6LOWPAN AĞLARI İÇİN
ÖZGÜN BİR YÖNETİM MİMARİSİ ÖNERİSİ

Yekta TÜRK

DOKTORA TEZİ
Bilgisayar Mühendisliği Anabilim Dalı
Danışman: Dr. Öğr. Üyesi Ali AKMAN

İstanbul
T.C. Maltepe Üniversitesi
Fen Bilimleri Enstitüsü
Ağustos, 2018

JÜRİ VE ENSTİTÜ ONAYI

Yekta TÜRK'ün "6lowpan Ağları İçin Özgün Bir Yönetim Mimarisi Önerisi" başlıklı tezi 27.09.2018 tarihinde aşağıdaki jüri tarafından değerlendirilerek "Maltepe Üniversitesi Lisansüstü Eğitim ve Öğretim Yönetmeliği"nin ilgili maddeleri uyarınca, Bilgisayar Mühendisliği Anabilim Dalında Yüksek Lisans/Doktora tezi **oy birliğiyle / oy çokluğuyla** olarak kabul edilmiştir.

	Unvanı. Adı ve soyadı	İmza
Üye (Tez Danışmanı)	: Dr. Öğr. Üyesi Ali AKMAN	
Üye	: Prof. Dr. Raif ÖNVURAL	
Üye	: Prof. Dr. Ahmet Mesut RAZBONYALI	
Üye	: Prof. Dr. Ebru MENŞUR ALKOY	
Üye	: Dr. Öğr. Üyesi Serap ÇEKLİ	

Prof. Dr. İlter BÜYÜKDIĞAN
Enstitü Müdürü

İLKE VE KURALLARA UYUM BEYANI

Bu tezin bana ait özgün bir çalışma olduğunu; çalışmamın hazırlık, veri toplama, analiz ve bilgilerin sunumu olmak üzere tüm aşamalarda bilimsel etik ilke ve kurallara uygun davrandığımı; bu çalışma kapsamında elde edilmeyen tüm veri ve bilgiler için kaynak gösterdiğimi ve bu kaynaklara kaynakçada yer verdiğimi; çalışmanın Maltepe Üniversitesinde kullanılan "bilimsel intihal tespit programı" ile tarandığını ve öngörülen standartları karşıladığını beyan ederim. Herhangi bir zamanda, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun saptanması durumunda, ortaya çıkacak tüm ahlaki ve hukuki sonuçlara razı olduğumu bildiririm.

20/08/2018

Yekta TÜRK

iii

6LoWPAN Ağları İçin Bulut Tabanlı Yönetim Sistemi

ORIJINALLIK RAPORU

%11	%10	%8	%8
BENZERLIK ENDEKSI	İNTERNET KAYNAKLARI	YAYINLAR	ÖĞRENCI ÖDEVLERI

BIRINCIL KAYNAKLAR

1	www.linux-phreak.biz İnternet Kaynağı	%2
2	Submitted to The Scientific & Technological Research Council of Turkey (TUBITAK) Öğrenci Ödevi	%1
3	Yekta Tiirk, Ali Akman. "A Management Model for Low Powered Wireless Personal Area Networks", 2018 IEEE 19th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), 2018 Yayın	<%1
4	www.zoeken-engine.com İnternet Kaynağı	<%1
5	Submitted to Texas A&M University, College Station Öğrenci Ödevi	<%1
6	Submitted to Colorado Technical University Online Öğrenci Ödevi	<%1

TEŞEKKÜR

Tez danışmanım olarak her türlü yönlendirmeyi sağlayan değerli hocam Dr. Öğr. Üyesi Ali Akman'a teşekkürlerimi sunarım. Her zaman yanımda olup beni destekleyen sevgili eşim Gülin'e ve bugünlere gelmemi sağlayan aileme içten teşekkürlerimi sunarım.

Yekta TÜRK

Ağustos 2018

ÖZ

6LOWPAN AĞLARI İÇİN ÖZGÜN BİR YÖNETİM MİMARİSİ ÖNERİSİ

Yekta TÜRK
Doktora Tezi
Bilgisayar Mühendisliği Anabilim Dalı
Danışman: Dr. Öğr. Üyesi Ali AKMAN
Maltepe Üniversitesi Fen Bilimleri Enstitüsü, 2018

Nesnelerin ve nesnelerden oluşan ağların nasıl yönetileceği önemli bir tasarım sorunudur. Kaynakları sınırlı olan 6LoWPAN nesneleri düşük güçte çalışır. İletim ortamları güvenilir olmayabilir. Bu cihazlar gerçek hayattaki kullanımlarında dış etkenlere açık ortamlarda konumlanmaktadır. Yönetimsel durumları izlenmeden uzun süre çalışıyor olmaları birtakım fiziksel ve güvenlik riskleri içermektedir. Bu nedenlerle 6LoWPAN ağında bulunan nesnelerin nasıl yönetileceğinin mimari yapısının belirlenmesi gerekmektedir.

Ağ yönetimi, temel olarak cihazların izlenmesini, hataların bulunmasını ve ağdaki parametrelerin değiştirilmesini içerir. Ağdaki cihazlar ağ yönetim sistemleri kullanılarak izlenir ve böylece ağın hizmet sağladığı servislerin verimli şekilde devam etmesi sağlanır. Geleneksel IP tabanlı ağ cihazlarının yönetiminde, SNMP en yaygın kullanılan protokoldür. SNMP protokolü üzerinde yapılacak olan bazı değişiklikler, protokolün kaynakları sınırlı olan 6LoWPAN nesnelerinde ve ağlarında kullanımı sağlar.

Bu tez çalışmasında, 6LoWPAN protokolü ile çalışan ağların yönetimi için bir yönetim mimarisi geliştirilmiştir. Çalışma kapsamında, ağ cihazlarının yönetiminde kullanılan SNMPv3 protokolü, IPv6 çalışan nesneler için uyarlanmıştır. Nesnelerin oluşturdukları yönetimsel bilgiler ile algılamayla oluşturdukları veriler ayrıştırılmıştır ve yönetimsel bilgilerin, uyarlanmış bir SNMPv3 protokolü ile buluta aktarımı sağlanmıştır. Nesnelerin ve nesnelerden oluşan ağların yönetimi, gerçekleştirilmektedir. Oluşturulan mimari ile birlikte, yönetimsel bilgiler için bulut içerisinde veri işleme ve tahmini analizler yapılmaktadır. Analizler sonucu bulut içerisindeki algoritmalar çalıştırılarak, izleme ve yönetim işlemleri, ihtiyaçlara cevap veren servislere dönüştürülmektedir.

Anahtar Sözcükler: Nesnelerin İnterneti; Bulut Tabanlı Yönetim; SNMP; 6LoWPAN

ABSTRACT

A NOVEL MANAGEMENT ARCHITECTURE PROPOSAL FOR 6LOWPAN

Yekta TURK
PhD Thesis
Computer Engineering Department
Thesis Advisor: Dr. Ali AKMAN
Maltepe University Science and Engineering Graduate School, 2018

Management of IoT devices and IoT networks is an important design problem. 6LoWPAN objects with limited resources run at low power. The transmission media may not be reliable. These devices are located in open environments for outdoor use in real life use. They have some physical and security risks to be working for long periods of time without being monitored. For this reason, the architectural structure of how to manage the objects in the 6LoWPAN network needs to be determined.

Network management basically involves monitoring devices, finding faults, and changing network parameters. The devices on the network are monitored using network management systems so that the services that the network serves are kept efficient. In managing traditional IP-based network devices, SNMP is the most widely used protocol. Some changes to the SNMP protocol will allow the protocol to be used in 6LoWPAN objects and networks with limited resources.

In this thesis, a management architecture for the management of networks operating with the 6LoWPAN protocol has been developed. Within the scope of the study, the SNMPv3 protocol used in the management of network devices is adapted for IPv6 running objects. The management information they create and the data they create by the detection are separated and the management information is transferred to the cloud with a customized SNMPv3 protocol. The management of networks of objects and objects is realized. Along with the developed architecture, data is processed and forecasted in the cloud for administrative information. Analyzes are executed and algorithms run in cloud, so monitoring and management processes are transformed into services that respond to needs.

Keywords: Internet of Things; Cloud Based Management; SNMP; 6LoWPAN

İÇİNDEKİLER

JÜRİ VE ENSTİTÜ ONAYI ..... **Error! Bookmark not defined.**
İLKE VE KURALLARA UYUM BEYANI ..... **Error! Bookmark not defined.**
İNTİHAL RAPORU iii
TEŞEKKÜR v
ÖZ vi
ABSTRACT vii
İÇİNDEKİLER viii
TABLOLAR LİSTESİ x
ŞEKİLLER LİSTESİ xi
KISALTMALAR xiii
ÖZGEÇMİŞ xvi
BÖLÜM 1. GİRİŞ 1
Problem 3
İlgili Literatür 6
Katkılar 9
BÖLÜM 2. YÖNTEM 10
6LoWPAN Protokolü 10
Başlık Sıkıştırma ve 6LoWPAN 11
6LoWPAN Ağının Çalışması 14
6LoWPAN Bant Genişliği ve İletim Gecikmesi 16
SNMP Mimarisi 17
SNMP MIB'leri ve Protokol Operasyonları 19
SNMPv3 24
SNMPv3 Kullanıcı Tabanlı Güvenlik Modeli 26
Önerilen Yönetim Sistemi Mimarisi 27
Mimari Yapıdaki Süreçler ve Yönetimsel Objeler 32
Net-SNMP ve SNMPv3 Protokol Geliştirmeleri 36
BÖLÜM 3. DENEYSEL ÇALIŞMALAR VE BULGULAR 47
Verilerin Toplanması 48
Bulgular 49
Yorumlar 47
BÖLÜM 4. SONUÇ 54
Özet 54
Yargı 54

Öneriler 55
EK-1 56
Yönetim Scriptleri 56
EK-2 100
Iftop Geliştirmeleri 100
EK-3 101
Net-SNMP Konfigurasyonu 101
KAYNAKÇA 102

TABLOLAR LİSTESİ

Tablo-1 Gönderim baytı içindeki ilk iki bitin değerleri 11
Tablo-2 M Sıfır Olduğu Durumda S/SAM ve D/DAM Değerleri 14
Tablo-3 Mimari'de Kullanılan Yönetimsel Nesneler ve Tanımları..... 33
Tablo-4 Yönetim Verilerinin Alınması için Farklı Protokollerin Performans Karşılaştırması 47
Tablo-5 6LoWPAN Yönetimi için Değerlendirilen 4 Protokolün Karakteristikleri, Limitasyonları ve Faydaları 53

ŞEKİLLER LİSTESİ

Şekil-1 IP ve 6LoWPAN Protokol Yapıları 11
Şekil-2 IPv6 ile 6LoWPAN Dönüşümü Sağlayan Kenar Yönlendirici Protokol Yapısı .. 12
Şekil-3 6LoWPAN Gönderim Baytı Eklenmiş bir IPv6 Paketi 12
Şekil-4 6LoWPAN Sıkıştırma Başlığı 13
Şekil-5 6LoWPAN Başlığı Trafik Sınıfı ve Akış Etiketinin Sıkıştırılması 13
Şekil-6 Diğer Başlığın Sıkıştırıldığı Durumda Port Numaralarının Sıkıştırılması 15
Şekil-7 Bir 6LoWPAN Ağı 16
Şekil-8 TCP/IP ve SNMP 18
Şekil-9 Bir SNMP Öğesi ve Bileşenleri 19
Şekil-10 SNMP OID Ağacı 20
Şekil-11 Örnek Bir SNMP Alma Operasyonu 21
Şekil-12 SNMP Ağacı Üzerinde Örnek Bir Yürüme İşlemi 21
Şekil-13 Örnek Bir SNMP Toplu Alma Operasyonu 22
Şekil-14 Örnek Bir SNMP Trap Operasyonu 22
Şekil-15 Örnek Bir SNMP Yapılandırma Operasyonu 23
Şekil-16 SNMPv3 Mesaj Formatı 25
Şekil-17 Önerilen Yönetim Sistemi ve 6LoWPAN Ağı ile Etkileşimi 27
Şekil-18 Mimari Yapının Yüksek Seviye Tasarımı 28
Şekil-19 İş Süreç Katmanları 29
Şekil-20 Mimari Yapı İş Akış Modeli 29
Şekil-21 Sistem Modülleri 30
Şekil-22 İşlemler ve Eylem Süreçleri 34
Şekil-23 6LoWPAN Sensöründeki Operasyonel İşlem Akışı 36
Şekil 24 Net-SNMP Programının Yapısı 37
Şekil-25 Önerilen Yönetim Sistemi ile 6LoWPAN sensörünün SNMPv3 Başlık Sıkıştırması Yöntemi Üzerinden İletişim Kurması 38
Şekil-26 Data_set.c Dosyası Kodu 40
Şekil-27 Scalar.h Dosyası Kodu 45
Şekil-28 Scalar.c Dosyası Kodu 46
Şekil-29 6LoWPAN Testleri için Test Topolojisi 48
Şekil-30 Farklı IoT Protokolleri için Yönetim Bilgisi Sorgu Karşılaştırması 46

Şekil-31 Yönetim Sorguları için Protokollerin Performans Sonuçları 48

KISALTMALAR

6LoWPAN	: IPv6 over Low Powered Wireless Personal Area Networks (Düşük Güçlü Kablosuz Kişisel Alan Ağları üzerinden IPv6)
CoAP	: Constrained Application Protocol (Sınırlandırılmış Uygulama Protokolü)
CBC-DES	: Cipher Block Chaining on Data Encryption Standard (Veri Şifreleme Standardında Şifreli Blok Zincirleme)
DDOS	: Distributed Denial of Service (Dağıtık Servis Kesintisi Saldırısı)
DTLS	: Datagram Transport Layer Security (Datagram Aktarım Katmanı Güvenliği)
DSCP	: Differentiated Services Code Point (Farklılaştırılmış Hizmetler Kod Noktası)
ECN	: Explicit Congestion Notification (Açık Tıkanıklık Bildirimi)
HMAC	: Hash-based Message Authentication Code (Özet-Tabanlı Mesaj Doğrulama Kodu)
HTTP	: Hypertext Transfer Protocol (Hiper-Metin Transfer Protokolü)
ICMPv6	: Internet Control Messaging Protocol Version 6 (İnternet Kontrol Mesajı Protokolü Sürüm 6)
IEEE	: The Institute of Electrical and Electronics Engineer (Elektrik ve Elektronik Mühendisleri Enstitüsü)
IETF	: Internet Engineering Task Force (İnternet Mühendisliği Görev Grubu) Elektronik Mühendisleri Enstitüsü)
IP	: Internet Protocol (İnternet Protokolü)
IPv6	: Internet Protocol Version 6 (İnternet Protokolü Sürüm 6)
IoT	: Internet of Things (Nesnelerin İnterneti)
JSON	: Javascript Object Notation (Javascript Nesne Gösterimi)
LoRA	: Long Range Communication (Uzak Mesafe Haberleşme Teknolojisi)

MAC	: Medium Access Control (Ortam Erişim Kontrolü)
MD5	: Message Digest Algorithm 5 (Mesaj Özet algoritması 5)
MQTT	: Message Queuing Transport Telemetry (Mesaj Sıralamalı Telemetri Aktarımı)
MQTT-SN	: Message Queuing Transport Telemetry – Sensor Network (Sensör Ağları İçin Mesaj Sıralamalı Telemetri Aktarımı)
MTU	: Maximum Transmission Unit (Maksimum İletim Birimi)
NETCONF	: Network Configuration Protocol (Ağ Yapılandırma Protokolü)
NMS	: Network Management System (Ağ Yönetim Sistemi)
NTP	: Network Time Protocol (Ağ Zaman Protokolü)
OID	: Object Identifier (Obje Ayırtedici)
PDU	: Protocol Data Unit (Protokol Veri Birimi)
PLC	: Power Line Communication (Güç Hattı İletişimi)
QoS	: Quality of Service (Servis Hizmeti Kalitesi)
RESTful	: Representational State Transfer (Servis yönelimli mimari üzerine oluşturulan veri transfer yöntemini içeren uygulama protokolleri)
RF	: Radio Frequency (Radyo Frekansı)
RFC	: Request for Comment (IETF Standart Yorum Dokümanı)
SHA	: Secure Hash Algorithm (Güvenli Hash Algoritması)
SNMP	: Simple Network Management Protocol (Basit Ağ Yönetim Protokolü)
SNMPv3	: Simple Network Management Protocol Version 3 (Basit Ağ Yönetim Protokolü Sürüm 3)
SMI	: Structure of Management Information (Yönetim Verisinin Yapısı)
TCP	: Transmission Control Protocol (İletim Kontrol Protokolü)

TCP/IP	: Transmission Control Protocol/Internet Protocol (İletim Kontrol Protokolü ve Internet Protokolünün beraber çalışması)
TLS	: Transport Layer Security (İletim Latmanı Güvenliği)
TSM	: Transport Secure Mode (İletim Güvenlik Modu)
UDP	: User Datagram Protocol (Kullanıcı Birim Protokolü)
USM	: User Security Model (Kullanıcı Güvelik Modeli)

ÖZGEÇMİŞ

Yekta Türk

Bilgisayar Mühendisliği Anabilim Dalı

Eğitim

Derece	*Yıl*	*Üniversite, Enstitü, Anabilim/Anasanat Dalı*
Y.Ls.	2007	George Washington Üniversitesi, Fen Bilimler Enstitüsü Telekomünikasyon Anabilim Dalı
Ls.	2005	Anadolu Üniversitesi, Müh-Mimarlık Fakültesi Elektrik-Elektronik Mühendisliği Anabilim Dalı
Lise	2000	Gaziantep Anadolu Lisesi

İş/İstihdam

Yıl	*Görev*
2016 -	Mobil Şebeke Tasarım Uzmanı. Türk Telekom
2013- 2016	Network Uzmanı. Avea
2012- 2013	Network Mühendisi. Turkcell Superonline
2007- 2012	Mühendis. Türk Telekom

Mesleki Birlik/Dernek Üyelikleri

Yıl	*Kurum*
2007 -	Üye: Elektrik Mühendisleri Odası

Yayınlar ve Diğer Bilimsel Faaliyetler

"An Experimental Analysis of Differentiated Quality of Service Support for LTE Users," IEEE 5th International Conference on Electrical and Electronics Engineering (ICEEE 2018). Mayıs, 2018.

"A Management Model for Low Powered Wireless Personal Area Networks," IEEE 19th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2018). Haziran, 2018.

Kişisel Bilgiler

Doğum yeri ve yılı : Bursa,1982 Cinsiyet: Erkek
Yabancı diller : İngilizce
GSM / e-posta : .+ (90) 555 255 36 39 / yekta.turk@hotmail.com

BÖLÜM 1. GİRİŞ

Kablosuz sensör ağlar, insanlar, sistemler ve bilgisayarlar arasındaki aktivitelerin algılanmasını ve izlenmesini sağlayan ağlardır [1]. Bu ağlar, sensörler, uygulayıcılar, ağ geçitleri ve kullanıcılar gibi bileşenlerinin bir ve ya birkaçını içermektedirler. Veri, bir ve ya birkaç düğüm atlayarak ağ geçidine erişir ve ağ geçidi üzerinden internete ya da bir uydu gibi farklı bir sisteme iletilir. Kullanıcılar, kablosuz ağları programlayarak, gerekli algılama aktivitelerini başlatır. Sonrasında elde edilen veri işlenerek analiz edilir. Bu şekilde algılama ve tanımlama aktivitelerinin servislere dönüştürülmesini sağlanır.

Kaynakları sınırlı kablosuz sensör cihazlar için IP protokolünün uyarlanması, nesneler ile internetin bütünleşmesini hızlandırmıştır [2]. Kablosuz sensör ağlar ile gerçekleştirilen uygulamalarda internet protokollerinin kullanımı, birçok avantajı beraberinde getirmiştir. Öncelikle, IP tabanlı çalışan cihazlar internete veya diğer IP çalışan ağlara, arada bir ağ geçidi ve ya bir protokol çevirici cihaz olmadan, doğrudan bağlanabilir. Böylece IP ile çalışan kablosuz sensör cihazları, mevcut ağlardaki IP altyapısına kolaylıkla entegre olur. Ayrıca, IP tabanlı teknolojiler üzerinde on yıllardır çalışma yapılmaktadır. IP çalışan sistemler üzerinde yeterli seviyede yetkinliğe sahip birçok kişi bulunmaktadır. Bunların yansıra, IP teknolojileri herkese açık ve bedavadır. Herkes IP dokümanları ve standartlarına kolayca erişebilmektedir. Son olarak, IP tabanlı ağların yönetimi, işletilmesi ve IP tabanlı teknolojilerin kullanılması için mevcutta geliştirilmiş birçok sistem bulunmaktadır.

6LoWPAN'ın tanımlanması ile birlikte, IP protokolünün kablosuz sensör ağlara uyarlanması sağlanmıştır [3]. Böylece, IP protokolünü kullanabilen ve düşük güçte çalışan sınırlı cihazların, kişisel alan ağları için kullanılabilmesi sağlanmıştır. İnsanlar, sistemler ve bilgisayarlar arasındaki faaliyetleri tespit etmek ve izlemek için IEEE 802.15.4 MAC katman spesifikasyonuna uyan 6LoWPAN ağları, sensörler, aktüatörler ve ağ geçitlerinden bir ve ya daha fazlasını içerebilir. 6LoWPAN, ilk olarak 2,4 Ghz çalışan ve sınırlı bant genişliği olan IEEE 802.15.4 çalışan kablosuz sensör ağları ile birlikte geliştirilmiştir. Bu nedenle IEEE 802.15.4 protokolü içerisinde bazı alanlar 6LoWPAN protokol kullanımına uygun durumdadır. 6LoWPAN ile birlikte kablosuz sensör cihazları için TCP/IP protokol katmanı içerisinde bir adaptasyon katmanı

tanımlanır. Mevcutta 868 Mhz ve 915 Mhz bantlarında ve IEEE 802.15.4 protokolü üzerinde çalışan 6LoWPAN sistemleri de bulunmaktadır. Son yapılan geliştirmeler ile birlikte 6LoWPAN, 1 GHz altı düşük enerjili RF, Bluetooth Smart, PLC ve düşük enerjili Wi-Fi sistemleri üzerinde de çalışabilmektedir.

Ağ yönetimi, cihazların izlenmesini, hataların bulunmasını, ağdaki parametrelerin değiştirilmesini içerir. Böylece arızaların izole edilmesi, ağdaki operasyonların ve hizmetleri verimli şekilde devam etmesi sağlanır. Sınırlı kapasiteleri olan 6LoWPAN nesneleri düşük güçte çalışır ve iletim ortamları güvenilir olmayabilir. 6LoWPAN cihazları kullanım alanları gereği olarak, dış etkenlere açık ortamlarda konumlanabilir. Bu nedenlerden ötürü, uzun süreler boyunca durumlarının izlenmeden çalışmaları çeşitli riskler içermektedir.

6LoWPAN çalışan ağlardaki cihazların yönetimsel bilgilerine erişilebilmek için, bir yönetim protokolü gereksinimi bulunmaktadır [4]. Günümüzde pratik uygulamalarda, CoAP, MQTT, JSON vb. gibi uygulama yüklemek için kullanılan popüler IoT protokolleri, aynı zamanda cihaz yönetim için de kullanılmaktadır. SNMP, geleneksel IP tabanlı ağ cihazlarının yönetiminde yaygın olarak kullanılan bir protokoldür. SNMP, ağ cihazları arasında yönetim bilgilerinin alışverişi için kullanılan bir uygulama katmanı protokolüdür. SNMP temel olarak bir veri toplama protokolünü, bir veri tabanı şemasını ve veri nesnesi türlerini içerir. Geniş alan ağlarında sıklıkla kullanılan SNMP protokolü, aynı zamanda 6LoWPAN ağlarının yönetiminde de kullanılabilir [5].

6LoWPAN ağlarının yönetimi için belirlenmiş olan standart bir yönetim mimarisi mevcut değildir ve bu konuda birçok öneri bulunmaktadır [6]. 6LoWPAN yönetimi ile ilgili bir başka konu, ağ içerisinde çok fazla sayıda cihaz bulunmasının bir yönetim sistemi için önemli yük oluşturmasıdır. Çok fazla sayıda cihazın aynı anda yönetilmesi ayrı bir yönetim problemini oluşturmaktadır. 6LoWPAN çalışan nesnelerin ve 6LoWPAN çalışan ağların yönetimi için, bir yönetim sistemi mimarisi ve yönetim sistemi tasarımı konuları önerilere açıktır [7].

Bu tez çalışması iki ana kısımdan oluşmaktadır. Birincisi, bulut içerisindeki geliştirilen bir yönetim sistemi ile yönetimsel verilerin yorumlanması ve sistem içerisindeki önceden belirlenmiş algoritmalar ile 6LoWPAN ağındaki hata ve performans yönetimini gerçekleştirecek aksiyonların alınmasıdır. 6LoWPAN nesneleri bulut

içerisinden, önceden belirlenmiş görevler ile yönetilecektir. Yönetimsel bilgiler bulut içerisinde saklanarak, 6LoWPAN ağını yönetimi için veri işlenmesi ve tahmini analizler yapılacaktır. İkincisi, standartlaşmış bir ağ yönetim protokolü olan SNMPv3'ün protokol yapısı üzerinden bazı değişiklikler yapılarak, 6LoWPAN cihaz ve ağ yönetiminde kullanımının sağlanmasıdır. Protokol uyarlama işleminde bazı alanlar cihazlar üzerinde sabitlenerek çıkarılmıştır. Protokol içerisindeki bazı alanlara ait fonksiyonlar ise, sensörlerin bağlı olduğu yönetim sistemi içerisinde çalıştırılarak, protokol alanı üzerinden çıkarılması sağlanmıştır. Böylece SNMPv3 protokolünün çalışması sırasında kullandığı bant genişliğinin bir kısmından tasarruf edilerek SNMPv3'ün, 6LoWPAN gibi bant genişliğinin sınırlı olduğu ağlarda kullanımına uygun hale getirilmesi sağlanmıştır. Ayrıca, 6LoWPAN ağları içerisinde, işletmecilerin ve ağ yöneticilerinin deneyim sahibi oldukları, standardı olan SNMPv3 protokolü ile yönetim fonksiyonları sağlanmış olacaktır.

Problem

Nesnelerin interneti için geliştirilen farklı teknolojiler ve bu farklı teknolojiler ile çalışan nesnelerin günlük yaşamdaki kullanım alanları, araştırmacıların en çok ilgilendikleri konular arasındadır [8]. Günümüzde kişisel alan ağları ile ilgili olarak yapılan araştırma çalışmaları çoğunlukla, yeni servis çözümlerine, yeni uygulama yöntemlerine ve ya enerjiyi daha etkin kullanan yöntemler ortaya çıkarmaya odaklanmaktadır [9]. Ancak düşük enerjili ağ ağları için yönetim sistemi mimarisi ve yönetim sistemi tasarımı konusu önerilere açık bir konu olarak bulunmaktadır. Pek çok araştırmacı 6LoWPAN yönetimi için sistemler geliştirmeye çalışsa da mevcutta kesin bir yönetim mimarisi bulunmamaktadır.

6LoWPAN çalışma yapısında sensörlede oluşan algılama verileri, ağ geçitlerine, doğrudan ya da birkaç atlama yaparak ulaşmaktadır. Sonrasında verinin ağ geçidi üzerinden internete erişimi sağlanmaktadır. Sonuç olarak, elde edilen bilgi analiz edilerek, 6LoWPAN ağı içerisindeki algılama etkinlikleri bu servisin hizmet verdiği kişi ve ya ticari yapılar için belirli çözümlere dönüştürülür. Düşük güç ve sınırlı bant genişliği kullanarak çalışan 6LoWPAN cihazları, geniş alan ağ cihazlarına, iletme ve yönlendirme gibi özellikleri bakımından benzemektedirler. Bu durumda, sınırlı kaynaklara sahip 6LoWPAN cihazlarının, geleneksel ağlara benzer şekilde yönetilmesi güçlü bir seçenek olarak düşünülebilir. Ayrıca SNMP'nin üçüncü sürümü ile birlikte, ağ yönetimi için

gerekli performans, güvenlik ve uyumluluk gereksinimleri karşılayacak yetenekler protokole kazandırılmıştır [10]. SNMP'nin düşük kapasiteli ağlarda tıpkı büyük alan ağlarına benzer bir yaklaşımla doğrudan uygulanması, kaynak tüketimi açısından, küçük kapasiteli cihazlarda sorun oluşturmaktadır. Bu durumda SNMPv3 protokolü 6LoWPAN'larda kullanım için uyarlanmalıdır, çünkü 6LoWPAN perspektifinden bakıldığında, yönetimsel bilgilerin taşınması bant genişliğine önemli miktarda trafik yükü getirmekte ve nesnelerde önemli derecede enerji sarfiyatı oluşturmaktadır. Sınırlı bağlantı kapasitesine sahip ağlarda SNMPv3 kullanıldığında, protokolün standardında tanımlı bazı alanlar, veri aktarımı sırasında bant genişliğinden tasarruf sağlamak amacıyla yeniden düzenlenebilir. Modern, sensör ağlarının neredeyse tamamı, algıladıkları veriyi buluta gönderen ve daha sonrasında analiz için bu veriyi bulut içerisinde saklayan yapıda çalışmaktadır. Sensör ağların buluta kesintisiz bağlı olacakları düşünüldüğünde, SNMPv3 protokolüne özgü bazı özellikler buluta aktarılabilir ve böylece protokole standardına ait özelliklerin bulut yardımı ile korunması sağlanabilir.

Uyarlanmış bir SNMPv3 protokolü, 6LoWPAN cihazları üzerindeki yönetimsel bilgilerin bir yönetim sistemine aktarılması için kullanılabilir [11]. Bulut sunucu içerisinde 6LoWPAN ağının yönetimi için bir yönetim sistemi bulunmalıdır. Geleneksel IP tabanlı sistemlerin yönetiminde kullanılan yönetim sistemleri cihazların durumunu izlemek için kullanılmaktadır. Cihazlardan SNMP ile alınan veri, ağ yönetim sistemleri üzerinde görselleştirilir. Ağ yönetim sistemleri ile operatörler cihazlar üzerinde arıza çözme ve yapılandırma işlemlerini yapmaktadır. Geniş alan ağları ile karşılaştırıldığında, 6LoWPAN cihazlarını gözlemleyecek operatörlerin bulunması maliyetli bir durumdur.

6LoWPAN ağ yönetim sistemi bir operatöre bağlı kalmadan kendi kendisine ağı yönetebiliyor olmalıdır [12]. Sensör düğümlerinin sınırlı yeteneklerine uygun, etkili ve 6LoWPAN içerisindeki sınırlı bant genişliğini dikkate alan ve kaynakların kullanımı açısından hafif bir yönetim sistemi, tasarlanacak olan mimari özelliklerinden biridir. Sensörler için bir yönetim sistemi tasarlarken, başka bir odak noktası ise, ağ içerisinde çok fazla sayıda cihazın varlığıdır [13]. Sorunlu linklerin tespiti ve kalite problemlerin analizi, özellikle çok fazla sayıda cihaz içeren ağlar içerisinde, çok zorlu bir iştir. Ağın iletişiminin tamamen kaybolması durumunda, kök neden analizinin yapılması ve temel kesinti nedeninin belirlenmesi, bu gibi bir durumda zaman alıcı bir görevdir. Özellikle gerçek zamanlı sistemlerde, kritik olan bu işlemlerin, hızlı bir şekilde yapılması ve çözüm

yöntemlerinin çabuk bir şekilde uygulanması gerekmektedir. Ayrıca tasarlanacak olan yönetim sisteminde, ağ durumu değişikliklerini ve tüm ağ topolojisini izlenmeli, yönetim sistemi oluşacak olan ağ arızaları ile ayrı bir operatör müdahalesine gerek duymadan doğrudan ilgilenmelidir [14]. Bir başka dikkat edilmesi gereken nokta ise, bu nesnelerden yönetim bilgisini taşınırken temel güvenlik gereksinimlerinin karşılanmasıdır [15].

Bu tezdeki mimari yapı, uyarlanmış SNMPv3 protokolü ile buluta veri aktarılması ve bulut içerisinde bu verilerin yorumlanmasını içerir. Bulut içerisinde yönetim sistemi sadece veri gözlemlemesini sağlamayacak, aynı zamanda otomatik olarak bazı düzeltici ve önleyici aksiyonlar alacaktır. Yönetimsel aksiyonların yanı sıra, aynı zamanda bazı temel güvenlik analizlerini gerçekleştirip, temel ağ güvenliğinin ihlali durumunda gerekli önlemleri almaktadır. Ağı yüksek performansta ve kesintisiz bir şekilde yönetmek için, izlenen SNMPv3 nesnesinin durumu, bulut içerisinde yer alan yönetim sistemimde değerlendirilebilir. Değerlendirme sonucu sistemin alacağı aksiyonlar ile ağ sorunlarının çözülmesi veya ağ performansını artışı sağlanabilir. Önerilen sistem tüm ağın topolojisinin görüşüne sahiptir ve bu perspektife dayanarak karar verme yeteneğine sahiptir.

Sensör ağların çalışma yapısında ağ içerisindeki cihazlar, kendi pil durumlarına göre kendi uyku aralıklarını belirlemektedir [16]. Bu durumun yani cihazların uyku aralıklarına kendi başına karar vermelerinin, ağın genel enerji kullanımı üzerinde olumsuz etkisi olabilir, çünkü bir düğümün uyku durumunda olması, diğer düğümlerin atlama yaparak oluşturacağı yol seçimine etki etmektedir. Bu bağlantı ve yol değişiklikleri, istenmeyen bir durum oluşturabilir ve tüm ağda daha fazla enerji ve kaynak kullanımına neden olabilir. Önerilen mimari yapı, ağın tamamının enerji durumu ve ya cihazların tekil enerji durumu anlık olarak takip edebilir. Sistem, paket yönlendirme yolunu değiştirerek veya cihazların uyku sürelerini değiştirerek fazla enerji tüketiminin kullanımının önüne geçebilir. Böylece enerji verimli bir yapı oluşması sağlanır.

Bazı kablosuz cihazlarda dışarıdan gelen gürültünün durumuna göre uyguladıkları iletim gücü, kablosuz kanalı değiştirmek, vb. gibi fonksiyonel özellikler bulunmaktadır [17] [18] [19]. Önerilen sistem bu tür işlevsel işlemleri yine bulut içerisinden sağlar, bu nedenle bu kanal değişimi için kullanılan kaynakların cihazlarca tüketilmesinin önüne geçilebilmektedir. Problem anında yönlendirme protokolü manipüle edilebilir ve ya

cihazların uyku aralıkları değiştirilebilir. Kenar yönlendirici üzerindeki yönlendirme tablosu alınarak 6LoWPAN ağının tüm topolojisi çıkarılabilir. Manuel olarak müdahale gerektiren ve çok fazla gürültüye maruz kalan cihazlar belirlenebilir. QoS verileri alınır ve problemli linkler tespit edilir. Sistem içerisinde yer alan scriptler ile gürültü durumunda sensörlerde kanal değişimi yapılarak kaliteli veri transferine olanak sağlanır.

