ELİPTİK EĞRİ ŞİFRELEMENİN KISITLAMALI AĞLARDA İLETİM KATMANINDAKİ DTLS PROTOKOLÜNDE DOĞRULAMA VE ANAHTAR KARŞILAŞTIRMA AŞAMASINDA KULLANILMASI

İbrahim Karataş
181402102

YÜKSEK LİSANS TEZİ
Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Yüksek Lisans Programı
Danışman: Dr. Öğr. Üyesi Selim Bayraklı

İstanbul
T.C. Maltepe Üniversitesi
Fen Bilimleri Enstitüsü
Temmuz 2019

JÜRİ VE ENSTİTÜ ONAYI

İBRAHİM KARATAŞ'ın "Eliptik Eğri Şifrelemenin Kısıtlamalı Ağlardaki İletim Katmanındaki Dtls Protokolünde Doğrulama Ve Anahtar Karşılaştırma Aşamasında Kullanılması" başlıklı tezi 26.07.2019 tarihinde aşağıdaki jüri tarafından değerlendirilerek "Maltepe Üniversitesi Lisansüstü Eğitim ve Öğretim Yönetmeliği" nin ilgili maddeleri uyarınca Bilgisayar Mühendisliği Anabilim Dalı Yüksek Lisans/Doktora tezi oy birliğiyle/~~oy çokluğuyla,~~ başarılı/~~başarısız~~ olarak kabul edilmiştir.

	Unvanı. Adı ve Soyadı	İmza
Üye (Tez Danışmanı)	Dr. Öğr. Üyesi Hüseyin Fehmi Selim BAYRAKLI	.....
Üye	Dr. Öğr. Üyesi Volkan TUNALI	.....
Üye	Dr. Öğr. Üyesi Can EYÜPOĞLU	.....

Prof. Dr. İlter BÜYÜKDIĞAN
Enstitü Müdürü

Yrd. Doç. Dr.
Özgür ÖZKAN

	ŞEKİL ONAY SAYFASI	Doküman No	FR-105
		İlk Yayın Tarihi	20.12.2017
		Revizyon Tarihi	10.12.2018
		Revizyon No	01
		Sayfa	1/2

ŞEKİL ONAY SAYFASI

07/08/2019

FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRLÜĞÜNE,

Aşağıda bilgileri bulunan lisansüstü öğrencinin tezi şekil yönünden tarafımca incelenmiş ve Enstitüye teslim edilmesi uygun bulunmuştur.

Anabilim Dalı Başkanı
Prof.Dr. Ahmet Mesut RAZBONYALI
İmza

ÖĞRENCİ BİLGİLERİ	
ADI SOYADI	İBRAHİM KARATAŞ
ÖĞRENCİ NUMARASI	181402102
ANABİLİM DALI	BİLGİSAYAR MÜHENDİSLİĞİ
PROGRAMI	(x) YÜKSEK LİSANS ( ) DOKTORA ( ) SANATTA YETERLİK
DANIŞMANI	Dr. Öğr. Üyesi Selim BAYRAKLI
TEZ BAŞLIĞI	Eliptik Eğri Şifrelemenin Kısıtlamalı Ağlarda İletim Katmanındaki Dtls Protokolünde Doğrulama ve Anahtar Karşılaştırma Aşamasında Kullanılması
SAVUNMA TARİHİ	26/07/2019
e-posta	ikaratas2515@gmail.com

İç Kapak	☑ Var ☐ Yok
Jüri Onay Sayfası	☑ Var ☐ Yok
Etik İlke ve Kurallara Uyum Beyanı	☑ Var ☐ Yok
İntihal Raporu	☑ Var ☐ Yok
Teşekkür Sayfası	☑ Var ☐ Yok

Hazırlayan İlgili Birim	Kalite Koordinatörü Dr. Öğr. Üyesi Şafak GÜNDÜZ	Kurumsal Yetkili Prof. Dr. Belma AKŞİT

(Doküman No: FR-105; Yayın Tarihi 20.12.2017; Revizyon Tarihi: ; Revizyon No:00)

	ŞEKİL ONAY SAYFASI	Doküman No	FR-105
		İlk Yayın Tarihi	20.12.2017
		Revizyon Tarihi	10.12.2018
		Revizyon No	01
		Sayfa	2/2

