modül üzerinde değişiklik yapılan değerler olduğu görülmüştür. Bu durum Modbus/TCP protokolünde güvenlik açısından başarıyla yararlanıldığını göstermektedir. İlk olarak, Modbus master cihazında Şekil 5.11a'da gösterildiği gibi Write Single Register fonksiyonunda 1 numaralı adresteki değerin 54 olarak güncellenmesi istenmektedir. Bu komut slave cihaza başarıyla gönderilmiş olup iletişimin testi Şekil 5.11b'de görüldüğü üzere başarıyla tamamlanmıştır.



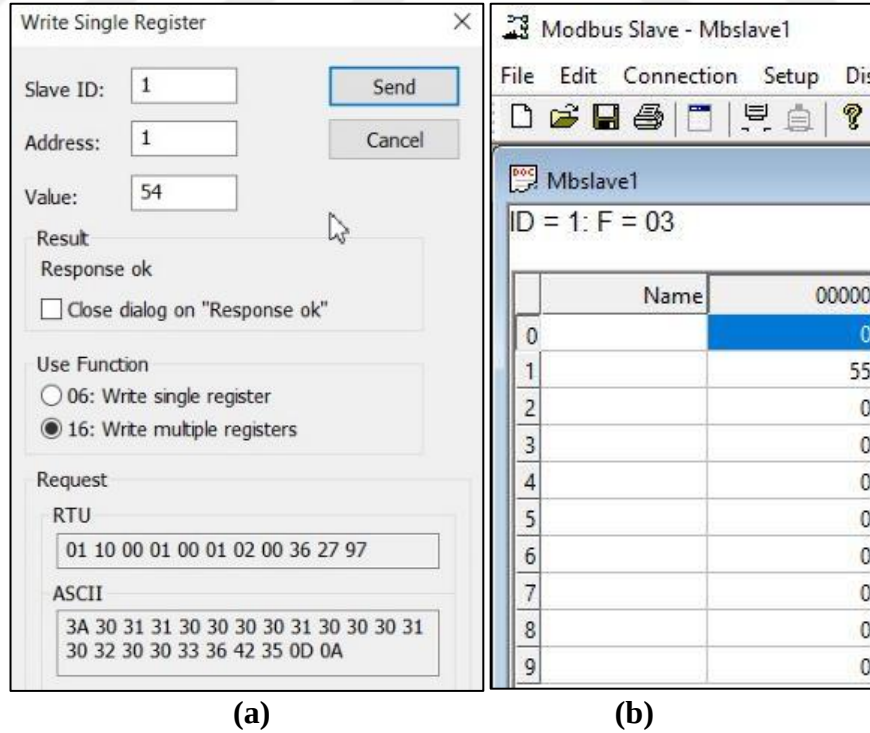
Şekil 5.11. Modbus master cihazından yollanan istek (a) ve slave taraftaki sonuç (b).

Ardından Modbus master cihazından çıkan paket, modül üzerinde yakalanıp üzerinde değişiklik yapılmıştır. Burada iletilmek istenen veri 54 = 0x36 değerine eşittir. Bu değer 0x37 ile değiştirilip slave cihazda 55 olarak okunacaktır. Paket yakalandığında üzerindeki değişiklik için Şekil 5.12'deki gibi tanımlama yapılmıştır:

```
class Modbus(Packet):
    name = 'Modbus'
    fields_desc = [
        ShortField("transaction", int('0e6b'),16),
        ShortField("protocol", int('0000'),16),
        ShortField("len", int('6'),16),
        XByteField("unit", int('1'),16),
        XByteField("func", int('6'),16),
        ShortField("refNumber", int('1'),16),
        ShortField("Data", int('0037'),16) ]
```

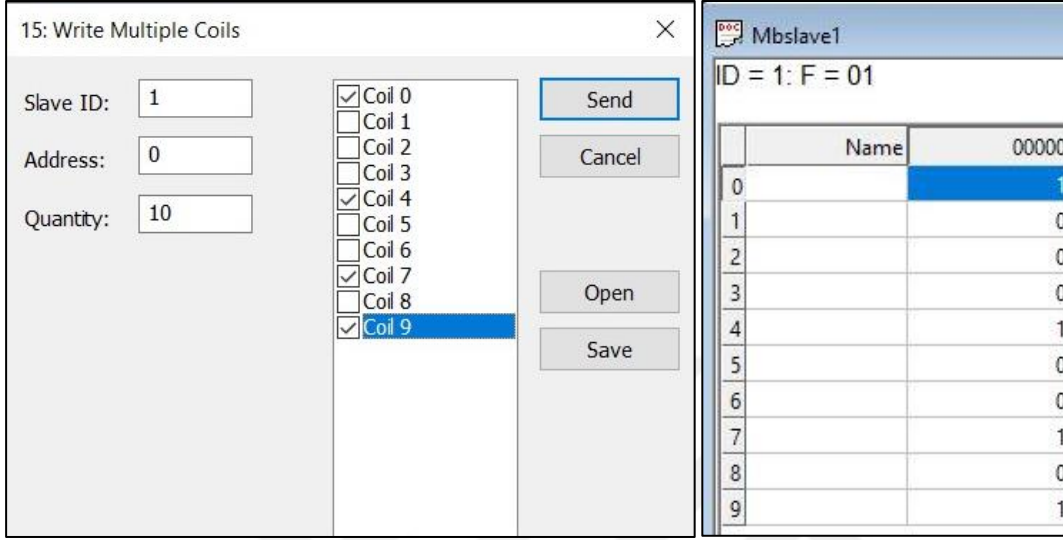
Şekil 5.12. Register değeri manipüle edilmiş Modbus/TCP paket yapısı.

Şekil 5.13a'da master cihaz tarafından gönderilen istek gösterilmektedir. Gönderilen istek paketindeki veri 54 olup modül üzerinde değişikliğe uğramış ve slave cihazda Şekil 5.13b'de gösterildiği gibi 55 olarak okunmuştur. Böylelikle Modbus/TCP fonksiyon kodu 6 olan Write Single Register verisine açıkça müdahale edilmiştir.



Şekil 5.13. Modbus master cihazından yollanan istek (a) ve slave taraftaki değiştirilmiş sonuç (b).

İkinci olarak, Modbus master cihazında Şekil 5.14a’da gösterildiği gibi Write Multiple Coils fonksiyonunda 0,4,7,9 numaralı Coil değerlerinin aktif (1) olması istenmektedir. Bu komut slave cihaza başarıyla gönderilmiş olup iletişimin testi Şekil 5.14b’de görüldüğü üzere başarıyla tamamlanmıştır.



(a)

(b)

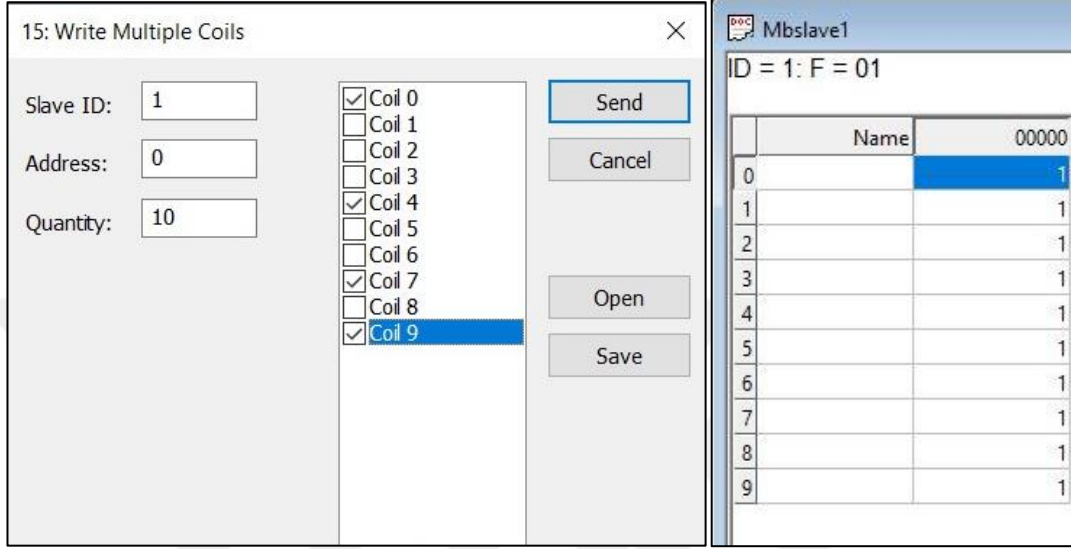
Şekil 5.14. Modbus master cihazından yollanan istek (a) ve slave taraftaki sonuç (b).

Sonrasında, Modbus master üzerinden paketin içeriğinde modül üzerinde değişiklik yapılmıştır. Bu değişiklik neticesinde tüm Coil değerlerinin aktif olması istenmiştir. Burada iletilmek istenen fonksiyon verisi 0x9102 değerine eşit olup Coil 0,4,7,9 değerleri aktif etmektedir. Paket yakalandığında bu değer 0xFFFF değeriyle değiştirilip tüm coil değerlerinin aktif olması sağlanmıştır. Değişiklik için Şekil 5.15’deki gibi paket üzerinde tanımlama yapılmıştır:

```
class Modbus(Packet):
    name = 'Modbus'
    fields_desc = [
        ShortField("transaction", int('4487'),16),
        ShortField("protocol", int('0000'),16),
        ShortField("len", int('9'),16),
        XByteField("unit", int('1'),16),
        XByteField("func", int('15'),16),
        ShortField("refNumber", int('0'),16),
        ShortField("bitCount", int('10'),16),
        ByteField("byteCount", int('2'),16),
        ShortField("Data", int('FFFF'),16) ]
```

Şekil 5.15. Coil değerleri manipüle edilmiş Modbus/TCP paket yapısı.

Şekil 5.16a’da master cihaz tarafından gönderilen istek gösterilmektedir. Gönderilen istek paketindeki aktif edilmek istenen değerler Coil 0,4,7,9 olup modül üzerinde değişikliğe uğramış ve slave cihazda Şekil 5.16b’de gösterildiği gibi tüm Coil değerleri aktif edilmiştir. Böylelikle Modbus/TCP fonksiyon kodu 15 olan Write Multiple Coils verilerine açıkça müdahale edilmiştir.



(a)

(b)

Şekil 5.16. Modbus master cihazından yollanan istek (a) ve slave taraftaki değiştirilmiş sonuç (b).

Bu iki farklı senaryo, Modbus/TCP protokolünün nasıl manipüle edildiğini açıkça göstermektedir. Bu durum, iletişim ortamındaki kötü niyetli kişiler tarafından istenmeyen mesajların hedefe gönderilmesinin mümkün olduğunu ve büyük bir güvenlik riski olduğunu göstermektedir. Bu güvenlik açığına karşın, bu çalışmada Modbus/TCP protokol verilerinin şifrelenmesi önerilmektedir.

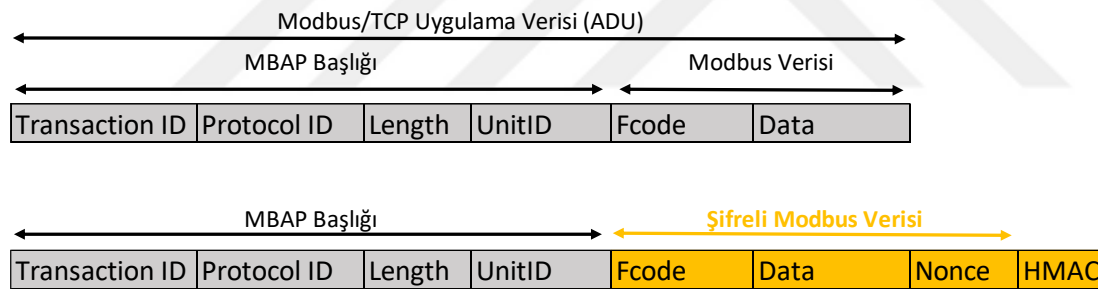
5.4. Modbus/TCP Paketinin Şifrelenmesi

Modbus/TCP verilerinin değiştirilip sistem üzerindeki etkisi gösterilmiştir. Verileri güvenli bir şekilde hedefe iletmek için bu aşamada iki farklı seçenek önerilmiş ve gerçekleştirilmiştir. Birinci seçenek yalnızca Modbus/TCP protokol verilerini şifrelemek ikinci seçenek ise Modbus/TCP başlık ve veri alanlarını şifrelemektir. Şifreleme işlemlerini gerçekleştirmek için sekiz farklı algoritma (Present, XTEA, Skinny, Grain, Simon, ChaCha20, Ascon, AES) kullanılmıştır. Bu algoritmalar hafif sıklet algoritmalar olduğu için tercih edilmiştir. AES algoritması bir standart olup

karşılaştırma için bu çalışmaya dahil edilmiştir. Modbus/TCP verileri şifrelendikten sonra gönderilen paketin kötü niyetli kişilerin eline geçmesi durumunda veriler şifreli olduğu için bir anlam ifade etmeyecektir. Ayrıca saldırganlar tarafından şifreli bir paket gönderilmesine karşın paketlere HMAC kodu eklenmiştir. HMAC kodu doğrulanmayan paketler engellenmiştir. Şifreleme işlemi yapılırken, paket uzunluğu, IP Checksum ve TCP Checksum değerleri değiştiği için yeniden hesaplanmıştır.

5.4.1. Modbus PDU şifrelenmesi

Bu senaryoda, Modbus/TCP protokolünün başlık alanından sonra gelen veri alanı şifrelemeye tabi tutulacaktır. Bu alan değişken uzunlukta olduğundan başlık verisinin bittiği indeksten son indekse kadar olan veriler işleme alınmaktadır. Şekil 5.17’de şifrelenen alanlar gösterilmektedir. Modbus MBAP başlığındaki Transaction ID, Protocol ID, Length, UnitID alanları şifrelenmeyip Modbus verisini içeren Fcode ve Data alanları şifrelenmiştir. Nonce değeri paketin tekrar gönderilip gönderilmediğini tespit etmek için eklenmiştir. Şifrelenen alanları doğrulamak için paketin sonuna bir HMAC kodu eklenmiştir.



Şekil 5.17. Modbus veri alanının şifrelenmesi.

Modbus master cihazından slave cihaza doğru yönlendirilen şifrelenmemiş bir Modbus/TCP paketi Şekil 5.18’de gösterilmiştir:

0000	45 00 00 49 26 3C 40 00 80 06 49 1F C0 A8 05 02	E..I&<@...I....
0010	C0 A8 05 01 FE 95 01 F6 93 0A 48 5F 59 BB 86 EA	H_Y...
0020	50 18 04 01 48 80 00 00 DD FB 00 00 00 1B 01 10	P...H.....
0030	00 00 00 0A 14 00 00 01 00 01 00 01 00 01 00	
0040	00 03 12 03 15 00 00 00 00	

Şekil 5.18. PDU alanı şifrelenmemiş Modbus/TCP paketi.

Şekil 5.18’deki bu paketin Simon algoritması ile Modbus PDU (veri alanı) şifrelenmiş hali Şekil 5.19’da verilmiştir.

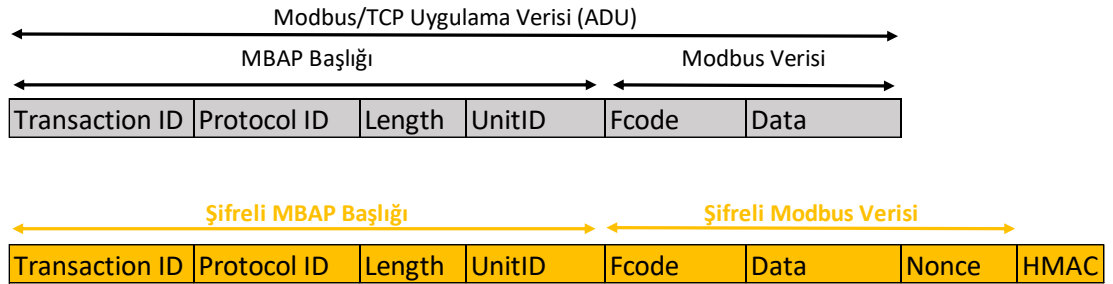
0000	45 00 00 59 26 12 40 00 80 06 49 39 C0 A8 05 02	E..Y&.@...I9....
0010	C0 A8 05 01 FE 90 01 F6 AC AD 96 26 1E DD 7D DF	&..}.
0020	50 18 04 01 EC CB 00 00 DD D7 00 00 00 1B 01 8B	P.....
0030	B8 A3 CD 3F 68 65 BA CB 59 F5 36 14 42 0F DD C6	...?he..Y.6.B...
0040	29 6D DD 4A 40 C7 CE 10 8C C0 F1 47 80 2B 57 21	)m.J@.....G.+W!
0050	09 A3 B5 50 93 FB 88 1B D5	

Şekil 5.19. PDU alanı şifrelenmiş Modbus/TCP paketi.