Önerilen çalışmada, ağı güvenlik saldırılarına karşı koruyacak olan mekanizmaların mimari sistem tarafından üstlenilmesi sağlanmıştır. Dağıtılmış güvenlik hizmet kesintisi saldırıları, paket manipüle etme, vb. gibi güvenlik saldırıları, sistem tarafından sürekli olarak hesaba katılır ve bir problem olup olmadığı ağ geçidinden iletilen trafik miktarı kontrol edilerek yorumlanabilir. Ayrıca yetkisiz erişimlerin reddi ve erişim türü kısıtlamaları da yine bulut üzerinden yapılarak sistemin erişim kontrolü gerçekleştirilebilir

Bu tez çalışmasında, IEEE 802.15.4 iletişimini sağlamak amacıyla Openlabs [20] firmasına ait modüller Raspberry PI [21] cihazlarına eklenmiştir. Bu modüller standart yapıda çalıştığı için kapasitesi daha büyük veya daha küçük olsun tüm ekipman tiplerinin bu iletişimi sağladığı varsayılmaktadır [22]. Daha küçük işlemci ve hafızaya sahip olan sensörlerde de benzer iletişim modülleri bulunmaktadır. SNMPv3 programı olarak Net-SNMP açık kaynak kodlu program kullanılmıştır [23]. Bu program üzerinde değişiklikler yapılmıştır. Yine aynı şekilde cihaz kapasitesinden bağımsız olarak tüm cihazların SNMPv3 protokolünü çalıştırması ve belirtilen değişikliklerin kaynak kodlar üzerinde yapılması ile geliştirdiğimiz sistem bu cihazlar üzerinde de uygulanabilir. Bu varsayımlar geliştirilen sistem üzerinde performans problemi oluşturmamaktadır ve bir sınırlılık olarak karşımıza çıkmamaktadır. Standart SNMPv3 çalıştıran tüm ekipman tiplerinde ilgili yönetim sistemi problemsiz çalışabilir.

İlgili Literatür

6LoWPAN ağlarının yönetimi konusunda literatürde birçok çalışma bulunmaktadır. Yönetim sorununu problemini bazı araştırmacılar ağ geçitlerinin yönetim protokolü çeviricisi olarak kullanılması önermişlerdir. Bu yaklaşımda 6LoWPAN ağ geçidine kadar standardı bulunan bir yönetim protokolü, ağ geçidinden sonra 6LoWPAN ağı içerisinde hafif sürüm bir yönetim protokolünün kullanılması önerilmektedir. Mukhtar ve arkadaşları (2008) ağ geçidine kadar SNMP protokolünün kullanılmasını

önermektedirler [24]. Bu çalışmada, 6LoWPAN ağı içerisinde kendi geliştirdikleri ve SNMP ile uyumlu çalışan bir protokol ile yönetimsel işlemler gerçekleştirilmektedir. Lindholm-Ventola ve arkadaşları (2014) ağ geçidine kadar SNMP protokolünün kullanılmasını ve 6LoWPAN ağı içerisinde ise CoAP protokolünün kullanılmasını önermektedirler [25]. 6LoWPAN ağ geçidi SNMP ile CoAP protokolü arasında eşleme yapmaktadır ve aynı zamanda bir protokol çevirici gibi davranmaktadır. Sheng ve arkadaşları (2015) CoAP ve HTTP protokollerinin arada bir çevirici kullanılarak 6LoWPAN ağının yönetimi için kullanılmasını incelemiştir [26]. Bu çalışmada, ağ geçidine kadar CoAP protokolü kullanılırken ağ içerisinde kendi geliştirdikleri ve http tabanlı çalışan bir mesajlaşma yöntemi önerilmiştir. Ağ geçitlerinin çevirici olarak davranması yönetim protokolü oturumlarının kesintisiz olmasının önüne geçmekte ve aynı zamanda bilgi iletiminin birebir işlenmesi konusunda risk içermektedir. Aynı zamanda iki protokolün çevirici ile birlikte kullanılması standartlaşmanın önüne geçmektedir.

IPv6 ile uyumlu çalışan mevcut yönetim protokollerinin 6LoWPAN gibi protokollere dayanan IoT ağlarına ne şekilde entegre edileceği de belirlenmelidir. SNMP, basitleştirilmiş MIB yapıları ve CoAP tabanlı yönetim metotları uygulanabilecek çözümler arasında sunulmaktadır [27]. Bir başka çalışmada araştırmacılar, mevcut IoT protokollerinin hafif sürümlerinin düşük güçteki kısıtlı ağların ve cihazların yönetimi için kullanılmasını önermişlerdir [28]. Mevcut IP tabanlı düşük kapasiteli ağların yönetimini SNMP ve NETCONF protokolleri ile düşük kaynak gereksinimleri sağlanarak 8-bit işlemcisi olan cihazlarda uygulanabileceğinden bahsedilmiştir. Benzer şekilde başka bir çalışmada, Lamazzi ve arkadaşları (2014) kısıtlı kapasiteli cihazların yönetimsel işlemlerindeki zorlukları incelemiş ve kullanılabilecek olan tüm uygun protokolleri cihaz yönetimi için değerlendirmişlerdir [29]. Bu çalışmada, 6LoWPAN ağlarının yönetimi için, SNMP dâhil olmak üzere bilinen tüm IoT protokollerinin kullanımı değerlendirilmiş ve bu protokoller üzerinde daha fazla geliştirme yapılması gerekliliğinden bahsedilmiştir. Bu tez çalışmasında benzer şekilde SNMPv3'ün hafifletişmiş bir uyarlanması kullanılmaktadır.

Mevcut protokollerin doğrudan 6LoWPAN ağlarının yönetimi için uygulanması ile ilgili çalışmalar bulunmaktadır. Karaman ve arkadaşları (2015) yaptıkları çalışmada CoAP protokolünü yönetimsel amaçla kullanmışlardır [30]. Yönetimsel verilerin

taşınması için bir yöntem olsa da CoAP protokolü daha çok uygulama bilgilerini taşımak için uygun bir yöntemdir. Yönetimsel faaliyetler için özel bir veri tabanı şeması içermemektedir [31]. Bunun yanı sıra, başka bir çalışmadan 6LoWPAN yönetimi için SNMPv3 doğrudan kullanılan protokol olmuştur [32]. Bu yaklaşımda SNMPv3 yüksek bant genişliği kullanarak, kısıtlı ağlar için önemli olan bant genişliğini kullanmaktadır.

Bazı çalışmalar SNMP protokolünün hafifletilmiş ve uyarlanmış sürümlerinin oluşturulması ve 6LoWPAN ağlarında kullanılmasında odaklanmıştır. Shanthini ve arkadaşları (2012) SNMP'nin ikinci sürümünü [33], Choi ve arkadaşları (2009) ise SNMP'nin birinci sürümünü uyarlayarak [34], 6LoWPAN ağında yönetim için kullanmışlardır. Bu tez çalışmasının ikinci kısmı olan protokol uyarlaması kısmında benzer bir yaklaşım bulunsa da, güvenlik özellikleri nedeni ile SNMPv3 protokolünün uyarlanması üzerine çalışılmıştır. Önerilen mimaride, SNMPv3 protokolünü kısıtlı bağlantılarda entegre etmek için SNMPv3 başlığını sıkıştıracak bir yaklaşım mevcuttur. 6LoWPAN'de başlık sıkıştırma, genel olarak başlık içerisindeki belirli alanların ve üstbilgiye benzer alanların sıkıştırılmasını ve ya kaldırılmasına olanak sağlayan bir tekniktir. [35] Schoenwaelder ve arkadaşları (2010), SNMP protokolünde ne şekilde optimizasyon yapılacağı üzerine önerilerde bulunmuştur. [36]

6LoWPAN ağ yönetim sistemi tasarımı üzerine çalışmalar bulunmaktadır. Turon (2005), çalışmasında 6LoWPAN ağındaki nesneleri gözlemleyebilecek ve gerektiğinde cihazlara uzaktan müdahale imkânı veren bir sistem tasarlamıştır. [37] Bu sistemde kendi kendini yönetme yeteneği bulunmamakta ve operatöre bağlı kalan bir çalışma yapısı mevcuttur. Tuan ve arkadaşlarının çalışmasında (2008), yönetim sistemi içerisinde bazı kurallar olması gerektiğine ve bu kurallara bağlı olarak tüm ağın yönetilmesi üzerine çalışılmıştır [38]. Bu çalışmada hangi kuralların ağ yönetiminde kullanılacağı ve katmanlı bir yönetim sistemi yapısında da bahsedilmektedir. Zhang ve arkadaşları (2010), çalışmalarında yönetim sistemi tasarımı haricinden bir mimari yapı bulunması gerekliği üzerine öneride bulunmuştur. [39] Bu çalışmada mimari yapının kendi kendini organize eden bir yapıda olması gerektiği belirtilmiştir. Minh ve arkadaşları (2012), kendi kendini yönetebilen ve 6LoWPAN ağlarında kullanılan bir yönetim sistemi oluşturmuştur. [40] Bu tez çalışmasında kendi kendini organize eden bir yönetim sistemi bulunmaktadır. Daha önce yapılan çalışmalardan farklı olarak bu tez çalışmasındaki ağı yöneten kurallar SNMPv3 objelerinin durumlarını izleyerek çalışmaktadır. Benzer çalışmalarda bir

protokol yapısı bulunmamaktadır. Ayrıca bu tez çalışmasında oluşabilecek olan durumlara göre kullanılacak olan kuralların hangileri olacağı detaylı şekilde belirtilmiştir.

6LoWPAN ağlarının bulut üzerinden yönetilmesi konularında çalışmalar bulunmaktadır. Bulut üzerinden yönetim için çok sayıda kaynakları kısıtlı IoT cihazlarının farklı uygulama protokollerine sahip farklı API türleri ile yönetilmesi için bir çalışma mevcuttur. [41] Bu konu ile birlikte bu tez çalışmamsıdaki, 6LoWPAN ağlarının bulut içerisinde yönetimi değerlendirilebilir. Başka bir çalışmada bulut altyapısı içinden, sensör yönetiminin verimli ve güvenilir gerçekleşmesini sağlayan, hafif ve RESTful web hizmetlerine dayanan bir yönetim katmanı tasarımı yapılmıştır [42]. Bir başka çalışmada, bulut tabanlı yapının, 6LoWPAN yönetimi için otonom yöntemlerin kullanılmasına odaklanılmıştır [43]. Bu çalışmada otomatize yöntemler kullanılarak aynı zamanda ağdaki iletim kalitesi bozuklukları tespit edilerek güzergâh değişiklikleri yapılabileceği gösterilmiştir. Bu tez çalışmasında, bulut içerisinde yer alan 6LoWPAN yönetim sistemi, ağı önceden hazırlanmış kurallara bağlı olarak scriptler ile yönetmektedir.

Katkılar

Bu tez çalışması ile "A Management Model for Low Powered Wireless Personal Area Networks" isimli bildiri ile IEEE 19th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2018) konferansına katkıda bulunulmuştur. "Management of Low Powered Personal Area Networks using Compression in SNMPv3" isimli bildiri ile ACS/IEEE 15th International Conference on Computer Systems and Applications (AICCSA 2018) konferansına katkıda bulunulmuştur. "A Self-Organizing Management Architecture for Low Powered Wireless Personal Area Networks", isimli yazı Springer Wireless Personal Communications isimli dergiden yanıt beklemektedir.

Ayrıca, tez çalışması sonucunda Türk Patent'e iki adet patent başvurusunda bulunulmuştur. TR2017/15714 sıra numaralı "Method and System for Cloud Management in LoWPANs" isimli ve TR2017/20155 sıra numaralı "System for Self-organizing Wireless Mesh Networks" isimli patent başvuruları yanıt beklemektedir.

BÖLÜM 2. YÖNTEM

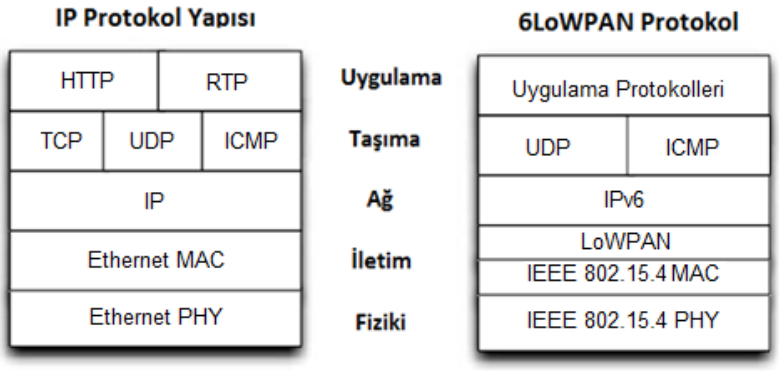
Bu bölümde, çalışmada kullanılan yöntem hakkında bilgilere yer verilmiştir. Öncelikle 6LoWPAN ağ yapısı, SNMPv3 ve ilgili SNMPv3 MIB türleri incelenmiştir. Sonrasında benzer bir SNMPv3 MIB modülü oluşturularak bunun kendi kendini organize edebilen bir 6LoWPAN yönetim sistemi için kullanılması sağlanmıştır. MIB türü benzer olsa da içerisinde bulunduracağı SNMPv3 objelerinin minimum sayıda olması ve sensörler içerisinde saklama alanı işgal etmemesine çalışılmıştır. Çalışmanın sonraki kısmında, bulut yapısı içerisinde çalışacak olan yönetim sistemi önerilmiş ve önerilen yönetim mimarisini oluşturmak için gerekli tüm bileşenler ortaya konulmuştur. Bulut içerisindeki yapıyı çalıştırmak için ilgili yorumlayıcı programlama dilleri incelenmiş ve ağ programlama konusunda daha yetenekli görünen PERL dili ile ilerlenmesine karar verilmiştir. Çalışacak olan sistemde bir veri tabanı ihtiyacı da bulunmaktadır. Bu konuda veri bütünlüğünü koruması nedeniyle My-SQL veri tabanı seçilmiştir.

6LoWPAN Protokolü

6LoWPAN protokol yapısı ile normal IP protokol yapısı hemen hemen aynıdır. 6LoWPAN, yalnızca IPv6'yı desteklemektedir. Bunun için 6LoWPAN adaptasyon katmanı adı verilen, IEEE 802.15.4 üzerinden IPv6 çalışmasını sağlayacak küçük bir adaptasyon katmanı tanımlanmıştır. [44]

Pratikte, gömülü aygıtlardaki 6LoWPAN için, 6LoWPAN adaptasyon katmanı ve IPv6 birlikte uygulanmaktadır. Böylece bu iki katman, ağ katmanının bir alternatifi olarak gösterilir. 6LoWPAN ile kullanılan en yaygın taşıma protokolü, aynı zamanda 6LoWPAN formatı ile de sıkıştırılabilen UDP'dir. TCP, performans, verimlilik ve karmaşık yapısı nedenleriyle 6LoWPAN'da genellikle kullanılmaz. Erişim kontrolü ve komşu bulma iletileri için ICMPv6 kullanılır. Uygulama protokolleri, genellikle uygulama özelinde farklılaşmaktadır. Zamanla daha fazla standart uygulama protokolü kullanılabilir hale getirilmektedir.

IPv6 ile 6LoWPAN biçimi arasındaki dönüşüm, kenar yönlendiriciler olarak adlandırılan ve 6LoWPAN sensör alanlarının kenarında konumlandırılan yönlendiriciler tarafından gerçekleştirilir. Kenar yönlendiricilerdeki dönüşüm, her iki yöne de yapılmaktadır. Şekil-1'de de görüleceği üzere, 6LoWPAN adaptasyonu bir kenar yönlendiricide 6LoWPAN ağ arabirimi sürücüsünün bir parçası olarak gerçekleştirilir. IPv6



Şekil-1 IP ve 6LoWPAN Protokol Yapıları

protokol yapısı açısından bu dönüşüm saydamdır. 6LoWPAN çalışan sensör ağının içerisindeki tüm sensörler ve kenar yönlendiriciler ana bilgisayarlar ve yönlendiriciler sıkıştırılmış alanların hangileri olduğunu bilmektedir. Böylece ağ içerisinde, IPv6 veya UDP üstbilgi formatlarıyla herhangi bir noktada çalışmaya ihtiyaç duymazlar.

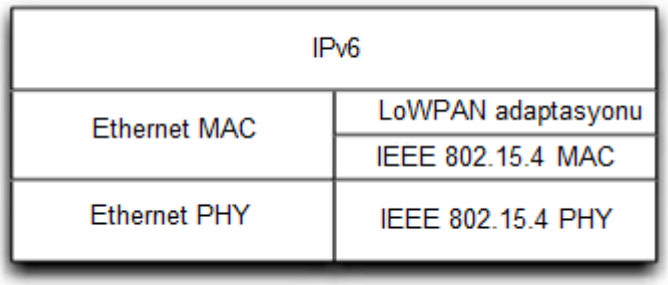
IEEE 802.15.4, 6LoWPAN paketlerini tanımlayan herhangi bir alan içermez. 6LoWPAN paketleri diğer tüm veri paketlerinden ve ya farklı türdeki 6LoWPAN paketleri birbirlerinden ayırt edilememektedir. Bu nedenle, 6LoWPAN adaptasyon katmanının ilk görevi, IEEE 802.15.4 tarafından 6LoWPAN paket türlerini ayırt edecek bir paket türü tanımlayıcısı sağlanmaktadır. Bu nedenle yükün ilk baytı gönderme baytı olarak kullanılır. Tablo-1'de farklı paket türleri için gönderme baytının değerleri belirtilmektedir. Gönderim baytı normal bir IPv6 paketine Şekil-2'deki gibi eklenmektedir.

Başlık Sıkıştırma ve 6LoWPAN

Normal bir IPv6 başlığının boyutu IEEE 802.15.4 tarafından sağlanan sınırlı paket boyutunun yarısını tüketmektedir. Parçalama ve yeniden birleştirme teknikleri kullanarak daha büyük paketler gönderilebilirken, 6LoWPAN tüm IPv6 paketlerini tek bir IEEE 802.15.4 paketine verimli bir şekilde yerleştirmektedir. Paketler parçalanmayı ve yeniden birleştirme gerektirmeyecek kadar küçük olabilir. Ancak bu durumda da, IEEE 802.15.4 ağlarında büyük IPv6 başlıklarının kullanılmak istenmemesinin nedenleri, pil ömrü süresi

00	6LoWPAN Paketi Değil
01	Normal Yollama
10	Mesh Başlığı
11	Fragmantasyon Başlığı

Tablo-1 Gönderim baytı içindeki ilk iki bitin değerleri

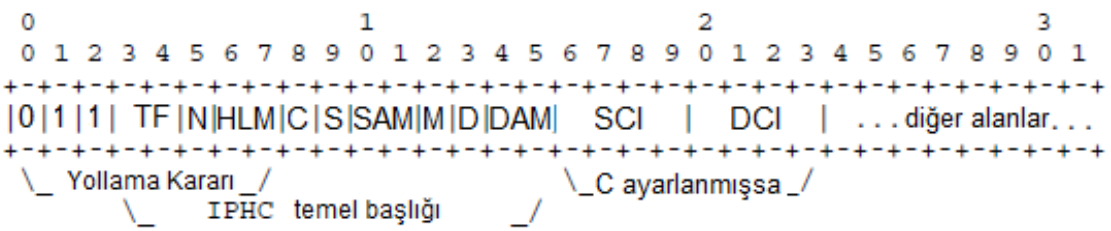


Şekil-2 IPv6 ile 6LoWPAN Dönüşümü Sağlayan Kenar Yönlendirici Protokol Yapısı

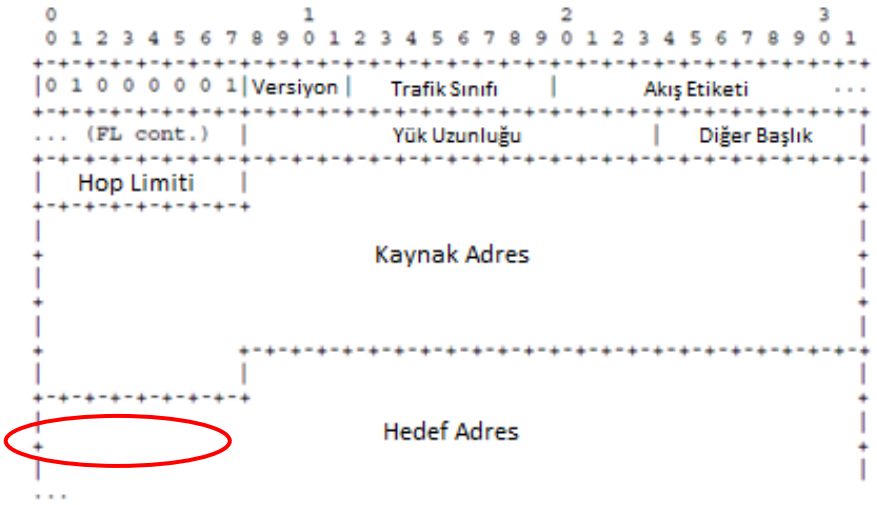
ve oldukça sınırlı kanal kapasitesinin sürekli olarak meşgul edilmemesi gerekliliğidir [45].

Ağ üzerinde defalarca gönderilen başlık bilgilerin büyük kısmı gereksizdir, yani sıkıştınlabilir. Örneğin, bir ağ üzerindeki son atlama sonrası ulaşılacak bir ana bilgisayara yapılan son atlamada, IPv6 hedef adresi alanı çok küçük bir set ile ifade edilebilir. Bu nedenle, son kullanıcı ve ev sahibinin anlaşması ile oluşturulan, 128 bitlik tam bir IPv6 başlığının tamamına ihtiyaç bulunmamaktadır. Gönderim baytının formatı Şekil-3'de gösterilmektedir.

Sıkıştırma işleminin IPv6 başlığındaki hangi alanları için yapıldığını belirten sıkıştırma başlığıdır. Sıkıştırma başlığı ve bu başlıkta bulunan alanlar Şekil-4'de gösterilmektedir. Bu başlık için 13 bitlik alan yeterlidir. 2 octetlik alan içerisinden 13 bit kullanılır. C biti (CID), kaynak ve hedef adres için context bilgisinin olduğu, üçüncü bir bayt eklenip eklenmediğini kontrol eder. C ayarlanmazsa, her ikisinde de "varsayılan değer" olan sıfır kullanılır. Yeni başlık formatında, IPv6 versiyon alanı tamamen çıkarılmıştır. N bitine değer atanmışsa, UDP başlığına da sıkıştırma uygulanır.



Şekil-3 6LoWPAN Gönderim Baytı Eklenmiş bir IPv6 Paketi

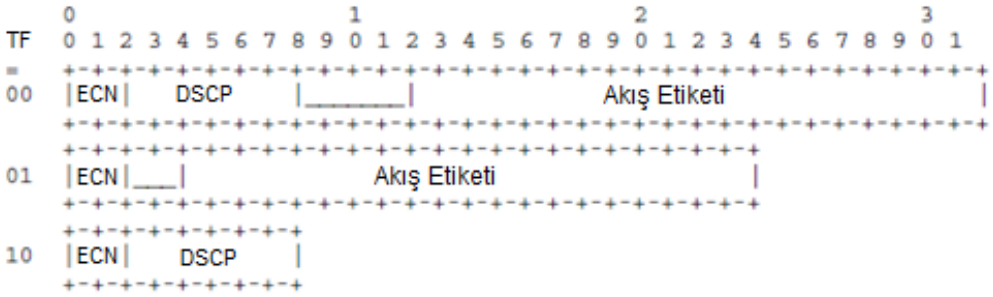


Şekil-4 6LoWPAN Sıkıştırma Başlığı

IPv6 başlığında bulunan "hop limit" alanı tamamen çıkarılmamıştır. Normalde bu alan 8 bittir ve en yüksek değer 255 olmaktadır. 6LoWPAN başlığında HLM alanına iki bit ayrılmıştır. 2 bitlik alandaki kodlama ile hop limit değeri 1,64 ve ya 255 olarak atanmaktadır.

TF bitleri, IPv6 başlığındaki trafik sınıfının ve akış etiketinin nasıl işleyeceğini kontrol eder. Trafik tıkanıklığı için ECN alanı, trafik sınıfı için DSCP alanı ve akış kontrolü alanları normal IPv6 başlığında her zaman gönderilmektedir. TF değeri 00 ise her üç alan da gönderilmektedir. ECN değerine ek olarak, TF değeri 01 ise, akış etiketi, TF değeri 10 ise trafik sınıfı da gönderilir. TF değeri 11 ise, tüm alanlar tamamen ortadan kaldırılabilir. Şekil-5'de bu değerlere karşı gönderilecek olan alanlar gösterilmiştir.

Kaynak ve hedef adreslerin sıkıştırılması sırasıyla S/SAM ve D/DAM tarafından kontrol edilir. Özel bir bayrak olan M biti ayarlanmış ise, hedefin adres birçoklu yayın adresidir. Tek noktaya yayın için alanlar şu şekilde tanımlanmalıdır: S alanı ve D alanı, ilgili adresin sıkıştırılmasının içeriğe dayalı olup olmadığını kontrol eder. SAM / DAM seçicileri, bu bitlere bağımlı alt modları kontrol eder. Bu alt modların bir kısmı, IPv6



Şekil-5 6LoWPAN Başlığı Trafik Sınıfı ve Akış Etiketinin Sıkıştırılması

S/SAM	Sıra İçerisinde	
0 00	128	Sıralı (adresin tamamı sıralı şekildedir, sıkıştırma yok)
0 01	64	FE80:0:0:0:sıra (link lokal + 64 bit arayüz ID)
0 10	16	FE80:0:0:0:0:0:sıra (link lokal + 16 bit kısa adres)
0 11	0	FE80:0:0:0:iletim (link lokal + iletim katmanı kaynak adresi)
1 00	-	rezerve
1 01	64	İçerik[0..63]:sıra(içerik + 64 bit arayüz ID)
1 10	16	İçerik[0..111]:sıra(içerik + 16 bit kısa adres)
1 11	0	İçerik[0..127] (içerik)

Tablo-2 M Sıfır Olduğu Durumda S/SAM ve D/DAM Değerleri

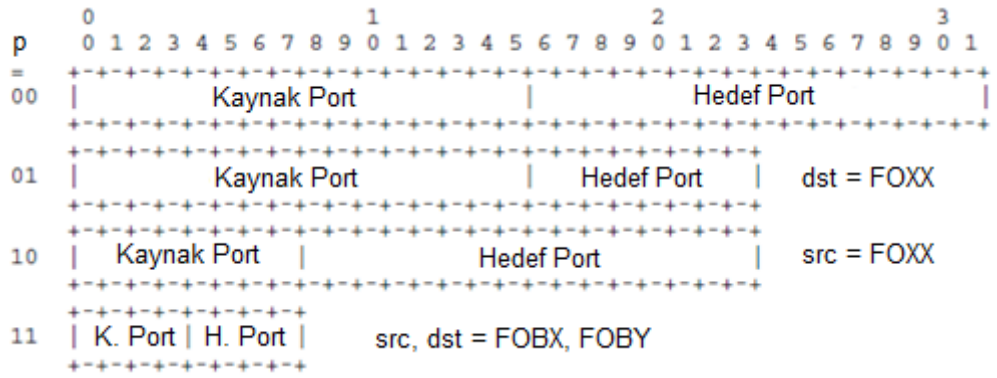
başlığının diğer sıkıştırılmamış kısımları ile birlikte taşınır. Daha sonra sabit bitler ile birlikte, adresi yeniden oluşturmak için kullanılır. Bu alan için ilgili değerler ve bu değerlere karşılık gelen açıklamalar Tablo-2'de gösterilmektedir. S/SAM değeri kaynak IPv6 adresindeki sıkıştırma oranını belirtir. D/DAM değeri ise hedef IPv6 adresindeki sıkıştırma oranını açıklamaktadır.

P alanı, UDP'nin kaynak ve hedef alanlarının sıkıştırılmasını kontrol eder. P değeri 00 ise bu alanlar tamamen gönderilir. En önemli bitler port numarası olarak yuvarlanabilir, 61616 değeri 0xF0B0 olarak değiştirilir. P alanındaki iki biti kullanarak, kaynak ve hedef port numaraları için sıkıştırma işlemi bağımsızca kontrol edilebilir.

6LoWPAN Ağının Çalışması

Bu bölümde, 6LoWPAN'ın pratikte önyükleme ve çalıştırma sırasında oluşan temel aşamalar kısa bir örnekle belirtilmiştir. Basit bir 6LoWPAN ağında, IPv6 İnternet'e bağlanmak için bir kenar yönlendiricisi, üç adet 6LoWPAN yönlendiricisi (R) ve üç adet 6LoWPAN sensörü (H) bulunmaktadır. Buna ek olarak, internette bir uzak sunucu mevcuttur. 6LoWPAN ağında IEEE 802.15.4 ve IPv6 yönlendirme kullanır. Aşağıdaki adımları izlemeyi kolaylaştırmak için, düğümlerin sahte IPv6 alt ağ önekleri ve adresleri kullanılmıştır.

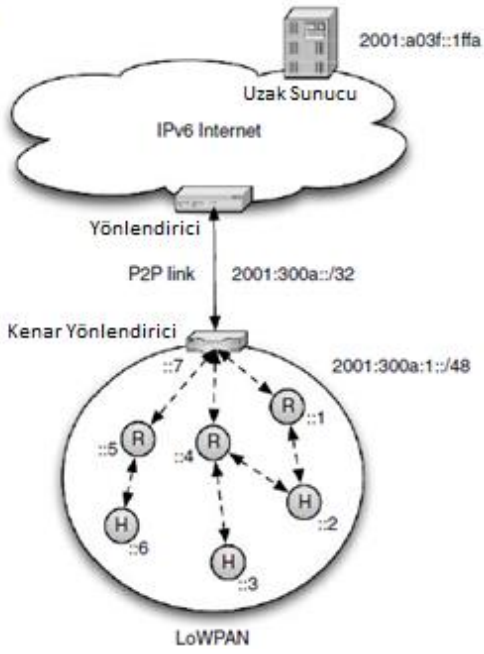
İnternete çıkan yönlendirici, ana hat bağlantısından otomatik adres konfigürasyonu için 2001: 300a :: /32 önekini duyurur. Kenar yönlendirici, 2001: 300a: 1 :: /48 IPv6



Şekil-6 Diğer Başlığın Sıkıştırıldığı Durumda Port Numaralarının Sıkıştırılması

önekini IEEE 802.15.4 kablosuz arabirimine yapılandırır. 6LoWPAN ve ana hat bağlantısı, farklı alt ağlardadır. 6LoWPAN'daki IEEE 802.15.4 kablosuz aygıtlar varsayılan bir kanal ve güvenlik anahtarı ayarları kullanırlar. Üç yönlendirici tarafından otomatik adres konfigürasyonu gerçekleştirilmesi için kenar yönlendirici, IPv6 ön ekinin duyurur. Üç yönlendirici kenar yönlendiriciye, 6LoWPAN komşu bulma metodunu kullanarak kaydolur. Her 6LoWPAN düğümünün kendi oluşturduğu 64 bit IID'si bulunmaktadır. Buna ek olarak, kenar yönlendiriciye kayıt esnasında alınan 16-bitlik IID ile birlikte, tam bir IPv6 adresi oluşturulur. IPv6 adresinin IID kısmı Şekil-6'da gösterilen:: 1 ve tam IPv6 adresi ise 2001: 300a: 1 :: 1 şeklindedir. Kenar yönlendiricinin verdiği IID'ler 16-bitlik rasgele sayılardan oluşturulmaktadır. Üç yönlendirici, aynı öneki ağ içerisinde duyurarak, üç sensörün kenar yönlendiriciye kayıt olmasını sağlar. 6LoWPAN ağındaki topoloji değişiminden, düğümlerin IPv6 adresleri etkilenmez.

Kenar yönlendiriciye kayıt olunmasını sağlayan ve yönlendiricileri duyuran komşu bulma trafiği, aynı zamanda yönlendirme algoritmasını da başlatır. Şekil-7'de örnek güzergâhlar kesik çizgilerle gösterilmiştir. Kesik çizgilerle gösterilen güzergâhlar üzerinden sensörler atlama yaparak kenar yönlendiriciye erişmektedir. Kenar yönlendirici ise geniş alan ağındaki bir yönlendiriciye doğrudan p2p bir bağlantı ile erişmektedir. Bu kenar yönlendirici ise internet üzerinden uzak bir sunucuya erişebilmektedir. Bu yapıda uzak sunucu IPv6 adresi kullanmaktadır. Eğer uzak sunucu IPv6 adresi dışında bir IP adresi kullanıyorsa bu durumda kenar yönlendiricide IPv6'dan ilgili adres tipine bir dönüşüm işleminin yapılması gerekmektedir. Bağlantı katmanı, aynı ağ içerisindeki kaynak ve hedef bilgisine sahiptir 6LoWPAN Düğümlerinin iletişimi sırasında IPv6 kaynak ve hedef adreslerinin tamamına ihtiyaç duyulmaz. (ör. :: 6'dan :: 5'e) Bir paket birden çok atlamalı



Şekil-7 Bir 6LoWPAN Ağı

olarak yönlendirilirse, sadece 16 bitlik kaynak ve hedef adresleri taşınır ve bu adresler paketi yönlendirmek için kullanılır. 6LoWPAN'ın dışına gönderilen paketler için ya tam bir IPv6 hedef adresi ve ya o adres için sıkıştırma içeriği bildirilirse, sıkıştırılmış bir 6LoWPAN adresi gerekmektedir. Örneğin, adresi 2001:300a:1: :6 olan 6LoWPAN sensörü adresi 2001:a03f: :1ffa olan uzak sunucuya bir paket göndersin. Kenar yönlendirici, sıkıştırılmış 6LoWPAN ve IPv6 adresini genişleterek, tam bir IPv6 başlığı haline getirir. Eğer UDP için sıkıştırma işlemi yapılmışsa, UDP üstbilgisi de başlığa eklenir. Gelen paketler ise kenar yönlendiricide işlenir. IPv6 ve UDP başlıklarında, mümkün olduğu kadar sıkıştırma işlemi yapılır.

6LoWPAN Bant Genişliği ve İletim Gecikmesi

Bir IEEE 802.15.4 ağında maksimum kullanılacak bant genişliği TP olarak adlandırılsın. Bu durumda linkteki gecikme aşağıdaki şekilde hesaplanır. [53].

$$TP = 8.\ (x)\ /\ \text{delay}\ (x) \qquad \textbf{(1)}$$

Bu denklemde, x teslim edilen baytların sayısıdır. Her paketin gecikmesi aşağıdaki şekildedir.

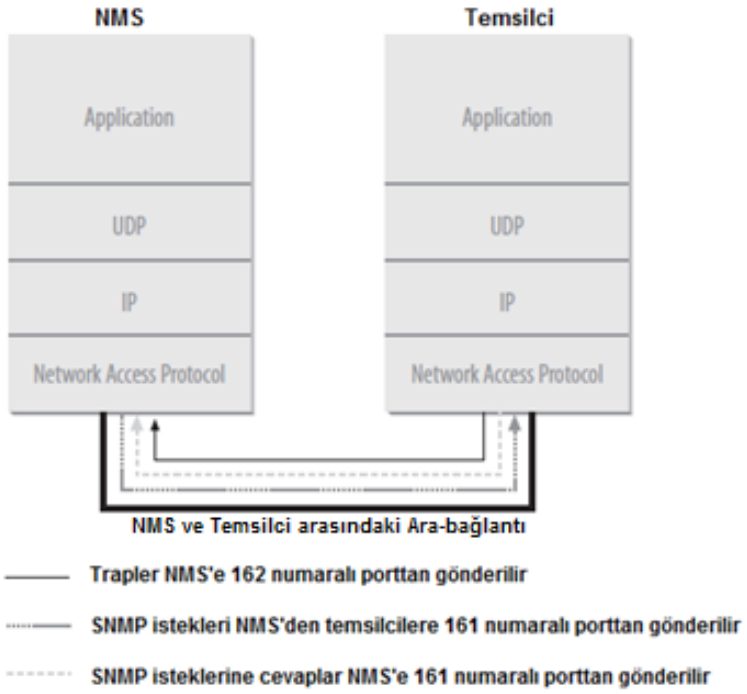
$$delay(x) = T_{BO} + T_{frame}(x) + T_{TA} + T_{ACK} + T_{IFS}(x) \quad \textbf{(2)}$$

T_{BO}, geri dönüş süresini gösterir. T_{frame} (x) zamanı bir çerçevenin *x* baytını iletmek için gerekli süredir. T_{TA} (*x*) ağ içerisindeki dönüş zamanıdır. T_{ACK} ise onay mesajının aktarım zamanı olarak ifade edilir. Son olarak, T_{IFS} (*x*) IEEE 802.15.4 orta erişim katmanındaki çerçeveler arası boşluk zamanıdır.