Öz (Başlık-Öz-Anahtar Sözcükler)	☑ Var ☐ Yok
Abstract (Title-Abstract-Key Words)	☑ Var ☐ Yok
İçindekiler	☑ Var ☐ Yok
Çizelgeler Listesi	☑ Var ☐ Yok
Şekiller Listesi (varsa)	☐ Şekil yok ☑ Uygundur ☐ Uygun Değildir
Kısaltmalar Listesi	☑ Var ☐ Yok
Tablolar Listesi (varsa)	☑ Tablo yok ☐ Uygundur ☐ Uygun Değildir
Ekler Listesi (varsa)	☑ Ek yok ☐ Uygundur ☐ Uygun Değildir
Özgeçmiş	☑ Var ☐ Yok
Sayfa Genişliği	☑ Uygundur ☐ Uygun Değildir
Yazı Tipi	☑ Uygundur ☐ Uygun Değildir
Referans Kullanımı	☑ Uygundur ☐ Uygun Değildir
Kaynakça Yazımı	☑ Uygundur ☐ Uygun Değildir
Ekler (varsa)	☑ Ek yok ☐ Uygundur ☐ Uygun Değildir

Dr. Öğr. Üyesi Erdal GÜVENOĞLU

İmza

Y. Hazar Akgül

Hazırlayan İlgili Birim	Kalite Koordinatörü Dr. Öğr. Üyesi Şafak GÜNDÜZ	Kurumsal Yetkili Prof. Dr. Belma AKŞİT

(Doküman No: FR-105; Yayın Tarihi 20.12.2017; Revizyon Tarihi: ; Revizyon No:00)

maltepe üniversitesi	ETİK İLKE VE KURALLARA UYUM BEYANI	Doküman No	FR-178
		İlk Yayın Tarihi	01.03.2018
		Revizyon Tarihi	
		Revizyon No	00
		Sayfa	1/1

Revizyon Takip Tablosu

REVİZYON NO	TARİH	AÇIKLAMA
00	01.03.2018	İlk yayın.

ETİK İLKE VE KURALLARA UYUM BEYANI

08/08/2019

Bu tezin bana ait, özgün bir çalışma olduğunu; çalışmamın hazırlık, veri toplama, analiz ve bilgilerin sunumu olmak üzere tüm aşamalarından 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.

İbrahim KARATAŞ

Hazırlayan	Kalite Koordinatörü	Kurumsal Yetkili
İlgili Birim	Dr. Öğr. Üyesi Şafak GÜNDÜZ	Prof. Dr. Belma AKŞİT

(Dokuman No: FR-178; Yayın Tarihi: 01.03.2018; Revizyon Tarihi: ; Revizyon No:00)

İNTİHAL RAPORU

ELİPTİK EĞRİ ŞİFRELEMENİN KISITLAMALI AĞLARDA İLETİM KATMANINDAKİ DTLS PROTOKOLÜNDE DOĞRULAMA VE ANAHTAR KARŞILAŞTIRMA AŞAMASINDA KULLANILMASI

ORIJINALLIK RAPORU

%9	%6	%1	%7
BENZERLIK ENDEKSI	İNTERNET KAYNAKLARI	YAYINLAR	ÖĞRENCI ÖDEVLERI

BIRINCIL KAYNAKLAR

1	Submitted to The Scientific & Technological Research Council of Turkey (TUBITAK) Öğrenci Ödevi	%2
2	sbe.maltepe.edu.tr İnternet Kaynağı	%1
3	www.turkhackteam.org İnternet Kaynağı	%1
4	Submitted to Karadeniz Teknik University Öğrenci Ödevi	%1
5	www2.itu.edu.tr İnternet Kaynağı	<%1
6	www.maltepe.edu.tr İnternet Kaynağı	<%1
7	www.bilgisayarkavramlari.com İnternet Kaynağı	<%1
8	www.eng.usf.edu	

Uygundur
Selim BAYRAKLI
İmza

TEŞEKKÜR

Tez çalışmamın her aşamasında benden desteklerini esirgemeyen değerli hocam Dr. Öğr. Üyesi Selim Bayraklı'ya, çalışmam süresince her türlü destek ve morali veren eşime ve aileme teşekkürlerimi sunarım.

İbrahim Karataş

Haziran 2019

ÖZ

ELİPTİK EĞRİ ŞİFRELEMENİN KISITLAMALI AĞLARDAKİ İLETİM KATMANINDAKİ DTLS PROTOKOLÜNDE DOĞRULAMA VE ANAHTAR KARŞILAŞTIRMA AŞAMASINDA KULLANILMASI

İbrahim Karataş
Yüksek Lisans Tezi
Bilgisayar Mühendisliği Anabilim Dalı
Bilgisayar Mühendisliği Yüksek Lisans Programı
Danışman: Dr. Öğr. Üyesi Selim Bayraklı
Maltepe Üniversitesi Fen Bilimleri Enstitüsü, 2019