Sarı renkli gösterilen alanlar şifrelenmiş verileri ifade etmektedir. Burada paketin uzunluğunun arttığı görülmüştür. Bunun sebebi blok şifrelemenin blok boyutunda şifreli çıktı üretmesi, nonce değerinin eklenmesi ve doğrulama için HMAC etiketinin pakete eklenmesidir.

5.4.2. Modbus ADU şifrelenmesi

Bu senaryoda ise Modbus/TCP protokolünün başlık ve veri alanlarının tamamı şifrelemeye tabi tutulacaktır. Şifrelemeye tabi tutulan bu alan değişken uzunlukta olduğundan TCP başlığının bittiği indeksten son indekse kadar olan veriler işleme alınmaktadır. Şekil 5.20’de şifrelenen alanlar gösterilmektedir. Modbus MBAP başlığındaki Transaction ID, Protocol ID, Length, UnitID alanları ve Modbus verisini içeren Fcode ve Data alanlarının tamamı şifrelenmiştir. Nonce değeri paketin tekrar gönderilip gönderilmediğini tespit etmek için eklenmiştir. Şifrelenen alanları doğrulamak için paketin sonuna önceki senaryoda olduğu gibi bir HMAC kodu eklenmiştir.



Şekil 5.20. Modbus başlık ve veri alanlarının şifrelenmesi.

Modbus master cihazından slave cihaza doğru yönlendirilen şifrelenmemiş bir Modbus/TCP paketi Şekil 5.21’de gösterilmiştir.

0000	45 00 00 37 25 14 40 00 80 06 4A 59 C0 A8 05 02	E..7%.@...JY....
0010	C0 A8 05 01 FD 78 01 F6 D5 2F F0 5A 52 6A 18 41	x.../.ZRj.A
0020	50 18 04 02 ED 13 00 00 00 01 00 00 00 09 01 0F	P.....
0030	00 00 00 0A 02 8C 00	

Şekil 5.21. ADU alanı şifrelenmemiş Modbus/TCP paketi.

Bu paketin Present algoritması ile Modbus ADU (başlık ve veri alanları) şifrelenmiş hali Şekil 5.22’de verilmiştir.

0000	45 00 00 47 25 14 40 00 80 06 4A 49 C0 A8 05 02	E..G%.@...JI....
0010	C0 A8 05 01 FD 78 01 F6 D5 2F F0 5A 52 6A 18 41	x.../.ZRj.A
0020	50 18 04 02 D0 9F 00 00 ED 5E 65 0A C1 E0 B0 8D	P.....^e.....
0030	D2 21 25 D8 8A 35 24 FA A0 38 EA 16 84 9C 60 DC	.!%..5\$.8....`.
0040	0D FF E9 33 48 16 05	

Şekil 5.22. ADU alanı şifrelenmiş Modbus/TCP paketi.

Sarı renkli gösterilen alanlar yine şifrelenmiş verileri ifade etmektedir. Burada bir önceki senaryoda olduğu gibi paketin uzunluğunun arttığı görülmüştür. Bunun sebebi de blok şifrelemeden dolayıdır. Böylelikle Modbus/TCP paketleri, şifreli olarak iletişim sağlamaya başlamıştır. Saldırganların paketi ele geçirmesi durumunda verilerin gizliliği korunmuş olacaktır.

5.5. Şifreleme Sürelerinin Ölçülmesi

Paketlerin şifrelenmesi, gizlilik parametresini sağlamaktadır. Ancak sistemin işleyişini bozmaması için en kısa sürede şifreleme işlemlerinin yapılması gerekmektedir. Bu durumu test etmek için şifreleme ve şifre çözme süreleri ölçülmüştür. Test için Modbus yazma fonksiyonları olan Write Multiple Coils (15) ve Write Multiple Registers (16) kullanılmıştır. Her iki fonksiyonun verilerini içeren 100 paket rastgele oluşturulup master cihazdan slave cihaza doğru gönderilmiştir. Burada, geliştirme test ortamında verilen topoloji kullanılmıştır. Modbus slave cihazının tarafına Raspberry Pi 4 8 GB ve Modbus master cihazının tarafına Orange Pi R1 LTS Plus cihazı bağlanmış olup üzerinde ayrı ayrı süre ölçümleri yapılmıştır.

Şekil 5.23’de Write Multiple Coils fonksiyonuna ait 10 adet coil değeri içeren 52 baytlık (ethernet başlığı hariç) bir Modbus/TCP paketi verilmiştir. Gönderilen tüm paketler aynı boyutta olup coils içerikleri rastgele oluşturulmuştur.

0000:	45 00 00 34 22 A7 40 00 80 06 4C C9 C0 A8 05 01	E..4".@...L.....
0010:	C0 A8 05 02 01 F6 DA 13 FE A3 1F A7 AC 12 E1 C3	
0020:	50 18 01 00 9A 17 00 00 00 0B 00 00 06 01 0F	P.....
0030:	00 00 00 0A	

Şekil 5.23. 10 adet coil değeri içeren Modbus/TCP paketi.

Aynı şekilde Write Multiple Registers fonksiyonuna ait 10 adet register değeri içeren 73 baytlık (ethernet başlığı hariç) bir Modbus/TCP paketi Şekil 5.24’de verilmiştir. Gönderilen tüm paketler aynı olup register değerleri rastgele oluşturulmuştur.

0000:	45 00 00 49 52 79 40 00 80 06 00 00 C0 A8 05 02	E..IRy@.....
0010:	C0 A8 05 01 DA 65 01 F6 82 93 F9 B3 9C BF CD 83	e.....
0020:	50 18 04 01 8B 8F 00 00 00 1A 00 00 00 1B 01 10	P.....
0030:	00 00 00 0A 14 28 09 AD DB 92 87 67 F9 0C 6C AD	(.....g..l.
0040:	C9 77 81 48 B7 DC 91 6F 93	

Şekil 5.24. 10 adet register değeri içeren Modbus/TCP paketi.

Aynı anda 100 adet register için veri alışverişi yapmak üzere Write Multiple Registers fonksiyonu Şekil 5.25’de gösterildiği gibi oluşturulmuş ve 241 baytlık (ethernet başlığı hariç) bir Modbus/TCP paketi elde edilmiştir.

Oluşturulan 3 farklı paket yapısı ile önerilen yöntemin performansı değerlendirilmiştir.

0000:	45 00 00 F1 EB D8 40 00 80 06 82 DA C0 A8 05 02	E.....@.....
0010:	C0 A8 05 01 EB AF 01 F6 C5 5A 81 5F E3 59 61 03	Z.._Ya.
0020:	50 18 03 FE 31 C5 00 00 00 5C 00 00 00 C3 01 0F	P...1....\.....
0030:	00 00 05 DC BC C0 C3 4E 7C 80 CC D3 35 72 46 FA	N ...5rF.
0040:	2A 57 CA C3 08 C9 AD B0 AA FD CD 20 06 12 E6 54	*W.....T
0050:	47 74 E1 61 6C D5 FB BB 28 3F 5F 78 E3 A5 6C 70	Gt.al...(?_x..lp
0060:	AD 72 10 A1 1E 45 9E 1E D9 C8 A1 A6 69 ED F2 B7	.r...E.....i...
0070:	C0 EA A4 57 33 18 4B E9 BF 62 82 D8 3D 74 A8 3E	...w3.K..b.=t.>
0080:	0A C7 A6 58 EC 76 39 FC 69 CF 9C 0A 45 D6 DD B6	...X.v9.i...E...
0090:	8A EC B3 12 C5 61 42 6A 06 8F 38 8E AC D7 24 38	aBj..8...\$8
00a0:	29 04 51 90 0A 04 91 81 3F 6E A9 AF 7A E8 21 54	).Q.....?n..z.!T
00b0:	D2 4D 10 F9 E4 59 E6 18 FE 46 48 44 69 E6 DB 97	.M...Y...FHDi...
00c0:	46 20 6D EB 15 5E D4 73 67 04 10 12 41 D8 1A 71	F m..^.sg...A..q
00d0:	58 E4 EE B1 5C 7A 61 83 59 CC 72 40 9B 2F 5D CB	X...\za.Y.r@./].
00e0:	CC DA 38 BA 9A EA A1 41 D4 3C 9F EC 8C 21 A5 37	..8....A.<...!.7
00f0:	00	

Şekil 5.25. 100 adet register değeri içeren Modbus/TCP paketi.

5.5.1. Raspberry Pi 4 tabanlı modül

Paketlerin şifrlenmesi iki farklı senaryoda test edilmiştir. İlk senaryoda yalnızca Modbus veri alanı şifrlenmiştir. Farklı Modbus fonksiyonları (15, 16) kullanılarak süre ölçümleri yapılmıştır. Bunun için Python ortamında bir betik yazılarak Modbus

fonksiyon verilerini içeren 100 adet paket kaynaktan hedefe gönderilmiştir. Elde edilen hesaplama süreleri, farklı algoritmaların performanslarını karşılaştırmak için kullanılmaktadır.

Tablo 5.1’de veri alanını içeren Modbus PDU için algoritma bazında ortalama hesaplama süreleri verilmiştir. Fonksiyon kodu 15 olan Write Multiple Coils fonksiyonu için en düşük ortalama şifreleme ve şifre çözme süresi Grain algoritmasıyla elde edilmiştir. Aynı fonksiyon için en yüksek süreler Present algoritmasıyla elde edilmiştir. Fonksiyon kodu 16 olan Write Multiple Registers fonksiyonu için en düşük şifreleme ve şifre çözme süreleri Simon, Grain ve AES algoritmalarına aittir. En yüksek hesaplama süresi ise Present algoritmasına aittir.

Tablo 5.1. Raspberry Pi 4 üzerinde Modbus PDU alanı işlem süreleri (s).

Algoritma	Senaryo-1 Write Multiple Coils (10 adet coil)		Senaryo-2 Write Multiple Registers (10 adet register)		Senaryo-3 Write Multiple Registers (100 adet register)	
	Şifreleme	Şifre Çözme	Şifreleme	Şifre Çözme	Şifreleme	Şifre Çözme
Present	0.002971	0.002779	0.005642	0.013344	0.004634	0.058730
XTEA	0.000676	0.000645	0.000831	0.001416	0.000964	0.005947
AES	0.000573	0.000536	0.000833	0.000764	0.000914	0.000806
Ascon	0.001720	0.001773	0.001564	0.002305	0.001865	0.006579
Grain	0.000281	0.000416	0.000569	0.001407	0.000709	0.008949
Skinny	0.002641	0.002646	0.002570	0.007286	0.002702	0.035163
ChaCha20	0.001186	0.001215	0.001234	0.001279	0.001543	0.004327
Simon	0.000458	0.000468	0.000741	0.001696	0.000934	0.008914

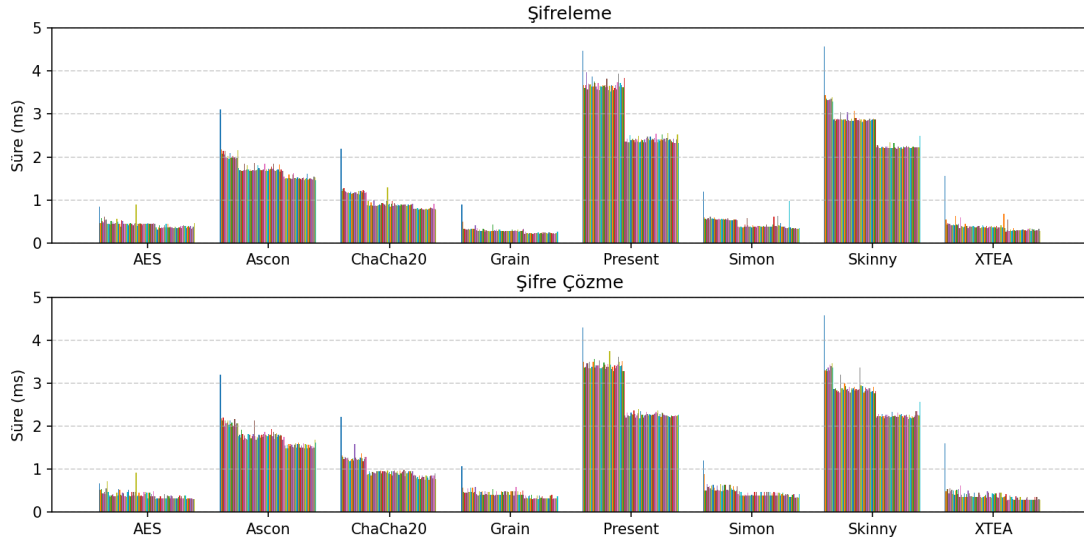
Tablo 5.2’de başlık ve veri alanını içeren Modbus ADU için hesaplama süreleri verilmiştir. Burada paket boyutunda başlık alanından dolayı 7 baytlık sabit bir artış olmuştur. Fonksiyon kodu 15 olan Write Multiple Coils fonksiyonu için en düşük ortalama şifreleme süresi Grain algoritması, şifre çözme süresi ise AES algoritmasıyla elde edilmiştir. Aynı fonksiyon için en yüksek süreler Present algoritmasıyla elde edilmiştir. Fonksiyon kodu 16 olan Write Multiple Registers fonksiyonu için en düşük şifreleme ve şifre çözme süresi AES algoritmasına aittir. En yüksek hesaplama süresinin ise yine Present algoritmasının olduğu dikkat çekmektedir.

Tablo 5.2. Raspberry Pi 4 üzerinde Modbus ADU alanı işlem süreleri (s).

Algoritma	Senaryo-1 Write Multiple Coils (10 adet coil)		Senaryo-2 Write Multiple Registers (10 adet register)		Senaryo-3 Write Multiple Registers (100 adet register)	
	Şifreleme	Şifre Çözme	Şifreleme	Şifre Çözme	Şifreleme	Şifre Çözme
Present	0.005594	0.005226	0.008531	0.013332	0.006992	0.067874
XTEA	0.000733	0.000699	0.000941	0.001621	0.001138	0.005990
AES	0.000603	0.000538	0.000823	0.000743	0.000873	0.000786
Ascon	0.001615	0.001757	0.001909	0.002355	0.002267	0.006365
Grain	0.000602	0.000706	0.000826	0.001705	0.000936	0.008949
Skinny	0.002738	0.002804	0.005020	0.007551	0.005180	0.034928
ChaCha20	0.001237	0.001218	0.001271	0.001281	0.001421	0.004085
Simon	0.000805	0.000792	0.001080	0.001744	0.001268	0.009276

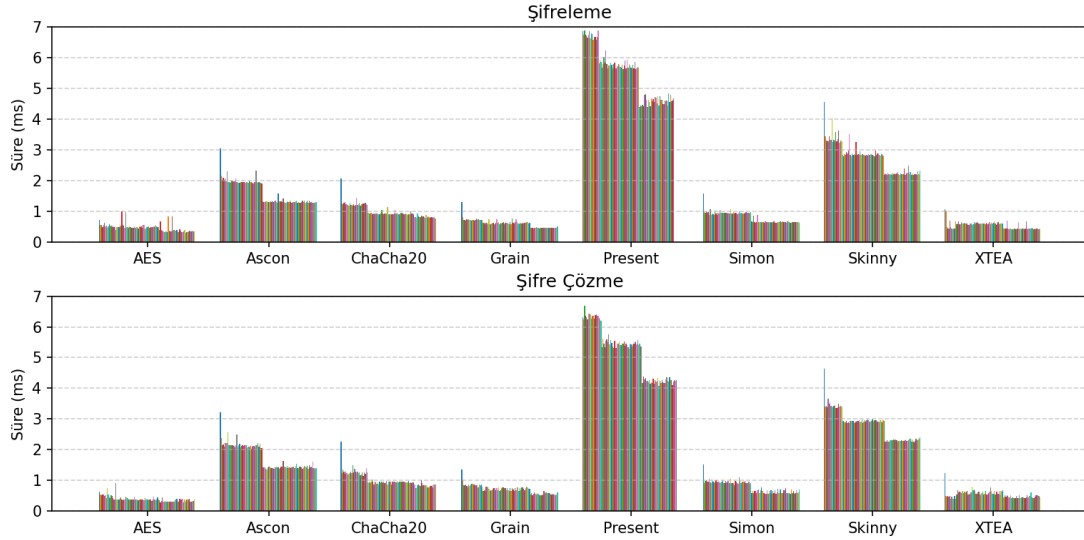
Şifreleme standardı olan AES algoritması, literatürde olduğu gibi [89] bazı senaryolarda diğer algoritmalarından daha iyi sonuç vermiştir. Bu durumun sebebi AES algoritmasının yıllardır yaygın kullanımı sayesinde hem yazılım hem de donanım için optimize edilmiş olmasıdır. Ayrıca, AES algoritmasının hafif sıklet versiyonu için yapılmış çalışmalar da vardır [90-93]. Diğer hafif sıklet şifreleme algoritmaları, genellikle araştırmalarda kullanıldığı için optimizasyon seviyeleri düşük olabilmektedir.