SNMP Mimarisi

SNMP, yöneticiler ve temsilciler arasındaki verilerin aktarımı için kullanıcı datagram protokolünü kullanır. RFC 768'de tanımlanan UDP, iletim denetimi protokolü gibi uçtan uca bağlantı kurulmasını gerektirmeden, temsilciler ile Ağ Yönetim İstasyonları (NMS) arasında bağlantı kurulmasını sağlar. Az sayıda komut içermesi, SNMP'nin "basit" olmasının nedenlerinden sadece bir tanesidir. Diğer basitleştirici faktörler, denetlenmeyen veya bağlantısız bir iletişim bağlantısına olan bağımlılığıdır. Bu basitlik, özellikle internet ağ yönetimi çerçevesi içerisinde yaygın kullanımına yol açmıştır. Bu çerçevede, yöneticilerin temsilcilerden tamamen bağımsız olarak şebekede yer alması nedeniyle bu yapı sağlam bir yapı olarak kabul edilir. Örnek olarak, bir SNMP temsilcisinde sorun oluşsa bile hem temsilci diğer fonksiyonlarına devam edecek hem de yönetici bu durumdan etkilenmeyecektir. Protokol seviyesinde uçtan uca bağlantı olmaması, kayıp paket ve yeniden gönderim kontrollerinin eksikliği, UDP'yi güvenilmez kılmaktadır. Verinin kaybolup kaybolmadığını belirlemek ve eğer kaybolduysa tekrar iletilmeleri sağlamak, SNMP uygulamasındaki basit bir zamanaşımı kuralı ile gerçekleştirilir. NMS bir UDP isteği gönderir. Temsilci bir yanıt bekler. NMS'in bekleyeceği süre önceden belirlenmiştir. Zaman aşımına varılırsa NMS, şebekede paketin kaybedildiğini varsayar ve isteği tekrar gönderir. Kayıp Paketleri tekrar gönderim sayısı NMS'de yapılandırılabilir. [46]

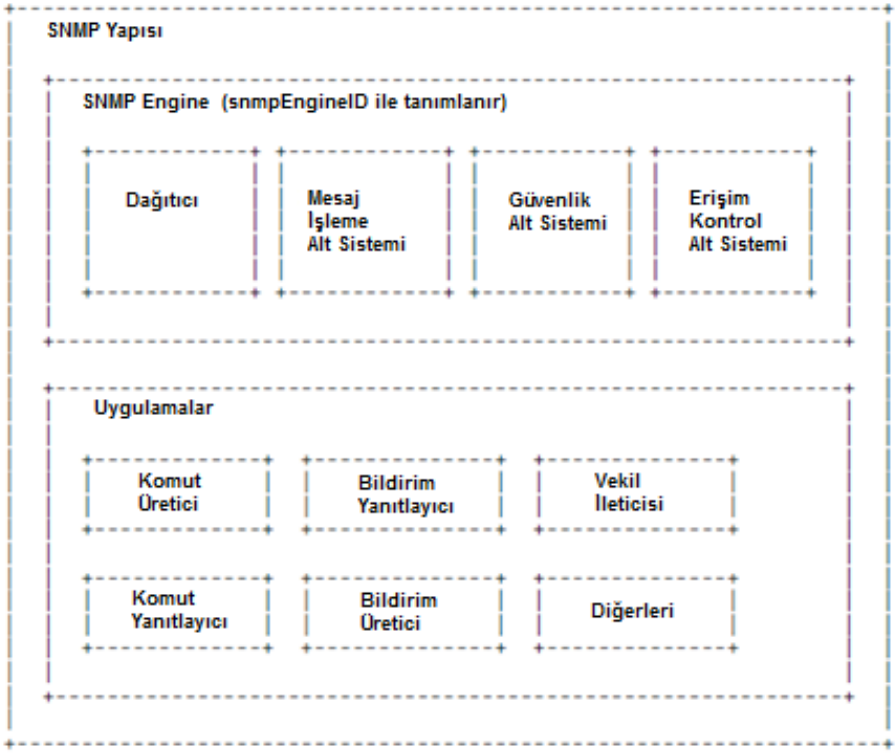
UDP'nin güvenilmez niteliği, normal bilgi talepleriyle ilgili olarak gerçek bir sorun değildir. En kötü ihtimalle NMS bir istekte bulunur ve hiçbir zaman bir yanıt almaz. Trap'ler için ise durum biraz farklıdır. Bir temsilci bir trap oluşturur ve trap asla NMS'e ulaşmaz, NMS'in, o ana kadar kaç adet trap gönderildiğini bilmesinin bir yolu yoktur. UDP'nin güvenilmez doğasına ters olarak, düşük bir yük gerektirdiğinden, ağın performansına olumsuz etkisi bulunmamaktadır. SNMP, TCP üzerinden de uygulanmıştır,



Şekil-8 TCP/IP ve SNMP

ancak ağır yoğunlukta olan bir ağ için, TCP üzerinden SNMP performansı etkileyebilir. Ayrıca SNMP, sorunlu şebekelerle çalışmak üzere tasarlanmıştır. Şekil-8, tüm TCP / IP iletişiminin temelini oluşturan TCP / IP protokol yapısı ve SNMP ilişkisini göstermektedir. SNMP, istekleri göndermek ve almak için UDP bağlantı noktası 161 ve yönetilen cihazlardan gelen trap'leri almak için 162 numaralı bağlantı noktasını kullanır. SNMP'yi uygulayan her cihaz, bunları kullanmalıdır. Port numaraları varsayılan olarak kullanabilir ve ya bazı satıcılar varsayılan port numaralarını değiştirebilir.

SNMP mimarisi ile SNMP öğelerinin hangi bileşenleri içermeleri gerektiği belirtilmiştir. Bu bileşenler Şekil-9'da gösterilmektedir. Her SNMP yazılımında bu mimari olmalıdır. Her bir SNMP öğesinde, bir SNMP Motoru (SNMP Engine) ve bir veya daha sayıda uygulama olabilir. SNMP motoru, mesaj gönderimi ve alınmasını, mesajların doğrulanmasını, şifrelemesini ve ayrıca yönetilen nesnelere erişimin kontrolünü sağlar. Bir SNMP motoru ve SNMP öğeleri arasında birebir bir ilişki vardır. Her öğe bir SNMP motoru içerir ve bu motora ait bir kimlik numarası tanımlıdır. Bir SNMP motoru, dağıtıcıyı, mesaj işleme alt sistemini, güvenlik alt sistemini ve erişim kontrolü alt sistemini içerir. Bir yönetim alanı içinde her snmpEngineID kimlik numarası birbirinden farklıdır. SNMP motorunun açık ve net olarak tanımlar. Bir SNMP motorunda yalnızca bir dağıtıcı vardır.



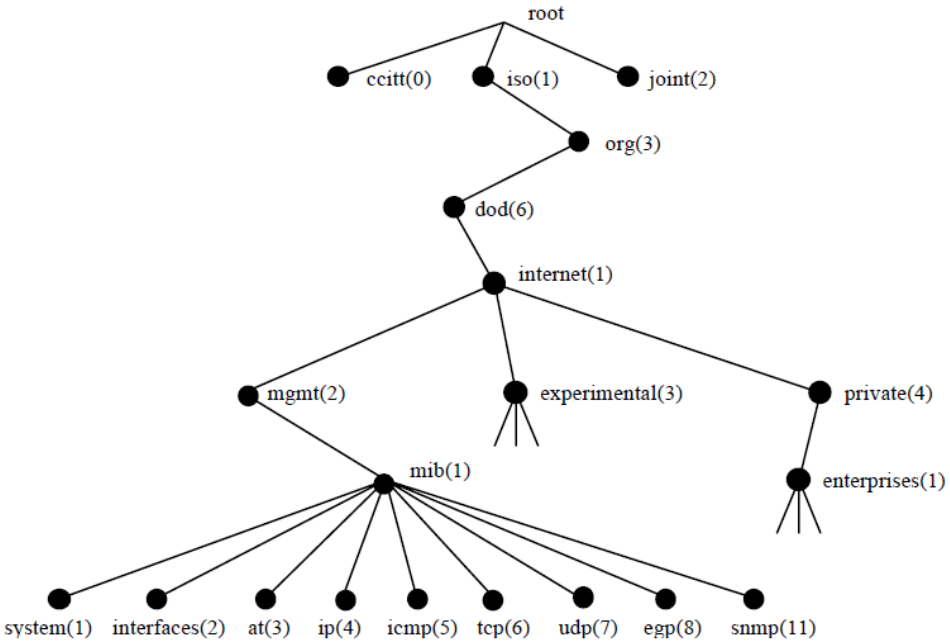
Şekil-9 Bir SNMP Öğesi ve Bileşenleri

SNMP motoru ile birden çok SNMP iletisinin eşzamanlı çalışmasını sağlar. Ağa SNMP mesajları gönderilmesi ve ağdan mesajların alınması için arayüz oluşturur. Bir SNMP mesajının sürümünü belirler ve ilgili mesaj işleme modeline iletir.

Mesaj İşleme Alt Sistemi, alınan ve gönderilen mesajları işlemekten sorumludur. Her SNMP sürümüne göre kendi içerisinde farklı mesaj işleme modelleri içerir. Güvenlik Alt Sistemi, kimlik doğrulama ve mesaj gizliliğini sağlar. Erişim Kontrolü Alt Sistemi yetkilendirme servisleri sunmaktadır. Uygulamalar, yönetimi izleyen ve değiştiren komut üreticileri, yönetim verisine erişim sağlayan komut yanıtlayıcıları, zaman uyumsuz mesajları başlatan bildirim üreticileri, zaman uyumsuz mesajları işleyen bildirim yanıtlayıcıları ve öğeler arası ileti sağlayan vekil ileticileridir. [47]

SNMP MIB'leri ve Protokol Operasyonları

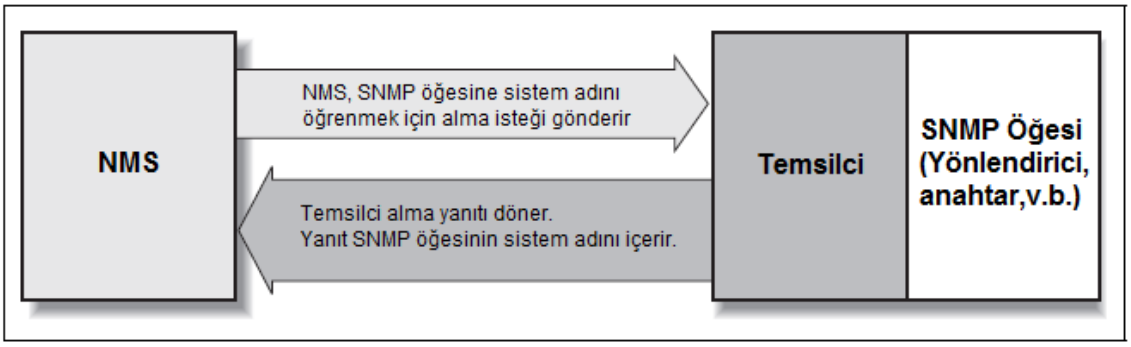
MIB'ler yönetilecek olan nesneleri SMI adlı bir çerçeveyi kullanarak tanımlamaktadır. SMI, yönetim bilgisinin nasıl gruplandırıldığını ve adlandırıldığını, izin verilen işlemleri, izin verilen veri türleri belirler. SMI, bir veri tabanı sistemi şemasına benzemektedir. SMI, yönetilen nesnelerin modelini ve nesneler üzerinde gerçekleştirilebilen işlemleri, ayrıca nesneler için izin verilen veri türlerin, belirler. Yönetilen nesneler var olan sistemlerdeki



Şekil-10 SNMP OID Ağacı

kaynakların soyutlanmış halidir. MIB'ler nesneleri gerçekten tanımlamaz. Bazı nesneler (örneğin bir sistemin konumu gibi) yalnızca bir durumu belirtirken, diğerleri (ağ bağlantıları gibi) birden fazla durumu ifade etmektedir. Aynı nesneye sahip ilgili nesneler, SNMP MIB'lerinde kavramsal tablolarda ifade edilir. Nesne ve nesnenin durumu, SNMP değişkeni olarak adlandırılır. Nesneler, SNMP ağacında, bir nesne tanımlayıcısı (OID) atayarak açıkça tanımlanır. OID'ler, hiyerarşik olarak UNIX veya PC-DOS dosya sistemi adları gibi organize edilmiştir ve negatif olmayan tamsayı dizileridir. Kullanımı kolaylaştırmak için, bir OID'nin her öğesi metinsel bir ad ile ifade edilebilir. OID'ler, yalnızca yönetilen nesneleri değil, herhangi bir şeyi özdeşleştirmek için kullanılabilir. Bazı OID'ler yer tutucularının OID hiyerarşisini düzenlemelerine yardımcı olması için kullanılır. Şekil-10'da ilgili yer tutucular ve hiyerarşi belirtilmiştir. MIB'ler SNMP'nin yönetim bilgisini nasıl düzenlediğini belirler. SNMP yönetim bilgisini toplamak için bazı operasyonel işlemler kullanır. [48]

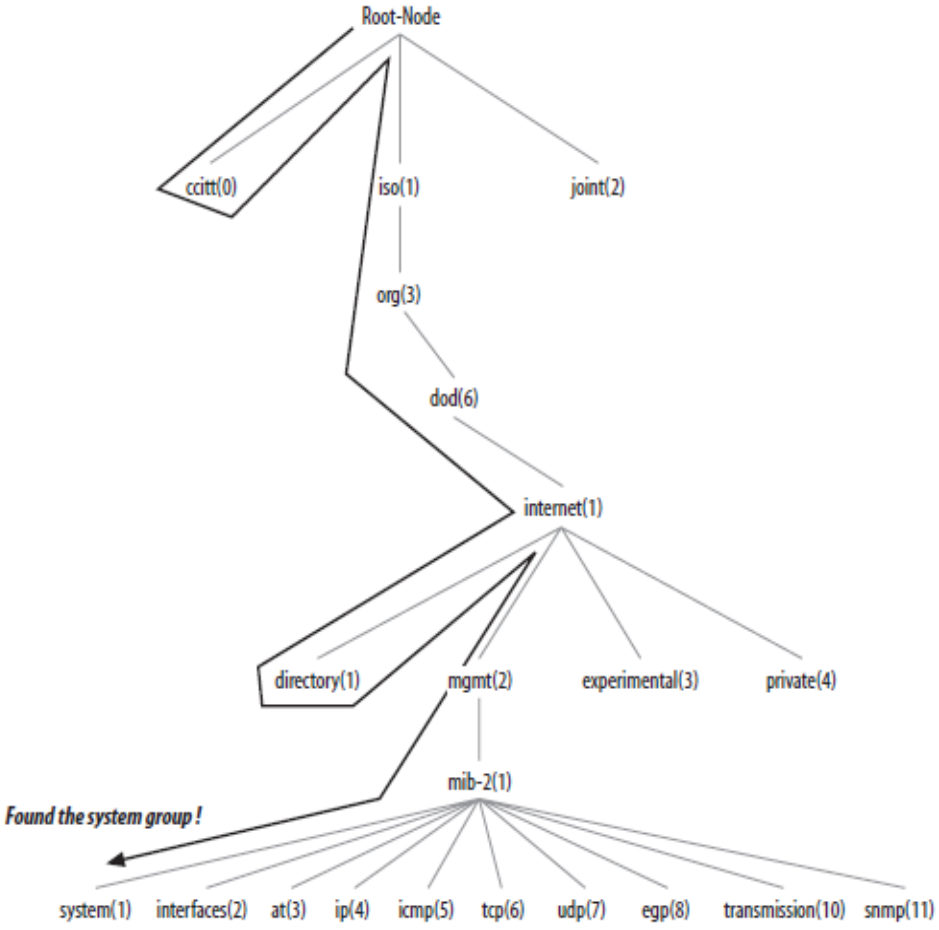
SNMP operasyonlarından birincisi alma (get) isteğidir. Temsilciye isteği gönderen NMS tarafından başlatılır. Temsilci, talebi alır ve işler. Cihazlar ağır yük altındaysa bazen talebe yanıt veremeyebilir ve onu bırakmak zorunda kalacaklar. Aracı talep edilen bilgilerin toplanmasında başarılı olduğu takdirde, bilgileri bir yanıt ile NMS'e gönderir. Örnek bir alma operasyonu işlemi Şekil-11'de gösterilmiştir. Alma isteğinde NMS'in öğedeki aradığı



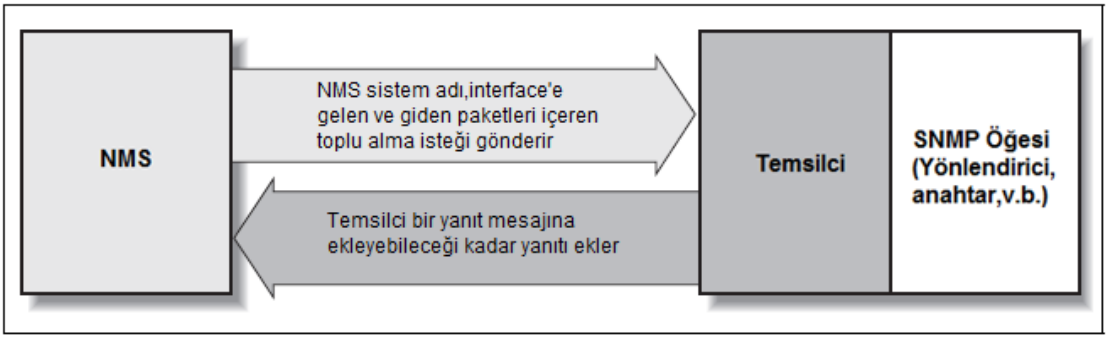
Şekil-11 Örnek Bir SNMP Alma Operasyonu

şeyi bulmasını sağlayan bir istek değişken listesidir. Değişken listesi MIB içerisindeki hangi nesnelerin talep edildiğini içerir.

Nesne ağacında yürüme ile sırasıyla tüm nesnelerin alınmasını sağlayan diğerini alma (getnext) işlemidir. Yürüme işlemi Şekil-12'de gösterilmektedir. Şekil-12'de en üstten başlayarak aşağıda doğru koyu renkte belirtilen işaret takip edilerek yürüme işlemi ve aşamaları takip edilebilir. Bir grup komutu almak için bir dizi komut yayınlamanıza izin



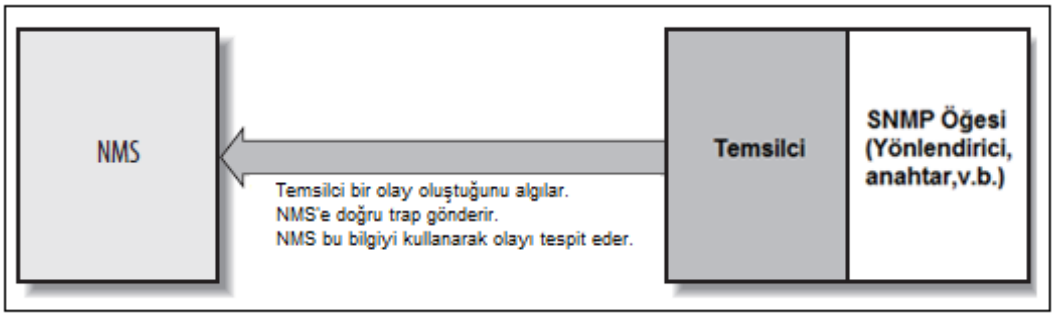
Şekil-12 SNMP Ağacı Üzerinde Örnek Bir Yürüme İşlemi



Şekil-13 Örnek Bir SNMP Toplu Alma Operasyonu

verir. Bir MIB'in tamamından ve bir kısmında değerler alınmasını sağlar. Alınmak istenen her bir MIB nesnesi için ayrı bir getnext isteği ve yanıt üretilir. Alma işlemi bir seferde tek bir MIB nesnesini almak için kullanışlıdır. Ancak bu şekilde herhangi bir şeyi yönetmeye çalışmak, zaman kaybı olabilir. Birkaç nesneyi almak gerektiğinde, diğerini alma komutun devreye girer. Diğerini alma işlemi bir cihazdan, bir süre boyunca birden fazla nesneyi almanızı sağlar. Ne zaman ki NMS, yeni yayınladığı diğerini alma komutu için aracıdan bir yanıt alırsa, bu durumda başka bir diğerini alma komutu gönderir. MIB'nin sonuna ulaşıldığını ve daha fazla cisim olmadığını belirten aracı bir hata döndürene kadar bu işleme devam eder. Bir kerede çok sayıda nesnenin alınmasını sağlayan toplu alma (getbulk) işlemidir. Bu işlem Şekil-13'de gösterilmektedir. Standart alma işlemi ile bir seferde birden fazla MIB nesnesi alınabilir, ancak mesaj boyutları temsilcinin yeteneklerine bağlıdır. Getbulk işleminin bir avantajı, temsilci talep edilen tüm yanıtları geri alamazsa, hata iletisi dönmez. Bununla birlikte, getbulk işlemi, temsilcinin yanıtı mümkün olduğu kadar cevaplamasını sağlar.

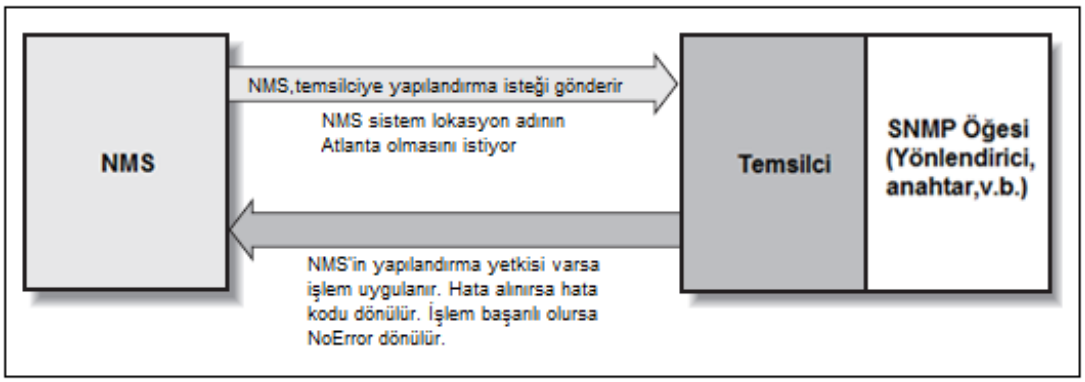
SNMP operasyonlarından ikincisi trap operasyonudur. Trap, bir temsilcinin NMS'e SNMP öğesine istenmeyen bir durum olduğunu söylemesinin bir yoludur. Şekil-14, trap oluşumunu göstermektedir. Temsilci Trap oluşturur ve hedefine gönderir. Trap hedef adresi



Şekil-14 Örnek Bir SNMP Trap Operasyonu

genellikle NMS'nin IP adresidir. NMS'den temsilciye herhangi bir onay gönderilmediğinden, temsilcinin gönderdiği trap'in NMS'e ulaşıp ulaşmadığını anlamasının bir yolu yoktur. Trap'ler ile ekipmanlar bir şeylerin yanlış olduğunu anlatır ve neler olduğunu tahmin edilmesine izin verir. Örnek olarak trap oluşabilecek durumlar, temsilcinin çalıştığı yerde bir ağ arabiriminin çalışmaması, yanlış olduğunu anlatır ve neler olduğunu tahmin edilmesine izin verir. Örnek olarak trap oluşabilecek durumlar, temsilcinin çalıştığı yerde bir ağ arabiriminin çalışmaması, temsilcinin çalıştığı yerde bir ağ arabirimi çalışmaya başlamasının bildirimi, cihaza gelen bağlantı isteğine rağmen bağlantının kurulamaması, bir anahtarın veya yönlendiricinin üzerindeki fanın çalışmaması, vb. SNMPv3'teki trap girişleri, bildirim, bildirim filtresi, hedef adres ve hedef parametreleri yapılandırılarak oluşturulur. Bildirim türü belirlenir ve bildirim tek bir etiket içerir. Etiket, bir tuzak (trap) almak için bir dizi hedef adresi tanımlar. Bildirim filtresi, bir tuzak nesne tanımlayıcıları koleksiyonuna (OID'lere) erişimi tanımlar. Hedef adres, bir yönetim uygulamasının adresini ve bildirimleri göndermek için kullanılacak diğer özellikleri tanımlar. Hedef parametreler, belirli bir yönetim hedefine bildirimlerin gönderilmesinde kullanılacak olan, mesaj işleme ve güvenlik parametrelerini tanımlar. [49]

Üçüncü bir SNMP operasyonu yapılandırma operasyonudur. Set komutu, yönetilen bir nesnenin değerini değiştirmek veya yeni bir nesne oluşturmak için kullanılır. MIB tablosunda okunur-yazılır nesneler, bu komut kullanılarak değiştirilebilir ve ya salt okunur nesneler bu komut ile oluşturulabilir. NMS'nin bir seferde birden fazla nesneyi ayarlaması mümkündür. Şekil-15, yapılandırma isteğinin sıralamasını gösterir. Bu, get komutuna benzer şekildedir. Herhangi bir değişkenin içerik bilgisini almak yerine, değerini değiştirir. Aşağıdaki örnek, sistem lokasyon değişkeni sorgulanır ve bir değere ayarlanmaktadır.



Şekil-15 Örnek Bir SNMP Yapılandırma Operasyonu

Diğer bir operasyon ilse SNMP bildirim operasyonudur. SNMP bildirimi, birden fazla NMS'ye trap gönderilmesine ihtiyaç duyulduğunda kullanılmaktadır. Bir bilgi gönderildiğinde, alıcı olayın alındığını kabul ederek gönderene bir yanıt gönderir. Bu davranış, get ve set istekleri ile aynıdır. SNMP rapor işlemi, SNMPv2 taslak sürümü içinde tanımlanmıştır ancak asla olmadı uygulamamıştır. SNMPv3 standardının bir parçasıdır ve SNMP motorlarının birbirleriyle iletişim kurması için kullanılmaktadır.

SNMPv3

Güvenlik, başlangıcından beri SNMP'nin en büyük zayıflığı olmuştur. SNMP sürüm 1 ve 2'deki doğrulama yöntemi, NMS ve SNMP öğesi arasında, açık metin olarak gönderilen bir paroladan ibarettir. Düz metin şifrelerinin gerçek anlamda güvenlik sağlamamaktadır. Parola, topluluk (community) bilgileri de edinilirse ağdaki cihazlardan bilgi almak için kullanabilir, konfigürasyonları değiştirebilir ve hatta kapatmalarını sağlanabilir. SNMPv3, daha önceki sürümlerdeki güvenlik sorununu adresler. En önemli değişiklik, Sürüm 3'ün yöneticiler ve temsilciler görüşünü terk etmesidir. Hem yöneticilere hem de temsilciler bu sürümde SNMP öğeleri denir. Her öğe bir SNMP motoru ve bir veya daha fazla SNMP uygulamasından oluşur. Bu yeni kavramlar önemlidir çünkü bunlar basitçe bir dizi mesajdan ziyade bir mimariyi tanımlamaktadır. [50]

Şekil-16 bir SNMPv3 paketinin başlık ve taşınan PDU olmak üzere tamamını göstermektedir. Bir SNMPv3 mesaj (paket) formatında aşağıdaki alanlar bulunur:

msgVersion: mesajın SNMP sürümü 3 olarak ayarlanır.

msgID, yönetici ve temsilci arasındaki mesajlaşmaların koordinasyonu için kullanılır.

msgMaxSize, bir SNMP göndericisinin desteklediği maksimum mesaj boyutudur.

msgFlags, bir rapor PDU'su oluşturulup oluşturulmayacağını belirten 8 bitlik bir değerdir. Gizlilik kullanılıp kullanılmadığı ve kimlik doğrulamanın kullanılıp kullanılmadığını belirtir.

msgSecurityModel, iletinin göndereni tarafından hangi güvenlik modelinin kullanıldığını belirtir. Değerler sırasıyla SNMPv1, SNMPv2c ve SNMPv3 için 1, 2 ve 3'tür.

msgSecurityParameters, güvenlikle ilgili bilgiler bu alan içermektedir.

contextEngineID, bir SNMP öğesini benzersiz olarak tanımlar

contextName, bir SNMP motoru içindeki belirli bir içeriği tanımlar.

scopedPDU, Bir içerik EngineID, bağlam adı ve SNMP PDU'dan oluşan bir veri bloğudur.

Bir SNMPv3 mesajındaki msgSecurityParameters aşağıdaki gibidir:

msgAuthoritativeEngineID, yetkili motorun snmpEngineID'si.

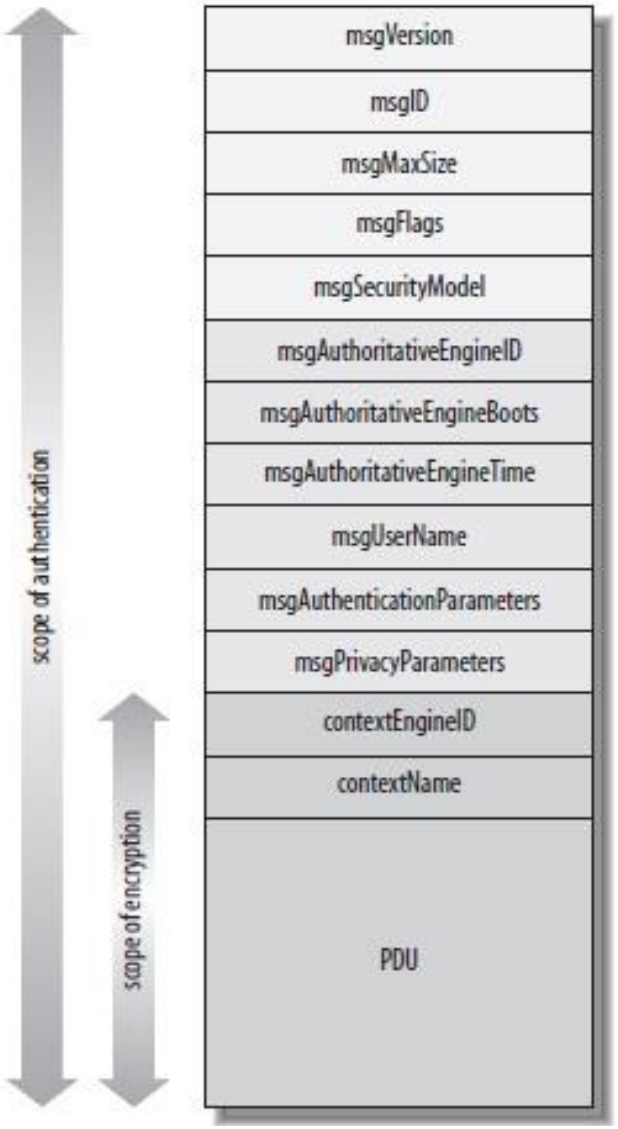
msgAuthoritativeEngineBoots, yetkili motorun kaç defa başlatıldığıdır.

msgAuthoritativeEngineTime, yetkili motorun zaman bilgisidir.

msgUserName, kimliği doğrulayan ve iletiyi şifreleyen kullanıcı adıdır.

msgAuthenticationParameters, kimlik doğrulamak için gerekli yöntemi ve bilgileri içerir. Şifreleme yapılmadığında bu değer boştur. Şu anda RFC standartlarına göre bu alan, MD5 ve ya SHA metotları olabilir.

msgPrivacyParameters, şifreleme kullanılmıyorsa, bu değer boştur. Aksi takdirde, bu alan veri şifreleme standardının şifreleme blok zincirleme modunun başlangıç değeri algoritmasıdır. (CBC-DES).



Şekil-16 SNMPv3 Mesaj Formatı

SNMPv3 Kullanıcı Tabanlı Güvenlik Modeli

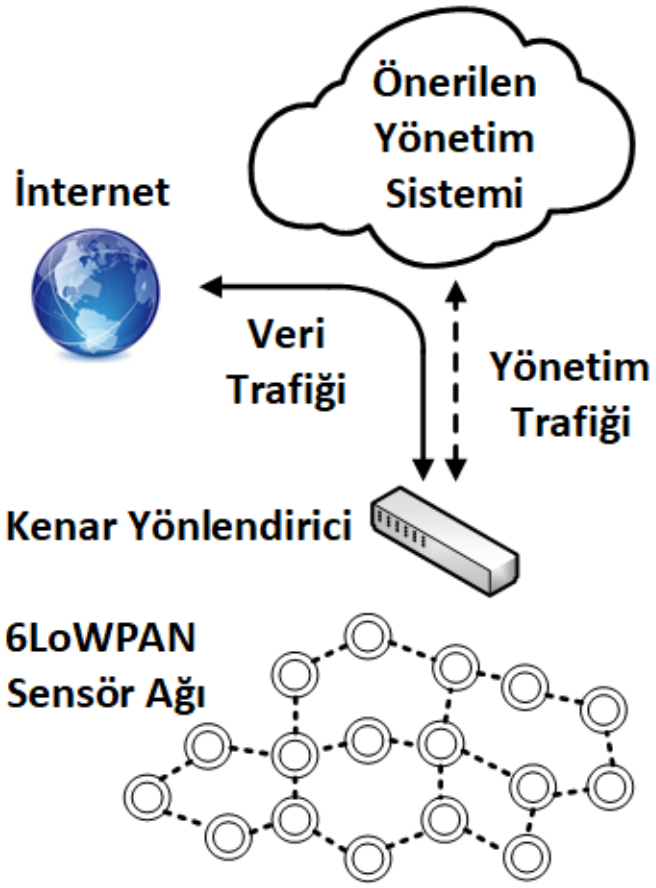
USM, güvenlik parametrelerinin (güvenlik seviyeleri, kimlik doğrulama ve gizlilik protokolleri ve anahtarları) hem aracı hem de yönetici tarafından yapılandırıldığı bir kullanıcı kavramını kullanır. USM kullanılarak gönderilen mesajlar, şifrelerin görüntülendiği ve SNMPv2'de kullanılan, topluluk tabanlı güvenlik ile gönderilen mesajlardan daha iyi korunması için geliştirilmiştir. USM'de için iki kriptografik işlev tanımlanmıştır: kimlik doğrulama ve şifreleme. Bu işlevleri desteklemek için bir SNMP motorunda iki adet değer gerektirir: bir gizlilik anahtarı (privKey) ve bir kimlik doğrulama anahtarı (authKey). Bu iki anahtarın ayrı değerleri aşağıdaki kullanıcılar için korunur. İlki, yönetim işlemlerinin yetkili olduğu bu SNMP motorundaki yerel kullanıcılardır. İkincisi, iletişimin istenildiği uzak bir SNMP motorundaki yetkili uzak kullanıcılardır. Bu değerler, her kullanıcı için saklanan kullanıcı özniteliklerdir. PrivKey ve authKey değerlerine SNMP üzerinden erişilemez. Veri bütünlüğü, verilerin yetkisiz birinin verinin değiştirilmesini ve ya ortadan kaldırılmasını önler. Veri doğrulama, veriyi alanın kimliği ve kökeninin belirtilmesidir. Veri gizliliği, verinin yetkisiz kişilere açıklanamıyor olması veya açıklanmamasıdır. USM ile, yönetici ile aracı arasında değiştirilen mesajlar, veri bütünlüğü denetimi ve veri kaynağı kimlik doğrulamasına sahiptir. Mesaj gecikmeleri ve mesaj tekrarları (normalde bağlantısız bir taşıma protokolüne bağlı olanın ötesinde) zaman göstergelerinin ve talep kimliklerinin kullanımına karşı korunur.