Nesnelerin İnterneti (IoT)'nin ortaya çıkışıyla birlikte milyarlarca nesnenin internet aracılığı ile birbirine bağlı olacağı ve yeni neslin interneti olacağına inanılmaktadır. Son yıllarda farklı altyapılarda çok çeşitli uygulamalar geliştirilmiş, özellikle sağlık, lojistik, endüstri, askeri alanda çalışmalar yoğunlaşmıştır. IOT için geliştirilen teknikler hala emekleme aşamasında olduğundan birçok zorlukla karşı karşıyadır. Bunlardan en önemlisi güvenlik problemidir. Bu durum Nesnelerin İnterneti'nin getirdiği birçok kolaylığın yanında bu cihazları önemli bir hedef haline getirmektedir. Çünkü bu cihazlar sürekli olarak birbirleriyle (M2M) ve insanlarla (M2H) iletişim halinde olacaktır. Bu iletişim sırasında insanlara ve çevreye ait pek çok önemli bilgi transfer halinde olacağı için bu iletişimin güvenli şekilde gerçekleşmesi elzemdir. Günümüz güvenlik yöntemlerinin Nesnelerin İnterneti ağlarına entegre edilmesi kısıtlı RAM, ROM oranı, düşük bant genişliği, düşük işlem gücü, düşük enerji beslemesi gibi sebeplerden dolayı mümkün görünmemektedir. IETF tarafından geliştirilen CoAP ve DTLS gibi protokoller buna çözüm getirse de DTLS protokolü simetrik şifreleme üzerine çalışmaktadır ve asimetrik şifreleme kullanılması halinde Sınıf-0 ve Sınıf-1 cihazlar için etkili çalışabilir durumda değildir. Bu çalışma kapsamında güvenliğin sağlanması için yapılan güncel çalışmalardan bahsedilmiş, Eliptik Eğri Kriptolojisi kullanarak verilerin güvenli bir şekilde internet ortamına açılması ve diğer yapılan çalışmalarla karşılaştırılması gerçekleştirilmiştir.

Anahtar Sözcükler: 1.Eliptik Eğri Şifreleme; 2.DTLS; 3.Nesnelerin İnterneti; 4.Kriptoloji.

ABSTRACT

USING ELLIPTIC CURVE CRYPTOGRAPHY FOR AUTHENTİCATION AND KEY COMPARISON OF THE DTLS PROTOCOL ON THE TRANSMISSION LAYER IN RESTRICTED NETWORKS

İbrahim Karataş
Master Thesis
Department of Computer Engineering
Computer Engineering Programme
Thesis Advisor: Asst. Prof. Selim Bayraklı
Maltepe University Graduate School of Science Engineering, 2019

With the emergence of the Internet of Things, it is believed that billions of objects will be interconnected and that the new generation will be the Internet of the next generation. In recent years, a wide range of applications have been developed in different infrastructures, especially in health, logistics, industry and military fields. The techniques developed for IoT are still in its infancy and face many challenges. The most important of these is the security problem. This makes these devices an important target in addition to the many conveniences provided by the Internet of Things. Because these devices will be in constant communication with each other (M2M) and people (M2H). During this communication, it is essential that this communication be carried out safely, since many important information about people and the environment will be transferred. It is not possible to integrate today's security methods into Internet of Things networks due to limited RAM, ROM ratio, low bandwidth, low processing power and low energy supply. Although IETF developed protocols such as CoAP and DTLS, the DTLS protocol works on symmetric encryption and is not effective for Class-0 and Class-1 devices if asymmetric encryption is used. The scope of this study, the current studies to ensure the security are mentioned and the data is securely opened to the internet environment by using Elliptic Curve Cryptology and compared with other studies.

Keywords: 1.Elliptic Curve Cryptography; 2.DTLS; 3.IoT; 4.Cryptology.