Şekil 5.26'da Raspberry Pi 4 tabanlı modül üzerinde gönderilen 10 adet coil değeri içeren Write Multiple Coils (fonksiyon kodu 15) fonksiyonu için 100 farklı Modbus PDU (veri alanı) şifreleme ve şifre çözme süreleri verilmiştir. Burada, tüm işlem süreleri 5 ms'nin altındadır. En hızlı işlem sürelerine bakılırsa AES, Grain, Simon ve XTEA algoritmaları 1 ms'nin altında olarak öne çıkmaktadır. En yavaş işlem süresine sahip algoritmalar ise Present ve Skinny olduğu görülmektedir.



Şekil 5.26. Raspberry Pi 4 üzerinde Senaryo-1 için Modbus PDU işlem süreleri.

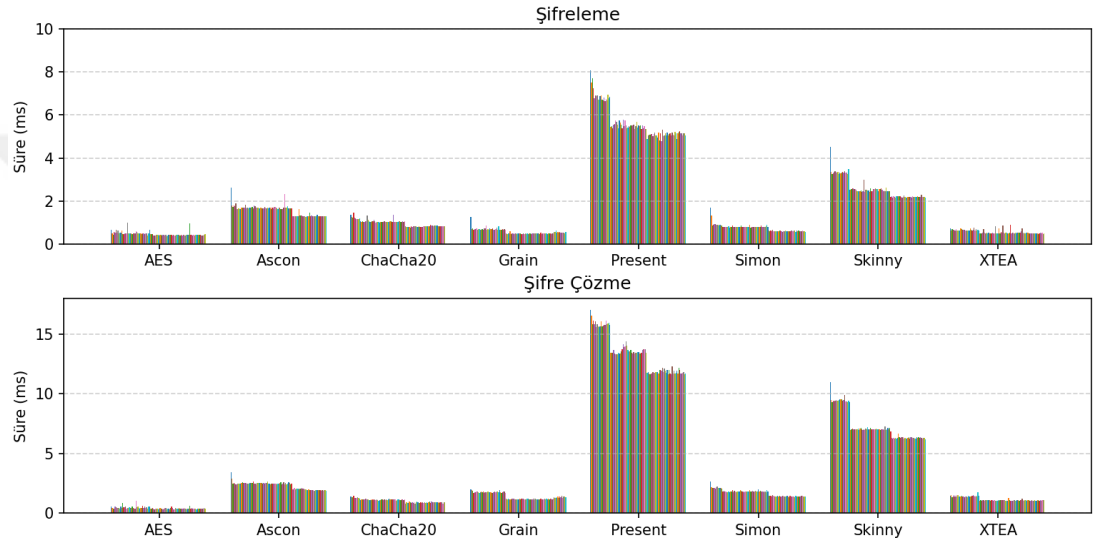
Şekil 5.27’de Raspberry Pi 4 tabanlı modül üzerinde gönderilen 10 adet coil değeri içeren Write Multiple Coils (fonksiyon kodu 15) fonksiyonu için 100 farklı Modbus ADU (başlık ve veri alanı) şifreleme ve şifre çözme süreleri verilmiştir.



Şekil 5.27. Raspberry Pi 4 üzerinde Senaryo-1 için Modbus ADU işlem süreleri.

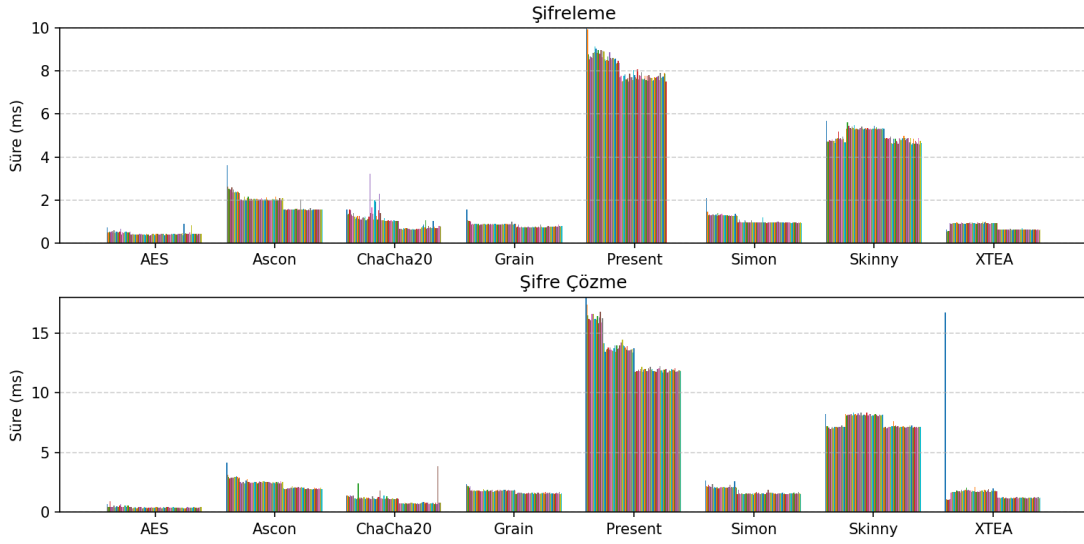
Burada, tüm işlem süreleri 7 ms’nin altındadır. Veri boyutu arttıkça bazı algoritmaların işlem süresinin arttığı da görülmektedir. En hızlı işlem sürelerine bakılırsa yine AES, Grain, Simon ve XTEA algoritmaları 1 ms’nin altında olarak öne çıkmaktadır. En yavaş işlem süresine sahip algoritmalar ise bir önceki senaryoda olduğu gibi Present ve Skinny olduğu görülmektedir.

Şekil 5.28’de Raspberry Pi 4 tabanlı modül üzerinde gönderilen 10 adet register değeri içeren Write Multiple Registers (fonksiyon kodu 16) fonksiyonu için 100 farklı Modbus PDU (veri alanı) şifreleme ve şifre çözme süreleri verilmiştir. Burada, tüm işlem süreleri 15 ms’nin altındadır. En hızlı işlem sürelerine bakılırsa; AES algoritması öne çıkmakta olup ChaCha20, Grain, Simon ve XTEA algoritmaları sonra gelmektedir. En yavaş işlem süresine sahip algoritmaların yine Present ve Skinny olduğu görülmektedir. Present algoritması şifreleme işlemleri için 10 ms’nin altındayken, şifre çözme işlemleri için 10 ms’nin üzerine çıktığı görülmektedir.



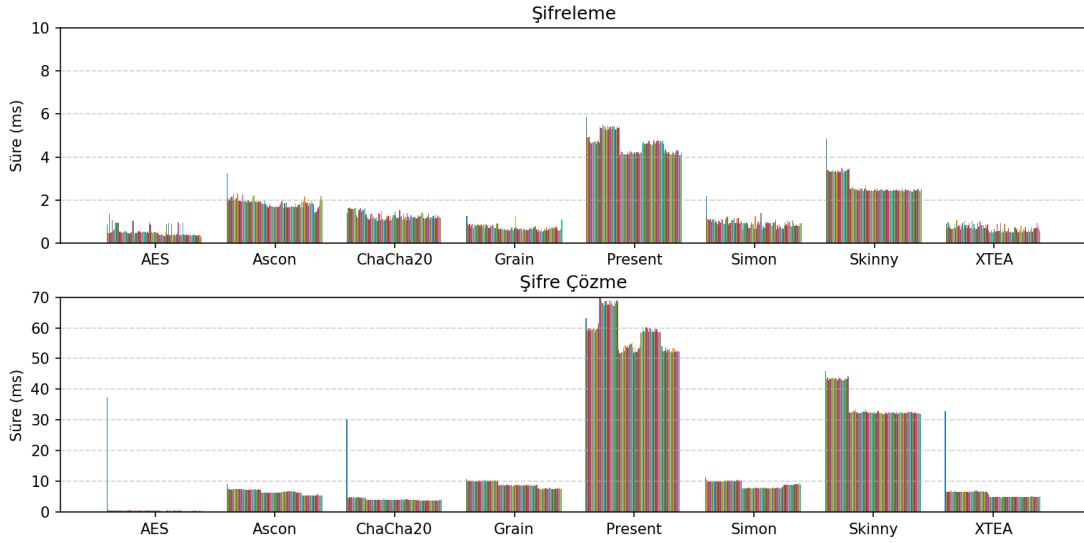
Şekil 5.28. Raspberry Pi 4 üzerinde Senaryo-2 için Modbus PDU işlem süreleri.

Şekil 5.29’da Raspberry Pi 4 üzerinde Modbus ADU (başlık ve veri alanı) için elde edilen şifreleme ve şifre çözme süreleri verilmiştir. Raspberry Pi 4 tabanlı modül üzerinde 10 adet register içeren Write Multiple Registers (fonksiyon kodu 16) fonksiyonu için 100 farklı paket gönderimi yapılmıştır. Elde edilen işlem süreleri 15 ms’nin altındadır. En hızlı işlem süreleri incelendiğinde; yine AES algoritması öne çıkmakta olup ChaCha20, Grain, Simon ve XTEA algoritmaları sonra gelmektedir. En yavaş işlem süresine sahip algoritmalar yine Present ve Skinny olmuştur. Present algoritması şifreleme işlemleri için 10 ms’nin altındayken, şifre çözme işlemleri için 15 ms’nin üzerine çıktığı elde edilmiştir.



Şekil 5.29. Raspberry Pi 4 üzerinde Senaryo-2 için Modbus ADU işlem süreleri.

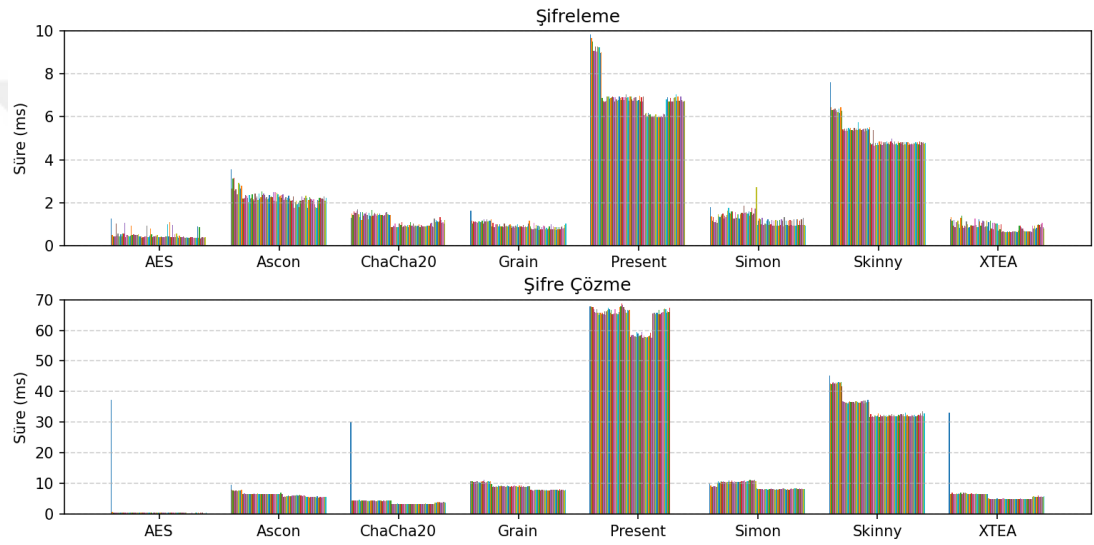
Şekil 5.30'da Raspberry Pi 4 tabanlı modül üzerinde gönderilen 100 adet register değeri içeren Write Multiple Registers (fonksiyon kodu 16) fonksiyonu için 100 farklı Modbus PDU (veri alanı) şifreleme ve şifre çözme süreleri verilmiştir.



Şekil 5.30. Raspberry Pi 4 üzerinde Senaryo-3 için Modbus PDU işlem süreleri.

Burada, şifre çözme işlem süreleri 70 ms'nin, şifreleme işlem süreleri ise 10 ms'nin altındadır. En hızlı işlem sürelerine bakılırsa; yine AES algoritması öne çıkmakta olup ChaCha20, Grain, Simon ve XTEA algoritmaları sonra gelmektedir. En yavaş işlem süresine sahip algoritmaların yine Present ve Skinny olduğu görülmektedir. Present algoritması şifreleme işlemleri için 10 ms'nin altındayken, şifre çözme işlemleri için 60 ms'nin üzerine çıktığı görülmektedir.

Şekil 5.31’de Raspberry Pi 4 tabanlı modül üzerinde gönderilen 100 adet register değeri içeren Write Multiple Registers (fonksiyon kodu 16) fonksiyonu için 100 farklı Modbus ADU (başlık ve veri alanı) şifreleme ve şifre çözme süreleri verilmiştir. Burada, şifre çözme işlem süreleri 70 ms’nin, şifreleme işlem süreleri ise 10 ms’nin altındadır. En hızlı işlem sürelerine bakılırsa; yine AES algoritması öne çıkmakta olup ChaCha20, Grain, Simon ve XTEA algoritmaları sonra gelmektedir. En yavaş işlem süresine sahip algoritmaların yine Present ve Skinny olduğu görülmektedir. Present algoritması şifreleme işlemleri için 10 ms’nin altındayken, şifre çözme işlemleri için 60 ms’nin üzerine çıktığı görülmektedir.



Şekil 5.31. Raspberry Pi 4 üzerinde Senaryo-3 için Modbus ADU işlem süreleri.

5.5.2. Orange Pi R1 LTS Plus tabanlı modül

Paketlerin şifrelenmesi önceki modülde olduğu gibi Orange Pi R1 LTS Plus üzerinde de iki farklı senaryoda test edilmiştir. Senaryolar, yalnızca Modbus PDU (veri alanı) ve Modbus ADU (başlık ve veri alanı) olarak iki farklı şekilde hazırlanmıştır. Modbus PDU için işlem süreleri incelendiğinde; fonksiyon kodu 15 olan Write Multiple Coils fonksiyonu için en düşük ortalama şifreleme ve şifre çözme süresi XTEA algoritmasıyla elde edilmiştir. Aynı fonksiyon için en yüksek süreler Present algoritmasıyla elde edilmiştir. Fonksiyon kodu 16 olan Write Multiple Registers fonksiyonu için en düşük şifreleme ve şifre çözme süresi yine XTEA algoritmasına aittir. En yüksek hesaplama süresi ise Present algoritmasına aittir.

Tablo 5.3’de veri alanını içeren Modbus PDU için algoritma bazında ortalama hesaplama süreleri verilmiştir.

Tablo 5.3. Orange Pi R1 LTS Plus üzerinde Modbus PDU alanı işlem süreleri (s).

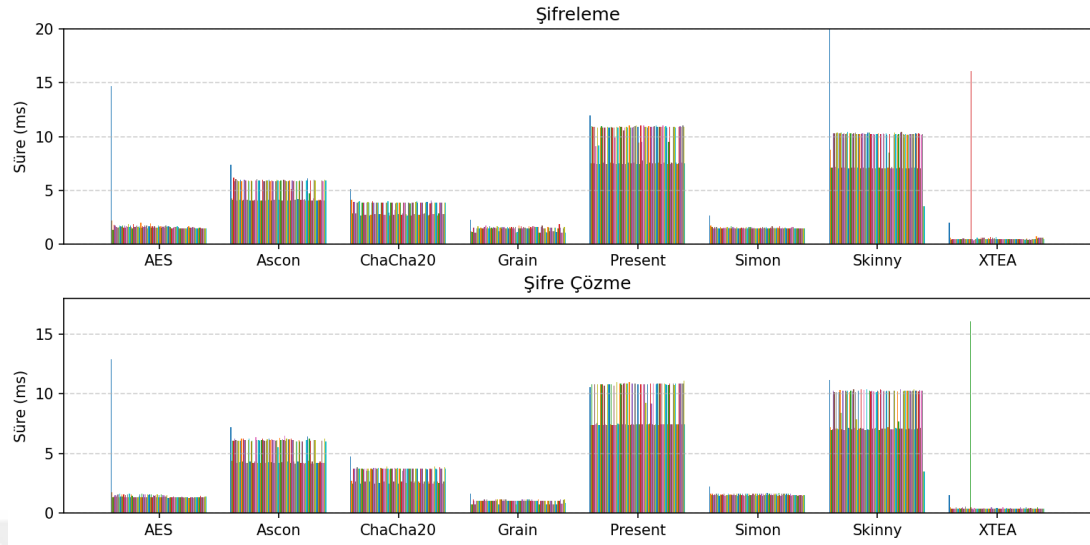
Algoritma	Senaryo-1 Write Multiple Coils (10 adet coil)		Senaryo-2 Write Multiple Registers (10 adet register)		Senaryo-3 Write Multiple Registers (100 adet register)	
	Şifreleme	Şifre Çözme	Şifreleme	Şifre Çözme	Şifreleme	Şifre Çözme
Present	0.009668	0.009118	0.023898	0.009700	0.084228	0.006658
XTEA	0.000643	0.000563	0.000841	0.000660	0.001674	0.000436
AES	0.001705	0.001504	0.001514	0.001498	0.001172	0.001147
Ascon	0.005153	0.005391	0.006808	0.004533	0.009918	0.002612
Grain	0.001458	0.000992	0.004897	0.002028	0.014101	0.000954
Skinny	0.009244	0.008709	0.016076	0.005658	0.047248	0.003842
ChaCha20	0.003439	0.003391	0.003440	0.003589	0.007070	0.001996
Simon	0.001500	0.001540	0.005342	0.002345	0.013601	0.001215

Tablo 5.4’de verilen Modbus ADU için işlem süreleri incelendiğinde; fonksiyon kodu 15 olan Write Multiple Coils fonksiyonu için en düşük ortalama şifreleme ve şifre çözme süresi XTEA algoritmasıyla elde edilmiştir. Aynı fonksiyon için en yüksek süreler Present algoritmasıyla elde edilmiştir. Fonksiyon kodu 16 olan Write Multiple Registers fonksiyonu için en düşük şifreleme ve şifre çözme süreleri AES ve XTEA algoritmalarına aittir. En yüksek hesaplama süresi ise Present algoritmasına aittir.