SNMPv3'te HMAC-MD5-96 ve HMAC-SHA-96 kimlik doğrulama protokollerini kullanılmaktadır. Buradaki mekanizma temel olarak, mesajın bütünlüğü korumak için bir hesaplama ile mesajın üzerine mesajın göndericisi tarafından bir parça eklemektir. Bu parça mesajın alıcısı tarafından tekrar hesaplanır ve aynı olmaları durumunda bütünlük doğrulanır. Mesajın kimin tarafından üretildiğinin doğrulanması için, kullanıcılarda HMAC modunda, yalnızca yetkili olan SNMP motorları tarafından bilinen gizli bir değer kullanılır. SNMPv3'te veri gizliliği, CBC-DES simetrik şifreleme protokolü kullanılarak Veri gizliliğini desteklemek için, bir şifreleme algoritması gereklidir. Mesajın uygun bir kısmı iletilmeden önce şifrelenir. Şifrelenen mesajın scopedPDU olarak belirtilen kısımdır. Zamanı gösteren bir değeri ve belirlenen gizli bir değer kullanılarak, mesajlar şifrelenir ve çözülür. Oluşturulan gizli değer, yetkili tüm SNMP motorları tarafından paylaşılır ve uygun kullanıcılar mesaj göndermek için bu değeri kullanır.

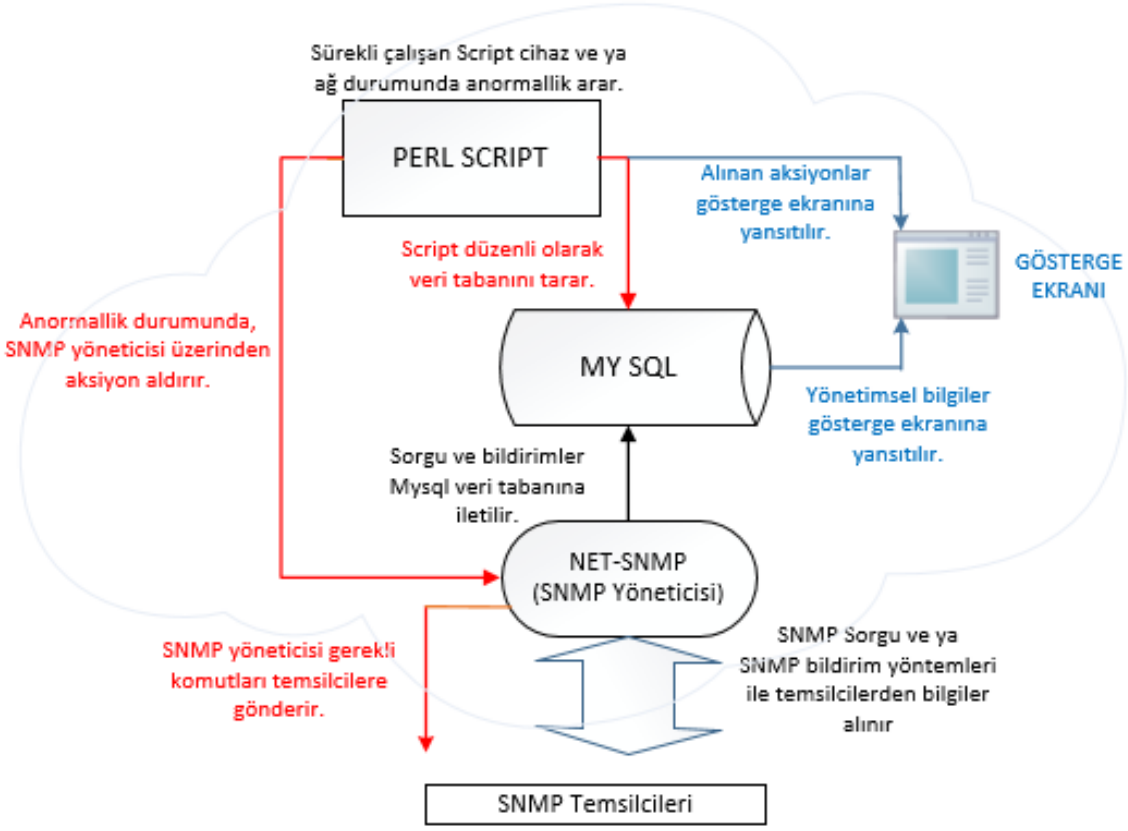
Önerilen Yönetim Sistemi Mimarisi

Tasarlanmış olan mimari yapı için izlenen metot ve çalışma yöntemleri bu bölümde anlatılmaktadır. Şekil-17 bir 6LoWPAN ağı ve önerilen mimari sistemin ağ ile etkileşimi göstermektedir. Önerilen mimari iki kısımdan oluşmaktadır. Birinci kısım şekil-17'de gösterilen ve bulut içerisinde yer alan yönetim sistemidir. Bu yönetim sistemi klasik yönetim sistemlerinin gerçekleştirdiği izleme faaliyetlerinin yanı sıra aynı zamanda daha önce belirlenmiş kurallar ile aksiyonlar alabilmektedir. Bulut içerisindeki yönetim sistemi modüler bir yapıda çalışmaktadır. Toplamda 3 adet modüle sahiptir. Çalışmanın ikinci kısmı ise, bulut içerisindeki yönetim sistemine verilerin iletilmesi için kullanılan SNMPv3 protokolünde yapılan değişiklikleri içermektedir. SNMPv3 protokol yapısında bazı alanlar bulut yardımı ile güncellenerek sıkıştırma işlemi uygulanmış ve bant genişliği kazanımı sağlanmıştır.

Uyarlanmış olan SNMPv3 ile sorgulanan bilgilerin saklanması için bir My-SQL veri tabanı kullanılmıştır. Veriler, My-SQL veri tabanı üzerinde sürekli olarak çalışan



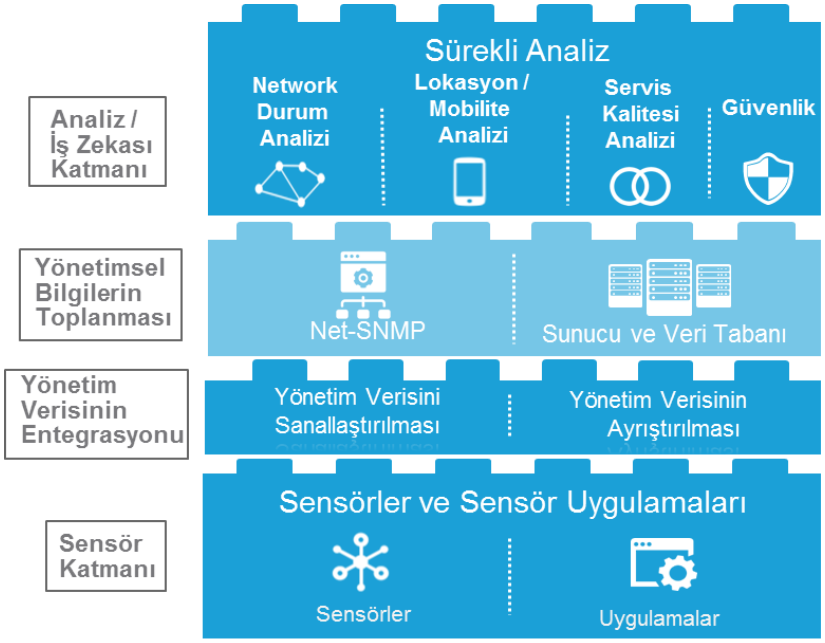
Şekil-17 Önerilen Yönetim Sistemi ve 6LoWPAN Ağı ile Etkileşimi



Şekil-18 Mimari Yapının Yüksek Seviye Tasarımı

PERL scriptler ile analiz edilmektedir. Mimari yapıda bulunan tüm modüller için PERL scriptleri Ek-1'de mevcuttur. Ek-2 içerisinde sadece bir adet sensorün çalışması için gerekli scriptler bulunmaktadır. Bu scriptler IP adresi değiştirilerek tüm sensörlere uygulanabilir. Son olarak sensör trafiklerini toplamak için kullanılan iftop programının kaynak kodunda yapılan değişiklikler Ek-2'de gösterilmektedir.

Yüksek seviye tasarımdaki bulut yapısı ve SNMPv3 çalışması Şekil-18'de gösterilmektedir. 6LoWPAN ağında çalışması üzere sıkıştırılmış olan SNMP protokolü yönetimsel bilgilerin taşınması için kullanılmaktadır. Scriptler SNMP yöneticisi içerisinde düzenli aralıklarla çalışır ve ilgili sorgu işlemlerini gerçekleştirir. Sorgu sonuçları My-SQL veri tabanına yazdırılır. Veri tabanında, her sensör için ayrı bir tablo tutulmaktadır. Tablolar, sensörler ağda aktif oldukça, bir tetikleme ile veri tabanında otomatik olarak oluşturulmaktadır. İlgili sorgu sonuçları sensörler için tutulan ilgili tablolara eklenir. Bulut içerisinde sürekli olarak izleme ve devamlı olarak değerlendirme işlemleri yapılmaktadır.



Şekil-19 İş Süreç Katmanları

Cihazlar için manuel olarak yapılacak bu analizler scriptler sayesinde kolaylıkla onlarca cihaz için yapılabilmektedir. Şekil-19 süreç katmanlarını yansıtmaktadır. Analiz içerisinde sadece yönetimsel bilgiler değil aynı zamanda ağın güvenlik durumu incelenmektedir. Cihazların kendi üzerinden yazılımsal ve ya donanımsal yazılım (firmware) olarak tuttukları fonksiyonlar da ağ içerisine taşınabilir. Buna bir örnek, kablosuz arayüzlerde cihazların gürültü durumunda iletimi farklı bir kanala taşımaları olabilir. Buna ek olabilecek başka bir örnek ise kablosuz güç artırımı olabilir. Buluta taşıma ile birlikte bu fonksiyonel özelliklerin cihazlar üzerinde hafıza ve kaynak kaplamasına gerek kalmayacaktır. Ayrıca bu özelliklerin her yeni firmware için çalışır

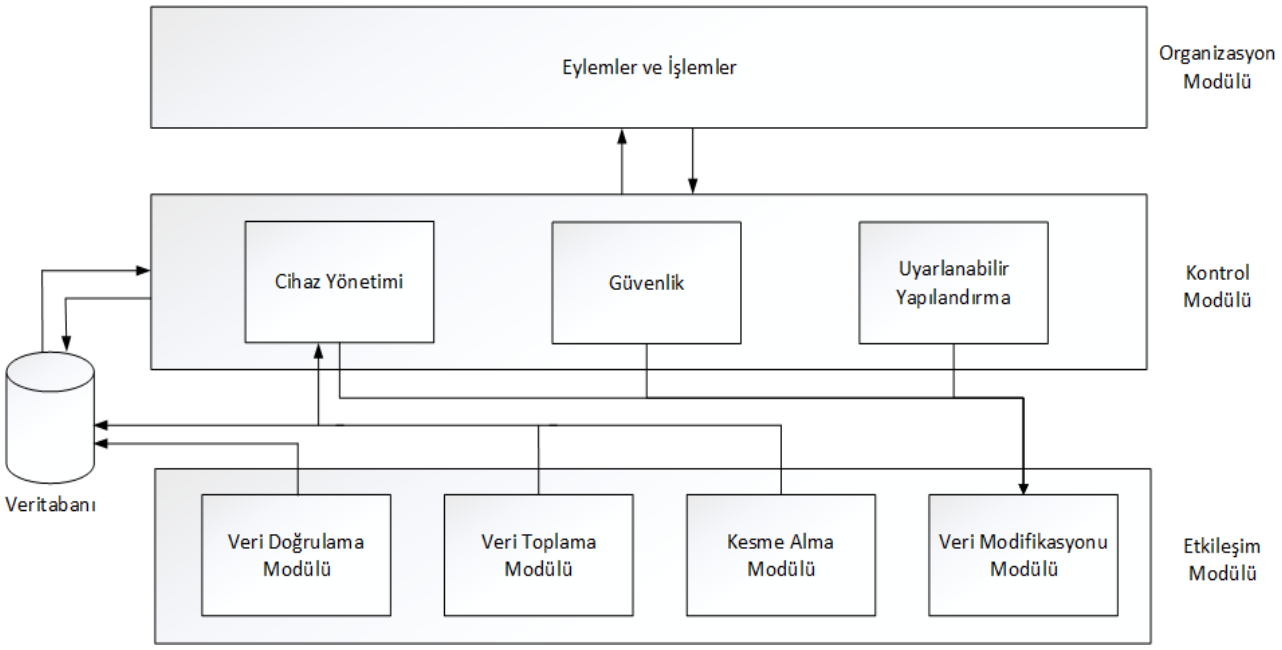


Şekil-20 Mimari Yapı İş Akış Modeli

olması ve özelliklerin geliştirilme ihtiyaçları, yazılımcılara iş yükü olarak gelecek bir husus olacaktır. Şekil-20 iş akışının ne şekilde ilerlediğini göstermektedir. Mimari sürekli olarak veri toplanmasını, bu verilerin analizini ve analizler sonucu ortaya çıkan aksiyonların alınmasını içerir.

Önerilen mimari yapı, sensör ve ya ağ geçidi olarak farklı kaynaklardan gelen veriyi toplayan, işleyen ve depolayan bir platformdur. Bu bölümde 2 temel kısım anlatılmıştır, bunlar tasarlanan mimari yapı ve bu tasarımın uygulama kısmıdır. İlk kısım aslında süreçleri içeren yönetim sistemidir. Bu sistem, cihazdaki veya ağdaki problemin tespit edilmesinde uygulanmaktadır. İkinci kısım, yönetim bilgilerini taşıyan ve 6LoWPAN düğümlerinde ile sunucu arasındaki SNMPv3 protokolü üzerindeki geliştirmeleri içerir.

Sistem mimarisi üç modül ve bir veri tabanı içerir. Her modülün içerisinde farklı işlevler sağlayan alt modüller vardır. Şekil-21, her ana modülü, bunların alt modüllerini ve diğer modüller ile olan ilişkilerini gösteren yüksek düzeyde bir görünümdür. Organizasyon modülü hiyerarşinin tepesinde konumlandırılmıştır. Tüm politikalar, farklı olay türleri için uygulanacak işlemler ve eylem listeleri bu bölümde saklanır. Uygulanacak olan yönetimsel eylemler, bu eylemlerin işlem sırası ve uygulanan eylemin türü bu seviyede güncellenebilir. Bir süreç, bir veya daha fazla işlemle bağlantılı olabilir



Şekil-21 Sistem Modülleri

ayrıca bir veya daha fazla eyleme de bağlı olabilir. Her işlemin sonunda, sorunu çözmek için en az bir eylem tanımlanmış olmalıdır. Bu modül ayrıca alt modüllere sağlanan verilerin işlenmesini de kaydeder. Böylece, aynı sorun tekrar oluştuğunda, bu kayıtların kontrol edilmesi ile birlikte sistemin her seferinde farklı bir çözüm uygulamasına izin verir. Böylece bir sonraki eylem için organizasyon modülünden farklı bir çözüm önerisi sunulmaktadır. Sistemin kullanılabilirliği, bu modülün ana sorumluluğudur. Bir altta bulunan kontrol modülü, organizasyon modülü ile iletişim kurar ve anormal bir durumda organizasyon modülünün tespit ettiği hangi kuralın çalışacağını ondan öğrenir. Örneğin, çoklu bir yayın saldırısı ağdan içeri veya ağın içinden dışarıya doğru algılanırsa, organizasyon modülü, uzak IP adresini engellemek için gerekli eylemi alt katmana yönlendirir.

Kontrol modülünde, 6LoWPAN'daki cihazlardan toplanan ve veri tabanına aktarılan veriyi sürekli olarak tarayan üç alt modül vardır. Bu üç alt modülden herhangi biri, bir sorun algılarsa, organizasyon modülünden ne yapılması gerektiğini belirlemesini ister. Bu üç alt modül, cihazın yönetimsel durumu, güvenliği ve uyarlanabilir yapılandırma gereksinimi, kontrol eder. İlgili kural, cihazda bulunan veri değiştirme servisi tarafından yürütülür ve sorgu sonuçları veri tabanında tutulur. Bir cihaz için aynı durum meydana geldiğinde uygulanacak kural her seferinde farklıdır. Aynı olay kısa bir süre içinde tekrar gerçekleşirse, bir sonraki kural uygulanır. Tüm kurallar uygulandıktan sonra bekleme süresi başlar ve bekleme süresi bittiğinde, ilk kuraldan tekrar başlanır. Bekleme süresi, güvenlik gibi her farklı olay türü için değişir. Veri depolama süresi sınırsız değildir ve belirli bir süre sonra veri tabanı temizlenir ve bu süre tanımlanan sistemde altı ay olarak belirlenmiştir.

Etkileşim modülü, düğümler ile bulutta bulunan sistem arasında iletişim sağlar. Etkileşim modülünde dört alt modül tanımlanmıştır ve hepsi doğrudan düğümlerle iletişim kurabilir. Veri doğrulama alt modülü, her düğümün çalışma durumunu inceler. Bir düğüm ilk kez kurulduğunda, bu alt modül veri tabanına bağlanır ve bu düğüm için yeni bir tablo oluşturur. Herhangi bir düğüm deaktive edildi ise, bu alt modül gereksiz veri depolamayı önlemek için veri tabanında o sensöre ait tabloyu siler. Veri toplama alt modülü, izlenen nesnelerin durumlarının sorgulanmasından ve sorgu sonucu elde edilen değerlerin veri tabanında o sensöre ait karşılık gelen tabloya yazılmasından sorumludur. Nadiren ortaya çıkan bazı problemler hemen harekete geçmeli ve düğümlerde bir bildirim

mekanizması tanımlanmalıdır. Kesme alma alt modülü, bu kesmeleri dinlemektedir. Veri modifikasyonu alt modülünün görevinde düğümlerde bulunan nesne değerlerinin ayarlanması sorumluluğu vardır. Önceden tanımlanmış değerler, ve görev numaraları belirlenmiş olan eylemleri yürütmek için düğümlere bu modül tarafından yazılır.

Mimari Yapıdaki Süreçler ve Yönetimsel Objeler

SNMP temsilcileri üzerinden alınacak yönetimsel bilgiler 13 adet olarak belirlenmiştir. 1 adet ek olarak bulunan time-window-check süreci özel durumlarda çalışmaktadır. Bunlar SNMP yapısında önceden OID atanmış nesnelerdir. Bu nesneler farklı MIB dosyalarına ait olabilir. Bu nesneler Tablo-3'de belirtilmektedir. Bulut içerisinde sürekli olarak izleme ve değerlendirme işlemleri yapılmaktadır. SNMP temsilcileri üzerinden alınacak yönetimsel bilgilerin tanımlanması bu çalışma içerisinde yer almıştır. Daha önce 6LoWPAN şebekesi için gerekli olan izlenecek nesneler belirlenmiştir [51]. Bu çalışmada bu nesnelerden birkaçı kullanılmıştır. Bunlar SNMP yapısında önceden OID atanmış nesnelerdir. Bu nesneler farklı MIB dosyalarına ait olabilir. Bu nesneler Tablo-3.'de belirtilmektedir.

Kendi kendini organize eden bir ağda, ağın kullanılabilirliğini iyileştirmek için bazı önleyici ve düzeltici eylemler bulutta yer alan sistem tarafından uygular. Önerilen mimari yapıda kontrol modülü, önceden tanımlanmış bazı nesnelerin durumunu sürekli olarak kontrol etmekten sorumludur. Sonuç olarak izlenen nesnenin durumuna bağlı olarak, her kontrol süreci başka bir süreci veya başka bir kontrol işlemini tetikleyebilmektedir. Ayrıca, bir kontrol işlemi, düğümde gerçekleşecek bir eylemi tetikleyebilir. Sistemde her bir kontrol adımı bir kontrol süreci olarak tanımlanmaktadır. Önerilen mimaride, bir eylemin veya sürecin uygulanması için kurallar ve bunların çalışma sırası tanımlanır ve organizasyon modülünde saklanır.

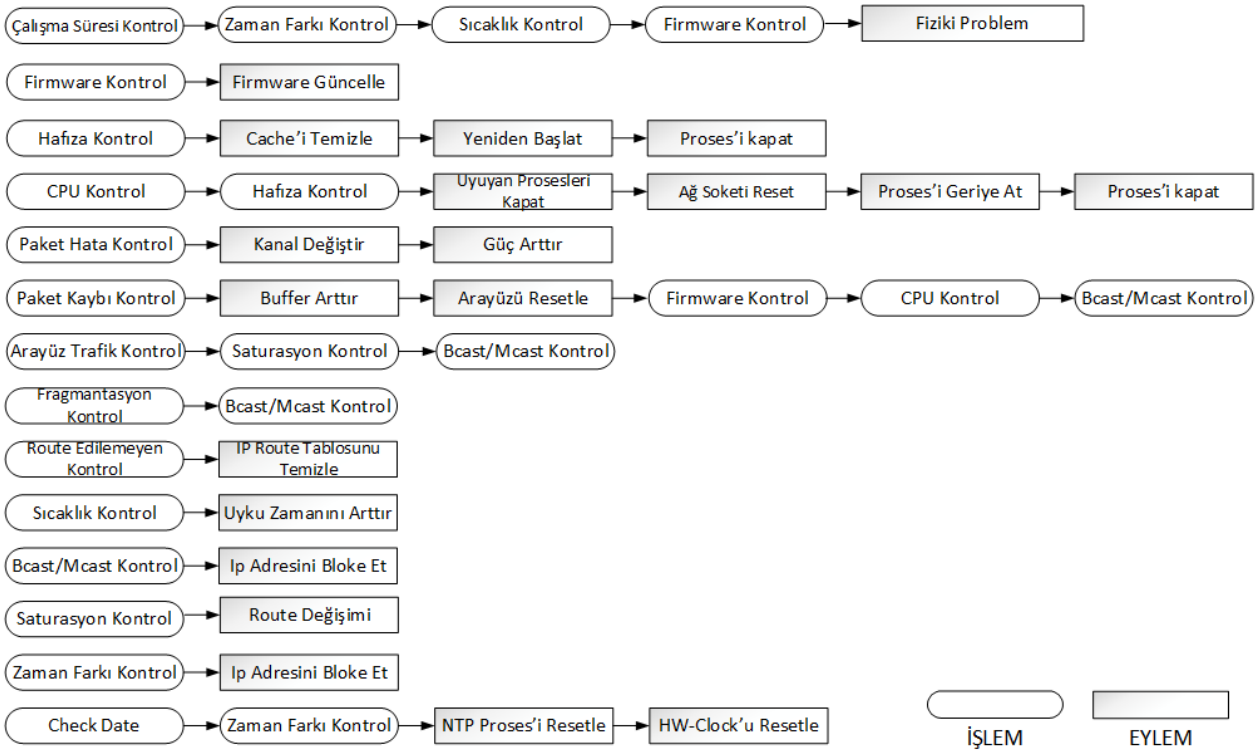
Sistem tarafından işletilen tüm işlemler ve eylemler Şekil-22'de gösterilmiştir. Düğümlerdeki nesnelerin durumunu kontrol eden ve her birinin farklı bir sorun olup olmadığını kontrol eden farklı işlemler durum. Her işlem tanımlanmış bir sorgunun bir sorgusuyla başlar ve her işlem bir eylemle bitmelidir. Tespitinde anormal bir durum, bir eylem doğrudan işlenebilir veya sorunun sonucu olarak ortaya çıkıp çıkmadığını kontrol etmek için başka bir işlem çağrılabilir. Tetiklenen eylem ve süreç sırası güncellenebilir.

Nesne	Tanım ve Gereksinim
HOST-RESOURCES-MIB::hrSystemUptime	Cihazın Ayakta Olma Süresi
IF-MIB:: ifAdminStatus	Interfece'in Yönetimsel Durumu
IF-MIB:: ifOperStatus	Interface Operasyonel Durumu
UCD-SNMP-MIB:: laLoad.1 UCD-SNMP-MIB:: laLoad.2 UCD-SNMP-MIB:: laLoad.3	Cihaz üzerindeki 1,5 ve 15 dakikalık işlemci yük durumunu gözlemlemek için kullanılmıştır.
UCD-SNMP-MIB::memTotalFree.0	Cihaz üzerindeki hafıza durumu belirlemek için kullanılmıştır.
IF::MIB::IfInErrors IF::MIB::IfOutErrors	Arayüzdeki gelen ve interface'den çıkan hatalı paket sayısı için kullanılmıştır
IF::MIB::IfInDiscards IF::MIB::IfOutDiscards	Arayüzdeki iptal ettiği ve göndermediği gelen ve giden paket sayısı için kullanılmıştır
IF::MIB::IfInOctets IF::MIB::IfOutOctets	Arayüzden gelen ve giden paket miktarını octet cinsinden belirlemek için kullanılmıştır.
IP-MIB::ipSystemStatsOutFragReqds	Fragmentasyon isteyen paket sayısını tespit etmek için kullanılmıştır.
IP-MIB::ipSystemStatsReasmFails	Fragmente edilmiş paketleri belirlemek için kullanılmıştır.
IP--MIB::iproutingDiscards	Yönlendirme tablosundan çıkan cihazları belirlemek için kullanılmıştır.
SNMPv2-MIB::sysDescr.0	Cihazdaki donanımdal yazılım (firmware) versiyonunu kontrol etmek için kullanılmıştır.
HOST-RESOURCES-MIB::hrSystemDate	Cihaz üzerindeki zaman bilgisi almak için kullanılmıştır. NTP sunucu'ya ulaşım olup olmadığı anlaşılır.

Tablo-3 Mimari'de Kullanılan Yönetimsel Objeler ve Tanımları

Ayrıca, her işlemin uygulama sıklığı ağ ve düğümlerde yönetimin gerekli ihtiyaçlarına bağlı olarak özelleştirilebilir.

Örnek bir durum olarak, cihaz çalışma süresinin uzunluğunu kontrol etme süreci takip edilebilir. Bu çalışma zamanı bir sorundu Cihazın diğer düğümlerle



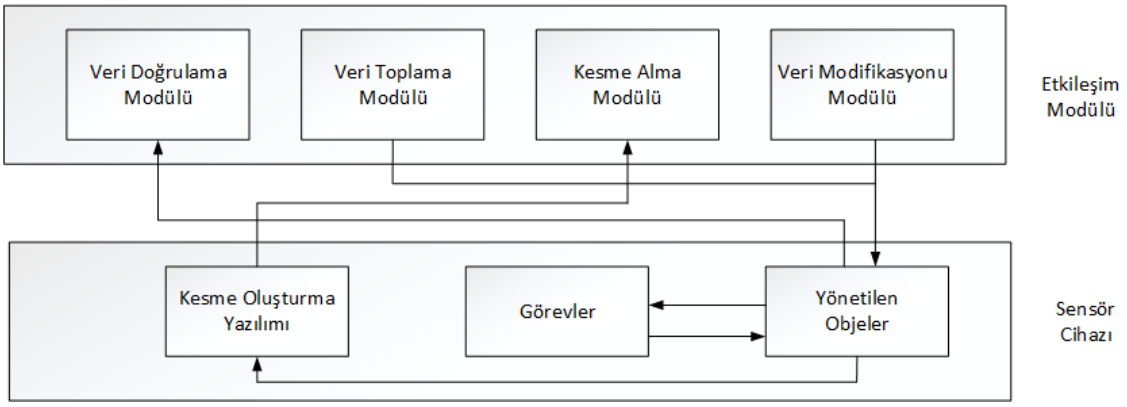
Şekil-22 İşlemler ve Eylem Süreçleri

karşılaştırıldığında daha kısa olduğu tespit edildi. Kısa çalışma süresi, yeniden başlatmanın bir işareti olabilir, bu nedenle olası bir neden dış etkenlerden dolayı cihazın aşırı ısındığını. Aynı sonuca art arda iki kez rastlandığında anormal bir durum olarak tanımlanır. Kontrol modülü sürekli olarak düğümün çalışma süresini kontrol eder. İkinci kısa çalışma süresinin tespitinde değer, kontrol modülü problem hakkında organizasyon modülünü bilgilendirir. Bu durumda, kontrol sıcaklığı olarak adlandırılan bir süreç organizasyon modülü olarak çalıştırıldı. Kontrol sıcaklığı sorgusundan sonra, eğer düğümün sıcaklığı yüksek ise, cihaz daha uzun bir uyku aralığı ile ayarlanır. Sıcaklık normal olarak tespit edilmezse hiçbir işlem yapılmayacaktır. Bir süre sonra kontrol modülü, çalışma zamanı için bir sonraki periyodik sorguyu uygular. Yine çalışma zamanı ile ilgili bir sorun varsa, bu sefer kontrol yazılımı süreç denir. Problemin, düğümün yüksek sıcaklığına bağlı olmadığına karar verildiğinden, kök nedeni bir bozuk veya eski bir firmware olabilir. Düğümün ürün yazılımı sürümünü firmware sorgusunda kontrol edilir. Cihazın bulunduğu tespit edilirse Önceki sürümde çalışan, güncel firmware düğüme yüklenir. Bir sonraki denetimde çalışma süresi sorunu devam ederse sorgu, bu durumda uzaktan yapılabilecek bir çözüm yoktur ve cihaz, fiziksel arızalı ve operatör olarak işaretlenmiştir.

Yönetim bilgilerinin toplanması için yeni bir öz organizasyon MIB tanımlanmıştır. Bu MIB var olan SNMP nesne tanımlayıcılarının yanı sıra bazı skaler değişkenler ve iki tablo içermektedir. Skaler nesneler üç grupta tanımlanmıştır. Bunlar sistem, cihaz ve arayüz nesneleridir. Cihaz özelliklerinin durumunu belirlemek mümkündür, kalitesi ağ bağlantıları ve yetkisiz güvenlik ihlalleri olup olmadığı. Seçme için belirlenen iki tablo var veri örneği. Birincisi, ağ cihazlarından yönlendirme tablosunu almak ve topolojiyi belirlemek için kullanılan yönlendirme tablosudur. İkinci tablo, düğümlerdeki ağ geçidi ve trafik istatistikleri üzerindeki trafik hacmini izleyen trafik tablosudur. Toplanacak tüm bilgiler cihaz özelliklerine bağlı olarak değişebilir.

Ağ geçidi cihazları, ancak gereksinimlere göre son cihazlardan sadece birkaç tür elde edilebilir. Önerilen çalışmada, bu nesneler problemlerin çoğunun temel nedenleri olarak belirlenmiştir ve bu nesnelerin toplamı, 6LoWPAN'da kendi kendini organizasyon görevlerini yürütmek için sistemin gerekli yeteneği kazanmasına yeterlidir. Herhangi bir uç cihaz, ağda ilk kez açıldığında, bu cihazın bağlı olduğu kenar yönlendirici ağ geçidi cihaza kullanacağı IPv6 adresini gönderir. Veri doğrulama servisi bu IPv6 adresini kenar yönlendirici üzerinden alarak bu düğüm için otomatik olarak ayrı bir veri tabanı tablosu oluşturulur. Tablo oluşturulduktan sonra, kontrol ve organizasyon modülleri, bu yeni cihaza uyguladıkları işlere sürekli olarak katar ve sorgularına bu yeni cihazı da ekleyerek çalışmalarına devam ederler. Cihazdaki önceden tanımlanmış olan nesne tanımlayıcısı, önceden belirlenen zaman frekansı ile veri toplama servisi tarafından alınır. Veri değiştirme hizmeti, aygıttaki daha önceden oluşturulmuş olan nesnenin durumunu ayarlar ve ilgili görevin başlatılmasını sağlar. İlgili nesne tanımlayıcısı (OID) ayarlandığında, gerekli olan eylemler sensör tarafından bu değer okunarak gerçekleştirir. Daha sonra ilgili OID durumunu güncellenir ve nesne üzerinde yazılı olan değer temizlenir. Tuzak jeneratörü sadece haberleşmede esnasında cihazda meydana gelen bildirimleri iletmek amacıyla bulunmaktadır. Ağdaki tüm yönetim işlemleri sıkıştırılmış SNMPv3 tarafından taşınmaktadır. Son olarak yönetim sistemi ile 6LoWPAN ağında bulunan cihazlardaki görevler, iş ilerleme süreci ve diğer etkileşimler Şekil-23'de sunulmuştur.

Bu bölümde önerilen kendi kendini organize eden mimari genel anlamda sunulmuştur. İlk olarak, modüller ve alt modüller yer alan iletişim yapıları ile modüllerin görevleri tanıtılmıştır. İkincisi, bir adım-adımda kendini organize eden bir ve yönetim aşamasında kullanacağımız MIB sunulmuştur. Son olarak, başka bir süreci veya eylemi



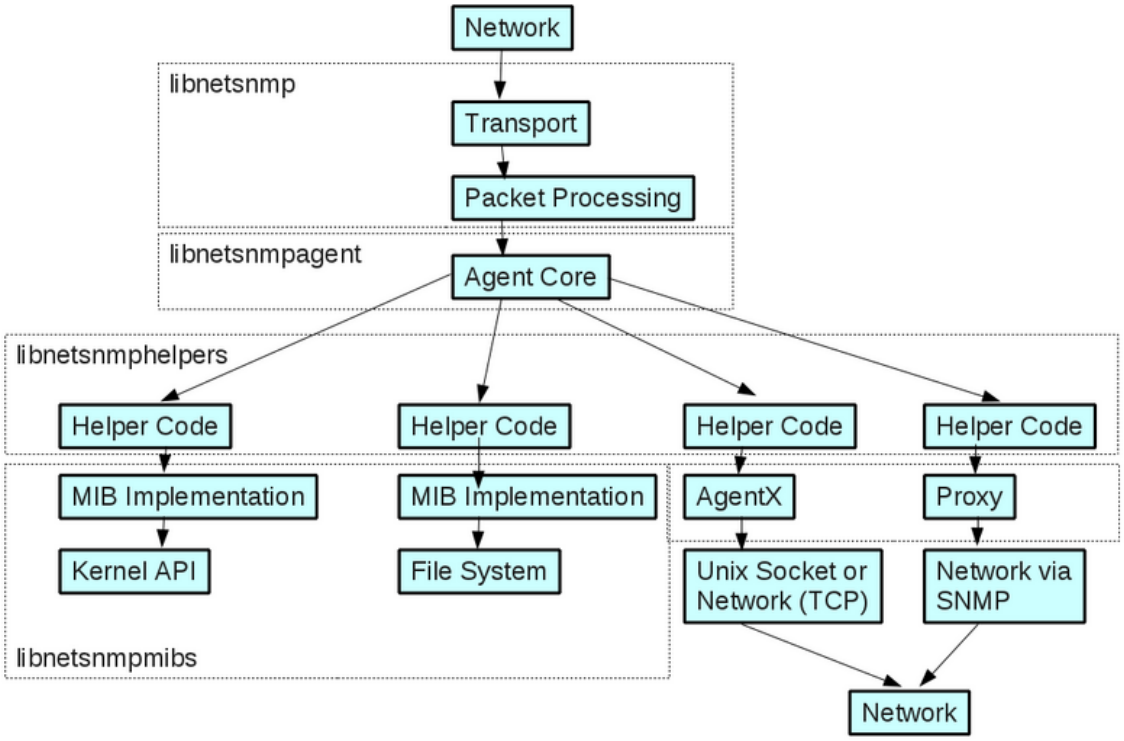
Şekil-23 6LoWPAN Sensöründeki Operasyonel İşlem Akışı

tetiklemek ve tanımlananlar için çözümler sağlamak için kullanılan süreçler problemler anlatılmaktadır. Düğümleri ve ağın tamamını kararlı kılmak için ve ayrıca düğümlerdeki hangi kaynaklara hangi talimatların verileceğine karar vermek için sunduğumuz teknikten bahsedilmiştir.