İÇİNDEKİLER

JÜRİ VE ENSTİTÜ ONAYI ..... **HATA! YER İŞARETİ TANIMLANMAMIŞ.**

ETİK İLKE VE KURALLARA UYUM BEYANI ..... **HATA! YER İŞARETİ TANIMLANMAMIŞ.**İV

İNTİHAL RAPORU ..... İV

TEŞEKKÜR ..... VİVİ

ÖZ ..... Vİİİİ

ABSTRACT ..... Vİİİ

İÇİNDEKİLER ..... VİİİX

ÇİZELGE LİSTESİ ..... Xİİ

ŞEKİLLER LİSTESİ ..... Xİİİİ

KISALTMALAR ..... XİV

ÖZGEÇMİŞ XVV

1. GİRİŞ 1

2. NESNELERİN İNTERNETİ'NİN YAPISI VE GÜVENLİĞİ 6

2.1. Güvenlik *8*

2.2. Kriptolojik Kavramlar *9*

2.2.1. Simetrik Şifreleme *10*

2.2.2. Asimetrik (Açık Anahtar) Şifreleme *12*

2.2.2.1. RSA Şifreleme 13

2.2.2.2. Diffie Hellman Anahtar Değişim Algoritması 14

2.2.2.3. Dijital İmza Algoritması 15

2.2.2.4. Eliptik Eğri Şifreleme 16

2.2.2.4.1. Eliptik Eğri Dijital İmza Algoritması 17

2.3. Nesnelerin İnternetinin Yapısı *19*

2.2.1. Kısıtlı Uygulama Protokolü (CoAP) *20*

2.2.1.1. Mesaj Katmanı Modeli 22

2.2.1.2. İstek/Yanıt Katman Modeli 23

2.2.1.3. CoAP Mesaj Formatı 25
2.2.2. Veri Birimi İletim Katmanı Güvenliği (DTLS) *26*
2.2.2.1. Yeniden Sıralama 27
2.2.2.2. Mesaj Boyutu 27
2.2.2.3. Paket Kaybı 28
2.2.2.4. DTLS El Sıkışma Protokolü 28
2.2.3. Düşük Güçlü Kablosuz Kişisel Alan Ağlarında IPv6 ..... *32*
2.2.4. Düşük Güçlü ve Kayıplı Ağlar İçin Yönlendirme Protokolü *33*
2.4. Contiki İşletim Sistemi ve Cooja Simülasyon Programı *34*
2.4.1. Contiki İşletim Sisteminin Yüklenmesi 34
2.4.2. Terminal Üzerinden "hello world" Uygulaması 35
2.4.3. Cooja Simülasyonu 36
2.4.3.1. Hello World Simülasyonu 38
2.4.3.2. IPv6-RPL-UDP Örneği 39
2.3.4.3. Algılayıcı Veri Toplama 40
2.3.4.4. Sınır Yönlendirici ile CoAP Protokolü Örneği 40

3. ELİPTİK EĞRİ KULLANARAK DTLS KATMANINDA EL SIKIŞMA AŞAMSINDA KİMLİK DOĞRULAMANIN GERÇEKLEŞTİRİLMESİ 42

3.1. Sistemin Tanıtımı *42*
3.2. Düğüm Seçimi *43*
3.3. DTLS Eliptik Eğri Şifreleme ile İstemci Sunucu Haberleşmesi *43*
3.4. RAM ROM Kullanımı *45*
3.5. Enerji Tüketimi *48*
3.6. Benzetim Yöntemiyle Elde Edilen Değerler *50*

4. SONUÇLAR 53

KAYNAKÇA 55

ÇİZELGELER LİSTESİ

Çizelge 2.1. Kısıtlı cihazların sınıflandırılması ……..……..7

Çizelge 2.2. Simetrik şifrelemenin olumlu ve olumsuz yönleri……..11

Çizelge 2.3. Nesnelerin interneti katman ve protokol yapısı……...20

Çizelge 2.4. İnternet ve IoT ile taşınan paket başlığı karşılaştırması…….….26

Çizelge 2.5. CoAP güvenlik mekanizmaları……..….26

Çizelge 3.1. Sensör düğümlerin hafıza değerleri……...…..43

Çizelge 3.2. Sistemin RAM ve ROM gereksinimi…….….45

Çizelge 3.3. İstemcinin harcadığı ROM değerleri…….…….……46

Çizelge 3.4. İstemcinin harcadığı RAM değerleri…….…….46

Çizelge 3.5. DTLS el sıkışma aşamasında ECC ve simetrik şifreleme modlarında tüketilen hafıza miktarı…..……...…….…….……47

Çizelge 3.6. Wismote simetrik şifreleme modunda hafıza tüketimi……..……...47

Çizelge 3.7. PowerTrace aracı ile 10sn ara ile düğümünden elde edilen veriler……49