Tablo 5.4. Orange Pi R1 LTS Plus üzerinde Modbus ADU işlem süreleri (s).

Algoritma	Senaryo-1 Write Multiple Coils (10 adet coil)		Senaryo-2 Write Multiple Registers (10 adet register)		Senaryo-3 Write Multiple Registers (100 adet register)	
	Şifreleme	Şifre Çözme	Şifreleme	Şifre Çözme	Şifreleme	Şifre Çözme
Present	0.012000	0.011964	0.032437	0.019661	0.087728	0.009777
XTEA	0.000745	0.000675	0.001023	0.000686	0.001736	0.000491
AES	0.001593	0.001484	0.001600	0.001490	0.001215	0.001145
Ascon	0.005237	0.005208	0.007739	0.006339	0.009583	0.003011
Grain	0.002352	0.001958	0.006023	0.002966	0.014375	0.001399
Skinny	0.009389	0.008836	0.015086	0.009865	0.047856	0.007230
ChaCha20	0.003300	0.003417	0.003440	0.003501	0.006485	0.001825
Simon	0.002443	0.002500	0.005422	0.003366	0.013016	0.001576

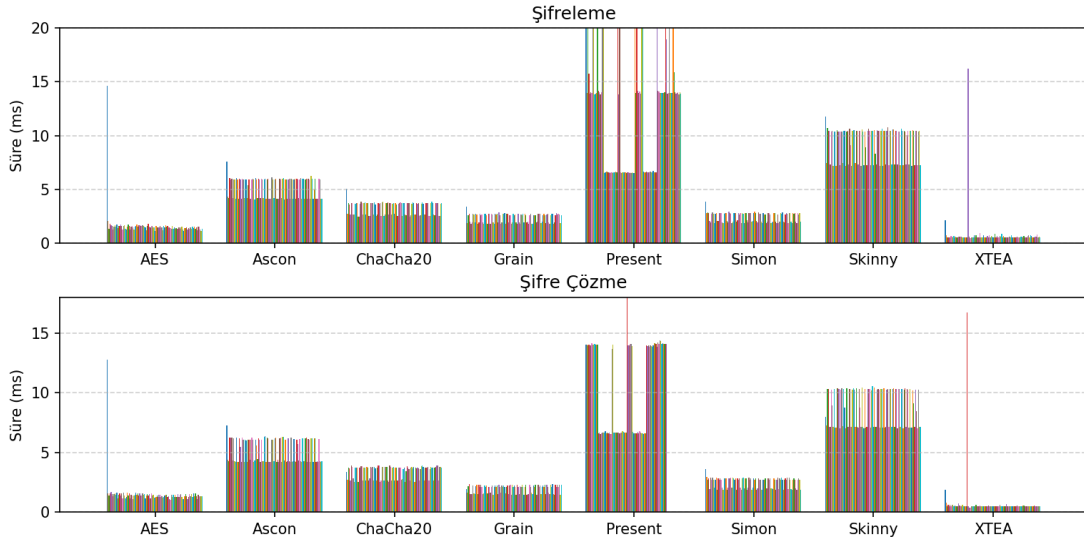
Şekil 5.32’de Orange Pi tabanlı modül üzerinde Senaryo-1 işlem süreleri verilmiştir.



Şekil 5.32. Orange Pi R1 LTS Plus üzerinde Senaryo-1 için Modbus PDU işlem süreleri.

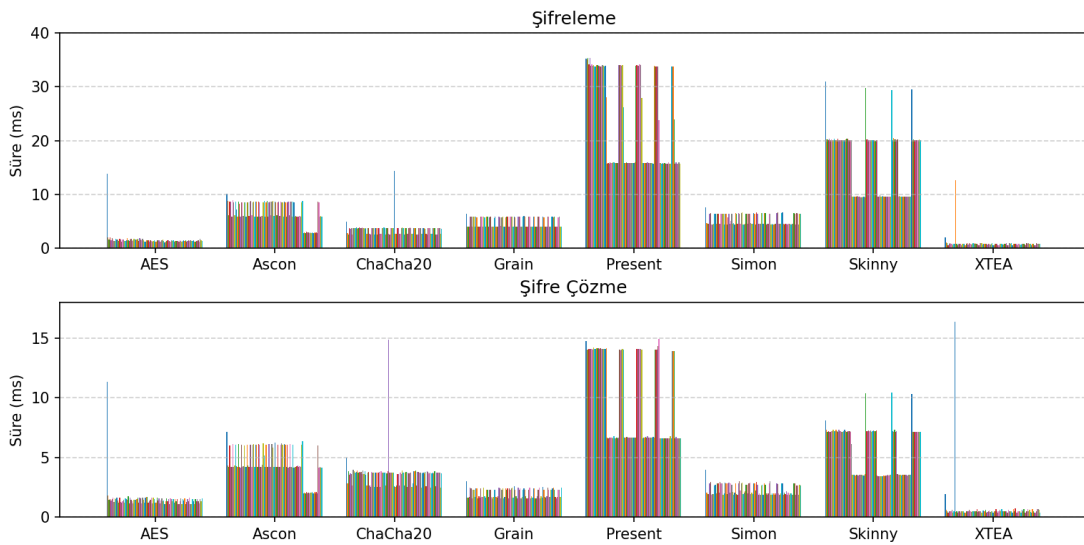
Orange Pi R1 LTS tabanlı modül üzerinde gönderilen 10 adet coil değeri içeren 100 farklı Modbus PDU (veri alanı) şifreleme ve şifre çözme süreleri incelendiğinde, tüm işlem süreleri 15 ms’nin altındadır. En hızlı işlem sürelerine bakılırsa AES, Grain, Simon ve XTEA algoritmaları 5 ms’nin altında olarak öne çıkmaktadır. En yavaş işlem süresine sahip algoritmalar ise Present ve Skinny olduğu görülmektedir.

Şekil 5.33’de Orange Pi R1 LTS Plus tabanlı modül üzerinde gönderilen 10 adet coil değeri içeren Write Multiple Coils (fonksiyon kodu 15) fonksiyonu için 100 farklı Modbus ADU (başlık ve veri alanı) şifreleme ve şifre çözme süreleri verilmiştir. Burada, tüm işlem süreleri 20 ms’nin altındadır. Veri boyutu arttıkça bazı algoritmaların işlem süresinin arttığı da görülmektedir. En hızlı işlem sürelerine bakılırsa yine AES, Grain, Simon ve XTEA algoritmaları 5 ms’nin altında olarak öne çıkmaktadır. En yavaş işlem süresine sahip algoritmalar ise bir önceki senaryoda olduğu gibi Present ve Skinny olduğu görülmektedir.



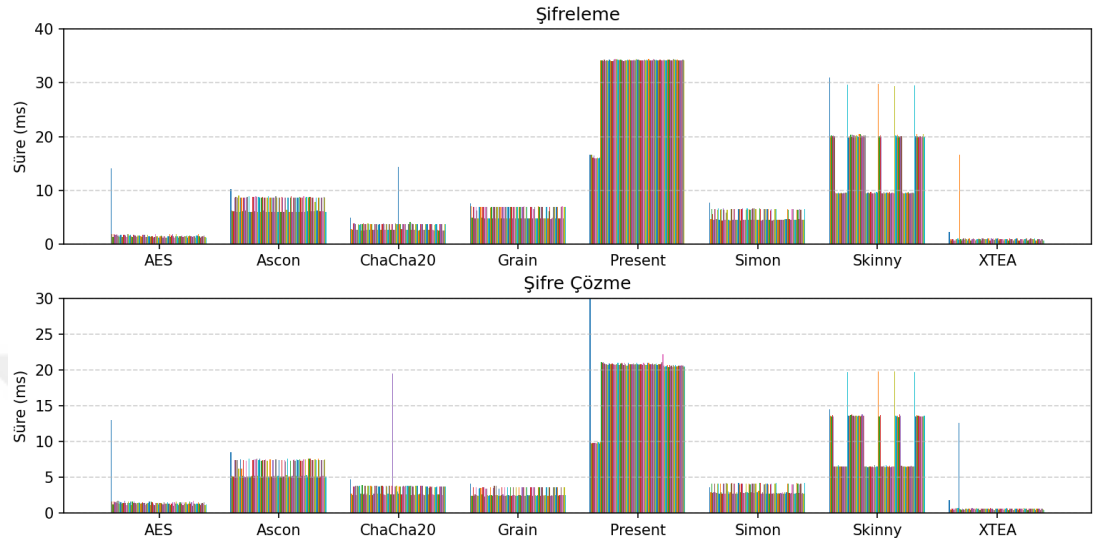
Şekil 5.33. Orange Pi R1 LTS Plus üzerinde Senaryo-1 için Modbus ADU işlem süreleri.

Şekil 5.34’de Senaryo-2 için elde edilen işlem süreleri verilmiştir. Orange Pi R1 LTS Plus tabanlı modül üzerinde gönderilen 10 adet register değeri içeren Write Multiple Registers (fonksiyon kodu 16) fonksiyonu için 100 farklı Modbus PDU (veri alanı) şifreleme ve şifre çözme süreleri incelendiğinde, tüm işlem süreleri 40 ms’nin altındadır. En hızlı işlem sürelerine bakılırsa; XTEA algoritması öne çıkmakta olup ChaCha20, Grain, Simon ve AES algoritmaları sonra gelmektedir. En yavaş işlem süresine sahip algoritmaların yine Present ve Skinny olduğu görülmektedir. Present algoritması şifrele çözme işlemleri için 15 ms’nin altındayken, şifreleme işlemleri için 30 ms’nin üzerine çıktığı görülmektedir.



Şekil 5.34. Orange Pi R1 LTS Plus üzerinde Senaryo-2 için Modbus PDU işlem süreleri.

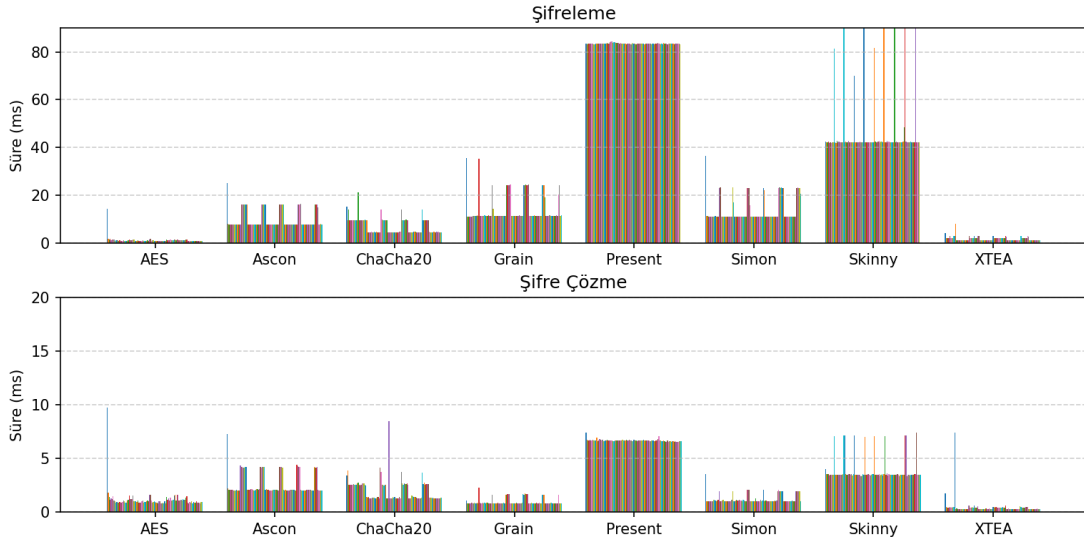
Şekil 5.35’de Orange Pi R1 LTS Plus tabanlı modül üzerinde gönderilen 10 adet register değeri içeren Write Multiple Registers (fonksiyon kodu 16) fonksiyonu için 100 farklı Modbus ADU (başlık ve veri alanı) şifreleme ve şifre çözme süreleri verilmiştir.



Şekil 5.35. Orange Pi R1 LTS Plus üzerinde Senaryo-2 için Modbus ADU işlem süreleri.

Senaryo-2 için elde edilen Modbus ADU işlem süreleri 40 ms’nin altındadır. En hızlı işlem sürelerine bakılırsa; yine XTEA algoritması öne çıkmakta olup ChaCha20, Grain, Simon ve AES algoritmaları sonra gelmektedir. En yavaş işlem süresine sahip algoritmaların yine Present ve Skinny olduğu görülmektedir. Present algoritması şifreleme işlemleri için 10 ms’nin altındayken, şifre çözme işlemleri için 15 ms’nin üzerine çıktığı görülmektedir.

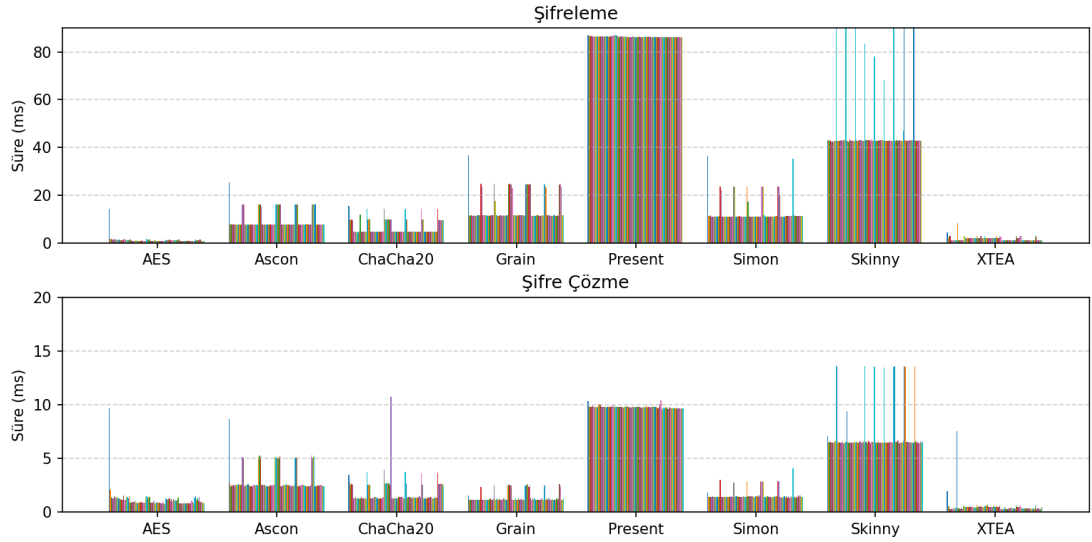
Şekil 5.36’da Orange Pi R1 LTS Plus tabanlı modül üzerinde gönderilen 100 adet register değeri içeren Write Multiple Registers (fonksiyon kodu 16) fonksiyonu için 100 farklı Modbus PDU (veri alanı) şifreleme ve şifre çözme süreleri verilmiştir.



Şekil 5.36. Orange Pi R1 LTS Plus üzerinde Senaryo-3 için Modbus PDU işlem süreleri.