Net-SNMP ve SNMPv3 Protokol Geliştirmeleri

Formül (1) ve (2)'de de belirtildiği üzere link kalitesinin düşmesine ve 6LoWPAN ağı içerisinde yer alan cihazların birbirine olan mesafesinin artmasına bağımlı olarak, link kapasitesinde düşüş meydana gelmektedir. Çalışmadaki temel amaçlardan biri, SNMPv3 paketlerinde sıkıştırma uygulayarak, kapasitesi düşen linklerdeki yönetim paketlerinin fragmantasyona maruz kalmamasını sağlamaktır.

SNMPv3 üzerinde yapılan geliştirmeler için, açık kaynak kodlu bir uygulama olan Net-Snmp programı kullanılmıştır [52]. Net-Snmp aynı zamanda C ile SNMP programlarının geliştirilebilmesi için bir kütüphane sunmaktadır. Bu programın yapısı şekil-24'te gösterilmektedir. Program Linux üzerinde yüklenerek çalıştırıldığı için kernel içerisinde çalışmaktadır. Kaynak kod üzerinde değişiklikler yapıldıktan sonra programı yeniden işletim sistemi üzerine kurmak gerekmektedir. Şekil-24'de mavi ile gösterilenler alt modüllerdir. Bu yapıda ağ üzerinden soket programlama ile alınan bilgilerin 4 ana alt modüle dağıtıldığı görünmektedir. Toplamda bulunan 4 adet kütüphane ile gerekli tüm alt modüller oluşturulmuştur. Kütüphaneler şekil-24'te gri ile belirtilmektedir. Kısaca açıklaman gerekirse libnetsnmphelpers kütüphanesi ile yazılan helper modülleri gelen bilgiyi 4 alt modüle dağıtır. Bunlar sırasıyla, Linux kernel'ine, MIB dosyalarına, SNMP paketini işleyen modüllere ve soket yapısına aktarılır. Net-SNMP geliştirmeleri iki alt

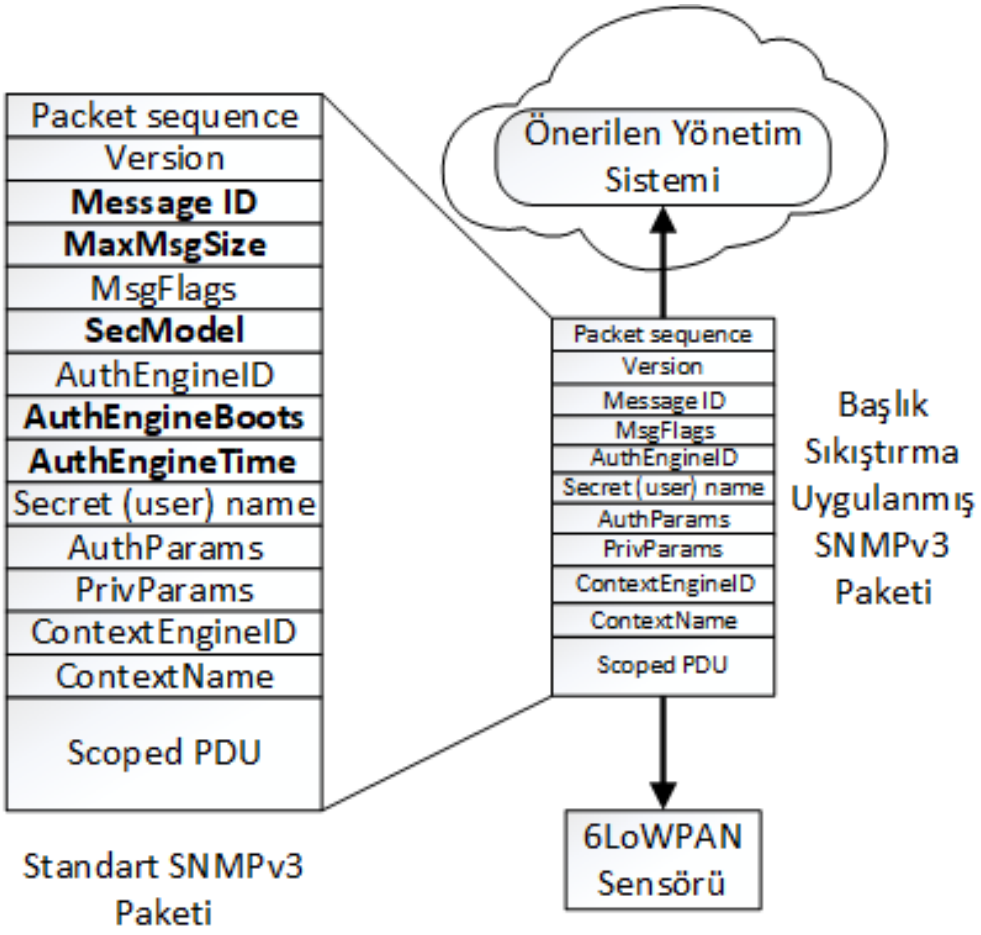


Şekil 24 Net-SNMP Programının Yapısı

modül içerisinde yapılmıştır. Bunlar MIB Implementation ve File System alt modülleridir.

Önerilen mimari yapı, gerçek zamanlı olarak birden çok kaynaktan gelen ve toplanan veriyi toplayan, işleyen ve depolayan bir platformdur. Bu bölümde tespit edilen problemler için ilgili kuralların iletildiği bir demo versiyonunun uygulaması gösterilmektedir. Net-SNMP ile yazılan mevcut SNMPv3 yazılımı üzerinde de bazı değişiklikler uygulanmıştır.

Bu değişiklikler kısıtlı kapasiteye sahip cihazlar ve bağlantılar için protokolün uyumlu çalışmasını ve protokolün sensör ağlarında daha iyi performans göstermesini sağlamak için uygulanmıştır. Değiştirilen ve modifiye edilen alanların, standart SNMPv3 paketi içindeki yeri şekil-25'de kalın olarak gösterilmiştir. Bu alanlar, sensör ağlara doğru yönlendirilemez ve bu şekilde ağ içerisinde bant genişliği kazanımı sağlanır. Toplamda SNMPv3 paketinde beş adet değişiklik yapılmıştır. Yapılan değişikliklerdeki amaç, SNMPv3 standardında tanımlanan özellikleri korumak ve bunların buluttaki yönetim sunucusuna standart yapı korunarak aktarılmasıdır. Değişikliklerin ilk kısmı, SNMPv3 standardında tanımlanan ve değerlerini SNMPv3'ün sunucu ve aracılar arasında çalışan protokol yapısının bulundurduğu iki alanın çıkarılmasını içerir. Bu alanlardan ilki



Şekil-25 Önerilen Yönetim Sistemi ile 6LoWPAN sensörünün SNMPv3 Başlık Sıkıştırması Yöntemi Üzerinden İletişim Kurması

SNMPv3 başlığında bulunan MaxMsgSize değeridir. Bu alan SNMPv3 sunucusunda ve istemci arasında paket büyüklüğünün belirli bir sayıyı aşmaması için kullanılmaktadır. Yönetim sisteminde paket boyutu SNMPv3 paketleri için önceden belirlenmiş bir eşik değerini geçemeyecek şekilde tasarlanmıştır. Bu yüzden maksimum mesaj boyutu alanının tekrar tekrar şebekeye gönderilmesi gerek yoktur. SecModel alanı önceden belirlenmiştir ve kullanıcı güvenlik modeli olarak seçilmiştir bu sebeple, hem sunucu hem de düğümler bu modelin farkındadır. Ayrıca bu alan uygulama sırasında ağda hiçbir zaman değiştirilmemektedir. Bu durumda, bu alanı da tekrar göndermek gerekli değildir.

Düğümler oluşturulan algılama verilerini aktarmak için buluta bağlı olduklarından, bulut içerisinde yönetim sisteminde, yönetim protokolünün bazı özellikleri ele alınabilir ve bu özellikler bulut içerisine transfer edilebilir. Yönetim sistemi mevcutta tanımlanmış Uptime sorguları vasıtasıyla düğümlerin çalışma süresini sürekli

olarak kontrol eder ve bir veri tabanında saklar. Bulut içerisindeki sistem bu iki parametreyi uzaktan yorumlar. Önerilen mimari yöntem, protokol içerisinde yer alan msgAuthoritativeEngineBoots ve msgAuthoritativeEngineTime alanları ile sağlanan güvenlik yöntemlerini sağlamaktadır. Sürekli olarak cihazdan uptime verisi toplandığı için cihazın yeniden başlayıp başlamadığı bilgisi yönetim sisteminde tutulabilir. Ayrıca zaman bilgisi de diğer toplanan bir bilgidir. Bu bilgide yönetim sistemi tarafından incelenerek tutarsızlık durumunda bir güvenlik ihlali olup olmadığı sitem sorgulayabilir.

Bu durumda bu iki alan mevcut protokol operasyonlarında çıkarılarak bu alanlar tarafından işgal edilen bant genişliğinden tasarruf edilir. Sistem zaman penceresi işlemini kontrol etmek için son sorguyu tekrarlar ve bir zamanlama problemi varsa bu durum ikinci sorguda ortaya çıkar. Sunucunun ve cihazın tarih ve saat bilgisi tutarsızsa, bu bir güvenlik saldırısı olarak, yorumlanır ve sonuç olarak ilgili düğümün IP adresi ve ya dışarıdan ilgili düğüme doğru gelen IP adresi engellenir. Bir diğer konu ise cihazın çalışma zamanının her sorgu esnasında güncellemesidir. Sorgular arasında büyük bir tutarsızlık tespit edilirse, bu durum yine bir güvenlik ihlali olarak işaretlenir.

Son değişiklik ise, farklı SNMPv3 paketlerini ayırmak için kullanılan mesaj tanımlayıcı alanının uzunluğunu azaltılmasıdır. Sebebi 6LoWPAN ağlarında, geleneksel ağlardan çok daha az yönetim sorgusu gerekmesidir. Bu sistemler, büyük alan ağları ile karşılaştırıldığında, yapıları gereği çok daha az yönetim ihtiyacı içerirler. Bu nedenle 6LoWPAN içerisinde kullanılacak olan SNMPv3 için çok daha az operasyonel ve sorgulama işlemleri gerekecektir ve bu nedenle daha düşük paket numaralama alanı kullanılabilir. Tüm bu değişiklikler bize ortalama SNMPv3 başlığından ortalama olarak yüzde on beş oranında kazanç sağlamaktadır ve bu yaklaşık her paket için yirmi baytlık bir alan kazancını ifade etmektedir. Yönetici ve temsilci arasında içerik değişiminin yapıldığı ilk iki SNMPv3 paketindeki kazanç oranı daha yüksektir ancak nesne içerisindeki bilginin taşındığı paketlerde kazanç yaklaşık olarak %12 civarındadır.

Net-snmp programı yeni geliştirilen MIB için gerekli iki dosya C kodunda yazılmıştır. İlk olarak, trafik bilgilerinin SNMPv3 ile sensörlerden toplanabilmesi için geliştirilmiş olan data_set.c dosyası şekil-26'da mevcuttur. Bu dosya ile içerisinde 4 kolon ve 6 satır bulunan bir SNMPv3 tablosu oluşturulmuştur. Tablonun içerisinde cihazın arayüzlerindeki gelen ve giden yönündeki trafik bilgileri bulunmaktadır.

```
#include <net-snmp/net-snmp-config.h>
#include <net-snmp/net-snmp-includes.h>
#include <net-snmp/agent/net-snmp-agent-includes.h>
#include <string.h>
#include <stdlib.h>
#define firstEntryRow 2

    netsnmp_table_data_set *table_set;

void init_data_set(void)
{
    Netsnmp_Node_Handler netSnmpIETFWGTable_handler;

    Oid my_registration_oid[] = { 1, 3, 6, 1, 4, 1, 8072, 2, 2, 1 };

    DEBUGMSGTL(("example_data_set","Initalizing example dataset table\n"));

    table_set = netsnmp_create_table_data_set("netSnmpIETFWGTable");

    netsnmp_table_row *row;

    table_set->allow_creation = 1;

    netsnmp_table_dataset_add_index(table_set, ASN_OCTET_STR);

    netsnmp_table_set_multi_add_default_row(table_set, 2, ASN_OCTET_STR, 1, NULL,
    0, 3, ASN_OCTET_STR, 1, NULL, 0,4, ASN_OCTET_STR, 1, NULL, 0, 5,
    ASN_OCTET_STR, 1, NULL, 0, 0);

    netsnmp_register_table_data_set(netsnmp_create_handler_registration
    ("netSnmpIETFWGTable", netSnmpIETFWGTable_handler, my_registration_oid,
    OID_LENGTH(my_registration_oid), HANDLER_CAN_RWRITE), table_set, NULL);

    row = netsnmp_create_table_data_row();

    netsnmp_table_row_add_index(row, ASN_OCTET_STR, "1",strlen("1"));

    netsnmp_set_row_column(row, 2, ASN_OCTET_STR, "Russ Mundy" , strlen("Russ
    Mundy" ));

    netsnmp_mark_row_column_writable(row, 2, 1);

    netsnmp_set_row_column(row, 3, ASN_OCTET_STR, "David Harrington",
                           strlen("David Harrington"));

    netsnmp_mark_row_column_writable(row, 3, 1);

    netsnmp_table_dataset_add_row(table_set, row);

#ifdef ADD_MORE_DATA

    row = netsnmp_create_table_data_row();

    netsnmp_table_row_add_index(row, ASN_OCTET_STR, "snmpconf",
    strlen("snmpconf"));

    netsnmp_set_row_column(row, 2, ASN_OCTET_STR, "David Partain",
    strlen("David Partain"));
```

Şekil-26 Data_set.c Dosyası Kodu

```
59      netsnmp_mark_row_column_writable(row, 2, 1);
60
61      netsnmp_set_row_column(row, 3, ASN_OCTET_STR, "Jon Saperia",
62      strlen("Jon Saperia"));
63
64      netsnmp_mark_row_column_writable(row, 3, 1);
65      netsnmp_table_dataset_add_row(table_set, row);
66  #endif
67
68      DEBUGMSGTL(("example_data_set", "Done initializing.\n"));
69  }
70
71  int netSnmpIETFWGTable_handler(netsnmp_mib_handler *handler,
72  netsnmp_handler_registration *reginfo, netsnmp_agent_request_info *reqinfo,
73  netsnmp_request_info *requests)
74  {
75        netsnmp_table_row *row,*tempRow;
76
77        if(table_set != NULL)
78        {
79
80              for(tempRow=table_set->table->first_row;
81                    tempRow;tempRow=tempRow->next     )
82              {
83                    row=netsnmp_table_data_set_get_first_row(table_set);
84                    netsnmp_table_dataset_remove_row(table_set,row);
85              }
86              }
87
88      int EntryRow=2;
89      FILE *fin;
90      fin=fopen("/home/yekta/pktstat.txt","r");
91
92      char line[255];
93      char line1[255];
94      char line2[255];
95      char line3[255];
96      char *singleValue[50];
97      char *deliValue[50];
98      char *ucuncuValue[50];
99      char *dorduncuValue[50];
100
101     int i;
102     int d=0;
103
104       for(i=0;i<=7;i++){
105           singleValue[i] = &d;
106           deliValue[i] = &d;
107             ucuncuValue[i] = &d;
108             dorduncuValue[i] = &d;
109       }
110
111     fgets(line, sizeof(line), fin);
112     fgets(line, sizeof(line), fin);
113     singleValue[0] = strtok(line, " ");
114
115     for(i=1;i<=7;i++){
116       singleValue[i] = strtok(NULL, " ");
```

Şekil-26 (Devamı) Data_set.c Dosyası Kodu

```
117         }
118
119     if(fgets(line1, sizeof(line1), fin)!=NULL){
120               deliValue[0] = strtok(line1, " ");
121
122              for(i=1;i<=7;i++){
123               deliValue[i] = strtok(NULL, " ");
124               }
125         }
126
127     if(fgets(line2, sizeof(line2), fin)!=NULL){
128         ucuncuValue[0] = strtok( line2, " ");
129
130         for(i=1;i<=7;i++){
131               ucuncuValue[i] = strtok(NULL, " ");
132               }
133         }
134
135     if(fgets(line3, sizeof(line3), fin)!=NULL){
136         dorduncuValue[0] = strtok( line3, " ");
137
138         for(i=1;i<=7;i++){
139               dorduncuValue[i] = strtok(NULL, " ");
140               }
141               }
142
143         row = netsnmp_create_table_data_row();
144
145         netsnmp_table_row_add_index(row, ASN_OCTET_STR, "1", strlen("1"));
146
147         netsnmp_set_row_column(row, 2, ASN_OCTET_STR, singleValue[0],
148          strlen(singleValue[0]));
149
150         netsnmp_mark_row_column_writable(row, 2, 1);
151
152         netsnmp_set_row_column(row, 3, ASN_OCTET_STR, deliValue[0],
153        strlen(deliValue[0]));
154
155         netsnmp_mark_row_column_writable(row, 3, 1);
156
157         netsnmp_set_row_column(row, 4, ASN_OCTET_STR, ucuncuValue[0],
158        strlen(ucuncuValue[0]));
159
160         netsnmp_mark_row_column_writable(row, 4, 1);
161
162         netsnmp_table_dataset_add_row(table_set, row);
163
164         netsnmp_set_row_column(row, 5, ASN_OCTET_STR, dorduncuValue[0],
165          strlen(dorduncuValue[0]));
166
167         netsnmp_mark_row_column_writable(row, 5, 1);
168
169         netsnmp_table_dataset_add_row(table_set, row);
170
171          row = netsnmp_create_table_data_row();
172
173         netsnmp_table_row_add_index(row, ASN_OCTET_STR, "2", strlen("2"));
174
```

Şekil-26 (Devamı) Data_set.c Dosyası Kodu

```
175         netsnmp_set_row_column(row, 2, ASN_OCTET_STR, singleValue[1],
176          strlen(singleValue[1]));
177
178         netsnmp_mark_row_column_writable(row, 2, 1);
179
180         netsnmp_set_row_column(row, 3, ASN_OCTET_STR, deliValue[1],
181          strlen(deliValue[1]));
182
183         netsnmp_mark_row_column_writable(row, 3, 1);
184
185         netsnmp_set_row_column(row, 4, ASN_OCTET_STR, ucuncuValue[1],
186          strlen(ucuncuValue[1]));
187
188         netsnmp_mark_row_column_writable(row, 4, 1);
189
190         netsnmp_set_row_column(row, 5, ASN_OCTET_STR, dorduncuValue[1],
191          strlen(dorduncuValue[1]));
192
193         netsnmp_mark_row_column_writable(row, 5, 1);
194
195         netsnmp_table_dataset_add_row(table_set, row);
196
197          row = netsnmp_create_table_data_row();
198
199         netsnmp_table_row_add_index(row, ASN_OCTET_STR, "3", strlen("3"));
200
201         netsnmp_set_row_column(row, 2, ASN_OCTET_STR, singleValue[2],
202          strlen(singleValue[2]));
203
204         netsnmp_mark_row_column_writable(row, 2, 1);
205
206         netsnmp_set_row_column(row, 3, ASN_OCTET_STR, deliValue[2],
207          strlen(deliValue[2]));
208
209         netsnmp_mark_row_column_writable(row, 3, 1);
210
211         netsnmp_set_row_column(row, 4, ASN_OCTET_STR, ucuncuValue[2],
212          strlen(ucuncuValue[2]));
213
214         netsnmp_mark_row_column_writable(row, 4, 1);
215
216         netsnmp_set_row_column(row, 5, ASN_OCTET_STR, dorduncuValue[2],
217          strlen(dorduncuValue[2]));
218
219         netsnmp_mark_row_column_writable(row, 5, 1);
220
221         netsnmp_table_dataset_add_row(table_set, row);
222
223          row = netsnmp_create_table_data_row();
224
225          netsnmp_table_row_add_index(row, ASN_OCTET_STR, "4", strlen("4"));
226
227         netsnmp_set_row_column(row, 2, ASN_OCTET_STR, singleValue[3],
228          strlen(singleValue[3]));
229
230         netsnmp_mark_row_column_writable(row, 2, 1);
231
232         netsnmp_set_row_column(row, 3, ASN_OCTET_STR, deliValue[3],
```

Şekil-26 (Devamı) Data_set.c Dosyası Kodu

```
233          strlen(deliValue[3]));
234
235         netsnmp_mark_row_column_writable(row, 3, 1);
236
237         netsnmp_set_row_column(row, 4, ASN_OCTET_STR, ucuncuValue[3],
238          strlen(ucuncuValue[3]));
239
240         netsnmp_mark_row_column_writable(row, 4, 1);
241
242         netsnmp_set_row_column(row, 5, ASN_OCTET_STR, dorduncuValue[3],
243        strlen(dorduncuValue[3]));
244
245         netsnmp_mark_row_column_writable(row, 5, 1);
246
247         netsnmp_table_dataset_add_row(table_set, row);
248
249         row = netsnmp_create_table_data_row();
250
251         netsnmp_table_row_add_index(row, ASN_OCTET_STR, "5", strlen("5"));
252
253         netsnmp_set_row_column(row, 2, ASN_OCTET_STR, singleValue[4],
254          strlen(singleValue[4]));
255
256         netsnmp_mark_row_column_writable(row, 2, 1);
257
258         netsnmp_set_row_column(row, 3, ASN_OCTET_STR, deliValue[4],
259          strlen(deliValue[4]));
260
261         netsnmp_mark_row_column_writable(row, 3, 1);
262
263         netsnmp_set_row_column(row, 4, ASN_OCTET_STR, ucuncuValue[4],
264          strlen(ucuncuValue[4]));
265
266         netsnmp_mark_row_column_writable(row, 4, 1);
267
268         netsnmp_set_row_column(row, 5, ASN_OCTET_STR, dorduncuValue[4],
269          strlen(dorduncuValue[4]));
270
271         netsnmp_mark_row_column_writable(row, 5, 1);
272
273         netsnmp_table_dataset_add_row(table_set, row);
274
275         row = netsnmp_create_table_data_row();
276
277         netsnmp_table_row_add_index(row, ASN_OCTET_STR, "6", strlen("6"));
278
279         netsnmp_set_row_column(row, 2, ASN_OCTET_STR, singleValue[5],
280          strlen(singleValue[5]));
281
282         netsnmp_mark_row_column_writable(row, 2, 1);
283
284         netsnmp_set_row_column(row, 3, ASN_OCTET_STR, deliValue[5],
285          strlen(deliValue[5]));
286
287         netsnmp_mark_row_column_writable(row, 3, 1);
288
289         netsnmp_set_row_column(row, 4, ASN_OCTET_STR, ucuncuValue[5],
290          strlen(ucuncuValue[5]));
```

Şekil-26 (Devamı) Data_set.c Dosyası Kodu

```
291
292         netsnmp_mark_row_column_writable(row, 4, 1);
293
294         netsnmp_set_row_column(row, 5, ASN_OCTET_STR, dorduncuValue[5],
295          strlen(dorduncuValue[5]));
296
297         netsnmp_mark_row_column_writable(row, 5, 1);
298
299         netsnmp_table_dataset_add_row(table_set, row);
300
301          return SNMP_ERR_NOERROR;
302    }
```

Şekil-26 (Devamı) Data_set.c Dosyası Kodu

Tablo haricinde skaler objelerin taşınması içinde C ile geliştirme yapılmıştır. Cihazlar üzerinde aksiyon alınırken bu skaler objeler kullanılmaktadır. Scalar1.c doyası için gereken C başlık dosyası şekil-27'de bulunmaktadır. Skaler obje için C dosyası ise şekil-28'de gösterilmiştir.

Bu MIB dosyaları haricinde types.h üzerinde, snmp_api.c snmp_api.h, snmpclinet.c ve snmpusm.c dosyaları üzerinde uyumluluk sağlamak üzere değişiklikler yapılmıştır. Yapılan değişiklikler çok fazla sayıda olduğu için yer verilmemiştir. Tüm dosyalar Net-snmp programının içerisine kopyalanıp, program ile birlikte derlenerek, sonrasında yönetim sistemine ve cihazlara yüklenmiştir.

```
1   #ifndef SCALAR1_H
2   #define SCALAR1_H
3
4   void init_scalar1(void);
5   Netsnmp_Node_Handler handle_scalar1;
6
7   #endif /* SCALAR1 H */
```

Şekil-27 Scalar.h Dosyası Kodu

```
#include <net-snmp/net-snmp-config.h>
#include <net-snmp/net-snmp-includes.h>
#include <net-snmp/agent/net-snmp-agent-includes.h>
#include "scalar1.h"

void
init_scalar1(void)
{
    const oid scalar1_oid[] = { 1,3,6,1,4,1,5100,1,1 };
    DEBUGMSGTL(("scalar1", "Initializing\n"));
    netsnmp_register_scalar(netsnmp_create_handler_registration("scalar1",
    handle_scalar1, scalar1_oid, OID_LENGTH(scalar1_oid),
    HANDLER_CAN_RWRITE));
}

int handle_scalar1(netsnmp_mib_handler *handler, netsnmp_handler_registration
*reginfo, netsnmp_agent_request_info *reqinfo,netsnmp_request_info
*requests)
{
   int i;
   FILE *fptr = fopen("/home/yekta/Desktop/data.txt","r+");
   fscanf (fptr,"%d",&i);
   fclose(fptr);
   int ret;
   switch(reqinfo->mode) {

        case MODE_GET:
            snmp_set_var_typed_value(requests->requestvb, ASN_INTEGER,
            &i,sizeof(i));
            break;

        case MODE_SET_RESERVE1:

           ret = netsnmp_check_vb_type(requests->requestvb, ASN_INTEGER);
            if ( ret != SNMP_ERR_NOERROR ) {
                netsnmp_set_request_error(reqinfo, requests, ret );
            }
            FILE *fpt = fopen("/home/yekta/Desktop/data.txt","r+");
            fprintf(fpt, "%d", ret);
            fclose(fpt);
            break;

        case MODE_SET_ACTION:

           ret = *requests->requestvb->val.integer;
            if ( ret != SNMP_ERR_NOERROR ) {
                netsnmp_set_request_error(reqinfo, requests, ret );
            }
            FILE *fptr = fopen("/home/yekta/Desktop/data.txt","r+");
            fprintf(fptr, "%d", ret);
            fclose(fptr);
            break;

        default:
    }

    return SNMP_ERR_NOERROR;
}
```

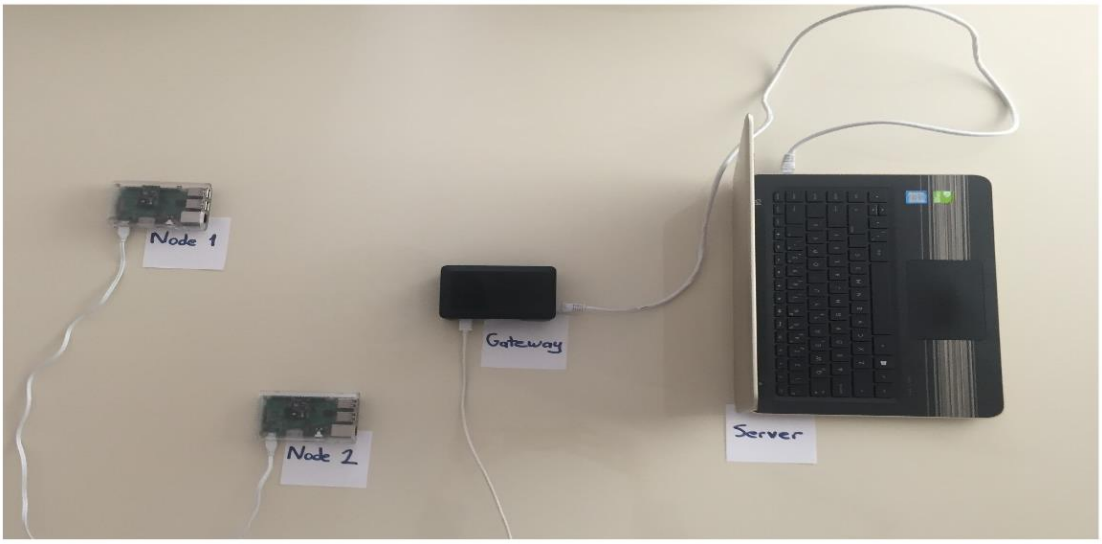
Şekil-28 Scalar.c Dosyası Kodu

BÖLÜM 3. DENEYSEL ÇALIŞMALAR VE BULGULAR

Bu bölümde, önce araştırmanın amaçlarını yansıtan belli başlıklar halinde, elde edilen bulgulara ve bulguların anlamlandırılmaya çalışıldığı yorumlara yer verilmiştir. Verilerin toplanması için gerçek hayattaki bir test düzeneğinin kurulması sağlanmıştır. Veri tabanı seçimi yapılırken ilişkisel veri tabanı ve ilişkisel olmayan veri tabanı şeklinde iki adet seçenek bulunmaktadır. Ticari uygulamalarda kapasite ve kullanım artışı düşünülerek ilişkisel olmayan veri tabanları kullanılmaktadır. Bu tez çalışmasında ticari bir amaç düşünülmediği ve tasarlanan sistemin bilimsel olarak çalışabilirliğini göstermek amaca ile ilişkisel bir veri tabanı seçilmiştir. Bu durum sistemin çalışması için bir performans kısıtlaması getirmemektedir. Ancak ticari uygulamalar ile karşılaştırıldığında yöntem farklı içermektedir.

Yönetim sistemin Debian Linux tabanlı bir işletim sistemi üzerinde çalışmasıdır. Sistem performansı Debian Linux işletim sistemi ile ilişkilidir. Çalışma kapsamında ilgili yönetimsel scriptler PERL programlama dili ile yazılmıştır. Bu açıdan yorumlayıcı bir programlama dili olan PERL dilinin veri işleme hızı ve yeteneği sistem hızına etki etmektedir. 6LoWPAN ağı için kullanılan cihazlar Raspberry PI cihazlarıdır. Bu cihazların seçilmesindeki amaç gösterim için ağ kurulumunun kolay bir şekilde sağlanmasıdır. 6LoWPAN ağ cihazları açısından yönetim sisteminin çalışma hızı, sorgular üzerinden alınan tepki zamanı, Raspberry PI cihazının performansı ile ilişkilidir.

Sunulan mimariyi çalıştırmak için, düğümleri ve ağ geçidini temsil eden üç adet Raspberry Pi cihazı kullanılarak bir demo topolojisi oluşturulmuştur. Openlabs IEEE 802.15.4 radyo modülü Raspberry Pi cihazlarını 6LoWPAN çalışan hale dönüştürmek için cihazlara takılmıştır. IEEE 802.15.4 protokolü, bu demoda 2.4 Ghz bandında çalıştırılmıştır. Net-SNMP açık kaynak kodlu uygulaması SNMPv3 protokolü üzerinde değişiklikler yapılması için kullanılmıştır. Kendi kendini organize eden MIB'de içerisindeki trafik tablosu için ve ağ geçidindeki bant genişliği izlemek için, açık kaynak iftop programı üzerinde geliştirmeler yapılmıştır [54]. Iftop uygulaması çalışırken çok fazla kaynak tüketmektedir, bu yüzden iftop, sadece SNMPv3 isteği alındığında düğümde başlatılır veri alındıktan sonra iftop uygulaması kapatılacak şekilde ayarlama yapılmıştır. Bir Shell scripti ile düzenleme ile birlikte giriş-çıkış trafiği ve kaynak IP bilgisi alındıktan sonra, iftop programı otomatik olarak kapatılmaktadır. Her düğümde bir metin dosyası



Şekil-29 6LoWPAN Testleri için Test Topolojisi

oluşturulmuştur. Bu metin dosyası, Shell script içerisinde bulunan bir komut dosyası tarafından sürekli olarak izlenir ve düğüm açılır açılmaz çalışmaktadır.

Komut dosyasında çalıştırılacak komutların bir listesi de bulunmaktadır. Her bir alınacak aksiyon için bir sıra numarası mevcuttur. Bu metin dosyasına sunucu tarafından bir rakam yazıldığında, bu numaraya karşılık gelen komut düğümde yürütülür. Örnek olarak, 14 bir yeniden başlatma işlemidir. Veri değiştirme hizmeti, bu metin dosyasına 14 sayısını yazdırırsa, düğüm 14 numaranın yazıldığını fark edecek ve düğümdeki yeniden başlatma komutunu çalıştırılacaktır.

Verilerin Toplanması

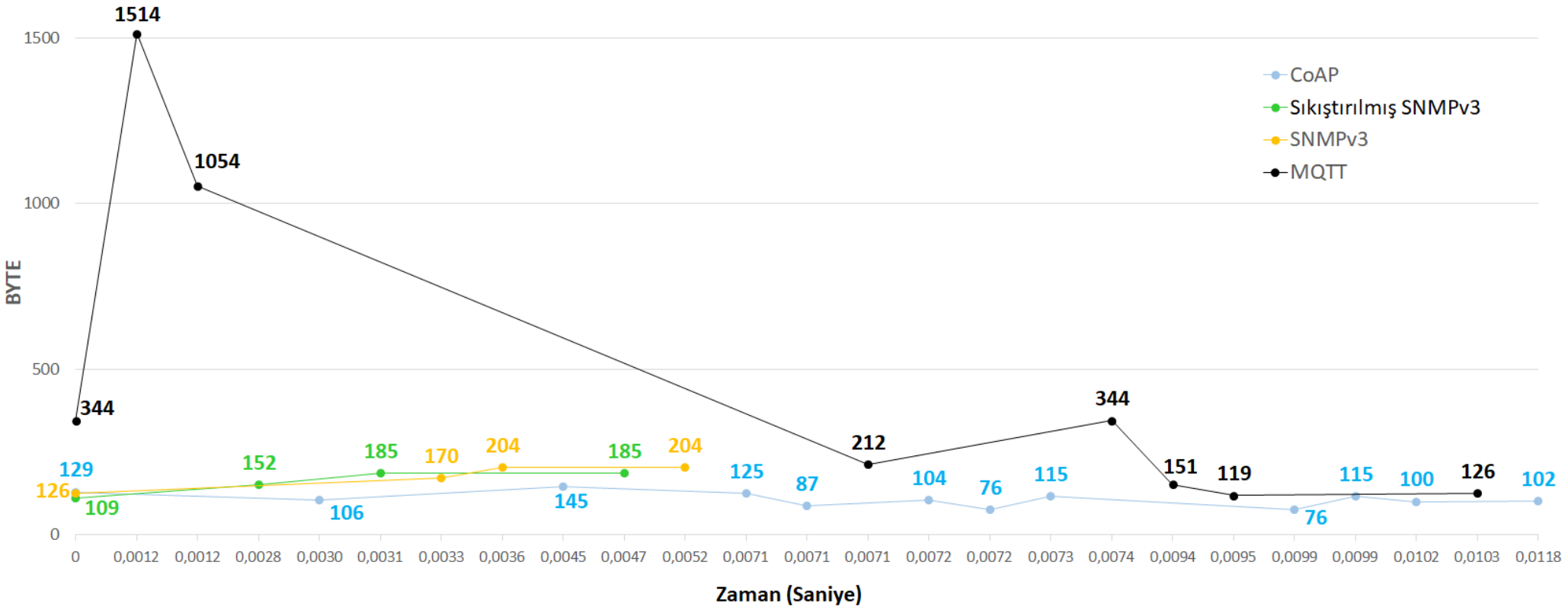
Şekil-29, bir 6LoWPAN ağ ağının oluşturulması için bir ağ geçidine bağlı iki düğüm ile oluşturulan test topolojisini göstermektedir. Bu mimari içerisinde, 4 protokol yönetim bilgilerinin toplanması için test edilmiş ve daha sonra performans sonuçları karşılaştırılmıştır. Karşılaştırmada kullanılan ilk protokol, önerilen yöntem olan başlık sıkıştırılması uygulanmış olan SNMPv3 protokolüdür ve veri güvenliği için USM güvenlik modunda çalışılmıştır. İkinci olarak, değerlendirme amacıyla gerçek protokol arasındaki farkı gözlemlenmesi için standart SNMPv3 seçilmiştir. Karşılaştırmamızda kullanılan başka bir protokol uygulama verilerinin iletiminde kullanılan ve popüler bir nesnelerin interneti protokolü olan MQTT'dir[55]. MQTT, sensör iletişimi için tasarlanmış hafif bir mesajlaşma protokolüdür. Ancak MQTT, TCP protokolünde çalışması nedeniyle doğrudan 6LoPWAN üzerinde uygulanamamaktadır. Bu nedenle, ağ geçidi ve düğümler arasındaki MQTT mesajlarının iletişimi için MQTT-SN protokolünün

desteğini alınmıştır. PAHO [56], Python MQTT kütüphanesi kullanılarak, sunucudan ve ağ geçidinden ve düğümlerden gelen istekler için yanıtlar yürütülmektedir. MQTT mevcutta kimlik doğrulama ve şifreleme mekanizması varsayılan olarak, kullanıcı adı / şifre ve şifreleme ile kimlik doğrulaması ekledik ve üzerine güvenlik için TLS şifreleme ve doğrulama özellikleri eklenmiştir. Karşılaştırmalı analiz değerlendirilmesinin sonunda CoAP protokolü kullanılmıştır ve CoAP üzerine kimlik doğrulama ile datagram taşıma katmanı güvenliği DTLS şifreleme özellikleri eklenmiştir. Libcoap [57], CoAP sunucularının işlevlerini sağlayan bir kütüphanedir ve bu protokolde geliştirme için kullanılmıştır.