Çizelge 3.8. Wismote düğümünün mW cinsinden tükettiği enerji miktarı…….……49

ŞEKİLLER LİSTESİ

Şekil 1.1. Nesnelerin interneti kullanım alanları……..……..…1

Şekil 2.1. Nesnelerin interneti sisteminin yapısı…….……..6

Şekil 2.2. McCumber bilgi güvenliği modeli……..……...8

Şekil 2.3. Güvensiz kanal üzerinden simetrik şifreleme……...10

Şekil 2.4. Şifreleme ve çözümleme işlemi….……11

Şekil 2.5. Asimetrik şifreleme yöntemi…...……..……...13

Şekil 2.6. RSA şifreleme yöntemi…….……...14

Şekil 2.7. Diffie Hellman anahtar değişim algoritması…….15

Şekil 2.8. Dijital imza algoritması……..……..16

Şekil 2.9. Eliptik eğri üzerinde iki noktanın toplanması……..17

Şekil 2.10. Nesnelerin interneti ağ yapısı……..…….….19

Şekil 2.11. CoAP mimarisi…….…….21

Şekil 2.12. CoAP katman yapısı……..…….22

Şekil 2.13. Güvenilir mesaj iletimi….……..23

Şekil 2.14. Güvenilir olmayan mesaj iletimi……23

Şekil 2.15. GET metodunun başarılı ve başarısız yanıt sonuçları……24

Şekil 2.16. Ayrı yanıt ile GET isteği……..……..24

Şekil 2.17. CoAP mesaj yapısı…….……25

Şekil 2.18. DTLS tekrar iletim örneği……..……28

Şekil 2.19. DTLS el sıkışma protokolü.……...31

Şekil 2.20. Güvenli DTLS'in paket düzeni….……...……..32

Şekil 2.21. 6LoWPAN ağlarının yapısı……...……..……...33

Şekil 2.22. Contiki işletim sistemi……..……..……....35

Şekil 2.23. Contiki hello world örneği……..……....35

Şekil 2.24. Contiki standart kod yapısı……36

Şekil 2.25. Cooja simülasyon programı……..……..……....37

Şekil 2.26. Cooja simülatörü ana ekranı…...……..…….38

Şekil 2.27. Cooja hello world simülasoyonu…….....……....37

Şekil 2.28. IPv6-RPL-UDP protokolleri ile düğümler arası iletişim ……..……....39

Şekil 2.29. Düğümlerden veri toplama……..……...40

Şekil 2.30. CoAP protokolü aracılığı ile sınır yönlendiri ve server iletişimi. ……..41

Şekil 3.1. DTLS ile uçtan uca şifreleme…… ……..……42

Şekil 3.2. DTLS el sıkışma aşamasındaki mimari……44

Şekil 3.3. Eliptik eğri ile DTLS el sıkışma aşamasının gerçekleştirilmesi ……..44

Şekil 3.4. DTLS'in ECC ve simetrik şifreleme modlarının kullandığı toplam hafıza miktarının KB cinsinden karşılaştırılması……48

Şekil 3.5. Cooja simülasyon ortamında Powertrace aracı ile güç tüketiminin ölçülmesi..48

Şekil 3.6. Eliptik eğri şifreleme kullanılarak düğümlerin haberleşmesinin Cooja simülasyon programı ile gerçeklenmesi …….....50

Şekil 3.7. Yapılan ECC çalışmalarının SŞ ve RSA ile RAM ve ROM kullanımı açısından karşılaştırılması ……...51

KISALTMALAR

ACK	Kabul Edilebilir	Acknowledgement
AES	Gelişmiş Şifreleme Sistemi	Advanced Encryption System
CoAP	Kısıtlı Uygulama Katmanı	Constrained Application Protocol
CON	Onaylanabilir	Confirmable
DES	Veri Şifreleme Standardı	Data Encryption Standart
DoS	Hizmet Aksatma	Denial of Service
DTLS	Veri Birimi İletim Katmanı Güvenliği	Datagram Transport Layer Security
ECC	Eliptik Eğri Şifreleme	Elliptic Curve Cryptography
ECDH	Eliptik Eğri Diffie Hellman	Elliptic Curve Diffie Hellman
ECDSA	Eliptik Eğri Dijital İmza	Elliptic Curve Digital Signature
HTTP	Üstün Metin Transfer Protokolü	Hyper Text Transfer Protocol
IETF	İnternet Mühendisliği Görev Grubu	Internet Engineering Task Force
IoT	Nesnelerin İnterneti	Internet of Things
IPv6	İnternet Protokolü Sürüm 6	Internet Protocol Version 6
M2H	Makine-İnsan İletişimi	Machine to Human
M2M	Makine-Makine İletişimi	Machine to Machine
PKC	Açık Anahtar Şifreleme	Public Key Cryptography
RPL	Düşük Güç Yönlendirme Protokolü	Routing Protocol for Low Power
RSA	Rivest Shamir Adleman	Rivest Shamir Adleman
SSL	Güvenli Soket Katmanı	Secure Socket Layet
TCP	Taşıma Kontrol Protokolü	Transmission Control Protocol
UDP	Kullanıcı Veribloğu Protokolü	User Datagram Protocol
6LowPAN	Düşük Güçlü Kişisel Alan Ağı	Low Power Wireless Personal Area Network

ÖZGEÇMİŞ

İbrahim Karataş

Bilgisayar Mühendisliği Anabilim Dalı

Eğitim

Derece	*Yıl*	*Üniversite, Enstitü, Anabilim/Anasanat Dalı*
Ls.	2013	Eskişehir Osmangazi Üniversitesi, Fen Edebiyat Fakültesi Matematik-Bilgisayar Bilimleri Anabilim Dalı
Lise	2008	Hatice Bayraktar Anadolu Lisesi