Burada, şifreleme işlem süreleri 90 ms'nin, şifre çözme işlemleri ise 10 ms'nin altındadır. En hızlı işlem sürelerine bakılırsa; yine XTEA algoritması öne çıkmakta olup ChaCha20, Grain, Simon ve AES algoritmaları sonra gelmektedir. En yavaş işlem süresine sahip algoritmaların yine Present ve Skinny olduğu görülmektedir. Present algoritması şifre çözme işlemleri için 10 ms'nin altındayken, şifreleme işlemleri için 80 ms'nin üzerine çıktığı görülmektedir.

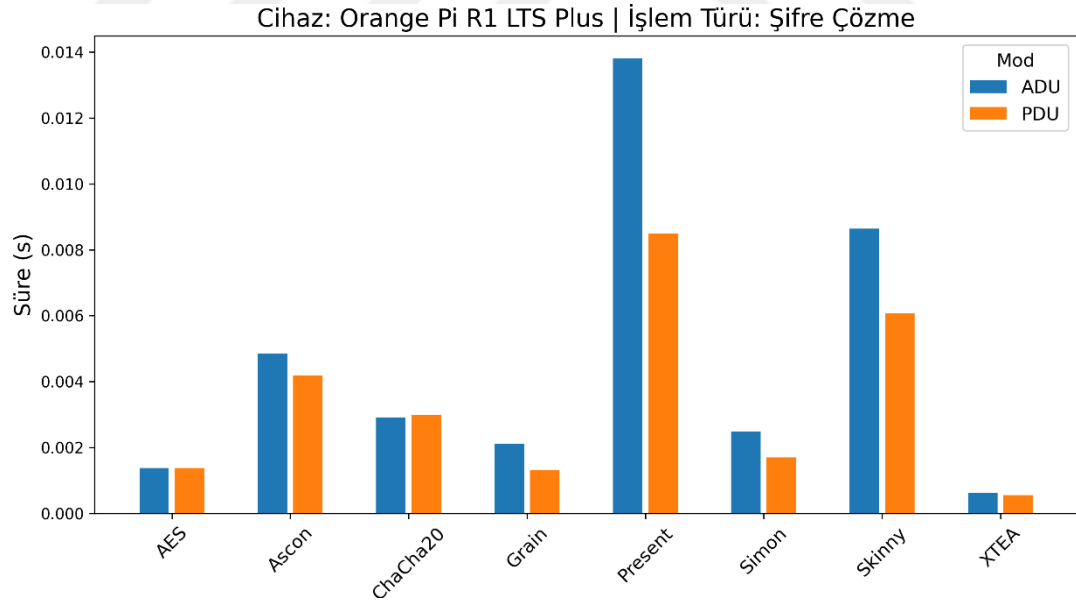
Şekil 5.37'de Orange Pi R1 LTS Plus Senaryo-3 için süreler gösterilmektedir. Son olarak Orange Pi R1 LTS Plus tabanlı modül üzerinde gönderilen 100 adet register değeri içeren Write Multiple Registers (fonksiyon kodu 16) fonksiyonu için 100 farklı Modbus ADU (başlık ve veri alanı) şifreleme ve şifre çözme süreleri incelendiğinde, şifreleme işlem süreleri 90 ms'nin, şifre çözme işlemleri ise 15 ms'nin altındadır. En hızlı işlem sürelerine bakılırsa; yine XTEA algoritması öne çıkmakta olup ChaCha20, Grain, Simon ve AES algoritmaları sonra gelmektedir. En yavaş işlem süresine sahip algoritmalar yine Present ve Skinny olmuştur. Present algoritması şifre çözme işlemleri için 10 ms'nin altındayken, şifreleme işlemleri için 80 ms'nin üzerine çıktığı elde edilmiştir.



Şekil 5.37. Orange Pi R1 LTS Plus üzerinde Senaryo-3 için Modbus ADU işlem süreleri.

5.5.3. Genel karşılaştırma

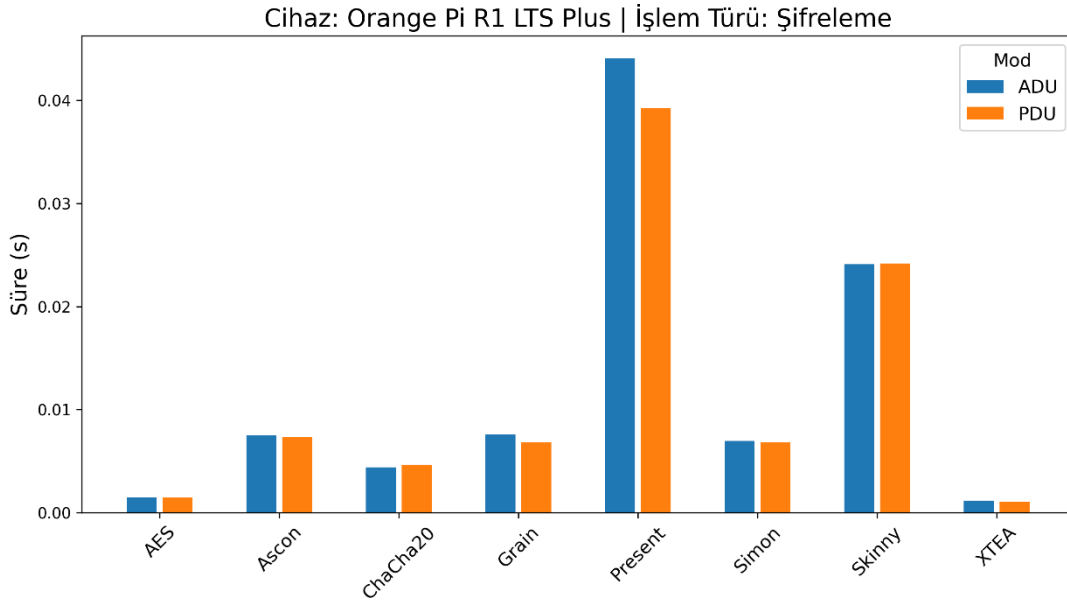
Bu alt başlıkta tüm senaryolardaki işlem sürelerinin ortalamaları hesaplanıp cihaz ve şifreleme modu bazında değerlendirilmiştir. Şekil 5.38’de Orange Pi R1 LTS Plus için şifre çözme işlem süreleri verilmiştir.



Şekil 5.38. Orange Pi R1 LTS Plus üzerinde hesaplanan şifre çözme işlem süreleri.

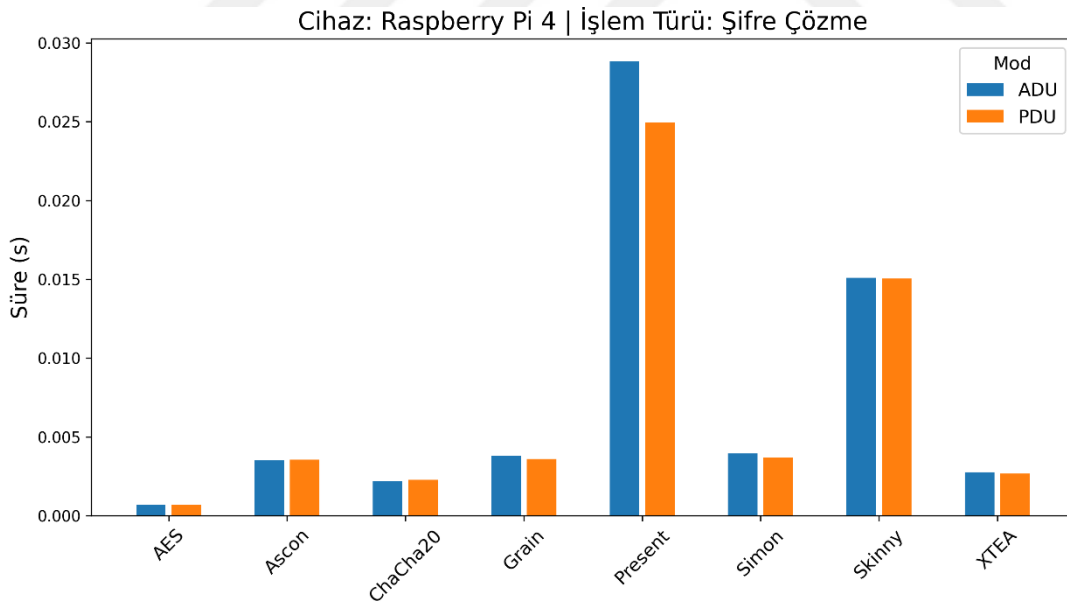
Orange Pi R1 LTS Plus için ADU ve PDU şifre çözme süreleri kıyaslandığında; Skinny algoritmasında yaklaşık 1.5 kat, Present algoritmasında ise yine 1.5 kat fark olduğu ve diğer algoritmalar için süre farkının az olduğu görülmektedir. Şekil 5.39’da Orange Pi R1 LTS Plus için şifreleme işlem sürelerine bakıldığında; neredeyse tüm

algoritmalar için ADU ve PDU alanları için süreler arasında farkın az olduğu ve Present algoritmasının en yüksek işlem süresine sahip olduğu görülmektedir.



Şekil 5.39. Orange Pi R1 LTS Plus üzerinde hesaplanan şifre çözme işlem süreleri.

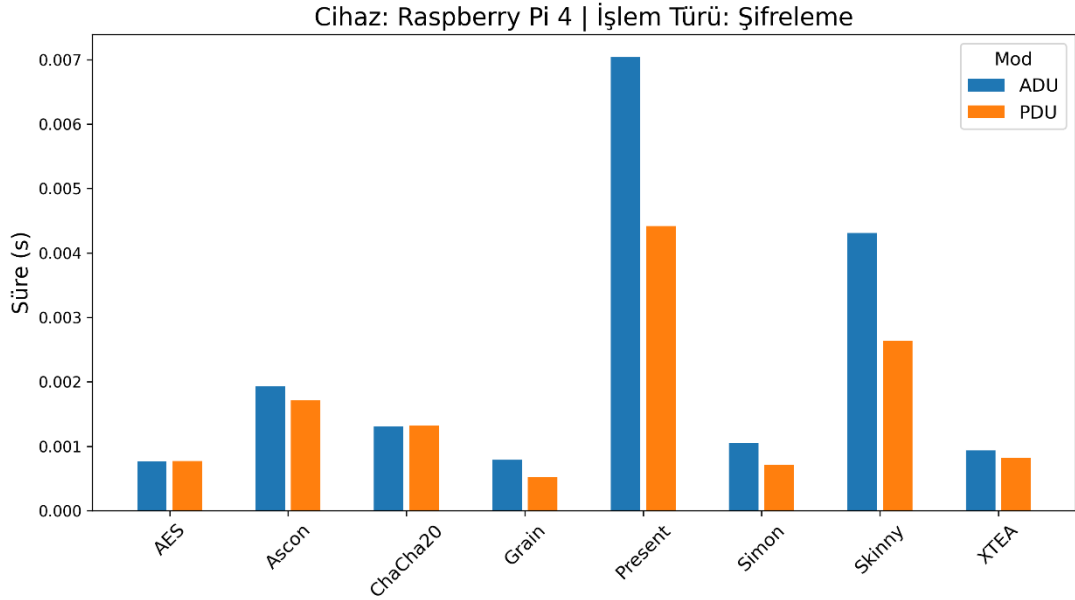
Şekil 5.40’da Raspberry Pi 4 için şifre çözme işlem süreleri verilmiştir.



Şekil 5.40. Raspberry Pi 4 üzerinde hesaplanan şifre çözme işlem süreleri.

Raspberry Pi 4 için şifre çözme sürelerine bakıldığında; neredeyse tüm algoritmalar için ADU ve PDU alanları için süreler arasında farkın az olduğu ve Present algoritmasının en yüksek işlem süresine sahip olduğu görülmektedir.

Şekil 5.41’de Raspberry Pi 4 için şifreleme işlem sürelerine bakıldığında; AES, Ascon, ChaCha20, Grain, Simon, XTEA algoritmaları için ADU ve PDU alanları için süreler arasında farkın az olduğu, Skinny algoritmasında yaklaşık 1.5 kat fark olduğu, Present algoritmasında ise yine yaklaşık 1.5 kat fark olduğu görülmektedir.



Şekil 5.41. Raspberry Pi 4 üzerinde hesaplanan şifreleme işlem süreleri.

Tüm elde edilen bulgulara bakıldığında Present ve Skinny algoritmalarının gecikme süresinin fazla olduğu ve AES, XTEA, ChaCha20, Grain ve Simon algoritmalarının yerine göre en düşük gecikme sürelerine sahip olduğu görülmektedir.

5.6. Güvenlik Analizi

Bu başlık altında Modbus/TCP protokolü ve hafif sıklet şifreleme algoritmalarının güvenlik analizi ayrı ayrı verilmiştir. Önerilen yöntem uygulandığında, Modbus/TCP protokolünün ilgili güvenlik açıklarına karşı nasıl güvenli iletişim sağladığı açıklanmıştır. Hafif sıklet şifreleme algoritmaları ise önerilen yöntemden bağımsız olarak güvenlik açıkları değerlendirilmiştir.

5.6.1. Modbus/TCP güvenlik analizi

Bu çalışmada verilen saldırı test ortamı düşünüldüğünde kötü niyetli kişiler Modbus/TCP paketlerini ele geçirebilir. Ancak önerilen yöntem ile Modbus/TCP protokolünde olan aşağıdaki güvenlik açıklarına karşı güvenlik tedbirleri alınmıştır:

- Kimlik Doğrulama eksikliği; Modbus/TCP protokolünün kendi yapısında paketlerin gönderimi ve alımı için bir kimlik doğrulama mekanizması yoktur. Bu çalışmada önerilen yöntem ile sisteme erişmeye çalışan yetkisiz kişiler şifreleme algoritmalarının ve hash fonksiyonlarının anahtarlarına sahip olmadıklarından dolayı sisteme gönderecekleri paketler engellenir.
- Şifreleme eksikliği; Modbus/TCP protokolünde veriler açık metin (plaintext) halinde iletilir. Bu çalışmada paketler iki farklı şekilde çeşitli hafif sıklet şifreleme algoritmaları ile şifrelenmiştir. Bu sayede paketler saldırganlar tarafından ele geçirilse bile anlamlı olmayacaktır.
- Ortadaki Adam Saldırısı; Modbus/TCP protokolü paketin değiştirilip değiştirilmediğin kontrol eden bir mekanizmaya sahip değildir. Saldırganların cihazlar arasındaki iletişimi manipüle etmesine karşın paketteki veriler şifrelenmiştir. Saldırganlar iletilen paketteki verileri değiştirebilmek için anlamlı bir Modbus/TCP paketi yakalamaları gerekir.
- Tekrarlama saldırısı; Modbus/TCP protokolünde bir paket tekrarlı olarak hedefe iletildiğinde her defasında pakete yanıt verilir. Ele geçirilen bir paketin defalarca gönderilmesini engellemek için her pakete bir nonce değeri eklenmiştir. Teslim edilen her paketin nonce değeri hedef cihazda saklanır ve yeni gelen paketlerin nonce değeri saklanan değerlerle karşılaştırılır. Eğer aynı değer varsa ise ilgili paketler engellenir.
- Bütünlük doğrulama eksikliği; Modbus/TCP protokolünde gönderilen paketin yetkili kaynak cihazdan geldiğini doğrulayan bir mekanizma yoktur. Bu çalışmada kaynak cihazdan çıkan ve şifrelenen veriler bir anahtar yardımıyla HMAC kodu üretilir. Bu kod pakete eklenir ve alıcı tarafta aynı anahtarla doğrulanması sağlanır. Eğer hesaplanan HMAC kodu geçersiz ise paketin bir değişikliğe uğradığı anlaşılır ve paket engellenir. PDU alanının şifrelenmesi senaryosunda HMAC kodu hesaplanırken yalnızca PDU alanları işleme tabi tutulursa paketi ele geçiren bir saldırgan MBAP alanındaki verileri değiştirerek paketi hedefe gönderebilir. Bu durum güvenlik açığı oluşturabilmekte olup önlem almak için HMAC kodu hesaplanırken MBAP alanındaki veriler de hash işlemine tabi tutulmalıdır.

5.6.2. Hafif sıklet şifreleme algoritmaları güvenlik analizi

Hafif sıklet şifreleme algoritmaları kaynak kısıtlı cihazlar için geliştirilmiştir. Bu nedenle bazı güvenlik açıkları oluşabilmektedir. Tablo 5.5’de kullanılan şifreleme algoritmalarıyla ilgili güvenlik açığı oluşturabilecek bazı önemli durumlar ve saldırı türleri belirtilmiştir.

Tablo 5.5. Şifreleme algoritmaları güvenlik problemleri.