Gerçekleştirilen testlerde veriler bir paket analiz programı üzerinden alınmıştır. Ham verilerden SNMPv3 paketine ait kısımlar, paket analiz programının loğlarından tespit edilmiştir. Aynı zamanda cihazların geliştirilen yönteme yanıt verme süreleri test içerisinde hesaplanmaktadır. Ağ içerisindeki herhangi bir sensör cihazının yönetim sistemine tepki süresi isteğin yönetim sisteminden çıkması, ağa ulaşması, ağ içerisinden ilerleyerek ilgili cihaza erişilmesi ve cevabın yönetim sistemine döndüğü sürenin tamamı olarak tespit edilmiştir. Ayrıca test sonuçlarında paketlerdeki sadece uygulama katmanı paket büyüklüğü değil, IPv6 paketinin tamamının büyüklüğü değerlendirilmiştir.

Bulgular

Şekil-30, dört protokolün her birinde yürütülen çalışma zamanı sorgularının tepki süresi açısından ve bant genişliği kullanımı davranışını gösterir. Başlık sıkıştırma yapılmış olan SNMPv3'ün performansı için, daha az mesaj gönderimi ve hızlı yanıt süresi hemen fark edilebilir. Standart SNMPv3 çalıştırıldığında yine daha az mesajlaşma ve hızlı yanıt süreleri gözlenmektedir, ancak beklendiği gibi mesaj boyutları daha büyük ve yanıt süresi sıkıştırılmış SNMPv3'ten daha yavaştır. Aynı sistemi çalıştırdığımızda elde ettiğimiz sonuçlar TLS desteği ile MQTT-SN siyah seride gösterilmektedir. Sertifika değişimi işlemi nedeniyle, her sorgu için çok yüksek boyutlu 2 paket oluşturulmuştur. 1514 baytlık ve 1054 baytlık 2 adet paket içerisinde sertifika değişim işlemlerinin alıcı ve sunucu arasında gerçekleştiği iletişimi göstermektedir. Asıl olarak bu iki paket bant genişliğini önemli ölçüde işgal etmektedir. MQTT-SN'de gönderilen paketlerin %75'i fragmantasyon'a uğrayacaktır. Bu 6LoWPAN ağları için düşünüldüğünde tamamen bir



Şekil-30 Farklı IoT Protokolleri için Yönetim Bilgisi Sorgu Karşılaştırması

	Başlık Sıkıştırma Uygulanmış SNMPv3	SNMPv3	MQTT	CoAP
Sorgu Başına Düşen Ortalama Bant Genişliği (Bayt)	786,4	874,5	3873,2	1288,6
Sorgu Başına Düşen Ortalama Paket Sayısı	4,8	4,8	8,1	12
Sorgu Başına Düşen Ortalama Tepki Süresi (ms)	6,03	6,69	14,57	77,6

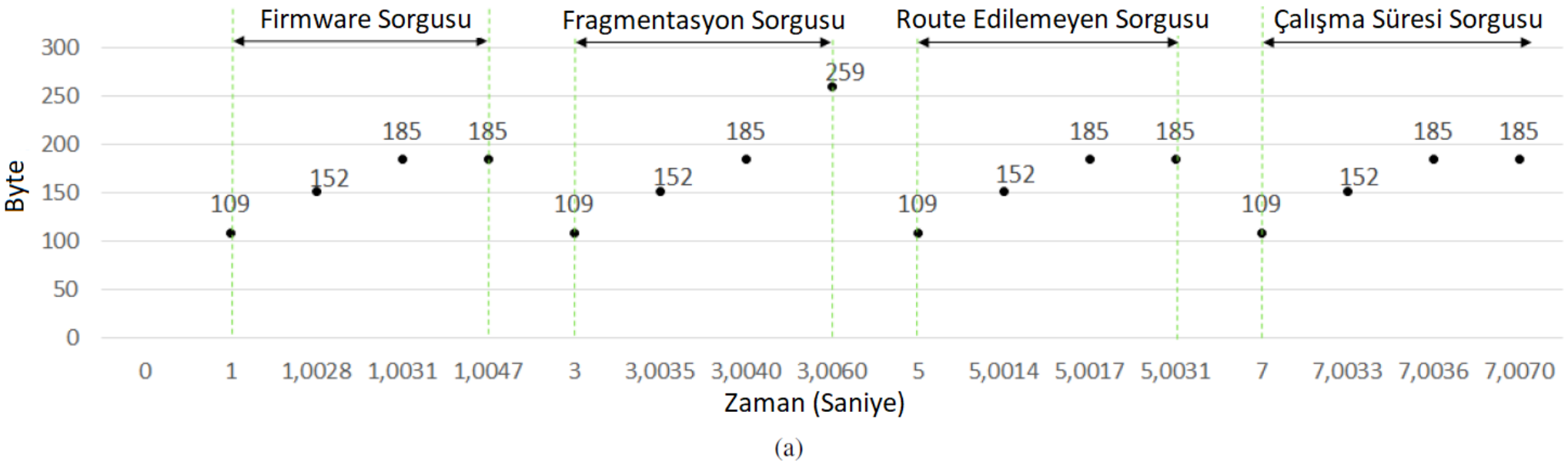
Tablo-4 Yönetim Verilerinin Alınması için Farklı Protokollerin Performans Karşılaştırması

problem konusu oluşturmaktadır. Mavi serideki gösterilen son protokol ise, DTLS desteği ile CoAP protokolünün çalışma performansıdır. CoAP ile çalışırken, orada uygulanan şifrelemeden kaynaklanan çok fazla mesajlaşma gereksinimi bulunur. Çok fazla mesajlaşmanın bu durumda yanıt süresine de olumsuz etki etmektedir. Sistemin zamanı aynı zamanda. Paket boyutlarının çok yüksek olmadığı gözlenebilir. Tablo-4, tüm protokoller için sistem çalıştırıldığındaki performans durumlarını özetlemektedir. Bant genişliği kullanımı, yanıt süresi ve her sorgu için gönderilen paketlerin sayısı ortalamaları karşılaştırmak için dört protokol için de tüm süreçler çalıştırılmıştır.

Diğer yapılan bir test ise yine aynı dört protokolü kullanarak dört adet farklı sorgunun yapıldığı ve paketlerin wireshark kullanılarak gözlenmesidir. Bu test için sonuçlar Şekil-31'de bulunmaktadır. Yapılan dört sorgu sırasıyla cihazdaki firmware'in güncel olup olmadığını kontrol eden sorgudur. İkinci sorgu ise linklerde herhangi bir fragmantasyon olup olmadığını kontrol eden sorgudur. Üçüncü sorgu link erişim problemi, link kalite problemi, sensör işleme problemi veya herhangi bir sebepten ötürü yönlendirme problemi olup olmadığını kontrol eder. Son olarak, cihazın ne kadar süredir alıştığını kontrol eden uptime sorgusu bulunmaktadır. Her protokol tipi için bu dört sorgu arka arkaya 6LoWPAN ağında çalıştırılmıştır.

Yorumlar

Testler sırasında MQTT'yi ağ geçidine kadar kullanılmıştır ve MQTT-SN ise ağ geçidi ile düğümler arasında kullanılmıştır. Ancak, MQTT-SN'nin yeterli güvenlik gereksinimleri bulunmamaktadır. Bu durumda, sunucu ile yalnızca ağ geçidine kadar olan ve kısmen güvenli bir iletişim gerçekleştirilmektedir. Ayrıca, sınırlı kapasite göz önünde

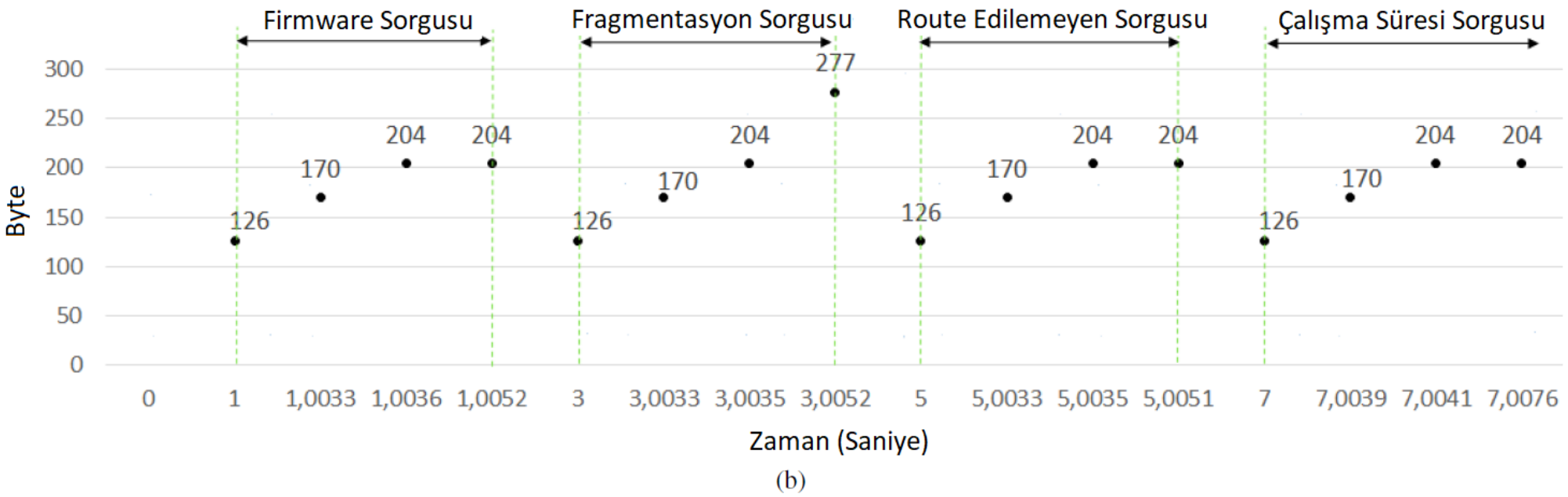


(a)

Şekil-31 Yönetim Sorguları için Protokollerin Performans Sonuçları

(a) Protokol I: Başlık Sıkıştırması Uygulanmış SNMPv3 (b) Protokol II: SNMPv3

(c) Protokol III: MQTT (d) Protokol IV: CoAP

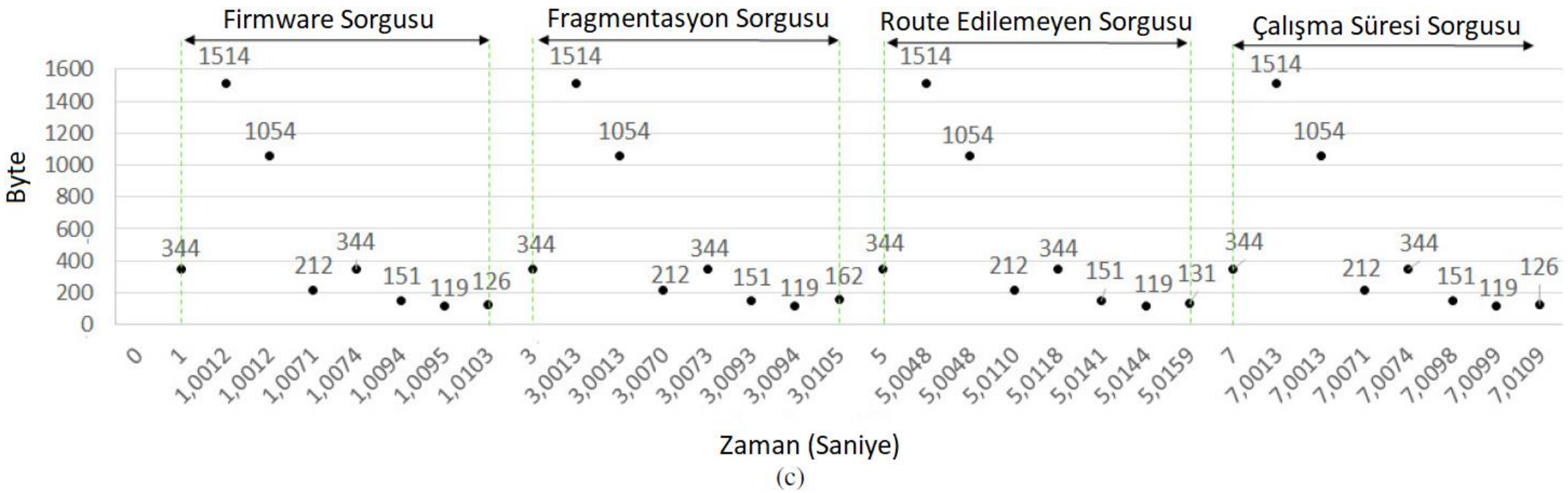


(b)

Şekil-31 (Devamı) Yönetim Sorguları için Protokollerin Performans Sonuçları

(a) Protokol I: Başlık Sıkıştırması Uygulanmış SNMPv3 (b) Protokol II: SNMPv3

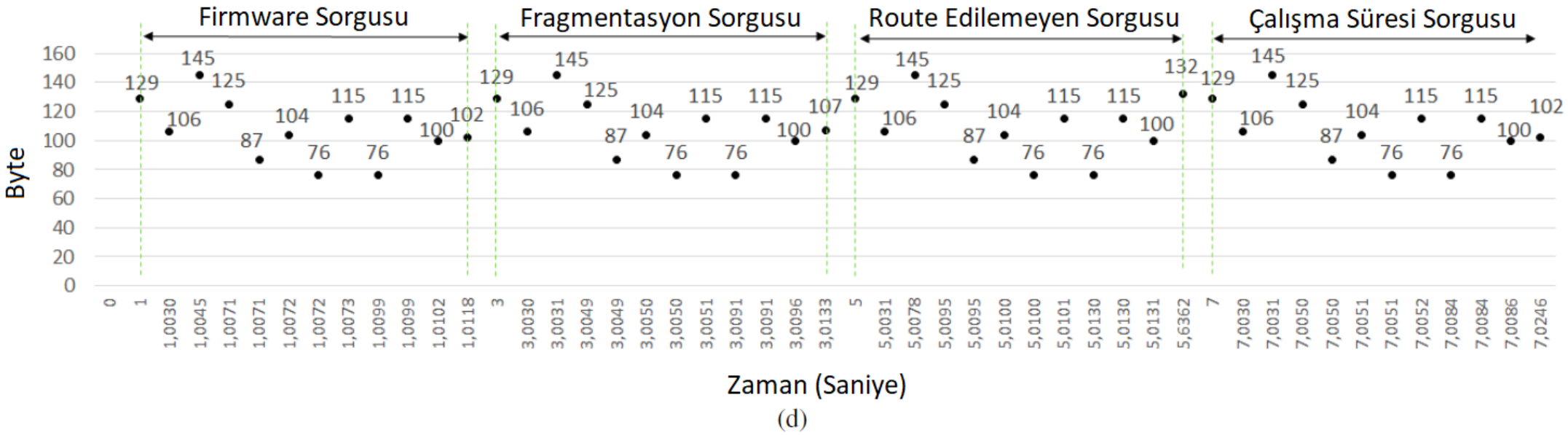
(c) Protokol III: MQTT (d) Protokol IV: CoAP



Şekil-31 (Devamı) Yönetim Sorguları için Protokollerin Performans Sonuçları

(a) Protokol I: Başlık Sıkıştırması Uygulanmış SNMPv3 (b) Protokol II: SNMPv3

(c) Protokol III: MQTT (d) Protokol IV: CoAP



Şekil-31 (Devamı) Yönetim Sorguları için Protokollerin Performans Sonuçları

(a) Protokol I: Başlık Sıkıştırması Uygulanmış SNMPv3 (b) Protokol II: SNMPv3

(c) Protokol III: MQTT (d) Protokol IV: CoAP

bulundurulduğunda TLS sertifika değişimi için çok geniş bir bant genişliği gerekmektedir. CoAP mevcut çalıştığı durumda, DTLS olmadan veri aktarımını yapmaktadır. Bu çalışma modunda çok az bant genişliği gerekmektedir. Sonuçlarda görüleceği üzere, DTLS oturumunu başlatmak ve devam ettirmek için çok fazla sayıda mesaj gönderilmesi gerekir. UDP çalışan ağında daha az mesajlaşma ile çalışma çevre daha uygun bir yaklaşımdır. Bu kadar sık mesajlaşma trafiği ayrıca uç cihazlar tarafından ilgili sorgu için verilen yanıt süresini de olumsuz etkiler. Başlık sıkıştırılmış SNMPv3 ve standart SNMPv3, gerekli sorgular için aynı sayıda ileti göndermektedir. Diğer protokoller ile karşılaştırıldığında son cihazlar tarafından daha az mesajlaşma yapılmaktadır.

6LoWPAN ağındaki yönetim işlemleri için dört IoT protokolü karşılaştırılmıştır. Her protokolün yönetimsel amaçlar için kullanıldıklarında karakterleri, limitasyonları ve kullanım durumundaki faydaları Tablo-5'de özet olarak belirtilmiştir. Birbirleri ile karşılaştırılabilmesi adına Tablo-5'de ilgili başlık altına kısa ifadeler eklenmiştir.

Test sisteminde en yüksek veri transferine izin vermek için testler 2.4 GHz'de frekansında yapılmıştır. 6LoWPAN'ın çalışabileceği 868 Mhz frekansı da bulunmaktadır. Mesafe ve harici nedeniyle etkiler, bu frekansta ulaşılan bant genişliği azaltmaktadır. Sıkıştırma yapılırsa daha fazla bant genişliği elde edilmektedir. Yüksek paket boyutları 6LoWPAN ağlarında parçalanmaya neden olur ve bu çok istenmeyen bir durumdur. Sebebi ise parçalanma ile oluşan birden fazla paketin tekrar bir araya getirilmesi sensör cihazında kaynak tüketmektedir. SNMPv3. Başlık sıkıştırması parçalanma kaybını önlemek için kullanılabilir. Başlık sıkıştırması ile her bir SNMPv3 paketinde yüzde 15'lik bir bant genişliği kazancı sağlanır. Başlık sıkıştırma yaklaşımındaki en özenli kazanım, bazı paketler için parçalanmayı önleme imkânı sağlamasıdır. Başlık sıkıştırma sayesinde paket kaybından kaçınılmaktadır.

Protokol	Karakteristik	Limitasyon	Fayda
Protokol I: Başlık Sıkıştırması Uygulanmış SNMPv3	Kullanıcı güvenlik modeli uygulandı. Mesaj tanımlayıcı alanı daraltıldı ve dört alan SNMPv3 paketinden çıkarılmıştır.	Standart SNMPv3 ile karşılaştırıldığında nispeten daha yavaş tepki süresi	Bant Genişliğinde % 10 oranında kazanç. Yönetim için daha az paket kullanımı. İyi tanımlanmış yönetim verisi yapısı. SNMPv3'e göre parçalanmaya daha az eğilimli
Protokol II: SNMPv3	Kullanıcı güvenlik modeli uygulanmıştır.	Ekstra hafıza gerektirir. Birçok düğümden yönetimsel veri almak için ideal değil.	Sorgularda birkaç paket gönderilir. İyi tanımlanmış yönetim verileri yapısı bulunmaktadır.
Protokol III: MQTT	Aktarım katmanı güvenliği ile birlikte kullanılır 6LoWPAN içinde MQTT-SN çalışmalıdır.	TLS sertifika değişikliği büyük paketler oluşturur. Bu istenmemektedir. Ağ geçidi ile uç düğümler arasında MQTT-SN gerekir. Kısmen güvenlidir.	Birçok sensörden aynı anda yöneim bilgisi almak için uygundur. Programcılar algılama verileri göndermek için MQTT kullanır.Geliştiriciler için idealdir.
Protokol IV: CoAP	DTLS desteği ile uygulanır. DTLS protokol standardı içerisinde tanımlıdır.	DTLS oturumu için çok fazla paket trafiği mevcuttur.Paketlerin işlem ve tepki süresini yavaşlatır.	Gönderilen paketler diğer protokollerle karşılaştırıldığında en küçük boyuttadır.

Tablo-5 6LoWPAN Yönetimi için Değerlendirilen 4 Protokolün Karakteristikleri, Limitasyonları ve Faydaları

BÖLÜM 4. SONUÇ

Özet

Bu çalışmada, SNMPv3 protokolüne dayalı olarak kendini organize eden bir 6LoWPAN mimarisini açıklanmıştır. İlk sürüm SNMP protokolünün yayınlanmasının üzerinden uzun zaman geçmiştir ve bu nedenle bu protokolü, 6LoWPAN ağları ve ağ içerisindeki cihazlar ile uyumlu hale getirmek için bazı değişiklikler yapılması gerekmektedir. 6LoWPAN ağları, SNMPv3'te bazı değişiklikler yapıldığında tıpkı geleneksel ağlar gibi yönetilebilmektedir. Hedefimiz protokolde bazı uyarlamalar yaparak SNMPv3 benzeri bir protokol oluşturmak değildir. Yaklaşımımızda SNMPv3 protokolü, standardında bazı esnemeler yapılarak ve tanımlanmış bazı özelliklerin bulut içerisine aktarılması ile birlikte, sensör ağları için kullanışlı ve verimli bir hale getirilmiştir.

Yargı

Günümüzün sensör ağları için bulutla olan kesintisiz bağlantının vazgeçilmez olduğunu düşündüğümüzde, kendi kendini organize eden bir yönetim sistemi için, protokol içerisinden kaldırılmış bazı alanların bulut içerisinden sağlanması mantıklıdır. Sistemin en kritik modülü, ağın ihtiyacına bağlı olarak kuralları tanımlayan organizasyon modülüdür ve ağın güç açısında ayakta kalan süresine doğrudan etkisi bulunmaktadır. Saha uygulamalarında, yönetim bilgisi alma sıklığı ihtiyaçlara göre değişebilir. Yönetim seviyesi ve cihazların kesintisiz çalışması ihtiyacı arasında da önemli bir karar verilmesi gerekmektedir, çünkü çok sık yönetim verisi alınması cihazın pil ömrüne olumsuz etki etmektedir.

Kendi kendine organizasyon amaçları için SNMPv3 protokolünü kullanma konusunda hem pratik hem de teorik bilgiler önemli bilgiler sağlanmıştır. Gözlemlerde SNMP'nin sensör ağlarının mevcutta veri aktarımı için kullanılan bulut bağlantı özelliğini kullanılması ve protokol özelliklerinin bulut içerisine aktarılması gerektiğini, böylece bu protokolün ideal olarak düşük güçlü cihazlarla kullanılabilecek hale geldiğidir. Ayrıca, başlık sıkıştırmanın yansıra aynı zamanda paket içerisinde taşınan bilginin uygun bir veri sıkıştırma yöntemi ile birlikte uygulanması ile bant genişliğinden sağlanan tasarruf artırılabilir. Yapılan testlerde başlık sıkıştırma işleminin başarılı bir 6LoWPAN ağı yönetimi için yeterli olduğu tespit edilmiştir. Bu çalışmada, önerilen ve geliştirdiğimiz bir sunucu üzerine kurulan, kendi kendini yönetebilen yönetim sistemin 6LoWPAN ağlarını başarılı bir

şekilde yönetebildiği görülmüştür. Bu tez çalışmasında ilgili literatür kısmında belirtilen, çeşitli araştırmacılar tarafından yapılan çalışmalarda 6LoWPAN yönetim sistemi için gereksinimler, geniş çerçeveler içerisinde sunulmuştur. Ancak gerçek hayat uygulamalarına bakıldığında, bu geniş çerçevenin gerçek hayata nasıl uygulanacağına dair detaylı çalışmalar bulunmamaktadır. Bu tez çalışmasında 6LoWPAN ağları için kullanılabilecek olan, SNMPv3 tabanlı olarak çalışan ve gerçek hayattaki kullanım için uygulanabilecek bir yönetim sistemi, uygulama detayları ile birlikte açıklanmıştır. Bu çalışmada belirtilen yöntem haricinde çalışmadaki kurallara bağlı kalarak farklı donanımlar üzerinde farklı bir uygulama tasarlanabilir. Bu çalışmadaki ağı yönetim kuralları ve başlık sıkıştırma işlemlerine tamamen bağlı kalınması durumunda 6LoWPAN ağı başarılı bir şekilde yönetilebilecektir.

Öneriler

Bu çalışma çeşitli yönlerde genişletilebilir. Mevcut sistem sırayla bir sonraki kuralı seçmek için geliştirilmiştir. Kural kararını etkilemek için bir sıralama faktörü kullanılarak sisteme ayrı bir hesaplama modülü eklenerek başka bir çalışma yapılabilir. Doğru kuralı seçmek için makine öğrenme yaklaşımı, geleceğe yönelik değerlendirmemiz altındadır. Aynı zamanda makine öğrenmesine ek olarak, cihaz yönetimi otomatize şekilde yönetebilecek olan, bir yapay zekâ karar yöntemi konuyu farklı noktaya taşıyabilir. Yönetimsel verileri işleyen sistem gelecekte tüm farklı IoT protokollerini işleyecek şekilde çalışan bir sistem tasarlanabilir. Tüm yönetim sisteminin buluttan alınarak, 6LoWPAN ağ geçidi üzerine aktarılması da ayrı bir konu olarak düşünülebilir. Bu çalışmada SNMPv3 paketi içerisinde başlık sıkıştırma işleminin nasıl uygulanabileceği incelenmiştir. Başlık sıkıştırmaya ek olarak SNMPv3 paketinde taşınan yük kısmı için ayrı bir yük sıkıştırma çalışması yapılabilir.

Yapılan çalışmada 6LoWPAN ağındaki nesneleri yönetmek için bir yönetim sistemi tasarlanmıştı. Farklı bir sistem tasarlanarak farklı IoT teknoloji tipleri ile çalışan cihazları da kapsayacak bir sistem tasarlanabilir. Bu teknoloji tipleri LoRA, Zigbee veya mobil şebeke üzerinden çalışan nesneleri içerebilir. Ayrıca geliştirilecek olan bu sistem sadece başlık sıkıştırma yapılmış SNMPv3 protokolü değil, CoAP, MQTT, vb. gibi diğer tüm IoT protokolleri ile birlikte çalışabilecek şekilde genişletilebilir.

EK-1

Yönetim Scriptleri

Geliştirilen sistemde 2 tür script bulunmaktadır. Bunlardan ilki yönetim sistemi içerisinde çalışacak olan PERL scriptleridir. Diğerleri ise sensör cihazların içerisinde çalışacak olan Shell scriptleridir. İlk olarak, yönetim sistemi içerisinde sorgulama işlemleri için geliştirilen PERL scriptleri bu kısımda belirtilmiştir. Tüm scriptler aşağıdaki yöntemler kullanılarak yazılmıştır ve belirtilen scriptlerin başlangıcında aşağıdaki yöntemlerin tamamı kullanılmıştır. Kullanılan PERL yöntemleri aşağıdaki şekildedir.

```
#! /usr/local/bin/perl

use strict;
use warnings;
use SNMP;
use Socket;
use DateTime;
use POSIX qw(strftime);
use HTTP::Date qw( time2iso str2time );
use DBI;
use DBD::mysql;
use Data::Dumper;
use 5.10.0;
use Data::Dumper qw(Dumper);
use Cache::FileCache;
use Cache::MemoryCache;
use Path::Tiny;
use autodie;
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların güncel tarih ve zaman bilgisini sorgular ve veri tabanına yazar.

```
my $sorguhedef = 'localhost';
my $ilgilimib  = '.1.3.6.1.2.1.25.1.2';
my $oturum;
my $sorgugetir;
my $yenisorgu;

&SNMP::initMib();
my %snmpparms;
$snmpparms{DestHost} = $sorguhedef;
$snmpparms{UseSprintValue} = '1';
$oturum = new SNMP::Session(%snmpparms);
$yenisorgu = new SNMP::Varbind([$ilgilimib]);
$sorgugetir = $oturum->get($yenisorgu);
```

```
if ($oturum->{ErrorNum}) {
      die "Got $oturum->{ErrorStr} querying $sorguhedef for
      $ilgilimib.\n";
      }

my (my $year, my $month, my $day) = $sorgugetir =~ m{^(
\d{4}) (\d{2})(\d{2})};

my $time      = str2time("$year\-$month\-$day");
my $isodate   = time2iso($time);
my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI-connect("DBI:mysql: database =
$veritabanim;host=$sorgulanan;port=$myport",'root','admin')
or die "Connection Error: $DBI::errstr\n";

my $sorgum  = "INSERT INTO node1s (query,result) VALUES
('date',?);";

my $dbislem= $veritabanimh->prepare($sorgum );

$dbislem->execute($isodate) or die "Couldn't execute
statement: $DBI::errstr; stopped";

my $now_string = localtime;

my ($serveryear, $servermonth, $serverday) = $now_string =~
m{^(\d{4})(\d{2})(\d{2})};

my $servertime = str2time("$serveryear\-$servermonth\-
$serverday");

my $serverdate = time2iso($servertime);
my $count = 0;
exit;
$veritabanimh->disconnect();
exit 0;
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların arayüzünde herhangi bir paket düşüşü olup olmadığını sorgular ve veri tabanına yazar.

```
my $sorguhedef = 'localhost';
my $ilgilimib  = '.1.3.6.1.2.1.2.2.1.13.1';
my $mib1  = '.1.3.6.1.2.1.2.2.1.19.1';
my $oturum;
my $sorgugetir;
my $sorgugetir1;
my $yenisorgu;
my $yenisorgu1;
```

```
&SNMP::initMib();
my %snmpparms;
$snmpparms{DestHost} = $sorguhedef;
$snmpparms{UseSprintValue} = '1';
$oturum = new SNMP::Session(%snmpparms);

$yenisorgu = new SNMP::Varbind([$ilgilimib]);
$yenisorgu1 = new SNMP::Varbind([$mib1]);
$sorgugetir = $oturum->get($yenisorgu);
$sorgugetir1 = $oturum->get($yenisorgu1);

if ($oturum->{ErrorNum}) {
      die "Got $oturum->{ErrorStr} querying $sorguhedef for
      $ilgilimib.\n";
      }

my $ifDiscards = $sorgugetir. ' ' .$sorgugetir1;
my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI->connect("DBI:mysql: database
=$veritabanim;host=$sorgulanan;port=$myport",'root','admin'
) or die "Connection Error: $DBI::errstr\n";

my $sorgum  = "INSERT INTO node1s (query,result) VALUES
('interface discards',?);";

my $dbislem= $veritabanimh->prepare($sorgum );

$dbislem->execute($ifDiscards) or die "Couldn't execute
statement: $DBI::errstr; stopped";

my $count = 0;
exit;
$veritabanimh->disconnect();
exit 0;
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların arayüzünde herhangi bir paket bozulması olup olmadığını sorgular ve veri tabanına yazar.

```
my $sorguhedef = 'localhost';
my $ilgilimib  = '.1.3.6.1.2.1.2.2.1.14.1';
my $mib1  = '.1.3.6.1.2.1.2.2.1.20.1';
my $oturum;
my $sorgugetir;
my $yenisorgu;
my $sorgugetir1;
my $yenisorgu1;

&SNMP::initMib();
```

```
my %snmpparms;
$snmpparms{DestHost} = inet_ntoa(inet_aton($sorguhedef));
$snmpparms{UseSprintValue} = '1';
$oturum = new SNMP::Session(%snmpparms);

$yenisorgu = new SNMP::Varbind([$ilgilimib]);
$yenisorgu1 = new SNMP::Varbind([$mib1]);
$sorgugetir = $oturum->get($yenisorgu);
$sorgugetir1 = $oturum->get($yenisorgu1);

if ($oturum->{ErrorNum}) {
     die "Got $oturum->{ErrorStr} querying $sorguhedef for
     $ilgilimib.\n";
     }

my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI->connect("DBI:mysql:database
=$veritabanim;host=$sorgulanan;port=$myport",'root','admin'
) or die "Connection Error: $DBI::errstr\n";

my $sorgum  = "INSERT INTO node1s (query,result) VALUES
('interface err status',?);";

my $dbislem = $veritabanimh->prepare($sorgum );

$dbislem->execute($sorgugetir." ".$sorgugetir1) or die
"Couldn't execute statement: $DBI::errstr; stopped";

my $discards = $sorgugetir.' ' .$sorgugetir1;

my $count = 0;
exit;
$veritabanimh->disconnect();
exit 0;
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların işlemci durum bilgisini sorgular ve veri tabanına yazar.

```
my $sorguhedef = 'localhost';
my $ilgilimib  = '.1.3.6.1.4.1.2021.10.1.3.1';
my $mib1  = '.1.3.6.1.4.1.2021.10.1.3.2';
my $mib2 = '.1.3.6.1.4.1.2021.10.1.3.3';
my $oturum;
my $sorgugetir;
my $sorgugetir1;
my $sorgugetir2;
my $yenisorgu;
my $yenisorgu1;
```

```
my $yenisorgu2;

&SNMP::initMib();
my %snmpparms;
$snmpparms{DestHost} = $sorguhedef;
$snmpparms{UseSprintValue} = '1';
$oturum = new SNMP::Session(%snmpparms);
$yenisorgu = new SNMP::Varbind([$ilgilimib]);
$yenisorgu1 = new SNMP::Varbind([$mib1]);.
$yenisorgu2 = new SNMP::Varbind([$mib2]);
$sorgugetir = $oturum->get($yenisorgu);

if ($oturum->{ErrorNum}) {
     die "Got $oturum->{ErrorStr} querying $sorguhedef for
     $ilgilimib.\n";
     }

my $cpuload = $sorgugetir;
my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI->connect("DBI:mysql:databas
e=$veritabanim;host=$sorgulanan;port=$myport",'root','admin
') or die "Connection Error: $DBI::errstr\n";

my $sorgum  = "INSERT INTO node1s (query,result) VALUES
('cpu usage %',?);";

my $dbislem = $veritabanimh->prepare($sorgum );

$dbislem->execute($cpuload) or die "Couldn't execute
statement: $DBI::errstr; stopped";

my $count = 0;
exit;
$veritabanimh->disconnect();
exit 0;
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların hafıza durum bilgisini sorgular ve veri tabanına yazar.