İş/İstihdam

Yıl	*Görev*
2011 - 2012	Web Yazılım - Kitle Bilişim
2013 - 2014	Araştırma Görevlisi - Kastamonu Üniversitesi
2015 -	MEB- Öğretmen

Kişisel Bilgiler

Doğum yeri ve yılı : Kartal/İstanbul 1990 Cinsiyet: Erkek
Yabancı diller : İngilizce
GSM / e-posta : 0554 714 42 77 / ikaratas2515@gmail.com

BÖLÜM 1. GİRİŞ

Gelişen teknoloji ile birlikte kablosuz bağlantı aracılığıyla kısa mesafede birbirleriyle haberleşen ve bilgiyi toplama, işleme ve iletme gibi birçok özelleştirilmiş görevi yerine getiren sensör düğümlerinden oluşan kablosuz sensör ağları ortaya çıkmıştır. Bu yapılar kısıtlı kaynaklarla (RAM miktarı, ROM miktarı, batarya kapasitesi ve işlem gücü) donatılmış ve genellikle bir ya da daha fazla ana istasyona sahip; sağlıktan inşaat sektörüne, kimyadan çevre izlemeye, savaş alanlarının gözetim altında tutulmasına, fabrika otomasyonundan evleri uzaktan kontrol etmeye kadar geniş bir yelpazede kullanılmaktadır. Şekil 1.1'de IoT'nin kullanım alanları gösterilmektedir.



Şekil 1.1. Nesnelerin interneti kullanım alanları

Nesnelerin İnterneti (IoT) kavramı mümkün olan her şeyin internete ve dolayısıyla birbirine bağlanabileceği fikriyle ortaya çıkmıştır. Yeni neslin interneti olarak adlandırılan bu ağ, bir çok teknoloji, sensör yapısı, iletişim, ağ, akıllı bilgi işleme teknolojisinin birbirine entegre olması ile oluşmuştur [1]. Milyarlarca cihazı birbirine bağlayacak olan bu ağ yapısında sensörler aracılığı ile bilgiler toplanarak internet ortamına aktarılacak ve burada işlenerek çeşitli karar verme yapılarında kullanılacaktır. Bu kadar yaygın olarak kullanılacak bu cihazlar işlemci gücü, batarya ömrü, bant

genişliği vb. kısıtlardan dolayı geleneksel internette bulunan bir çok güvenlik tehditiyle başa çıkmak zorunda kalacaktır. Bu nedenle nesnelerin interneti yapısının gelişmesinin önündeki en önemli engel güvenlik problemidir. Birçok hassas bilginin bu ağ ve cihazlar üzerinden taşınılacağı düşünüldüğünde güvenliğin ne kadar hassas bir konu olduğu daha iyi anlaşılacaktır. Özellikle sağlık ve askeri alanda kullanılan uygulamalarda bu güvenlik ihtiyacı artmaktadır. Örneğin, sağlık sektöründe kullanılan insülin pompası bu duruma en güzel örneklerden biridir. İnsülin pompası, hastanın şeker miktarını düzenli olarak ölçerek gerektiğinde gerekli miktarda insülini hastaya enjekte etmekte ve kan şekerinin istenen seyirde olmasını sağlamaktadır. Ayrıca bu cihazlar internete bağlı şekilde hastanın şeker ölçümlerini rapor olarak göndermektedir. Bu gibi cihazların güvenliği sağlanmadığında çok büyük sonuçları olacağı açıktır.

İki düğüm arasında yapılan iletişimde güvenliğin sağlanması için gerekli olan bazı şartlar vardır. Bunlar; veri gizliliği, veri bütünlüğü, kaynak doğrulama, veri güncelliği, anahtar doğrulama, servis sürekliliğidir. Bu şartların sağlanması için geleneksel internette kullanılan yöntemler kablosuz sensör cihazların sahip olduğu kısıtlar nedeniyle mümkün olmamaktadır. Bu nedenle ya bu cihazlar için ayrıca protokoller geliştirilmesi ve yeni şifreleme tekniklerinin bulunması ya da var olan teknolojilerin hafifletilmesi gerekmektedir. Doğrulama ve anahtar yönetimi gibi güvenlik servisleri kablosuz sensör ağlarda iletişim kurma sürecinde kritik öneme sahiptir. Günümüzde kullanılan internet gibi geleneksel ağlarda güvenli iletişimi sağlamak için açık anahtar şifreleme (PKC) yapısı sayesinde pek çok güvenlik servisi ve protokolünün kullanılmasına olanak sağlamaktadır (SSL, IPsec, TLS gibi). Örneğin, simetrik anahtarların dağıtımında ve çoklu kullanıcıları doğrulama mesajlarında açık anahtar şifreleme sıklıkla kullanılmaktadır. Fakat bu açık anahtar şifreleme tabanlı güvenlik yöntemleri henüz kaynak kısıtlı sensör cihazlara belirtilen nedenlerden dolayı tam anlamıyla uygulanabilmiş değildir.