Algoritma	Güvenlik Problemleri
Present	lineer ve diferansiyel kriptanaliz, yan kanal saldırıları
XTEA	lineer ve diferansiyel kriptanaliz saldırıları, yan kanal saldırıları
AES	güvensiz şifreleme kipi seçimi, yan kanal saldırıları
Ascon	nonce tekrar kullanımı, yan kanal saldırıları
Grain	IV tekrar kullanımı, lineer ve diferansiyel kriptanaliz saldırıları, yan kanal saldırıları
Skinny	diferansiyel ve lineer kriptanaliz, yan kanal saldırıları
ChaCha20	tekrarlı nonce kullanımı, yan kanal saldırıları
Simon	diferansiyel ve lineer kriptanaliz, yan kanal saldırıları

Present algoritmasının tur sayısı düşürüldüğünde kriptanaliz saldırılarına karşı savunmasız hale gelebilmektedir. Ayrıca güç analizi ve zamanlama saldırıları gibi yan kanal saldırılarına karşı savunmasız olabilir [94, 95]. Bu durumda ek bir güvenlik yöntemi uygulanmalıdır. XTEA algoritmasının da tur sayısı düşürüldüğünde lineer ve diferansiyel kriptanaliz saldırılarına karşı tam koruma sağlamadığı görülmektedir [96, 97]. XTEA algoritması da yan kanal saldırılarına maruz kalabilir. AES algoritması teorik olarak güçlü bir yapıya sahiptir. Ancak şifreleme kipi olarak; ECB kipi veya CTR modunda tekrarlı nonce kullanımı güvenlik açığı oluşturabilmektedir. Bu durumlara dikkat edilmeli veya güvenli olarak kabul gören GCM modu seçilmelidir. AES algoritmasına yönelik diferansiyel ve güç analizi saldırıları yapılmıştır [98, 99]. Ascon algoritmasında aynı nonce değerinin tekrar kullanılması durumunda verilerin gizliliği tehlikeye girebilir. Ascon algoritmasına yönelik yan kanal saldırıları içeren çalışmalar yapılmıştır [100, 101]. Grain algoritmasında; başlatma vektörünün tekrarlı kullanımı ve tur sayısının azaltılmış sürümlerinin tercih edilmesi güvenlik açığı oluşturabilmektedir [102]. Grain algoritmasına yönelik diferansiyel saldırılar da mevcuttur [103]. Skinny algoritmasında tur sayısının

azaltılmasında diferansiyel ve lineer kriptanaliz saldırıları başarılı olmuştur [104]. Skinny algoritmasına yönelik yan kanal saldırılarını önlemek için de çalışmalar yapılmıştır [105]. ChaCha20 algoritmasında tekrarlı anahtar ve nonce değeri kullanımı ve yan kanal saldırıları güvenlik problemi oluşturabilir [106, 107]. Simon algoritmasında tur sayısının azaltıldığı uygulamalarda diferansiyel ve lineer kriptanaliz saldırıları ve yan kanal saldırıları başarılı olmuştur [108, 109].

Tüm şifreleme algoritmalarının parametre üretiminde; hatalı kütüphane kullanımı, anahtarların güvensiz bir şekilde saklanması, basit rastgele sayı üretimi, güçsüz anahtar türetme fonksiyonları gibi işlemlerin olması güvenlik sorunları oluşturabilmektedir. Yan kanal saldırıları tüm şifreleme algoritmalarını tehdit edebilmektedir. Bu saldırılara karşı koruma sağlamak için maskeleye, gürültü ekleme, donanımsal rastgele sayı üreteçleri gibi tedbirler uygulanmalıdır. Bu hususlar, algoritmaların kodlanması aşamasında göz önünde bulundurulmalıdır.

5.7. Değerlendirme

Modbus/TCP istemci-sunucu mantığıyla çalışan bir endüstriyel iletişim protokolüdür. Bu protokol, verileri TCP/IP ağı üzerinden iletmektedir. Python dili geniş kütüphane desteği ile Modbus/TCP ve diğer ağ operasyonları için uygun geliştirme ortamı sunmaktadır. Bu çalışmada, Python ortamında Modbus/TCP paketleri oluşturulmuş, iletim halindeki Modbus/TCP paketleri bir modüldede şifrelenmiş ve başka bir modülde şifreleri çözülmüştür. Şifreleme ve şifre çözme işlemleri Ubuntu tabanlı Raspberry Pi ve Orange Pi cihazları üzerinde yapılmıştır.

Modbus/TCP protokolünün EKS'de en sık kullanılan protokol olması, birçok saldırganın hedeflerini kendisine yöneltmiştir. Bu çalışmada Modbus/TCP protokolünün güvenlik açıkları vurgulanmış ve bazı güvenlik açıkları anlatılarak protokolün güvenlik zafiyeti olduğu gösterilmiştir. Modbus/TCP protokolü günümüzde kolaylıkla saldırıya uğrayabilir. Güvenlik açıklarının temel sebebi Modbus/TCP protokolünün paketlerinin açık metin halinde iletişim sağlamasıdır. Buradan hareketle bu çalışmada Modbus/TCP protokolü verilerinin iki farklı senaryoda şifreli olarak iletilmesi amaçlanmıştır. Şifreleme işlemleri için hafif sıklet şifreleme algoritmaları tercih edilmiştir. Bu algoritmaların performansını karşılaştırmak amacıyla işlem süreleri ölçülmüştür. Yapılan ölçümlerde Modbus fonksiyonları olarak Write Multiple Coils ve Write Multiple Register kullanılarak her

algoritma için 100 adet aynı türden paket yollanmıştır. Bu paketlerin şifrenmesi ve şifresinin çözülmesi esnasında ayrı ayrı süreler ölçülmüştür. Elde edilen sonuçlarda en iyi performans gösterenler; AES, Grain, Simon, ChaCha20 ve XTEA algoritmalarıdır. Ascon algoritması bu algoritmaların performansına yakın düzeyde performans sergilemiştir. Present ve Skinny algoritmaları tüm senaryolarda en yüksek hesaplama süresini vermiştir. Ancak test ortamı üzerinde paketler, Present ve Skinny algoritmaları ile başarıyla şifrenip hedefe gönderilmiş aynı zamanda başarıyla paketlerin şifreleri çözülmüştür. Hesaplama sürelerinin yüksek çıkması paketin hedefe ulaşmasını engellemiştir. Bu noktada kullanılan modülün de etkisi olduğundan söz edilebilir. Raspberry Pi 4 tabanlı modülde yapılan şifreleme ve şifre çözme işlemleri, Orange Pi R1 LTS Plus tabanlı modüle göre daha hızlıdır. Bunun temel sebebi Raspberry Pi 4 cihazının daha yüksek donanım özelliklerine sahip olmasıdır.

Modbus/TCP paketlerinin şifrenmesi esnasında paket boyutlarının aynı olmasına rağmen bazı ölçüm durumlarında süreleri normalden uzun sürdüğü görülmektedir. Bu noktada işletim sisteminin yürütmekte başka işlemler varsa o anki donanım iş yükü fazla olabilmektedir. Bu durum hesaplama süresinde önceki hesaplama sürelerine göre gecikmeye sebep olabilir. Modbus/TCP paketi, şifreli olarak iletildiğinde gizlilik parametresi sağlanmış olur. Ancak, sisteme erişebilen kötü niyetli bir kişi bu paketi yakalayabilir. Paketi defalarca kez tekrarlayıp hedefe göndererek bir saldırı oluşturabilir. Bu duruma karşı, şifreleme esnasında paketin içerisine bir nonce değeri eklenmiştir. Bu değer şifre çözme aşamasında kontrol edilir; eğer daha önce böyle bir değer gelmişse paket engellenir, ilk defa gelen bir değer ise paket hedefe yönlendirilir. Böylelikle tekrarlama saldırısına karşı önlem alınmıştır. Modbus/TCP paketinin bütünlüğünü doğrulamak için de kendi yapısında bir mekanizma yoktur. Her ne kadar IP ve TCP başlıklarında checksum alanları olsa da bu bilgiler saldırganlar tarafından tekrar düzenlenebilir. Bu duruma karşı olarak pakete HMAC kodu eklenmiştir. HMAC kodunun hesaplanabilmesi için önceden paylaşımlı bir anahtar gerekmektedir. Anahtar kullanımı ile bir kişinin bu değeri yeniden hesaplamasının önüne geçilmiştir.

Modbus PDU ve Modbus ADU şifreleme modları açısından karşılaştırma yapıldığında ise genellikle işlem sürelerinin farketmediği, ancak Present ve Skinny algoritmalarının hesaplama sürelerinde farklılıklar olduğu görülmektedir. Bu durumda Present ve Skinny algoritmaları dışında kullanım yapıldığında Modbus ADU şifreleme modu tercih edilmesi daha hızlı bir çözüm olacaktır. Modbus PDU şifreleme modu

kullanıldığında ise HMAC kodu hesaplanırken MBAP başlık alanından yapılabilecek bir manipölasyona karşılık Modbus ADU alanının tamamı işleme tabi tutulmalıdır. Şifreleme algoritmalarının güvenlik problemleri göz ardı edilmeden uygulama yapılmalıdır. Her ne kadar Modbus/TCP için güvenli bir iletişim sağlanmaya çalışılsa da algoritmaların kendi güvenlik açığı sistemi tehlikeye atabilir.

Modbus/TCP protokolü, bu çalışmada önerilen özelliklerle güvenliği artırılmış bir protokol haline gelmiştir. Kimlik doğrulama, bütünlük doğrulama, şifreli iletişim gibi özellikler protokole dahil edilmiştir. Ancak hafif sıklet şifreleme algoritmalarına yönelik belirtilen problemler dışında güncel saldırılara karşı tedbirler göz önünde bulundurulmalıdır. Mevcut sistemlerde Modbus/TCP protokolü kullanan endüstriyel cihazlarda herhangi bir değişiklik yapılmadan yalnızca önerilen modülün araya eklenmesiyle güvenlik sağlanmış olur.

6. SONUÇ VE ÖNERİLER

Bu çalışmada, endüstriyel kontrol sistemlerinin güvenliğini artırmak amacıyla Modbus/TCP protokolü üzerinde bir yaklaşım önerilmiştir. Modbus/TCP protokolü yapısı gereği şifreleme, bütünlük kontrolü, kimlik doğrulama mekanizmalarına sahip değildir. Bu eksiklikler, endüstriyel ağ ortamındaki verilerin yetkisiz erişim sonucunda ele geçirilmesine olanak sağlamaktadır. Bu durumu engellemek için bu çalışmada Modbus/TCP protokol verilerini şifrelemek için bir modül önerilmiştir. Önerilen modülde hafif sıklet şifreleme algoritmaları kullanılarak Modbus/TCP paketlerinin şifrelenmesi ve şifrelerinin çözülmesi işlemlerinde uygulanabilirliği incelenmiştir.

Uygulama kapsamında, Raspbery Pi 4 ve Orange Pi R1 LTS Plus cihazları üzerinde Ubuntu işletim sisteminde Python programlama dili kullanılmıştır. Yapılan performans testlerinde aynı boyutta ve farklı Modbus fonksiyonlarını içeren 100'er adet Modbus/TCP paketi master cihazdan slave cihaza gönderilmiştir. Gönderilen her paket modül üzerinde hafif sıklet şifreleme algoritmaları ve AES algoritması kullanılarak şifrelenmiş diğer modül üzerinde ise şifresi çözülmüştür. Bu işlemler yapılırken geçen her bir paket için işlem süreleri ölçülmüştür. Elde edilen sonuçlar, işlem sürelerinin kullanılan algoritmaya göre ve sistemin o anki kullanımına bağlı olarak değiştiğini göstermektedir. Bu bilgiler doğrultusunda, hafif sıklet şifreleme algoritmalarının düşük işlem süresine sahip olması endüstriyel ağ ortamı için uygun bir çözüm olduğu sonucuna varılmıştır.

Sonuç olarak, bu çalışma Modbus/TCP protokolü kullanan endüstriyel sistemlerde güvenliğini artırmak için bir modül üzerinde hafif sıklet şifreleme algoritmalarının efektif bir şekilde kullanılabileceğini göstermektedir. Önerilen yöntem, paketteki verilerin bütünlüğünü ve gizliliğini korumakta, verilerin tekrar gönderilmesini engellemekte olup sistemin işleyişini aksatacak bir performans kaybına sebep olmamaktadır. Bu çalışmada önerilen yöntem ile EKS'lerde kullanılabilecek güvenlik çözümlerine bir alternatif sunulmuştur.

6.1. Önerilen Yöntemin Sınırlamaları ve Zorlukları

Bu çalışmada, Raspberry Pi 4 ve Orange Pi R1 LTS Plus cihazları üzerinde Modbus/TCP paketlerinin hafif sıklet şifreleme algoritmaları kullanılarak performans analizi yapılmıştır. Ancak önerilen yaklaşımın bazı sınırlamaları ve zorlukları bulunmaktadır. Çalışma, yalnızca Modbus/TCP paketlerine yönelik olup Modbus RTU ve Modbus ASCII paketlerine yönelik değildir.

Çalışma, iki adet bilgisayara yüklenmiş Modbus benzetim programları kullanılarak basit bir topoloji üzerinde gerçekleştirilmiştir. Gerçek bir endüstriyel sistemde test amaçlı çalışma yapmak zordur. Çünkü sistemin işleyişinin kısa süreli de olsa kesilmesi büyük kayıplara sebep olabilir. Çalışma yapılan topolojide ölçülen işlem sürelerine karşın gerçek zamanlı bir sistem üzerinde ortaya çıkacak sonuçların gözlenme imkanı sunulamamaktadır.

Diğer bir sınırlama ise, şifreleme algoritmalarında kullanılan başlangıç parametrelerinin sabit tutulmasıdır. Ayrıca kullanılan şifreleme algoritmalarının donanım destekli sürümlerinin (AES hariç) olmaması performansın tam olarak ölçülmesini sağlamamaktadır. Son olarak, çalışmada sekiz adet şifreleme algoritması tercih edilmiş olup bu algoritmaların matematiksel temelleri detaylı olarak ele alınmamıştır.

Belirtilen bu sınırlamalara rağmen çalışma, endüstriyel kontrol sistemlerinde Modbus/TCP protokolü için hafif sıklet şifreleme tabanlı yöntemlerin uygulanabilir olduğunu gösteren önemli bir alt yapı oluşturmaktadır.

6.2. Gelecek Çalışmalar İçin Öneriler

Bu çalışma, Modbus/TCP protokolünün güvenliğini artırmaya yönelik bir yaklaşım önermiştir. Elde edilen sonuçlar önerilen yöntemde kullanılan şifreleme algoritmalarının endüstriyel sistemlerde güvenlik tedbirlerine katkı sunduğunu göstermiştir. Bu çalışmaya ek olarak gelecekte güçlendirilmiş çalışmalar yapılabilir.

İlk olarak farklı hafif sıklet şifreleme algoritmaları kullanılarak performansları denenebilir. İşlem sürelerini en aza indirmek amacıyla donanım tabanlı hızlandırma yöntemleri kullanılabilir. Şifreleme algoritmalarının performansı kadar anahtar yönetimi mekanizmalarının güvenliği de kritik derecede önemlidir. Gelecek çalışmalarda, anahtar değişim protokolleri ile bu çalışmanın yapısı bütünleştirilebilir.

Sonuç olarak, gelecek çalışmaların donanım ve yazılım seviyesinde optimizasyonlara odaklanması, farklı şifreleme algoritmalarının dahil edilmesi ve anahtar yönetim protokollerinin kullanılması Modbus/TCP protokolünü kullanan endüstriyel ağ ortamının güvenliğini üst seviyelere çıkararak, siber dayanıklılığı yüksek sistemlerin geliştirilmesine katkı sunacaktır.