```
my $sorguhedef = 'localhost';
my $ilgilimib  = '.1.3.6.1.4.1.2021.4.5.0';
my $mib1  = '.1.3.6.1.4.1.2021.4.6.0';
my $oturum;
my $sorgugetir;
my $sorgugetir1;
my $yenisorgu;
my $yenisorgu1;

&SNMP::initMib();
```

```
my %snmpparms;
$snmpparms{DestHost} = $sorguhedef;
$snmpparms{UseSprintValue} = '1';
$oturum = new SNMP::Session(%snmpparms);
$yenisorgu = new SNMP::Varbind([$ilgilimib]);
$yenisorgu1 = new SNMP::Varbind([$mib1]);
$sorgugetir = $oturum->get($yenisorgu);
$sorgugetir1 = $oturum->get($yenisorgu1);

if ($oturum->{ErrorNum}) {
      die "Got $oturum->{ErrorStr} querying $sorguhedef for
      $ilgilimib.\n";
      }

my ($result_total, $kb) = split /[ ]/, $sorgugetir;
my ($result_free, $kb1) = split /[ ]/, $sorgugetir1;
my $total = int($result_total);
my $free = int($result_free);
my $memUsage =  ( $free / $total ) * 100;
my $memUsage = eval sprintf('%.2f', $memUsage);
my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI->connect("DBI:mysql:database=
$veritabanim;host=$sorgulanan;port=$myport",'root','admin')
or die "Connection Error: $DBI::errstr\n";

my $sorgum  = "INSERT INTO node1s (query,result) VALUES
('memory usage %',?);";

my $dbislem = $veritabanimh->prepare($sorgum );

$dbislem->execute($memUsage) or die "Couldn't execute
statement: $DBI::errstr; stopped";

my $count = 0;
exit;
$veritabanimh->disconnect();
exit 0;
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazlarda fragmentasyon durumunu sorgular ve veri tabanına yazar.

```
my $sorguhedef = 'localhost';
my $ilgilimib  = '.1.3.6.1.2.1.4.31.1.1.26.1';
my $oturum;
my $sorgugetir;
my $yenisorgu;

&SNMP::initMib();
my %snmpparms;
```

```
$snmpparms{DestHost} = $sorguhedef;
$snmpparms{UseSprintValue} = '1';
$oturum = new SNMP::Session(%snmpparms);
$yenisorgu = new SNMP::Varbind([$ilgilimib]);
$sorgugetir = $oturum->get($yenisorgu);

if ($oturum->{ErrorNum}) {
      die "Got $oturum->{ErrorStr} querying $sorguhedef for
      $ilgilimib.\n";
      }

my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI->connect("DBI:mysql:database
=$veritabanim;host=$sorgulanan;port=$myport",'root','admin'
) or die "Connection Error: $DBI::errstr\n";

my $sorgum  = "INSERT INTO node1s (query,result) VALUES
('frag fails',?);";

my $dbislem = $veritabanimh->prepare($sorgum );

$dbislem->execute($sorgugetir) or die "Couldn't execute
statement: $DBI::errstr; stopped";

my $count = 0;
exit;
$veritabanimh->disconnect();
exit 0;
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların arayüzlerinden gelen ve ya giden trafik miktarı bilgisini sorgular ve veri tabanına yazar.

```
my $sorguhedef = 'localhost';
my $ilgilimib  = '.1.3.6.1.2.1.2.2.1.10.1';
my $mib1  = '.1.3.6.1.2.1.2.2.1.16.1';
my $oturum;
my $sorgugetir;
my $yenisorgu;
my $sorgugetir1;
my $yenisorgu1;

&SNMP::initMib();
my %snmpparms;
$snmpparms{DestHost} = inet_ntoa(inet_aton($sorguhedef));
$snmpparms{UseSprintValue} = '1';
$oturum = new SNMP::Session(%snmpparms);
$yenisorgu = new SNMP::Varbind([$ilgilimib]);
$yenisorgu1 = new SNMP::Varbind([$mib1]);
$sorgugetir = $oturum->get($yenisorgu);
```

```
$sorgugetir1 = $oturum->get($yenisorgu1);

if ($oturum->{ErrorNum}) {
      die "Got $oturum->{ErrorStr} querying $sorguhedef for
      $ilgilimib.\n";
      }

my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI->connect("DBI:mysql:database
=$veritabanim;host=$sorgulanan;port=$myport",'root','admin'
) or die "Connection Error: $DBI::errstr\n";

my $sorgum  = "INSERT INTO posts (query,result) VALUES
('interface traffic',?);";

my $dbislem = $veritabanimh->prepare($sorgum );

$dbislem->execute($sorgugetir." ".$sorgugetir1) or die
"Couldn't execute statement: $DBI::errstr; stopped";
print $sorgugetir;

my $count = 0;
exit;
$veritabanimh->disconnect();
exit 0;
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların açık kalma süresini sorgular ve veri tabanına yazar.

```
my $sorguhedef = 'localhost';
my $ilgilimib  = '.1.3.6.1.2.1.25.1.1.0';
my $oturum;
my $sorgugetir;
my $yenisorgu;

&SNMP::initMib();
my %snmpparms;
$snmpparms{DestHost} = $sorguhedef;
$snmpparms{UseSprintValue} = '1';
$oturum = new SNMP::Session(%snmpparms);

$yenisorgu = new SNMP::Varbind([$ilgilimib]);
$sorgugetir = $oturum->get($yenisorgu);

if ($oturum->{ErrorNum}) {
      die "Got $oturum->{ErrorStr} querying $sorguhedef for
      $ilgilimib.\n";
      }
```

```
my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';
my $veritabanimh = DBI->connect("DBI:mysql:database
=$veritabanim;host=$sorgulanan;port=$myport",'root','admin'
) or die "Connection Error: $DBI::errstr\n";

my $sorgum  = "INSERT INTO node1s (query,result) VALUES
('uptime',?);";

my $dbislem = $veritabanimh->prepare($sorgum );

$dbislem->execute($sorgugetir) or die "Couldn't execute
statement: $DBI::errstr; stopped";

my $count = 0;
exit;
$veritabanimh->disconnect();
exit 0;
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların yönlendirme tablolarında bulunan bir sebepten ötürü yönlendirme problemine uğrayan paket olup olmadığını sorgular ve veri tabanına yazar.

```
my $sorguhedef = 'localhost';
my $ilgilimib  = '.1.3.6.1.2.1.4.23.0';
my $oturum;
my $sorgugetir;
my $yenisorgu;

&SNMP::initMib();
my %snmpparms;
$snmpparms{DestHost} = $sorguhedef;
$snmpparms{UseSprintValue} = '1';
$oturum = new SNMP::Session(%snmpparms);
$yenisorgu = new SNMP::Varbind([$ilgilimib]);
$sorgugetir = $oturum->get($yenisorgu);

if ($oturum->{ErrorNum}) {
     die "Got $oturum->{ErrorStr} querying $sorguhedef for
     $ilgilimib.\n";
     }

my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI->connect("DBI:mysql:database
=$veritabanim;host=$sorgulanan;port=$myport",'root','admin'
) or die "Connection Error: $DBI::errstr\n";
```

```
my $sorgum  = "INSERT INTO node1s (query,result) VALUES
('route discards',?);";

my $dbislem = $veritabanimh->prepare($sorgum );

$dbislem->execute($sorgugetir) or die "Couldn't execute
statement: $DBI::errstr; stopped";

my $count = 0;
exit;
$veritabanimh->disconnect();
exit 0;
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların firmware bilgisini sorgular ve veri tabanına yazar.

```
my $sorguhedef = 'localhost';
my $ilgilimib  = '.1.3.6.1.2.1.1.1.0';
my $oturum;
my $sorgugetir;
my $yenisorgu;

&SNMP::initMib();
my %snmpparms;
$snmpparms{DestHost} = $sorguhedef;
$snmpparms{UseSprintValue} = '1';
$oturum = new SNMP::Session(%snmpparms);
$yenisorgu = new SNMP::Varbind([$ilgilimib]);
$sorgugetir = $oturum->get($yenisorgu);

if ($oturum->{ErrorNum}) {
     die "Got $oturum->{ErrorStr} querying $sorguhedef for
     $ilgilimib.\n";
     }

my $data=$sorgugetir;
my @values = split(' ', $data);
my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI >connect("DBI:mysql:database
=$veritabanim;host=$sorgulanan;port=$myport",'root','admin'
) or die "Connection Error: $DBI::errstr\n";

my $sorgum  = "INSERT INTO node1s (query,result) VALUES
('firmware',?);";

my $dbislem = $veritabanimh->prepare($sorgum );

$dbislem->execute($values[5]) or die "Couldn't execute
statement: $DBI::errstr; stopped";
```

```
if ( $values[5] ne "hej"){
      #update firmware fonksiyonunu çağır
      $dbislem1->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";
      }

my $count = 0;
exit;
$veritabanimh->disconnect();

if ( $values[5] != "hej"){
      #update firmware fonksiyonunu çağır
      }

exit 0;
```

Aşağıdaki PERL script, 6LoWPAN şebekesinde broadcast ve ya multicast yayın olup olmadığını sorgular. Bu scriptin çalışması için 2 adet text dosyaı oluşturulması gerekmektedir. Bu dosyaların isimleri bu scriptte bwlog2.txt ve bwlogfile.txt olarak belirtilmiştir. Bu text dosyası isimleri değiştirilebilir.

```
my $sorguhedef = 'localhost';
my $ilgilimib  = '.1.3.6.1.4.1.8072.2.2.1';
open (my $log, '>', '/home/yekta/bwlogfile.txt');
my $oturum;
my $sorgugetir;
my $yenisorgu;

&SNMP::initMib();
my %snmpparms;
$snmpparms{DestHost} = inet_ntoa(inet_aton($sorguhedef));
$snmpparms{UseSprintValue} = '1';
$oturum = new SNMP::Session(%snmpparms);
my $s = new SNMP::Session(DestHost => 'localhost');

print $log Dumper ($s->gettable('netSnmpIETFWGTable'));
close $log;

#Log dosyasında ikinci kısım trafik kimden geliyor

my $beginString = "'1.50'";
my $endString = "}";
my $fileContent;
my $filename = '/home/yekta/bwlogfile.txt';

#ilk kısmı loglara yaz

open(my $fh, $filename) or die"no read '$filename' $!\n";

while(<$fh>) {
      if(/$beginString/../$endString/) {
```

```
      $fileContent .= $_ unless(/$beginString/ or
      /$endString/) ;
            }
      }
close($fh);

open (my $log2, '>', '/home/yekta/bwlog2.txt');
print $log2 Dumper $fileContent;
close $log2;
my $filename2 = '/home/yekta/bwlog2.txt';

open(my $fh2, $filename2) or die"no read '$filename2'
$!\n";

my $substr="hair1";
my $substr2="hair2";
my $substr3="hair3";
my $substr4="hair4";
my @username;

while (my $row = <$fh2>) {
  chomp $row;
      if (index($row, $substr) != -1) {
           my @fields = split />/, $row;
           my @fields2 = split /[\\']/, $fields[1];
           $username[0] = $fields2[2];
            }
      if (index($row, $substr2) != -1) {
           my @fields = split />/, $row;
           my @fields2 = split /[\\']/, $fields[1];
           $username[1] = $fields2[2];
            }
      if (index($row, $substr3) != -1) {
           my @fields = split />/, $row;
           my @fields2 = split /[\\']/, $fields[1];
           $username[2] = $fields2[2];
            }
      if (index($row, $substr4) != -1) {
           my @fields = split />/, $row;
           my @fields2 = split /[\\']/, $fields[1];
           $username[3] = $fields2[2];
            }
      }

close $fh2;

##### Log dosyasındaki 2. kısım gelen trafik 2 saniyelik

my $beginString1 = "'1.51'";
my $endString1 = "}";
my @intraffic;
my $fileContent1;
```

```
#ikinci kısmı loglara yaz

$filename = '/home/yekta/bwlogfile.txt';

open($fh, $filename) or die"no read '$filename' $!\n";

while(<$fh>) {
  if(/$beginString1/../$endString1/) {
     $fileContent1 .= $_ unless(/$beginString1/ or
     /$endString1/) ;
     }

close($fh);
open ($log2, '>', '/home/yekta/bwlog2.txt');
print $log2 Dumper $fileContent1;
close $log2;

$filename2 = '/home/yekta/bwlog2.txt';
open($fh2, $filename2) or die"no read '$filename2' $!\n";

while (my $row = <$fh2>) {
  chomp $row;
     if (index($row, $substr) != -1) {
          if ($row =~ m{K}){
               my @fields = split />/, $row;
               my @fields2 = split /['K]/, $fields[1];
               $intraffic[0] = $fields2[1];
               $intraffic[0] = $intraffic[0] + 0;
               $intraffic[0] = $intraffic[0] * 1000;
                }
          else{
               my @fields = split />/, $row;
               my @fields2 = split /['b]/, $fields[1];
               $intraffic[0] = $fields2[1];
               $intraffic[0] = $intraffic[0] + 0;
                }
          }
     if (index($row, $substr2) != -1) {
          if ($row =~ m{K}){
               my @fields = split />/, $row;
               my @fields2 = split /['K]/, $fields[1];
               $intraffic[1] = $fields2[1];
               $intraffic[1] = $intraffic[1] + 0;
               $intraffic[1] = $intraffic[1] * 1000;

                }
          else{
               my @fields = split />/, $row;
               my @fields2 = split /['b]/, $fields[1];
               $intraffic[1] = $fields2[1];
               $intraffic[1] = $intraffic[1] + 0;
                }
          }
```

```
      if (index($row, $substr3) != -1) {
           if ($row =~ m{K}){
                my @fields = split />/, $row;
                my @fields2 = split /['K]/, $fields[1];
                $intraffic[2] = $fields2[1];
                $intraffic[2] = $intraffic[2] + 0;
                $intraffic[2] = $intraffic[2] * 1000;
                }
           else{
                my @fields = split />/, $row;
                my @fields2 = split /['b]/, $fields[1];
                $intraffic[2] = $fields2[1];
                $intraffic[2] = $intraffic[2] + 0;
                }
           }

      if (index($row, $substr4) != -1) {
           if ($row =~ m{K}){
                my @fields = split />/, $row;
                my @fields2 = split /['K]/, $fields[1];
                $intraffic[3] = $fields2[1];
                $intraffic[3] = $intraffic[3] + 0;
                $intraffic[3] = $intraffic[3] * 1000;
                }
           else{
                my @fields = split />/, $row;
                my @fields2 = split /['b]/, $fields[1];
                $intraffic[3] = $fields2[1];
                $intraffic[3] = $intraffic[3] + 0;
                }
           }
 }

close $fh2;

######Log dosyasındaki 3. kısım giden trafik 2 saniyelik

$beginString1 = "'1.52'";
$endString1 = "}";

my @outtraffic;
my $fileContent2;

#üçüncü kısmı loglara yaz

$filename = '/home/yekta/bwlogfile.txt';

open($fh, $filename) or die"no read '$filename' $!\n";

while(<$fh>) {
  if(/$beginString1/../$endString1/) {
      $fileContent2 .= $_ unless(/$beginString1/ or
      /$endString1/) ;
```

```
            }
      }

close($fh);
open ($log2, '>', '/home/yekta/bwlog2.txt');
print $log2 Dumper $fileContent2;
close $log2;
$filename2 = '/home/yekta/bwlog2.txt';
open($fh2, $filename2) or die"no read '$filename2' $!\n";

while (my $row = <$fh2>) {
  chomp $row;

      if (index($row, $substr) != -1) {
            if ($row =~ m{K}){
                  my @fields = split />/, $row;
                  my @fields2 = split /['K]/, $fields[1];
                  $outtraffic[0] = $fields2[1];
                  $outtraffic[0] = $outtraffic[0] + 0;
                  $outtraffic[0] = $outtraffic[0] * 1000;
                   }
      else{
                  my @fields = split />/, $row;
                  my @fields2 = split /['b]/, $fields[1];
                  $outtraffic[0] = $fields2[1];
                  $outtraffic[0] = $outtraffic[0] + 0;
            }
      }

if (index($row, $substr2) != -1) {
            if ($row =~ m{K}){
                  my @fields = split />/, $row;
                  my @fields2 = split /['K]/, $fields[1];
                  $outtraffic[1] = $fields2[1];
                  $outtraffic[1] = $outtraffic[1] + 0;
                  $outtraffic[1] = $outtraffic[1] * 1000;
            }
      else{
                  my @fields = split />/, $row;
                  my @fields2 = split /['b]/, $fields[1];
                  $outtraffic[1] = $fields2[1];
                   $outtraffic[1] = $outtraffic[1] + 0;

            }
      }

if (index($row, $substr3) != -1) {
            if ($row =~ m{K}){
                  my @fields = split />/, $row;
                  my @fields2 = split /['K]/, $fields[1];
                  $outtraffic[2] = $fields2[1];
                  $outtraffic[2] = $outtraffic[2] + 0;
                  $outtraffic[2] = $outtraffic[2] * 1000;
```

```
            }
      else{
                  my @fields = split />/, $row;
                  my @fields2 = split /['b]/, $fields[1];
                  $outtraffic[2] = $fields2[1];
                  $outtraffic[2] = $outtraffic[2] + 0;
            }
      }

if (index($row, $substr4) != -1) {
            if ($row =~ m{K}){
                  my @fields = split />/, $row;
                  my @fields2 = split /['K]/, $fields[1];
                  $outtraffic[3] = $fields2[1];
                  $outtraffic[3] = $outtraffic[3] + 0;
                  $outtraffic[3] = $outtraffic[3] * 1000;
            }
      else{
                  my @fields = split />/, $row;
                  my @fields2 = split /['b]/, $fields[1];
                  $outtraffic[3] = $fields2[1];
                        $outtraffic[3] = $outtraffic[3] + 0;
             }
      }
 }

close $fh2;

# Önce in yönünde en çok trafik oluşturanı bul

my $index1 = 0;
my $maxval = $intraffic[ $index1 ];

for my $i ( 0 .. $#intraffic )
      {
     $maxval < $intraffic[$i] and
     $index1 = $i
      }

#sonra out yönündeki trafiği kontrol et

my $index = 0;
my $maxval1 = $outtraffic[ $index ];

for my $i ( 0 .. $#outtraffic )
      {
     $maxval1 < $outtraffic[$i] and
     $index = $i
      }

#en büyük olanı bul ve dosyaya yaz

if(($intraffic[$index1]) > ($outtraffic[$index])){
```

```
print "$username[$index1]";
}

else {
print "$username[$index]";
}

exit;
```

Sorgulama scriptleri sensör cihazlarından gerekli nesnelerin durum bilgilerini toplayıp veritabanındaki ilgili tabloya yazmaktadır. Sonrasında veritabanını tarayan ve aksiyon alan scriptler çalışmaktadır Kontrol işlemini yapıp sonrasında gerekli aksiyonu çağıran scriptler aşğıda belirtilmektedir.

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların güncel tarih ve zaman bilgisinide sorun olup olmadığını kontrol eder. Sorun mevcutsa ilgili aksiyon sürecini veya işlemini işletir.

```
sub set_value{

my ($p1) = @_;
my $sorguhedef = 'localhost';
my %snmpparms;
$snmpparms{DestHost} = inet_ntoa(inet_aton($sorguhedef));
$snmpparms{UseSprintValue} = 1;
my $oturum = new SNMP::Session(%snmpparms);
my $ilgilimib = 'netSnmpExampleInteger';
my $instance = '0';
my $value = $p1;
my $yenisorgu = new
SNMP::Varbind([$ilgilimib,$instance,$value]);
$oturum->set($yenisorgu);
}

my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI->connect("DBI:mysql:database
=$veritabanim;host=$sorgulanan;port=$myport",'root','admin'
) or die "Connection Error: $DBI::errstr\n";

my $sth = $veritabanimh->prepare( "SELECT result FROM posts
WHERE query='date'" );

my $sorgum 1 = "INSERT INTO posts (query,result) VALUES
('alarm','service denial risk');";

my $sorgum 2 = "INSERT INTO posts (query,result) VALUES
('action','update ntp process');";
```

```
my $dbislem2 = $veritabanimh->prepare($sorgum 2);
my $dbislem1 = $veritabanimh->prepare($sorgum 1);
$sth->execute();
my @row;
my @arrayo;
my $i=0;
my $count=0;

while ( @row = $sth->fetchrow_array ) {
     push @arrayo,@row;
     }

for $i (1 .. $#arrayo){
     if ($arrayo[$i] lt $arrayo[$i-1]){
          $count++;
          }
          $count = $count % 3;
     }

if($count == 1){
     $dbislem1->execute() or die "Couldn't execute
     statement: $DBI::errstr; stopped";

     my $output = `./gw1bandwidthcontrol.pl`;
          }

if($count == 2){
     $a=14;
     set_value($a);

     $dbislem1->execute() or die "Couldn't execute
     statement: $DBI::errstr; stopped";

     $dbislem2->execute() or die "Couldn't execute
     statement: $DBI::errstr; stopped";
          }

$sth->finish();
$veritabanimh->disconnect();
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların arayüzünde herhangi bir paket düşüşü olup olmadığını kontrol eder. Sorun mevcutsa ilgili aksiyon sürecini veya işlemini işletir.

```
sub set_value{

my ($p1) = @_;
my $sorguhedef = 'localhost';
my %snmpparms;
$snmpparms{DestHost} = inet_ntoa(inet_aton($sorguhedef));
$snmpparms{UseSprintValue} = 1;
```

```
my $oturum = new SNMP::Session(%snmpparms);
my $ilgilimib = 'netSnmpExampleInteger';
my $instance = '0';
my $value = $p1;
my $yenisorgu = new
SNMP::Varbind([$ilgilimib,$instance,$value]);
$oturum->set($yenisorgu);
}

my $a;
my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI->connect("DBI:mysql:database
=$veritabanim;host=$sorgulanan;port=$myport",'root','admin'
) or die "Connection Error: $DBI::errstr\n";

my $sth = $veritabanimh->prepare( "SELECT result FROM posts
WHERE query='interface discards'" );

my $sorgum 1 = "INSERT INTO posts (query,result) VALUES
('alarm','link quality');";

my $sorgum 2 = "INSERT INTO posts (query,result) VALUES
('action','restart interface');";

my $sorgum 3 = "INSERT INTO posts (query,result) VALUES
('action','increase mtu');";

my $sorgum 4 = "INSERT INTO posts (query,result) VALUES
('action','increase buffer');";

my $dbislem4 = $veritabanimh->prepare($sorgum 4);
my $dbislem3 = $veritabanimh->prepare($sorgum 3);
my $dbislem2 = $veritabanimh->prepare($sorgum 2);
my $dbislem1 = $veritabanimh->prepare($sorgum 1);
$sth->execute();

my @row;
my @arrayo;
my $count=0;
my $actioncount=0;
my $indiscards=0;
my $outdiscards=0;
my $indiscards1=0;
my $outdiscards1=0;

while ( @row = $sth->fetchrow_array ) {
     push @arrayo,@row;
     }

for my $i (0 .. $#arrayo){
```

```
      my @son =  split /[ ]/, $arrayo[$i];
      my @sonbir =  split /[ ]/, $arrayo[$i-1];
      my $indis = $sonbir[0];
      my $outdis = $sonbir[1];
      my $indis1 = $son[0];
      my $outdis1 = $son[1];
      $indis = $indis +1;
      $outdis = $outdis+1;
      $indis1 = $indis1+1;
      $outdis1 = $outdis1+1;

      if ((($indis1 != 0)) & ($indis > $indis1)){
      $count++;
            }
      }

$count = $count % 7;

if($count == 1){
      # firmware güncelleme
      $a= 5;

      $dbislem1->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";

      my $output = `./gw1firmwarecontrol.pl`;
      set_value($a);
      }

if($count == 2){
      # arayüzü yeniden başlat
      $a= 8;

      $dbislem1->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";

      $dbislem2->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";

      set_value($a);
      }

if($count == 3){
      # kontrol cpu kullanımı

      $dbislem1->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";
      my $output = `./gw1cpucontrol.pl`;
      }

if($count == 4){
      # buffer'ı arttır
      $a= 8;
```

```
        set_value($a);

        $dbislem1->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";

        $dbislem3->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";
        }

if($count == 5){
        # kontrol uygulama mtu değeri
        $a=03;
        set_value($a);
        $dbislem1->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";

        $dbislem4->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";
        }

if($count == 6){
        # kontrol multicast / broadcast
        $dbislem1->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";
        my $output = `./mcast-bcast-control.pl`;
        }

$sth->finish();
$veritabanimh->disconnect();
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların arayüzünde herhangi bir paket bozulması olup olmadığını kontrol eder. Sorun mevcutsa ilgili aksiyon sürecini veya işlemini işletir.

```
sub set_value{

my ($p1) = @_;
my $sorguhedef = 'localhost';
my %snmpparms;
$snmpparms{DestHost} = inet_ntoa(inet_aton($sorguhedef));
$snmpparms{UseSprintValue} = 1;
my $oturum = new SNMP::Session(%snmpparms);
my $ilgilimib = 'netSnmpExampleInteger';
my $instance = '0';
my $value = $p1;
my $yenisorgu = new
SNMP::Varbind([$ilgilimib,$instance,$value]);
$oturum->set($yenisorgu);
}

my $a;
```

```
my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI->connect("DBI:mysql:databas
e=$veritabanim;host=$sorgulanan;port=$myport",'root','admin
') or die "Connection Error: $DBI::errstr\n";

my $sth = $veritabanimh->prepare( "SELECT result FROM posts
WHERE query='interface err status'" );

my $sorgum 1 = "INSERT INTO posts (query,result) VALUES
('alarm','high link error');";

my $sorgum 2 = "INSERT INTO posts (query,result) VALUES
('action','change channel');";

my $dbislem2 = $veritabanimh->prepare($sorgum 2);

my $sorgum 3 = "INSERT INTO posts (query,result) VALUES
('action','increase power');";

my $dbislem3 = $veritabanimh->prepare($sorgum 3);
my $dbislem1 = $veritabanimh->prepare($sorgum 1);
$sth->execute();
my @row;
my @arrayo;
my $count=0;
my $actioncount=0;

while ( @row = $sth->fetchrow_array ) {
push @arrayo,@row;
      }

for my $i (0 .. $#arrayo){
      my ($inerrors, $outerrors) = split /[ ]/,
      $arrayo[$i];

      my ($inerrors1, $outerrors1) = split /[ ]/,
      $arrayo[$i-1];

      if (($inerrors > ($inerrors1*2))||($outerrors >
      ($outerrors1*2))){
                  $count = $count+2;
                  }

      elsif (($inerrors > $inerrors1)||($outerrors >
      $outerrors1)){
      $count=$count+1;
                  }
      }

$count = $count % 3;
```

```
if($count == 1){
      $a=19;
      set_value($a);
      $dbislem1->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";
      $dbislem2->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";
      # kanal değişimi yap
      }

if($count == 2){
      # gücü arttır
      $a=20;
      set_value($a);
      $dbislem1->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";
      $dbislem3->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";
      }

$sth->finish();
$veritabanimh->disconnect();
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların işlemci durum bilgisini kontrol eder. Sorun mevcutsa ilgili aksiyon sürecini veya işlemini işletir.

```
sub set_value{

my ($p1) = @_;
my $sorguhedef = 'localhost';
my %snmpparms;
$snmpparms{DestHost} = inet_ntoa(inet_aton($sorguhedef));
$snmpparms{UseSprintValue} = 1;
my $oturum = new SNMP::Session(%snmpparms);
my $ilgilimib = 'netSnmpExampleInteger';
my $instance = '0';
my $value = $p1;
my $yenisorgu = new
SNMP::Varbind([$ilgilimib,$instance,$value]);
$oturum->set($yenisorgu);
}

my $a;
my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI->connect("DBI:mysql:database =
$veritabanim;host=$sorgulanan;port=$myport",'root','admin')
or die "Connection Error: $DBI::errstr\n";
```

```
my $sth = $veritabanimh->prepare( "SELECT result FROM posts
WHERE query='cpu usage %'" );

my $sorgum 2 = "INSERT INTO posts (query,result) VALUES
('action','kill sleeping process');";

my $sorgum 3 = "INSERT INTO posts (query,result) VALUES
('action','renice process');";

my $sorgum 4 = "INSERT INTO posts (query,result) VALUES
('action','kill process');";

my $dbislem4 = $veritabanimh->prepare($sorgum 4);
my $dbislem3 = $veritabanimh->prepare($sorgum 3);
my $dbislem2 = $veritabanimh->prepare($sorgum 2);

my $sorgum 1 = "INSERT INTO posts (query,result) VALUES
('alarm','high CPU');";

my $dbislem1 = $veritabanimh->prepare($sorgum 1);
$sth->execute();
my @row;
my $count=0;
my @arrayo;
my $action1count=0;

while ( @row = $sth->fetchrow_array ) {
push @arrayo,@row;
      }

# CPU kullanımı sınırların üzerinde mi kontrol edilir

for my $i (0 .. $#arrayo){
      #my ($oneminload, $fiveminload, $fifteenminload) =
      split /[ ]/, $arrayo[$i];

      #my ($oneminload) = split /[ ]/, $arrayo[$i];
      # check if one minute load is above %90
      #$arrayo[$i]= $arrayo[$i] + 0;
      if ($arrayo[$i] > 90){
            $count++;
            }
      $count = $count % 6;
      }

if($count == 1){

      # kontrol  zombie / sleeping process var mı
      $dbislem1->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";

      $dbislem2->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";
```

```
        $a=17;
        set_value($a);
        }

if($count == 2){
        # kontrol memory consumption / correption var mı
        $DBI::errstr; stopped";

        $dbislem1->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";

        my $output = `./gw1memorycontrol.pl`;
        }

if($count == 3){
        # kontrol tcp security
        $dbislem1->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";

        my $output = `./gw1bandwidthcontrol.pl`;
        }

if($count == 4){
        # renice ve ya cpulimit process yap
        $dbislem1->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";

        $dbislem3->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";

        $a=16;
        set_value($a);
        }

if($count == 5){
        # Yeniden başlatma / kapat process
        $a=10;
        set_value($a);
        $dbislem1->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";

        $dbislem4->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";
        }

$sth->finish();
$veritabanimh->disconnect();
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların hafıza durum bilgisini kontrol eder. Sorun mevcutsa ilgili aksiyon sürecini veya işlemini işletir.

```
sub set_value{

my ($p1) = @_;
my $sorguhedef = 'localhost';
my %snmpparms;
$snmpparms{DestHost} = inet_ntoa(inet_aton($sorguhedef));
$snmpparms{UseSprintValue} = 1;
my $oturum = new SNMP::Session(%snmpparms);
my $ilgilimib = 'netSnmpExampleInteger';
my $instance = '0';
my $value = $p1;
my $yenisorgu = new
SNMP::Varbind([$ilgilimib,$instance,$value]);
$oturum->set($yenisorgu);
}

my $a;
my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI->connect("DBI:mysql:database
=$veritabanim;host=$sorgulanan;port=$myport",'root','admin'
) or die "Connection Error: $DBI::errstr\n";

my $sorgum 1 = "INSERT INTO posts (query,result) VALUES
('alarm','high memory');";

my $sorgum 2 = "INSERT INTO posts (query,result) VALUES
('action','fluch cache');";

my $dbislem2 = $veritabanimh->prepare($sorgum 2);

my $sorgum 3 = "INSERT INTO posts (query,result) VALUES
('action','restart process');";

my $dbislem3 = $veritabanimh->prepare($sorgum 3);

my $sorgum 4 = "INSERT INTO posts (query,result) VALUES
('action','kill process');";

my $dbislem4 = $veritabanimh->prepare($sorgum 4);
my $dbislem1 = $veritabanimh->prepare($sorgum 1);

my $sth = $veritabanimh->prepare( "SELECT result FROM posts
WHERE query='memory usage %'" );

$sth->execute();
my @row;
my @arrayo;
my $count=0;
my $actioncount=0;
```

```
while ( @row = $sth->fetchrow_array ) {
      push @arrayo,@row;
      }

for my $i (0 .. $#arrayo){
      if (@arrayo[$i] > 90){
      $count++;
            }
      }

$count = $count % 4;

# Problem var şimdi $count değerini kontrol et aksiyon için

if($count == 1){
      #flushcache()fonksiyonunu çağır.
      $a=06;
      set_value($a);

      $dbislem1->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";

      $dbislem2->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";
      }

if($count == 2){
#restartprocess()fonksiyonunu çağır.
$a=18;
set_value($a);

      $dbislem1->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";

      $dbislem3->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";
      }

if($count == 3){
#shutprocess()fonksiyonunu çağır.
$a=11;
set_value($a);

      $dbislem1->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";

      $dbislem4->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";
      }

$sth->finish();
$veritabanimh->disconnect();
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazlarda fragmentasyon durumunu kontrol eder. Sorun mevcutsa ilgili aksiyon sürecini veya işlemini işletir.

```
sub set_value{

my ($p1) = @_;
my $sorguhedef = 'localhost';
my %snmpparms;
$snmpparms{DestHost} = inet_ntoa(inet_aton($sorguhedef));
$snmpparms{UseSprintValue} = 1;
my $oturum = new SNMP::Session(%snmpparms);
my $ilgilimib = 'netSnmpExampleInteger';
my $instance = '0';
my $value = $p1;
my $yenisorgu = new
SNMP::Varbind([$ilgilimib,$instance,$value]);

$oturum->set($yenisorgu);
}

my $a;
my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI->connect("DBI:mysql:database
=$veritabanim;host=$sorgulanan;port=$myport",'root','admin'
) or die "Connection Error: $DBI::errstr\n";

my $sth = $veritabanimh->prepare( "SELECT result FROM posts
WHERE query='frag fails'" );

my $sorgum 1 = "INSERT INTO posts (query,result) VALUES
('alarm','service denial risk');";

my $sorgum 2 = "INSERT INTO posts (query,result) VALUES
('action','increase mtu');";

my $dbislem2 = $veritabanimh->prepare($sorgum 2);
my $dbislem1 = $veritabanimh->prepare($sorgum 1);
$sth->execute();

my @row;
my @arrayo;
my $i=0;
my $count=0;
my $firstaction=0;
my $secondaction=0;

while ( @row = $sth->fetchrow_array ) {
     push @arrayo,@row;
     }
```

```
# fragmantasyon için bekleyen paket sayısını bul
for $i (0 .. $#arrayo){
      if ($arrayo[$i] > $arrayo[$i-1]){
            $count++;
            }
      $count = $count % 3;
      }

# Problem var, aksiyon için $count değerini kontrol

if($count == 1){
      #check multicast/broadcast fonksiyonunu çağır.
      $dbislem1->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";
      my $output = `./gw1bandwidthcontrol.pl`;
      }

if($count == 2){
      #check application mtu()fonksiyonunu çağır.
      $a=03;
      set_value($a);

      $dbislem1->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";

      $dbislem2->execute() or die "Couldn't execute
      statement: $DBI::errstr; stopped";
      }

$sth->finish();
$veritabanimh->disconnect();
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların arayüzlerinden gelen ve ya giden trafik miktarı bilgisini kontrol eder. Sorun mevcutsa ilgili aksiyon sürecini veya işlemini işletir.

```
sub set_value{

my ($p1) = @_;
my $sorguhedef = 'localhost';
my %snmpparms;
$snmpparms{DestHost} = inet_ntoa(inet_aton($sorguhedef));
$snmpparms{UseSprintValue} = 1;
my $oturum = new SNMP::Session(%snmpparms);
my $ilgilimib = 'netSnmpExampleInteger';
my $instance = '0';
my $value = $p1;
my $yenisorgu = new
SNMP::Varbind([$ilgilimib,$instance,$value]);
$oturum->set($yenisorgu);
}
```

```
my $a;
my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI->connect("DBI:mysql:database
=$veritabanim;host=$sorgulanan;port=$myport",'root','admin'
) or die "Connection Error: $DBI::errstr\n";

my $sth = $veritabanimh->prepare( "SELECT result FROM posts
WHERE query='interface traffic'" );

my $sorgum 1 = "INSERT INTO posts (query,result) VALUES
('alarm','link saturation');";

my $dbislem1 = $veritabanimh->prepare($sorgum 1);
$sth->execute();
my @row;
my @arrayo;
my $count=0;
my $actioncount=0;
my $intSpeed = 1;

while ( @row = $sth->fetchrow_array ) {
     push @arrayo,@row;
     }

for my $i (0 .. $#arrayo){

     my ($inUsage, $outUsage) = split /[ ]/, $arrayo[$i];
     my ($inUsage1, $outUsage1) = split /[ ]/, $arrayo[$i-
     1];

     my $inUsageNew = int($inUsage);
     my $inUsage1New = int($inUsage1);
     my $outUsageNew = int($outUsage);
     my $outUsage1New = int($outUsage1);
     $inUsageNew = $inUsageNew - $inUsage1New;
     $outUsageNew = $outUsageNew - $outUsage1New;
     my $intUseIN =  (($inUsageNew /450)/$intSpeed)*100;
     my $intUseOUT =  (($outUsageNew /450)/$intSpeed)*100;

     if (($intUseIN > 50)||($intUseOUT>50)){
          $count++;
          }

     $count = $count % 4;
     }

# Problem var, aksiyon için $count değerini kontrol

if($count == 1){
     #check_broadcast()fonksiyonunu çağır.
```

```
        $dbislem1->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";
        my $output = `./gw1mcast-bcast-control.pl`;
        }

if($count == 2){
        #check_ddos()fonksiyonunu çağır.
        $dbislem1->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";
        my $output = `./gw1bandwidthcontrol.pl`;
        }

if($count == 3){
        #change_routing()fonksiyonunu çağır.
        $dbislem1->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";
        }

$sth->finish();
$veritabanimh->disconnect();
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların açık kalma süresini kontrol eder. Sorun mevcutsa ilgili aksiyon sürecini veya işlemini işletir.