1985 yılında Neal Koblitz ve Victor Miller tarafından ilk defa Eliptik Eğri Kriptolojisi önerilmiştir. Eliptik Eğri Kriptolojisi (ECC) ayrık logaritma probleminin zorluğuna dayanmaktadır. RSA ile aynı güvenlik seviyesini çok daha düşük anahtar uzunluğu ile sağlamaktadır. Bu nedenle kaynak kısıtlı cihazlarda ECC kullanımı için önemli çalışmalar gerçekleştirilmiştir.

An Liu vd. tarafından yapılan TinyECC çalışmasında kablosuz sensör ağlarda kaynak kısıtlı cihazın kaynaklarının izin verdiği ölçüde farklı uygulamalara göre ayarlanabilen eliptik eğri tabanlı bir açık anahtar şifreleme için açık kaynak kodlu bir yazılım ortaya konulmuştur. TinyECC bellek kullanımı, çalışma zamanı ve kaynak tüketimi gibi gereksinimlere göre ayarlanabilmektedir. Yapılan çalışma kaynak tüketimi sebebiyle Wismote düğümünde gerçekleştirilmiştir. Bu çalışma ile 3 standart eliptik eğri algoritması entegre edilmektedir; Eliptik Eğri Diffie Hellman Anahtar Anlaşma Algoritması (ECDH), Eliptik Eğri Dijital İmza Algoritması (ECDSA), Eliptik Eğri Bütünleştirilmiş Şifreleme Şeması (ECIES) [2].

Piotr Szczechowiak vd. tarafından yapılan NANOECC adlı çalışmada kısıtlı kaynaklara sahip cihazlar için eliptik eğri kriptolojisinin kullanılabileceğini göstermiştir. Ancak elde edilen sonuçlara göre hala tam anlamıyla tatmin edici bir sonuç elde edilemediği görülmüştür. Yapılan çalışmadaki en etkili sonuçlar MICA2 (8-bit/7.3828-MHz ATmega128L) ve Tmote Sky (16-bit/8.192-MHz MSP-430) düğümlerinde elde edilmiştir [3].

Kai Zhao vd. tarafından yapılan çalışmada nesnelerin internetinde üç katmanlı sistem mimarisinde var olan güvenlik problemlerinden ve çözüm yollarından bahsetmişlerdir. Söz konusu güvenlik önlemleri arasında algılama katmanında anahtar yönetimi ve algoritma, güvenlik yönlendirme protokolü kimlik doğrulama ve erişim kontrolü üzerinde ayrıca durulmuştur. Nesnelerin İnterneti teknolojisini kullanan kısıtlı kaynaklara sahip cihazların simetrik ve asimetrik şifreleme teknikleri kullanarak güvenlik problemlerinin üstesinden gelinmeye çalışılmıştır. Güvenlik probleminin tek katmana değil bütün katmanlara ait problem olduğu ortaya konulmuştur [4].

Giederson Lessa dos Santos vd. tarafından gerçekleştirilen nesnelerin interneti için DTLS tabanlı güvenlik mimarisi çalışmasında güvenlik probleminin IOT gelişmesinde en büyük engel olduğu, kısıtlı işlem ve hafıza kapasitesi nedeniyle standart güvenlik mekanizmalarının desteklenmediği belirtilmiştir. Geliştirilen mimaride kısıtlı cihazlarda karşılıklı doğrulama ve iletişim için Nesnelerin İnterneti Güvenlik Destek Sağlayıcısı (IoTSSP) denilen üçüncü part bir cihaz kullanılarak Veri Bloğu Aktarım Katmanı Güvenliği (DTLS) kullanımına olanak sağlanmıştır [5].

Shadid Raza vd. tarafından gerçekleştirilen Lithe (IoT'de Düşük Yükte CoAP Güvenliği) çalışmasında DTLS ve CoAP'ın nesnelerin internetine uygulanması üzerine bir çalışma gerçekleştirmişlerdir. Ek olarak enerji tüketimini oldukça azaltan DTLS başlık sıkıştırma şeması önerilmiştir. Bu başlık sıkıştırma mekanizması uçtan uca güvenliği de etkilememiştir. Aynı zamanda DTLS işlemi gerçekleştirilirken değiştirilen byte miktarı da önemli ölçüde azalmıştır. DTLS tabanlı uygulama için Contiki işletim sistemi kullanılmıştır. Sonuçlar sıkıştırılmış DTLS mekanizması aktif hale getirildiğinde paket boyutu, enerji tüketimi, işlem zamanı ve ağ yanıt süresinde ciddi kazanç olduğu gözlenmiştir [6].