KAYNAKLAR

- [1] B. Karabacak. (2011). Kritik Altyapılar ve Kritik Altyapıların Korunması. Siber Savunma Sempozyumu, 24-25 Mayıs 2011 İstanbul.
- [2] X. Niu, M. M. Cook ve D. Pezaros. (2024). Examining the Suitability of Stream Ciphers for Modbus-TCP Encryption on Resource Constrained Devices. Proceedings of the 17th European Workshop on Systems Security (51-57) Atina, Yunanistan. <https://doi.org/10.1145/3642974.3652287>
- [3] E. Amer. (2024). Malicious Behavioural Detection in Scada Networks Based on Analyzing Modbus/Tcp Functions Sequences. SSRN.<https://doi.org/10.2139/ssrn.4792053>
- [4] D. S. Afenu, M. Asiri ve N. Saxena. (2024). Industrial Control Systems Security Validation Based on MITRE Adversarial Tactics, Techniques, and Common Knowledge Framework. Electronics, 13(5).<https://doi.org/10.3390/electronics13050917>
- [5] A. T. Al Ghazo ve R. Kumar. (2024). ANDVI: Automated Network Device and Vulnerability Identification in SCADA/ICS by Passive Monitoring. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 54(4), 2539-2550.<https://doi.org/10.1109/tsmc.2023.3345254>
- [6] R. S. Kurmi, A. Gupta ve R. Krishnamoorthy. (2023). A Technical Bas System Through Cetralised System for Grid Systems in Trnsmission of Data. 2023 International Conference on Power Energy, Environment & Intelligent Control (PEEIC) (1067-1071) Greater Noida, Hindistan. <https://doi.org/10.1109/peEIC59336.2023.10451583>
- [7] T. Ghosh, S. Bagui, S. Bagui, M. Kadzis ve J. Bare. (2023). Anomaly Detection for Modbus over TCP in Control Systems Using Entropy and Classification-Based Analysis. Journal of Cybersecurity and Privacy, 3(4), 895-913.<https://doi.org/10.3390/jcp3040041>
- [8] R. J. Rodríguez, S. Marrone, I. Marcos ve G. Porzio. (2023). MOSTO: A toolkit to facilitate security auditing of ICS devices using Modbus/TCP. Computers & Security, 132.<https://doi.org/10.1016/j.cose.2023.103373>
- [9] M. Tripathy, R. Niyogi, P. S. Kumar, G. B. Kumbhar, R. Singh ve V. Thakur. (2023). A Novel Approach for Detection of Cyber Attacks in Microgrid SCADA System. 2023 IEEE 3rd International Conference on Sustainable Energy and Future Electric Transportation (SEFET) (1-6) Bhubaneswar, Hindistan. <https://doi.org/10.1109/SeFeT57834.2023.10245046>
- [10] S. Banik, T. Banik, S. M. M. Hossain ve S. K. Saha. (2023). Implementing Man-in-the-Middle Attack to Investigate Network Vulnerabilities in Smart Grid Test-bed. 2023 IEEE World AI IoT Congress (AIIoT) (0345-0351) Seattle, WA, ABD. <https://doi.org/10.1109/AIIoT58121.2023.10174478>

- [11] T. Nishiuchi, S. Fujita, Y. Watanabe, M. Iwamoto ve K. Sawada. (2023). Packet Analysis and Information Theory on Attack Detection for Modbus TCP. IECON 2023- 49th Annual Conference of the IEEE Industrial Electronics Society (1-6) Singapur, Singapur. <https://doi.org/10.1109/iecon51785.2023.10312507>
- [12] H. Ochiai, M. D. Hossain, P. Chirupphapa, Y. Kadobayashi ve H. Esaki. (2023). Modbus/RS-485 Attack Detection on Communication Signals with Machine Learning. IEEE Communications Magazine, 61(6), 43-49. <https://doi.org/10.1109/mcom.002.2200553>
- [13] H. Wu, M. Jiang ve M. Cen. (2022). An Integrated Security Framework for OT System Based on Edge Computing. 2022 IEEE 21st International Conference on Ubiquitous Computing and Communications (IUCC/CIT/DSCI/SmartCNS) (9-16) Chongqing, Çin. <https://doi.org/10.1109/IUCC-CIT-DSCI-SmartCNS57392.2022.00016>
- [14] Z. Liu, T. Liang, W. Wang, R. Sun, S. Li ve I. Azeem. (2023). Design and Implementation of a Lightweight Security-Enhanced Scheme for Modbus TCP Protocol. Security and Communication Networks, 2023, 1-12. <https://doi.org/10.1155/2023/5486566>
- [15] F. Katulić, D. Sumina, S. Groš ve I. Erceg. (2023). Protecting Modbus/TCP-Based Industrial Automation and Control Systems Using Message Authentication Codes. IEEE Access, 11, 47007-47023. <https://doi.org/10.1109/access.2023.3275443>
- [16] H. Zhang, Y. Min, S. Liu, H. Tong, Y. Li ve Z. Lv. (2023). Improve the Security of Industrial Control System: A Fine-Grained Classification Method for DoS Attacks on Modbus/TCP. Mobile Networks and Applications, 28(2), 839-852. <https://doi.org/10.1007/s11036-023-02108-8>
- [17] W. Alsabbagh, S. Amogbonjaye, D. Urrego ve P. Langendörfer. (2023). A Stealthy False Command Injection Attack on Modbus based SCADA Systems. 2023 IEEE 20th Consumer Communications & Networking Conference (CCNC) (1-9) Las Vegas, NV, ABD. <https://doi.org/10.1109/ccnc51644.2023.10059804>
- [18] A. S. Mohammed, E. Anthi, O. Rana, N. Saxena ve P. Burnap. (2023). Detection and mitigation of field flooding attacks on oil and gas critical infrastructure communication. Computers & Security, 124. <https://doi.org/10.1016/j.cose.2022.103007>
- [19] P. Sindhwad ve F. Kazi. (2022). Exploiting Control Device Vulnerabilities: Attacking Cyber-Physical Water System. 2022 32nd Conference of Open Innovations Association (FRUCT) (270-279) Tampere, Finlandiya. <https://doi.org/10.23919/fruct56874.2022.9953826>
- [20] T. Martins ve S. V. G. Oliveira. (2022). Enhanced Modbus/TCP Security Protocol: Authentication and Authorization Functions Supported. Sensors (Basel), 22(20). <https://doi.org/10.3390/s22208024>

- [21] E. Damian Gutierrez Mlot, J. Saldana ve R. J. Rodriguez. (2022). Towards a Testbed for Critical Industrial Systems: SunSpec Protocol on DER Systems as a Case Study. 2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA) (1-4) Stuttgart, Almanya. <https://doi.org/10.1109/etfa52439.2022.9921522>
- [22] X. Luo ve Y. Li. (2019). Security Enhancement Mechanism of Modbus TCP Protocol. DEStech Transactions on Computer Science and Engineering, (iciti).<https://doi.org/10.12783/dtcse/iciti2018/29146>
- [23] F. R. Ametov, E. A. Bekirov ve M. M. Asanov. (2021). Organizing the information security in Modbus TCP interfaces for use in the energy complex. IOP Conference Series: Materials Science and Engineering, 1089(1).<https://doi.org/10.1088/1757-899x/1089/1/012007>
- [24] H.-C. Chang, C.-Y. Lin, D.-J. Liao ve T.-M. Koo. (2020). The Modbus Protocol Vulnerability Test in Industrial Control Systems. 2020 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC) (375-378) Chongqing, Çin. <https://doi.org/10.1109/CyberC49757.2020.00070>
- [25] P. Radoglou-Grammatikis, I. Siniosoglou, T. Liatifis, A. Kourouniadis, K. Rompoulos ve P. Sarigiannidis. (2020). Implementation and Detection of Modbus Cyberattacks. 2020 9th International Conference on Modern Circuits and Systems Technologies (MOCAST) (1-4) Bremen, Almanya. <https://doi.org/10.1109/mocast49295.2020.9200287>
- [26] J. Luswata, P. Zavarsky, B. Swar ve D. Zvabva. (2018). Analysis of SCADA Security Using Penetration Testing: A Case Study on Modbus TCP Protocol. 2018 29th Biennial Symposium on Communications (BSC) (1-5) Toronto, Kanada. <https://doi.org/10.1109/bsc.2018.8494686>
- [27] C. Alcaraz ve S. Zeadally. (2015). Critical infrastructure protection: Requirements and challenges for the 21st century. International Journal of Critical Infrastructure Protection, 8, 53-66.<https://doi.org/10.1016/j.ijcip.2014.12.002>
- [28] H. Farsi, A. Fanian ve Z. Taghiyarrenani. (2019). A novel online state-based anomaly detection system for process control networks. International Journal of Critical Infrastructure Protection, 27.<https://doi.org/10.1016/j.ijcip.2019.100323>
- [29] A. Ghaleb, S. Zhioua ve A. Almulhem. (2018). On PLC network security. International Journal of Critical Infrastructure Protection, 22, 62-69.<https://doi.org/10.1016/j.ijcip.2018.05.004>
- [30] J. M. McGinthy ve A. J. Michaels. (2019). Secure Industrial Internet of Things Critical Infrastructure Node Design. IEEE Internet of Things Journal, 6(5), 8021-8037.<https://doi.org/10.1109/jiot.2019.2903242>
- [31] E. Irmak ve İ. Erkek. (2018). Endüstriyel Kontrol Sistemleri ve SCADA Uygulamalarının Siber Güvenliği: Modbus TCP Protokolü Örneği. Gazi Üniversitesi Fen Bilimleri Dergisi Part C: Tasarım ve Teknoloji, 6(1), 1-16.<https://doi.org/10.29109/http-gujsc-gazi-edu-tr.364411>

- [32] Y. Lai, H. Gao ve J. Liu. (2020). Vulnerability Mining Method for the Modbus TCP Using an Anti-Sample Fuzzer. *Sensors* (Basel), 20(7).<https://doi.org/10.3390/s20072040>
- [33] B. Chen, N. Pattanaik, A. Goulart, K. L. Butler-purry ve D. Kundur. (2015). Implementing attacks for modbus/TCP protocol in a real-time cyber physical system test bed. 2015 IEEE International Workshop Technical Committee on Communications Quality and Reliability (CQR) (1-6) Charleston, SC, ABD. <https://doi.org/10.1109/cqr.2015.7129084>
- [34] O. N. Nyasore, P. Zavorsky, B. Swar, R. Naiyeju ve S. Dabra. (2020). Deep Packet Inspection in Industrial Automation Control System to Mitigate Attacks Exploiting Modbus/TCP Vulnerabilities. 2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS) (241-245) Baltimore, MD, ABD. <https://doi.org/10.1109/BigDataSecurity-HPSC-IDS49724.2020.00051>
- [35] F. Yi, L. Zhang, S. Yang ve D. Zhao. (2021). A Security-Enhanced Modbus TCP Protocol and Authorized Access Mechanism. 2021 IEEE Sixth International Conference on Data Science in Cyberspace (DSC) (61-67) Shenzhen, Çin. <https://doi.org/10.1109/dsc53577.2021.00016>
- [36] B. Lupton, M. Zappe, J. Thom, S. Sengupta ve D. Feil-Seifer. (2022). Analysis and Prevention of Security Vulnerabilities in a Smart City. 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC) (0702-0708) Las Vegas, ABD. <https://doi.org/10.1109/ccwc54503.2022.9720824>
- [37] L. Mahendra, R. K. S. Kumar, R. Hareesh, B. S. Bindhumadhava ve R. Kalluri. (2021). Deep Security Scanner for Industrial Control Systems. TENCON 2021 - 2021 IEEE Region 10 Conference (TENCON) (447-452) Auckland, Yeni Zelanda. <https://doi.org/10.1109/tencon54134.2021.9707347>
- [38] K. Chan, Y. Kim ve J.-Y. Jo. (2022). DER Communication Networks and Their Security Issues. 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC) (0785-0790) Las Vegas, NV, ABD. <https://doi.org/10.1109/ccwc54503.2022.9720774>
- [39] T. Cherifi, W. Cherifi, K. Challal, L. Abbad, M. Bouterfas ve A. Gramez. (2021). MUBus: A new secure industrial communication protocol based on ModBus and Ubus protocols. 2021 International Conference on Artificial Intelligence for Cyber Security Systems and Privacy (AI-CSP) (1-6) El Oued, Cezayir. <https://doi.org/10.1109/ai-csp52968.2021.9671085>
- [40] E. Holasova, K. Kuchar, R. Fujdiak, P. Blazek ve J. Misurec. (2021). Security Modules for Securing Industrial Networks. 2021 2nd International Conference on Electronics, Communications and Information Technology (CECIT) (1125-1132) Sanya, Çin. <https://doi.org/10.1109/cecit53797.2021.00199>
- [41] I. Ozcelik, M. Iskefiyeli, M. Balta, K. O. Akpınar ve F. S. Toker. (2021). CENTER Water: A Secure Testbed Infrastructure Proposal For Waste and Potable Water Management. 2021 9th International Symposium on Digital Forensics and Security (ISDFS) (1-7) Elazığ, Türkiye. <https://doi.org/10.1109/isdfs52919.2021.9486364>

- [42] Y. Ren, L. Rupasinghe, S. Khaksar, N. Ferdosian ve I. Murray. (2024). Secured Real-Time Machine Communication Protocol. *Network*, 4(4), 567-585.<https://doi.org/10.3390/network4040028>
- [43] G. Lazaridis, A. Drosou, P. Chatzimisios ve D. Tzovaras. (2024). Unraveling the Threat Landscape of CPS: Modbus TCP Vulnerabilities in the Era of I4.0. 2024 IEEE International Conference on Cyber Security and Resilience (CSR) (593-598) Londra, İngiltere. <https://doi.org/10.1109/csr61664.2024.10679453>
- [44] M. Almendli ve J. B. Mohasefi. (2024). Anomaly detection system based on deep learning for cyber physical systems on sensory and network datasets. *International Journal of Electrical and Computer Engineering (IJECE)*, 14(6).<https://doi.org/10.11591/ijece.v14i6.pp6827-6837>
- [45] R. Altschaffel, J. Dittmann ve L. Lingk. (2024). Information Hiding Detection in Industrial Control Systems-Statistical Analysis in Modbus TCP/IP. *SECURWARE 2024, The Eighteenth International Conference on Emerging Security Information, Systems and Technologies* (119-124) Nice, Fransa. <https://doi.org/>.
- [46] D. Zhang, Y. Gao, Q. Pan ve C. Yang. (2024). Ultra-Low Latency Security Hardening of Modbus/TCP Protocol Based on ZUC Cryptographic Algorithm. 2024 IEEE 12th International Conference on Information, Communication and Networks (ICICN) (25-30) Guilin, Guangxi, Çin. <https://doi.org/10.1109/icicn62625.2024.10761885>
- [47] X. Le, J. Li, Y. Zhao ve Z. Fan. (2025). A Security-Enhanced Scheme for ModBus TCP Protocol Based on Lightweight Cryptographic Algorithm. *Electronics*, 14(18).<https://doi.org/10.3390/electronics14183674>
- [48] M. Rashid, Y. Singh ve S. I. Manzoor. (2025). Methodology for Assessing Existing Vulnerabilities in Modbus Protocol. *Procedia Computer Science*, 259, 1983-1993.<https://doi.org/10.1016/j.procs.2025.04.154>
- [49] M. Tawfik ve W. Badawy. (2025). Adversarial Generation of Modbus TCP Packets Using WGANs for ICS Security Testing. 2025 International Telecommunications Conference (ITC-Egypt) (167-172) Cairo, Mısır. <https://doi.org/10.1109/ITC-Egypt66095.2025.11186573>
- [50] F. Trungadi *et al.* (2025). Securing Modbus in legacy industrial control systems: A decentralized approach using proxies, Post-Quantum Cryptography and Self-Sovereign Identity. *Journal of Information Security and Applications*, 94.<https://doi.org/10.1016/j.jisa.2025.104199>
- [51] Q.-T. Dao, L.-T. Nguyen, T.-K. Ha, V.-H. Nguyen ve T.-A. Nguyen. (2025). Investigation of Secure Communication of Modbus TCP/IP Protocol: Siemens S7 PLC Series Case Study. *Applied System Innovation*, 8(3).<https://doi.org/10.3390/asi8030065>
- [52] X. Li, M. Liu, R. Zhang, P. Cheng ve J. Chen. (2018). Demo Abstract: An Industrial Control System Testbed for the Encrypted Controller. 2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPs) (343-344) Porto, Portekiz. <https://doi.org/10.1109/iccps.2018.00045>