```
sub set_value{
my ($p1) = @_;
my $sorguhedef = 'localhost';
my %snmpparms;
$snmpparms{DestHost} = inet_ntoa(inet_aton($sorguhedef));
$snmpparms{UseSprintValue} = 1;
my $oturum = new SNMP::Session(%snmpparms);
my $ilgilimib = 'netSnmpExampleInteger';
my $instance = '0';
my $value = $p1;
my $yenisorgu = new
SNMP::Varbind([$ilgilimib,$instance,$value]);
$oturum->set($yenisorgu);
}

my $a;
my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI->connect("DBI:mysql:database
=$veritabanim;host=$sorgulanan;port=$myport",'root','admin'
) or die "Connection Error: $DBI::errstr\n";

my $sth = $veritabanimh->prepare( "SELECT result FROM posts
WHERE query='uptime'" );
```

```
my $sorgum 1 = "INSERT INTO posts (query,result) VALUES
('alarm','device reboot');";

my $sorgum 2 = "INSERT INTO posts (query,result) VALUES
('action','mark as physical');";

my $dbislem2 = $veritabanimh->prepare($sorgum 2);
my $dbislem1 = $veritabanimh->prepare($sorgum 1);
$sth->execute();

my @row;
my @arrayo;
my $i=0;
my $count=0;
my $firstaction=0;
my $secondaction=0;

while ( @row = $sth->fetchrow_array ) {
     push @arrayo,@row;
     }

# reboot check by splitting and comparing array variables

for $i (0 .. $#arrayo){
     my ($gun, $saat, $dakika, $saniye,$salise) = split
     /[:.]/, $arrayo[$i];

     my ($gunx, $saatx, $dakikax, $saniyex,$salisex) =
     split /[:.]/, $arrayo[$i - 1];

     if ($gun < $gunx){
          $count++;
          }

     elsif(($gun == $gunx) && ($saat < $saatx)){
          $count++;
          }

     elsif(($gun == $gunx) && ($saat == $saatx)&&($dakika
     < $dakikax)){
          $count++;
          }

     elsif(($gun == $gunx) && ($saat == $saatx)&& ($dakika
     == $dakikax)&& ($saniye < $saniyex)){
          $count++;
          }

     elsif(($gun == $gunx) && ($saat == $saatx)&& ($dakika
     == $dakikax)&&($saniye == saniyex)&&
     ($salise<$salisex)){
          $count++;
          }
```

```
        $count = $count % 4;
        }
if($count == 1){
        #check_temperature()fonksiyonunu çağır.

        $dbislem1->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";
        }

if($count == 2){
        #firmwarecheck()fonksiyonunu çağır.
        $a=5;
        set_value($a);

        $dbislem1->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";

        my $output = `./gw1firmwarecontrol.pl`;
        }

if($count == 3){
        #physicalproblem()fonksiyonunu çağır.

        $dbislem1->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";

        $dbislem2->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";
        }
}

$sth->finish();
$veritabanimh->disconnect();
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların yönlendirme tablolarında bulunan bir sebepten ötürü yönlendirme problemine uğrayan paket olup olmadığını kontrol eder. Sorun mevcutsa ilgili aksiyon sürecini veya işlemini işletir.

```
sub set_value{
my ($p1) = @_;
my $sorguhedef = 'localhost';
my %snmpparms;
$snmpparms{DestHost} = inet_ntoa(inet_aton($sorguhedef));
$snmpparms{UseSprintValue} = 1;
my $oturum = new SNMP::Session(%snmpparms);
my $ilgilimib = 'netSnmpExampleInteger';
my $instance = '0';
my $value = $p1;
my $yenisorgu = new
SNMP::Varbind([$ilgilimib,$instance,$value]);
```

```
$oturum->set($yenisorgu);
}

my $a;
my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';

my $veritabanimh = DBI->connect("DBI:mysql:database
=$veritabanim;host=$sorgulanan;port=$myport",'root','admin'
) or die "Connection Error: $DBI::errstr\n";

my $sth = $veritabanimh->prepare( "SELECT result FROM posts
WHERE query='route discards'" );

my $sorgum 1 = "INSERT INTO posts (query,result) VALUES
('alarm','route error');";

my $sorgum 2 = "INSERT INTO posts (query,result) VALUES
('action','clear ip route table');";

my $dbislem2 = $veritabanimh->prepare($sorgum 2);

my $sorgum 3 = "INSERT INTO posts (query,result) VALUES
('action','increase buffer');";

my $dbislem3 = $veritabanimh->prepare($sorgum 3);
my $dbislem1 = $veritabanimh->prepare($sorgum 1);
$sth->execute();

my @row;
my @arrayo;
my $i=0;
my $count=0;
my $firstaction=0;
my $secondaction=0;

while ( @row = $sth->fetchrow_array ) {
     push @arrayo,@row;
     }

for $i (0 .. $#arrayo){
     if ($arrayo[$i] > $arrayo[$i-1]){
          $count++;
          }
     }

$count = $count % 4;

if($count == 1){

     $dbislem1->execute() or die "Couldn't execute
     statement: $DBI::errstr; stopped";
```

```
        $a=1;
        set_value($a);

        $dbislem2->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";
        }

if($count == 2){
        #increaser buffer()fonksiyonunu ÇaĞIr
        $a=8;
        set_value($a);

        $dbislem1->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";

        $dbislem3->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";
        }

if($count == 3){
        #change routing()fonksiyonunu ÇaĞIr
        $dbislem1->execute() or die "Couldn't execute
        statement: $DBI::errstr; stopped";
        }

$sth->finish();
$veritabanimh->disconnect();
```

Aşağıdaki PERL script, 6LoWPAN şebekesindeki cihazların firmware bilgisini kontrol eder. Sorun mevcutsa ilgili aksiyon sürecini veya işlemini işletir.

```
sub set_value{
my ($p1) = @_;
my $sorguhedef = 'localhost';
my %snmpparms;
$snmpparms{DestHost} = inet_ntoa(inet_aton($sorguhedef));
$snmpparms{UseSprintValue} = 1;
my $oturum = new SNMP::Session(%snmpparms);
my $ilgilimib = 'netSnmpExampleInteger';
my $instance = '0';
my $value = $p1;
my $yenisorgu = new
SNMP::Varbind([$ilgilimib,$instance,$value]);
$oturum->set($yenisorgu);
}

my $veritabanim = 'ali';
my $sorgulanan = 'localhost';
my $myport = '3306';
```

```
my $veritabanimh = DBI->connect("DBI:mysql:database
=$veritabanim;host=$sorgulanan;port=$myport",'root','admin'
) or die "Connection Error: $DBI::errstr\n";

my $sth = $veritabanimh->prepare( "SELECT result FROM posts
WHERE query='firmware'" );

my $sorgum 1 = "INSERT INTO posts (query,result) VALUES
('alarm','outdated firmware');";

my $sorgum 2 = "INSERT INTO posts (query,result) VALUES
('action','update firmware');";

my $dbislem2 = $veritabanimh->prepare($sorgum 2);
my $dbislem1 = $veritabanimh->prepare($sorgum 1);
$sth->execute();

my @row;
my $count=0;
my @arrayo;

while ( @row = $sth->fetchrow_array ) {
     push @arrayo,@row;
     }

# reboot olup olmadığını kontrol et

my $i = 0;

if( @arrayo[$#arrayo] ne '3.16.39-1+deb8u2'){
     $dbislem1->execute() or die "Couldn't execute
     statement: $DBI::errstr; stopped";
     # updatefirmware() fonksiyonunu çağır.
     my $a=05;
     set_value($a);
     $dbislem2->execute() or die "Couldn't execute
     statement: $DBI::errstr; stopped";
     }

$sth->finish();
$veritabanimh->disconnect();
```

Aşağıdaki PERL script, 6LoWPAN şebekesinde broadcast ve ya multicast yayın olup olmadığını kontrol eder. Sorun mevcutsa ilgili aksiyon sürecini veya işlemini işletir.

```
my $sorguhedef = 'localhost';
my $ilgilimib  = '.1.3.6.1.2.1.31.1.1.1.8.1';
my $mib1  = '.1.3.6.1.2.1.31.1.1.1.12.1';
my $mib2 = '.1.3.6.1.2.1.31.1.1.1.9.1';
my $mib3 = '.1.3.6.1.2.1.31.1.1.1.13.1';
my $oturum;
```

```
my $sorgugetir;
my $sorgugetir1;
my $sorgugetir2;
my $sorgugetir3;
my $yenisorgu;
my $yenisorgu1;
my $yenisorgu2;
my $yenisorgu3;

&SNMP::initMib();
my %snmpparms;
$snmpparms{DestHost} = inet_ntoa(inet_aton($sorguhedef));
$snmpparms{UseSprintValue} = '1';
$oturum = new SNMP::Session(%snmpparms);
$yenisorgu = new SNMP::Varbind([$ilgilimib]);
$yenisorgu1 = new SNMP::Varbind([$mib1]);
$yenisorgu2 = new SNMP::Varbind([$mib2]);
$yenisorgu3 = new SNMP::Varbind([$mib3]);
$sorgugetir = $oturum->get($yenisorgu);
$sorgugetir1 = $oturum->get($yenisorgu1);
$sorgugetir2= $oturum->get($yenisorgu2);
$sorgugetir3= $oturum->get($yenisorgu2);

if ($oturum->{ErrorNum}) {
      die "Got $oturum->{ErrorStr} querying $sorguhedef for
      $mib.\n";
      }

$sorgugetir=1;

if(($sorgugetir>0)||($sorgugetir1>0)||($sorgugetir2>0)||($s
orgugetir3>0)){
      my $output = `./gw1bandwidthcontrol.pl`;
      }
```

Önceden belirtilen tüm scriptler yönetim sistemi içerisinde çalışacak olan PERL scriptlerdir. Bunlar haricinde sensör cihazların içerisinde çalışması gereken Shell scriptleri bulunmaktadır. Bu shell scriptlerin herbiri bir adet görevi yerine getirmektedir. Ancak Shell scriptlerin çalışabilmesi için cihaz içerisinde bulunan ve cihazın çalışması ile brlikte devreye giren bir temel PERL script bulunur. PERL script daha önceden tanımlanan snmplisten.txt dosyasındaki değeri takip etmektedir. Ayrıca bloke edilecek IP adresinin bulunduğu bir snmpBlockIP.txt text dosyası bulunmaktadır ve snmpBlockIP.txt dosyasında bulunan IP adresi bloke edilir. Aşağıda ilgili temel PERL script bulunmaktadır.

```
use warnings;
use strict;
```

```
use IPC::System::Simple qw(system capture);
use autodie;
use Cache::FileCache;
use Cache::MemoryCache;
use Path::Tiny;

sub ClearTxt{
my $filename = '/home/yekta/snmplisten.txt';

open(my $fh, '>', $filename) or die "Could not open file
'$filename' $!";

print $fh "0";
close $fh;
}

while(){
      my $dir = path("/home/yekta");
      my $file = $dir->child("snmplisten.txt");
      my $content = $file->slurp_utf8();
      my $file_handle = $file->openr_utf8();
      my $line = $file_handle->getline();

      if($line == 0){
            }

      if($line == 1){
            ClearTxt();
            system("/bin/bash/home/yekta/SnmpTasks/clearrout
            e.sh");
            }

      if($line == 2){
            ClearTxt();
            system("/bin/bash/home/yekta/SnmpTasks/blockip.s
            h");
            }

      if($line == 3){
            ClearTxt();
            system("/bin/bash/home/yekta/SnmpTasks/changemtu
            .sh");
            }

      if($line == 4){
            ClearTxt();
            system("/bin/bash/home/yekta/SnmpTasks/cpulimit.
            sh");
            }

      if($line == 5){
            ClearTxt();
```

```
      system("/bin/bash/home/yekta/SnmpTasks/firmwareu
      pdate.sh");
      }

if($line == 6){
      ClearTxt();
      my $filenames = '/proc/sys/vm/drop_caches';
      open(my $fhs, '>', $filenames) or die "Could not
      open file '$filenames' $!";

      print $fhs "1";
      close $fhs;
      }

if($line == 7){
      ClearTxt();
      system("/bin/bash/home/yekta/SnmpTasks/hwclockse
      t.sh");
      }

if($line == 8){
      ClearTxt();
      my $dirx = path("/proc/sys/net/core");
      my $filex = $dirx->child("rmem_max");
      my $contentx = $filex->slurp_utf8();
      my $handle_filex=$filex->openr_utf8();
      my $linex=$handle_filex->getline();
      $linex = $linex * 2;

      $dirx = path("/proc/sys/net/core");
      $filex = $dirx->child("rmem_max");
      my  $fh = $filex->openw_utf8();
      $fh->print($linex . "\n");
      close $fh;

      $filex = $dir->child("rmem_default");
      $fh = $filex->openw_utf8();
      $fh->print($linex . "\n");
      close $fh;
      }

if($line == 9){
      ClearTxt();
      system("/bin/bash/home/yekta/SnmpTasks//Scripts/
      testo.sh");
      }

if($line == 10){
      ClearTxt();
      system("/bin/bash/home/yekta/SnmpTasks/killcpuus
      e.sh");
      }
```

```
if($line == 11){
      ClearTxt();
      system("/bin/bash/home/yekta/SnmpTasks/killmemus
      e.sh");
      }

if($line == 12){
      ClearTxt();
      system("/bin/bash/home/yekta/SnmpTasks/killtcppr
      ocess.sh");
      }

if($line == 13){
      ClearTxt();
      system("/bin/bash/home/yekta/SnmpTasks/ntprestar
      t.sh");
      }

if($line == 14){
      ClearTxt();
      system("/bin/bash/home/yekta/SnmpTasks/ntpupdate
      .sh");
      }

if($line == 15){
      ClearTxt();
      system("/usr/bin/xterm reboot");
      }

if($line == 16){
      ClearTxt();
      system("/bin/bash/home/yekta/SnmpTasks/renicepro
      sses.sh");
      }

if($line == 17){
      ClearTxt();
      system("/bin/bash/home/yekta/SnmpTasks/zombieki
      l.sh");
      }

if($line == 18){
      ClearTxt();
      system("/bin/bash/home/yekta/SnmpTasks/restartif
      .sh");
      }

if($line == 19){
      ClearTxt();
      system("/bin/bash/home/yekta/SnmpTasks/changecha
      nnel.sh");
      }
```

```
      if($line == 20){
            ClearTxt();
            system("/bin/bash/home/yekta/SnmpTasks/increasep
            ower.sh");
            }
}
```

Aşağıdaki Shell script, yönlendirme tablosunu yenileme görevini işletir.

```
echo 1 > /proc/sys/net/ipv4/route/flush
```

Aşağıdaki Shell script, IP bloke etme görevini işletir.

```
BLOCKDB=/home/yekta/snmplistenblkip.txt
IPS=$(grep -Ev "^#" $BLOCKDB)
iptables -A INPUT -s $IPS -j DROP
iptables -A OUTPUT -d $IPS -j DROP
```

Aşağıdaki Shell script, MTU değerini değiştirme görevini işletir.

```
ifconfig eth0 mtu 1600 up
ifconfig wlan0 mtu 1600 up
sleep 600
ifconfig eth0 mtu 1500 up
ifconfig wlan0 mtu 1500 up
```

Aşağıdaki Shell script, işlemci içerisindeki bir işlemin işlemci kullanma yeteneğinin kısıtlanması görevini işletir. Bunun için öncelikle cpulimit programının sensör cihazına önceden yüklenmiş olması gerekmektedir.

```
pid=`ps aux | sort -nrk 3,3 | head -n1 |  awk '{print $2}'`
cpulimit -p $pid -l 50 -b
```

Aşağıdaki Shell script, firmware bilgisinin güncellenmesi görevini işletir.

```
myupdates=`dirname $0`/`basename $0`

if [ `whoami` != 'root' ]; then
        sudo $myupdates $1
        exit 1
else
        apt-get update
        apt-get upgrade -y --fix-missing
        apt-get dist-upgrade -y --fix-missing
        sync
        apt-get autoremove -y
        sync
        rpi-update
        sync
fi
```

```
if [ "$1" = 'reboot' ]; then
        sync
        reboot
        exit 1
fi
if [ "$1" = 'shutdown' ]; then
        sync
        shutdown -h -P now
        exit 1
fi
```

Aşağıdaki PERL script, cache içerisindeki bilginin temizlenmesi görevini işletir.

```
while(){
     my $dir= path("/home/yekta");
     my $file= $dir->child("snmplisten.txt");
     my $contect = $file->slurp_utf8();
     my $handle_file=$file->openr_utf8();
     my $line=$handle_file->getline();

     if($line == 6){
          my $filename = '/proc/sys/vm/drop_caches';

          open(my $fh, '>', $filename) or die "Could not
          open file '$filename' $!";

     print $fh "1";
          close $fh;
          $filename = '/home/yekta/snmplisten.txt';

          open($fh, '>', $filename) or die "Could not open
          file '$filename' $!";

          print $fh "0";
          close $fh;
          }
}
```

Aşağıdaki PERL script, cihaz içerisindeki buffer büyüklüğüğn artırılması görevini işletir.

```
while(){
     my $dir= path("/home/yekta");
     my $file= $dir->child("snmplisten.txt");
     my $content = $file->slurp_utf8();
     my $handle_file=$file->openr_utf8();
     my $line=$handle_file->getline();

     if($line == 8){

          $dir = path("/proc/sys/net/core");
```

```
                $file = $dir->child("rmem_max");
                $content = $file->slurp_utf8();
                $handle_file=$file->openr_utf8();
                $line=$handle_file->getline();
                $line = $line * 2;
                $dir = path("/proc/sys/net/core");
                $file = $dir->child("rmem_max");

                my  $fh = $file->openw_utf8();
                $fh->print($line . "\n");
                close $fh;

                $file = $dir->child("rmem_default");
                $fh = $file->openw_utf8();
                $fh->print($line . "\n");
                close $fh;

                my $filename1 = '/home/yekta/snmplisten.txt';

                open(my $fh1, '>', $filename1) or die "Could not
                open file '$filename1' $!";

                print $fh1 "0";
                close $fh1;
                }
}
```

Aşağıdaki Shell script, işlemciyi yüksek kullanan işlemin durdurulması görevini işletir.

```
pid=`ps aux | sort -nrk 3,3 | head -n1 |  awk '{print $2}'`
kill $pid
sleep 15
kill -9 $pid
```

Aşağıdaki Shell script, hafızayı yüksek kullanan işlemin durdurulması görevini işletir.

```
pid=`ps -A --sort -rss --format pid --no-headers | head -
n1`
kill $pid
sleep 15
kill -9 $pid
```

Aşağıdaki Shell script, zaman bilgisini kullanan işlemin güncellenmesi görevini işletir. Bu amaçla ilgili cihaza bir NTP programının yüklenmiş olması gerekmektedir.

```
ntpdate -u 0.us.pool.ntp.org
```

Aşağıdaki Shell script, zaman bilgisini kullanan işlemin yeniden başlatılması görevini işletir.

```
service ntp stop
service ntp start
service ntp restart
```

Aşağıdaki Shell script, sensör cihazının tamamen yeniden başlatılması görevini işletir.

```
ifconfig eth0 down
ifconfig eth0 up
```

Aşağıdaki Shell script, işlemciyi yüksek kullanan işlemin öncelik sırasının düşürülmesi görevini işletir.

```
pid=`ps aux | sort -nrk 3,3 | head -n1 |  awk '{print $2}'`
renice -n 19 -p $pid
```

Aşağıdaki Shell script, takılı durumda bulunan ve görevine devam edemeyen işlemlerin kapatılması görevini işletir.

```
pid='ps axu | awk '"[Zz] ~ $8 { system(sprintf("kill -HUP
%d", $2)); }''
kill pid
kill -HUP $(ps -A -ostat,ppid | grep -e '[zZ]'| awk '{
print $2 }')
```

EK-2

Iftop Geliştirmeleri

Açık kaynak kodlu olan Iftop programı trafik bilgisi toplamak için kullanılmıştır. 6LoWPAN uç cihazlar içerisinde bir metin dosyası oluşturulmuştur. Iftop üzerindeki eklemeler ile birlikte, programın toplandığı 6LoWPAN trafik bilgilerinin metin dosyasına aktarılması sağlanmıştır. Ui.c dosyası içerisindeki 2 fonksiyonda değişiklik yapılmış, böylece iftop programının pcap kütüphanesi kullanarak topladığı trafik bilgileri, pktstat.txt metin dosyasına aktarımı sağlanmıştır.

Pktstat.txt dosyası sonrasında net-snmp programının içerisine oluşturulan MIB dosyası sayesinde dâhil edilmiştir. Bu şekilde sıkıştırılmış SNMPv3 kullanarak, 6LoWPAN sensör cihazlarındaki gelen ve giden trafik bilgilerinin önerilen yönetim sistemine aktarımı sağlanmıştır. İlk değişiklik void draw_line_total() fonksiyonunun 23 ve 24. satırlarının arasına aşağıdaki kodlar eklenerek yapılmıştır. Bu kodlar ile trafik değerleri pktstat.txt metin dosyasına aktarılır.

```
FILE *fp = fopen("/home/yekta/pktstat.txt","a");
fputs(buf,fp);
fclose(fp);
```

void ui_print() fonksiyonunun 17 ile 18. satırlarının arasına aşağıdaki kodlar eklenmiştir, amaç dosyanın içeriğini her güncelleme öncesinde silmektir.

```
FILE *fp = fopen("/home/yekta/pktstat.txt","w");
fclose(fp);
```

void ui_print() fonksiyonunun 63 ile 64. satırlarının arasına aşağıdaki kodlar eklenmiştir. Aşağıdaki kodlar trafik bilgilerinin anlamlı şekilde kaydedilmesi için gerekir.

```
FILE *fp = fopen("/home/yekta/pktstat.txt","a");
fputs("\n",fp);
fputs(host1,fp);
fputs(host2,fp);
fclose(fp);
```

EK-3

Net-SNMP Konfigurasyonu

Net-snmp programının Debian LINUX üzerinde çalıştırılabilmesi için birtakım konfigürasyonların yapılması gerekmektedir. Program işletim sistemine yüklendikten sonra, bu kısımda beliritilen üç adet dosyanın LINUX işletim sisteminde bulunan /local/lib/share/snmp klasörü altına kopyalanması gerekmektedir. Program açıldığında, özelleştirilmiş konfigürasyon bilgilerini olup olmadığını kontrol etmek için snmp.conf dosyasını inceler. SNMPv3'ün çalışabilmesi için snmp.conf dosyasına aşağıdaki satırlar eklenmelidir.

```
mibs +ALL
defSecurityName newuser
defContext ""
defAuthType MD5
defPrivType DES
defSecurityLevel authPriv
defAuthPassphrase my_password
defPrivPassphrase my_password
defversion 3
```

Net-snmp programının ilk çalışması sırasında kontrol ettiği diğer bir dosya ise snmpd.conf dosyasıdır. Sıkıştırılmış SNMPv3'ün düzgğn çalışabilmesi için bu dosyaya aşağıdaki satırlar eklenmelidir.

```
agentAddress udp:161
rwcommunity public  localhost
rouser v3admin noauth
createUser v3admin MD5 adminpass
rwuser myuser
rwuser newuser
```

Son olarak net-snmp programı içerisindeki trap istemcisinin doğru çalışabilmesi için snmptrapd.conf dosyası aşağıdaki şekilde olmalıdır.

```
disableAuthorization yes
authCommunity log,execute,net public
authUser log,execute,net myuser
authUser log,execute,net v3admin
createUser -e 0x0102030405 v3admin MD5 adminpass
```

KAYNAKÇA

[1] I. F. Akyildiz ve M. C. Vuran, "Wireless Sensor Networks", Wiley, 2010.

[2] C. Yibo, K. Hou, H. Zhou, H. Shi, X. Liu, X. Diao, H. Ding, J. Li ve C.Vaulx, "6LoWPAN Stacks: A Survey", *International Conference on Wireless Communications, Networking and Mobile Computing*, 2011.

[3] G. Montenegro, N. Kushalnagar, J. Hui ve D. Culler, " Transmission of IPv6 Packets over IEEE 802.15.4 Networks", IETF RFC 4944, 2007.

[4] N. Kushalnagar, G. Montenegro ve C. Schumacher, " IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement and Goals", IETF RFC 4919, 2007.

[5] Z. Sheng, C. Mahapatra, C. Zhu ve V. C. M. Leung, "Recent Advances in Industrial Wireless Sensor Networks Toward Efficient Management in IoT", *IEEE Access*, sayı 3, 2015, sayfa 622 – 637.

[6] M. Ersue, D. Romascanu, J. Schoenwaelder ve A. Sehgal, "Management of Networks with Constrained Devices: Use Cases", IETF RFC 7548, 2015.

[7] M. Ersue, D. Romascanu, J. Schoenwaelder ve U. Herberg, "Management of Networks with Constrained Devices: Problem Statement and Requirements", IETF RFC 7547, 2015.

[8] H. Sundmaeker ve P. Guillemin, "Vision and Challenges for Realizing the Internet of Things", *European Commission - Information Society*, 2010.

[9] J. Gubbia, R. Buyyab, S. Marusic ve M. Palaniswami, "Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions", *Elsevier Future Generation Computer Systems*, sayı 29, no. 7, 2013, sayfa 1645-1660.

[10] U. Blumenthal ve B. Wijnen, "User-based Security Model (USM) for Version 3 of the Simple Network Management Protocol (SNMPv3)", IETF RFC 3414, 2012.

[11] M. Díaz, C. Martín ve B. Rubio, "State-of-the-art, Challenges and Open Issues in the Integration of Internet of Things and Cloud Computing", *Elsevier Journal of Network and Computer Applications,* sayı 67, 2016, sayfa 99-117.

[12] L. B. Ruiz, J. M. Nogueira ve A. A. F. Loureiro, "MANNA: A Management Architecture for Wireless Sensor Networks", *IEEE Communications Magazine,* sayı 41, no. 2, 2003, sayfa 116 – 125.

[13] I. Lee ve K. Lee," The Internet of Things (IoT): Applications, Investments and Challenges for Enterprises", *Elsevier Business Horizons,* sayı 58, no. 4, 2015, sayfa 431-440.

[14] J. A. Stankovic, "Research Directions for the Internet of Things", *IEEE Internet of Things Journal,* sayı 1, no. 1, 2014, sayfa 3 – 9.

[15] A. Meddeb , "Internet of Things Standards: Who Stands out from the Crowd?", *IEEE Communications Magazine,* sayı 54, no. 7, 2016, sayfa 40 – 47.

[16] C. Perera, C. H. Liu, S. Jayawardena ve M. Chen, "A Survey on Internet of Things From Industrial Market Perspective", *IEEE Access,* sayı 2, 2015, sayfa 1660 – 1679.

[17] J. Gomez ve A.T. Campbell, "A Case for Variable-range Transmission Power Control in Wireless Multihop Networks", *IEEE Infocom,* 2004

[18] E.-S. Jung ve N.H. Vaidya, "A Power Control MAC Protocol for Ad-hoc Networks", *Eighth Annual International Conference on Mobile Computing and Networking (MobiCom),* 2002.

[19] P. Karn, "A New Channel Access Protocol for Packet Radio", *American Radio Relay League Ninth Computer Networking Conference,* 1990.

[20] OpenLabs project. https://goo.gl/keokoY, son erişim: 30-October-2017, 2012.

[21] Raspberry PI Foundation. https://goo.gl/Hj0MZ9, son erişim: 29 Haziran 2018.

[22] E. Upton ve G. Halfacree," Raspberry Pi User Guide," Wiley, 2012

[23] Net-Snmp. https://goo.gl/QFDFK, son erişim: 20 Haziran 2018.

[24] H. Mukhtar, K. Kang-Myo, S. A. Chaudhry, A. H. Akbar, K. Ki-Hyung ve S. Yoo, "LNMP-Management Architecture for IPv6 based Low Power Wireless Personal Area Networks (6LoWPAN)", *IEEE Network Operations and Management Symposium,* 2008.

[25] H. Lindholm-Ventola ve B. Silverajan, "CoAP-SNMP Interworking in IoT Scenarios", *Tampere University of Technology Department of Pervasive Computing Report*, 2014.

[26] Z. Sheng, H. Wang, C. Yin, X. Hu, S. Yang ve V. C. M. Leung, "Lightweight Management of Resource-constrained Sensor Devices in Internet of Things ", *IEEE Internet of Things Journal*, sayı 2, no. 5, 2015, sayfa 402 – 411.

[27] M. Thoma, T. Braun, C. Magerkurth ve A. Antonescu, "Managing Things and Services with Semantics: A Survey", *Network Operations and Management Symposium (NOMS)*, 2014.

[28] A Sehgal, V. Perelman, S. Kuryla ve J. Schonwalder, "Management of Resource Constrained Devices in the Internet of Things", *IEEE Communications Magazine*, sayı 50, no. 12, 2012, sayfa 144 – 149.

[29] H. Lamaazi, N. Benamar, J. J. Antonio, L. Ladid ve D. El Ouadghiri, "Challenges of the Internet of Things: IPv6 and Network Management", *International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, 2014.

[30] D. Karaman, N. Gozuacik, M. O. Alagoz, H. Ilhan, U. Cagal ve O. Yavuz, "Managing 6LoWPAN Sensors with CoAP on Internet", *Signal Processing and Communications Applications Conference (SIU)*, 2015.

[31] Z. Shelby, K. Hartke ve C. Bormann, "The Constrained Application Protocol (CoAP)", IETF RFC 7252, 2014.

[32] S. Zehl ve T. Scheffler, "Secure Access and Management of Smart Objects with SNMPv3", *IEEE International Conference on Consumer Electronics (ICCE)*, 2013.

[33] J. Shanthini ve S. Vijayakumar, "Modified Simple Network Management Protocol for 6LoWPAN", *The International Conference on Modeling, Optimization and Computing*, 2012.

[34] H. Choi, N. Kim ve H. Cha, "6LoWPAN-SNMP: Simple Network Management Protocol for 6LoWPAN", *11th IEEE International Conference on High Performance Computing and Communications*, 2009.

[35] C. Bormann, “6LoWPAN-GHC: Generic Header Compression for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)”, IETF RFC 7400, 2014.

[36] J. Schoenwaelder, H. Mukhtar, S. Joo ve K. Kim, “SNMP Optimizations for Constrained Devices”, IETF Internet-Draft; 2010.

[37] M. Turon, ” MOTE-VIEW: A Sensor Network Monitoring and Management Tool", *The Second IEEE Workshop on Embedded Networked Sensors,* 2005.

[38] T. D. Le, W. Hu, S. Jha ve P. Corke, “Design and Implementation of a Policy-based Management System for Data Reliability in Wireless Sensor Networks”, *33rd IEEE Conference on Local Computer Networks (LCN),* 2008.

[39] W. Zhang ve H. Xu,” A Policy Based Wireless Sensor Network Management Architecture”, *Third International Conference on Intelligent Networks and Intelligent Systems,* 2010.

[40] T. C. Minh, B. Bellalta ve M. Oliver, “DISON: A Self-organizing Network Management Framework for Wireless Sensor Network”, *International Conference on Ad Hoc Networks,* 2012.

[41] B. Al-Shammari, N. Al-Aboody ve H. S.Al-Raweshidy, “IoT Traffic Management and Integration in the QoS Supported Network”, *Internet of Things Journal*, sayı 5, no. 1, 2018, sayfa 352-370.

[42] L. Hou, S. Zhao, Xing Li, P. Chatzimisios ve K. Zheng, “Design and Implementation of Application Programming Interface for Internet of Things Cloud”, *International Journal of Network Management*, sayı 27, 2017, sayfa 20-27.

[43] Y. Turk ve A. Akman, “A Management Model for Low Powered Wireless Personal Area Networks”, *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM),* 2018.

[44] Z. Shelby ve C. Bormann, “6LoWPAN: The Wireless Embedded Internet”, Wiley, 2009.

[45] Ed J. Hui ve P. Thubert, ” Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks”, IETF RFC 6282, 2011.

[46] J. Case, R. Mundy, D. Partain ve B. Stewart, “Introduction and Applicability Statements for Internet Standard Management Framework,” RFC 3410, 2002.

[47] K. McCloghrie, D. Perkins ve J. Schoenwaelder, “Structure of Management Information Version 2 (SMIv2),” RFC 2578, 1999.

[48] K. McCloghrie, D. Perkins ve J. Schoenwaelder, “Conformance Statements for SMIv2,” RFC 2580, 1999.

[49] R. Presuhn, J. Case, K. McCloghrie, M. Rose ve S. Waldbusser, “Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP),” RFC 3416, 2002.

[50] K. McCloghrie, D. Perkins ve J. Schoenwaelder, “Textual Conventions for SMIv2, RFC 2579, 1999

[51] J. Schoenwaelder, A. Sehgal,T. Tsou ve C. Zhou, “Definition of Managed Objects for IPv6 over Low-PowerWireless Personal Area Networks (6LoWPANs)”, IETF RFC 7388, 2014.

[52] B. Rockwood, “The Net-SNMP Programming Guide”, Cuddletech, 2004.

[53] B. Latre, P. De Mil, I. Moerman, B. Dhoedt, P. De Meester ve N. Van Dierdonck, “Throughput and Delay Analysis of Unslotted IEEE 802.15.4”, *Journal of Networks*, sayı 1, no. 1, 2006, sayfa 20-28.

[54] Iftop, https://goo.gl/DY4zX, son erişim: 10 Temmuz 2018.

[55] A. Banks ve R. Gupta., “MQTT Version 3.1.1.”, OASIS Standard, 2014.

[56] Eclipse Paho, https://goo.gl/Qex1Kj, son erişim: 15 Haziran 2018.

[57] O.Bergman, “libcoap: C-Implementation of CoAP”, https://libcoap.net/, 2013, son erişim: 15 Haziran 2018.