- [53] J. Sun *et al.* (2022). Data traceability scheme of industrial control system based on digital watermark. 2022 7th IEEE International Conference on Data Science in Cyberspace (DSC) (322-325) Guilin, Çin. <https://doi.org/10.1109/dsc55868.2022.00050>
- [54] Y. Jiang, L. Wang ve X. Zhang. (2019). Tobacco System Industrial Control System Security. 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS) (693-696) Singapore. <https://doi.org/10.1109/ccoms.2019.8821652>
- [55] Y. Chen, X. Gu, Y. Wang ve L. Ling. (2024). Industrial Robot Control System Integration Design and Application. 2024 International Conference on Advanced Control Systems and Automation Technologies (ACSAT) (81-84) Nanjing, Çin. <https://doi.org/10.1109/acsat63853.2024.10823968>
- [56] E. Zhang. (2022). The Application of Industrial Automation Computer Control System. 2022 International Conference on Computing, Robotics and System Sciences (ICRSS) (20-23) Macau Macau, Çin. <https://doi.org/10.1109/icrss57469.2022.00015>
- [57] Y. Yang, Y. Zhang, S. X. Ding ve L. Li. (2015). Design and Implementation of Lifecycle Management for Industrial Control Applications. IEEE Transactions on Control Systems Technology, 23(4), 1399-1410. <https://doi.org/10.1109/tcst.2014.2367316>
- [58] C. Lu *et al.* (2016). Real-Time Wireless Sensor-Actuator Networks for Industrial Cyber-Physical Systems. Proceedings of the IEEE, 104(5), 1013-1024. <https://doi.org/10.1109/jproc.2015.2497161>
- [59] A. F. Murillo ve S. Rueda. (2020). Access Control Policies for Network Function Virtualization environments in Industrial Control Systems. 2020 4th Conference on Cloud and Internet of Things (CIoT) (17-24) Niteroi, Brezilya. <https://doi.org/10.1109/CIoT50422.2020.9244205>
- [60] Y. Xie, W. Wang, F. Wang ve R. Chang. (2018). VTET: A Virtual Industrial Control System Testbed for Cyber Security Research. 2018 Third International Conference on Security of Smart Cities, Industrial Control System and Communications (SSIC) (1-7) Shanghai, Çin. <https://doi.org/10.1109/ssic.2018.8556732>
- [61] J. Rrushi ve P. A. Nelson. (2015). Big Data Computing for Digital Forensics on Industrial Control Systems. 2015 IEEE International Conference on Information Reuse and Integration (593-608) San Francisco, CA, ABD. <https://doi.org/10.1109/iri.2015.94>
- [62] Y. Li, J. Zhang, Y. Xiong, H. Wu ve F. Yang. (2025). Design of Industrial Equipment Remote Control System Based on Python. 2025 8th International Conference on Electronics Technology (ICET) (1-5) Chengdu, Çin. <https://doi.org/10.1109/icet64964.2025.11103226>
- [63] A. Farraj. (2025). On Using Zero Trust to Securing Industrial Control Systems in the Power Systems Industry. 2025 IEEE Texas Power and Energy Conference (TPEC) (1-5) College Station, TX, ABD. <https://doi.org/10.1109/tpec63981.2025.10906998>

- [64] M. S. Jawed ve M. Sajid. (2022). Cryptanalysis of Lightweight Block Ciphers using Metaheuristic Algorithms in Cloud of Things (CoT). 2022 International Conference on Data Analytics for Business and Industry (ICDABI) (165-169) Sakhir, Bahreyn. <https://doi.org/10.1109/icdabi56818.2022.10041583>
- [65] D. Ma ve Y. Shi. (2019). A Lightweight Encryption Algorithm for Edge Networks in Software-Defined Industrial Internet of Things. 2019 IEEE 5th International Conference on Computer and Communications (ICCC) (1489-1493) Chengdu, Çin. <https://doi.org/10.1109/iccc47050.2019.9064352>
- [66] R. Korolov, D. Yevseiev, S. Milevskyi, O. Umanskiy, I. Shyian ve Z. Sydorenko. (2024). Development of a Lightweight Encryption Algorithm Based on Extended Galois Fields. 2024 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA) (1-6) Istanbul, Türkiye. <https://doi.org/10.1109/hora61326.2024.10550599>
- [67] R. V. Chandrashekhar, J. Visumathi ve A. P. Anandaraj. (2022). Advanced Lightweight Encryption Algorithm for Android (IoT) Devices. 2022 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI) (1-5) Chennai, Hindistan. <https://doi.org/10.1109/accai53970.2022.9752555>
- [68] S. J. Hussain Pirzada, T. Xu ve L. Jianwei. (2020). Lightweight Encryption Algorithm Implementation for Internet of Thing Application. 2020 International Conference on Cyber Warfare and Security (ICWS) (1-6) Islamabad, Pakistan. <https://doi.org/10.1109/iccws48432.2020.9292373>
- [69] Z. M. Jawad Kubba ve H. K. Hoomod. (2019). A Hybrid Modified Lightweight Algorithm Combined of Two Cryptography Algorithms PRESENT and Salsa20 Using Chaotic System. 2019 First International Conference of Computer and Applied Sciences (CAS) (199-203) Bağdat, Irak. <https://doi.org/10.1109/cas47993.2019.9075488>
- [70] Z. A. Mohammed ve K. A. Hussein. (2023). Lightweight Cryptography Concepts and Algorithms: A Survey. 2023 Second International Conference on Advanced Computer Applications (ACA) (1-7) Misan, Irak. <https://doi.org/10.1109/aca57612.2023.10346914>
- [71] P. K, N. B, S. Rajalakshmi ve P. Duraisamy. (2023). A Review on Lightweight Cryptographic algorithms in Internet of Things. 2023 5th International Conference on Inventive Research in Computing Applications (ICIRCA) (1448-1451) Coimbatore, Hindistan. <https://doi.org/10.1109/icirca57980.2023.10220904>
- [72] P. Jha, H. Y. Zorkta, D. Allawi ve M. R. Al-Nakkar. (2020). Improved Lightweight Encryption Algorithm (ILEA). 2020 International Conference for Emerging Technology (INCET) (1-4) Belgaum, Hindistan. <https://doi.org/10.1109/incet49848.2020.9154170>
- [73] A. M. Raatul, T. Rahman, S. I. Mouno ve N. Mansoor. (2023). RMT: A Lightweight Encryption Algorithm For Blockchain-Based Certificate Verification. 2023 IEEE 9th International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (WIECON-ECE) (232-236) Kerala, Hindistan. <https://doi.org/10.1109/wiecon-ece60392.2023.10456497>

- [74] V. D. Rocha *et al.* (2022). Soft Error Reliability Assessment of Lightweight Cryptographic Algorithms for IoT Edge Devices. 2022 IEEE International Symposium on Circuits and Systems (ISCAS) (457-460) Austin, TX, ABD. <https://doi.org/10.1109/iscas48785.2022.9937998>
- [75] A. Bogdanov *et al.* (2007). PRESENT: An Ultra-Lightweight Block Cipher. In (Ed.), Secondary PRESENT: An Ultra-Lightweight Block Cipher (450-466) içinde. https://doi.org/10.1007/978-3-540-74735-2_31
- [76] R. M. Needham ve D. J. Wheeler. (1997). Tea extensions. Report (Cambridge University, Cambridge, UK, 1997).<https://doi.org/>.
- [77] C. Beierle *et al.* (2016). The SKINNY Family of Block Ciphers and Its Low-Latency Variant MANTIS. In (Ed.), Secondary The SKINNY Family of Block Ciphers and Its Low-Latency Variant MANTIS (123-153) içinde. https://doi.org/10.1007/978-3-662-53008-5_5
- [78] M. Ågren, M. Hell, T. Johansson ve W. Meier. (2011). Grain-128a: a new version of Grain-128 with optional authentication. International Journal of Wireless and Mobile Computing, 5(1).<https://doi.org/10.1504/ijwmc.2011.044106>
- [79] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks ve L. Wingers. (2015). The SIMON and SPECK lightweight block ciphers. Proceedings of the 52nd Annual Design Automation Conference (1-6) San Francisco, CA, ABD. <https://doi.org/10.1145/2744769.2747946>
- [80] D. J. Bernstein, "ChaCha, a variant of Salsa20," in *Workshop record of SASC*, 2008, vol. 8, no. 1: Lausanne, Switzerland, pp. 3-5.
- [81] Y. Nir ve A. Langley, "ChaCha20 and Poly1305 for IETF Protocols," 2070-1721, 2015.
- [82] NIST. (2023). Lightweight Cryptography Standardization Process: NIST Selects Ascon. <https://csrc.nist.gov/News/2023/lightweight-cryptography-nist-selects-ascon> adresinden 5 Mayıs 2025 tarihinde alınmıştır.
- [83] M. J. Dworkin *et al.* (2001). Advanced encryption standard (AES).<https://doi.org/10.6028/NIST.FIPS.197-upd1>
- [84] Raspberry Pi Foundation. (2021). Raspberry Pi 4. <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/> adresinden 10 Kasım 2021 tarihinde alınmıştır.
- [85] Xunlong Software Co. Ltd. (2023). Orange Pi R1 LTS Plus <http://www.orangepi.org/html/hardWare/computerAndMicrocontrollers/details/orange-pi-R1-Plus-LTS.html> adresinden 10 Temmuz 2023 tarihinde alınmıştır.
- [86] ModbusTools. (2021). Modbus Poll. https://www.modbus-tools.com/modbus_poll.html adresinden 1 Aralık 2021 tarihinde alınmıştır.
- [87] Enes Sanal Github. (2021). ModFlask. <https://github.com/enessanal/ModFlask> adresinden 1 Aralık 2021 tarihinde alınmıştır.

- [88] M. Varol ve M. İskefiyeli. (2025). A low cost compact network TAP device with Raspberry Pi 4. *Engineering Science and Technology, an International Journal*, 70. <https://doi.org/10.1016/j.jestch.2025.102118>
- [89] C. A. Lara-Nino, M. Morales-Sandoval ve A. Diaz-Perez. (2016). An evaluation of AES and present ciphers for lightweight cryptography on smartphones. 2016 International Conference on Electronics, Communications and Computers (CONIELECOMP) (87-93) Cholula, Meksika. <https://doi.org/10.1109/conielecomp.2016.7438557>
- [90] A. Bogdanov, F. Mendel, F. Regazzoni, V. Rijmen ve E. Tischhauser. (2014). ALE: AES-Based Lightweight Authenticated Encryption. In (Ed.), *Secondary ALE: AES-Based Lightweight Authenticated Encryption* (447-466) içinde. https://doi.org/10.1007/978-3-662-43933-3_23
- [91] S. A. Jassim; ve A. K. Farhan. (2022). Designing a New Lightweight AES Algorithm to Improve the Security of the IoT Environment. *Iraqi Journal of Computer, Communication, Control and System Engineering*, 96-108. <https://doi.org/10.33103/uot.ijccce.22.2.9>
- [92] M. Qasaimeh, R. S. Al-Qassas, F. Mohammad ve S. Aljawarneh. (2020). A Novel Simplified AES Algorithm for Lightweight Real-Time Applications: Testing and Discussion. *Recent Advances in Computer Science and Communications*, 13(3), 435-445. <https://doi.org/10.2174/2213275912666181214152207>
- [93] R. S. Salman, A. K. Farhan ve A. Shakir. (2022). Lightweight Modifications in the Advanced Encryption Standard (AES) for IoT Applications: A Comparative Survey. 2022 International Conference on Computer Science and Software Engineering (CSASE) (325-330) Duhok, Irak. <https://doi.org/10.1109/csase51777.2022.9759828>
- [94] A. J. Banu ve A. Prathiba. (2025). Non-Profiled Partial Nibble Recovery on Power Attack Resilient Adiabatic PRESENT Block Cipher Through AI Vulnerability Assessment. *IEEE Access*, 13, 85543-85562. <https://doi.org/10.1109/access.2025.3569570>
- [95] D. Bellizia, G. Scotti ve A. Trifiletti. (2016). Implementation of the PRESENT-80 block cipher and analysis of its vulnerability to Side Channel Attacks Exploiting Static Power. 2016 MIXDES - 23rd International Conference Mixed Design of Integrated Circuits and Systems (211-216) Lodz, Polonya. <https://doi.org/10.1109/mixdes.2016.7529734>
- [96] R. Chatterjee ve R. Chakraborty. (2020). A Modified Lightweight PRESENT Cipher For IoT Security. 2020 International Conference on Computer Science, Engineering and Applications (ICCSEA) (1-6) Gunupur, Hindistan. <https://doi.org/10.1109/iccsea49143.2020.9132950>
- [97] M. Hajari, S. A. Azimi, P. Aghdaie, M. Salmasizadeh ve M. R. Aref. (2015). Impossible differential cryptanalysis of reduced-round TEA and XTEA. 2015 12th International Iranian Society of Cryptology Conference on Information Security and Cryptology (ISCISC) (58-63) Reşt, İran. <https://doi.org/10.1109/iscisc.2015.7387899>

- [98] L. Han, N. Wu, F. Ge, F. Zhou, J. Wen ve P. Qing. (2020). Differential Fault Attack for the Iterative Operation of AES-192 Key Expansion. 2020 IEEE 20th International Conference on Communication Technology (ICCT) (1156-1160) Nanning, Çin. <https://doi.org/10.1109/icct50939.2020.9295779>
- [99] T. Dhar, S. Bhunia ve A. R. Trivedi. (2017). A solitary protection measure against scan chain, fault injection, and power analysis attacks on AES. 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS) (575-578) Boston, ABD. <https://doi.org/10.1109/mwscas.2017.8052988>
- [100] E. Tena-Sánchez, F. E. Potestad-Ordóñez, M. Martín-González, A. Casado-Galán ve A. J. Acosta. (2025). Low Entropy Masking Protection Scheme for Ascon Cipher to Counteract Side-Channel Attacks. 2025 40th Conference on Design of Circuits and Integrated Systems (DCIS) (43-48) Santander, İspanya. <https://doi.org/10.1109/dcis67520.2025.11281929>
- [101] H. Zhang, Y. Gao, Y. Zhou ve J. Ming. (2025). Side-Channel Collision Attacks Against ASCON. 2025 Design, Automation & Test in Europe Conference (DATE) (1-6) Lyon, Fransa. <https://doi.org/10.23919/date64628.2025.10993063>
- [102] V. A. Ghafari ve H. Hu. (2017). A New Chosen IV Statistical Attack on Grain-128a Cipher. 2017 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC) (58-62) Nanjing, Çin. <https://doi.org/10.1109/CyberC.2017.101>
- [103] I. Salam, T. H. Ooi, L. Xue, W.-C. Yau, J. Pieprzyk ve R. C. W. Phan. (2021). Random Differential Fault Attacks on the Lightweight Authenticated Encryption Stream Cipher Grain-128AEAD. IEEE Access, 9, 72568-72586. <https://doi.org/10.1109/access.2021.3078845>
- [104] X. Song, W. Jiang, Z. Li, L. Liu ve S. Wu. (2020). New collision paths for round-reduced SKINNY-Hash. China Communications, 17(6), 145-152. <https://doi.org/10.23919/jcc.2020.06.012>
- [105] J. Ge, Y. Xu, R. Liu, E. Si, N. Shang ve A. Wang. (2018). Power Attack and Protected Implementation on Lightweight Block Cipher SKINNY. 2018 13th Asia Joint Conference on Information Security (AsiaJCIS) (69-74) Guilin, Çin. <https://doi.org/10.1109/AsiaJCIS.2018.00020>
- [106] Z. Najm, D. Jap, B. Jungk, S. Picek ve S. Bhasin. (2018). On Comparing Side-channel Properties of AES and ChaCha20 on Microcontrollers. 2018 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS) (552-555) Chengdu, Çin. <https://doi.org/10.1109/apccas.2018.8605653>
- [107] D. E. R. Alam, H. Al Fitrah, M. Ayunasari ve D. Ogi. (2023). Implementation of ChaCha20 Algorithm with Elliptic-Curve-Diffie-Hellman in Server Room Monitoring System to Prevent MITM and Reused Key Attack. 2023 IEEE International Conference on Cryptography, Informatics, and Cybersecurity (ICoCICs) (71-76) Bogor, Endonezya. <https://doi.org/10.1109/ICoCICs58778.2023.10277110>

- [108] H. Tupsamudre, S. Bisht ve D. Mukhopadhyay. (2014). Differential Fault Analysis on the Families of SIMON and SPECK Ciphers. 2014 Workshop on Fault Diagnosis and Tolerance in Cryptography (40-48) Busan, Güney Kore. <https://doi.org/10.1109/fdtc.2014.14>
- [109] H. Enami, Y. Koderu, T. Kusaka ve Y. Nogami. (2021). Consideration of the side-channel attack to SIMON implemented on Arduino Uno. 2021 Ninth International Symposium on Computing and Networking Workshops (CANDARW) (412-415) Matsue, Japonya. <https://doi.org/10.1109/candarw53999.2021.00075>





ÖZGEÇMİŞ

Ad-Soyad : Murat VAROL

ÖĞRENİM DURUMU:

- **Lisans** : 2014, Sakarya Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü
- **Yükseklisans** : 2019, Ondokuzmayıs Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı
- **Doktora** : 2025, Sakarya Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar ve Bilişim Mühendisliği Anabilim Dalı

MESLEKİ DENEYİM:

- 2017 yılından beri Zonguldak Bülent Ecevit Üniversitesi'nde öğretim elemanı olarak çalışmaktadır.

TEZDEN TÜRETİLEN ESERLER:

- Varol M. ve İskefiyeli M. (2024, 10-11, Mayıs). Implementing Man-in-the-middle Attack to the ModBus-TCP Communication at SAU-CENTER Testbed. *Karaelmas International Science and Engineering Symposium (KISES 2024)*, Zonguldak, Turkey.
- Varol M. ve İskefiyeli M. 2025. A low cost compact network TAP device with Raspberry Pi 4, *Engineering Science and Technology, an International Journal*, 70, 102118.