Hakan Aydın vd. tarafından yayınlanan nesnelerinin interneti teknolojisi için güvenlik adlı çalışmada nesnelerin interneti için düşünülen güncel protokoller tanıtılmış, oluşabilecek güvenlik zafiyetleri her katman için incelenmiştir. 6LoWPAN, 6TiSCH, CoAP gibi protokollerin güvenliğin ön planda olduğu bir anlayışla tasarlanması gerektiği belirtilmiştir [7].

Bu tez kapsamında amacımız; nesnelerin internetinde var olan kısıtlı cihazların birbirleriyle (M2M) ve insanlarla (M2H) iletişim kurarken, veri alışverişi yaparken bunun güvenli bir şekilde yapılmasını sağlamaktır. Bunu yaparken var olan işlem gücü, enerji tüketimi, RAM, ROM kapasitesi gibi kısıtlar göz önüne alınmış ve bir yöntem önerilmiştir. Özellikle DTLS'te yapılan el sıkışma aşamasında (doğrulama ve anahtar yönetimi) kısıtlı cihazlara yük bindiği ve güvenliğin sağlanmasında sorun oluştuğu çalışmalarda görülmüştür. Bu tez çalışması ile eliptik eğri şifreleme yöntemi kullanılarak düğümlerin birbirini doğrulanması ve anahtar değişimi sağlanmış ve asimetrik şifreleme yönteminin kaynak kısıtına sahip düğümlerde kullanılabileceği gösterilmiştir. Ayrıca geliştirilen eliptik eğri şifreleme kütüphaneleri, RSA ve simetrik şifreleme yöntemleri kullandıkları hafıza miktarları (RAM, ROM) açısından kıyaslanmıştır. Yapılan çalışma Contiki işletim sisteminde Cooja benzetim ortamında uygulanmış ve değerlendirilmiştir.

İkinci bölümde, simetrik ve asimetrik şifreleme yöntemlerinin özelliklerinden, artı ve eksi yönlerinden bahsedilmiştir. Eliptik eğri şifreleme yöntemi üzerinde durulmuş ve diğer asimetrik şifreleme yöntemlerine göre artıları incelenmiştir. Ayrıca nesnelerin internetinin yapısı ve var olan protokollerden bahsedilmiştir. Contiki işletim sistemi ve Cooja simülasyon programı anlatılmış ve temel simülasyon işlemleri gösterilmiştir.

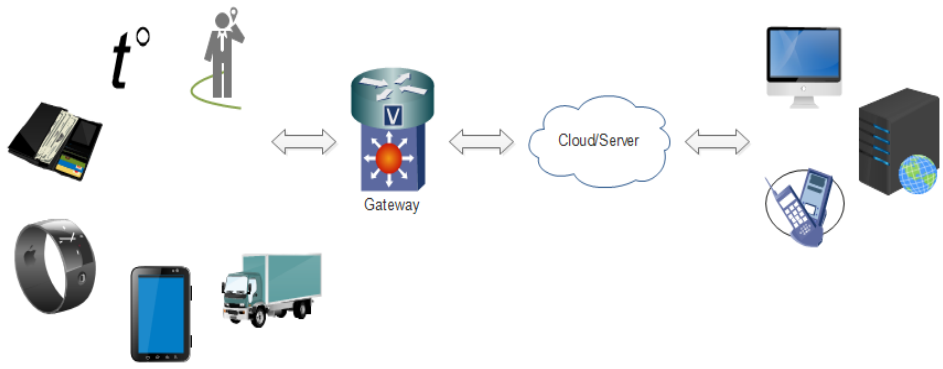
Üçüncü bölümde düğümler arası güvenli haberleşmeyi sağlamak için Cooja benzetim ortamında TinyDTLS kütüphanesi kullanılarak eliptik eğri şifreleme yöntemi test edilmiş ve simetrik şifreleme yöntemiyle kıyaslaması yapılmıştır. Ayrıca diğer eliptik eğri kütüphanelerinin kıyaslaması yapılmıştır.

Son bölümde benzetim ortamında elde edilen sonuçlar ortaya konulmuş ve değerlendirilmesi yapılmıştır. Ayrıca eliprik eğri şifrelemenin hala Sınıf-0 cihazlar için yeterince uygun olmadığı ve geliştirilmesi gerektiği ortaya konulmuştur.

BÖLÜM 2. NESNELERİN İNTERNETİ'NİN YAPISI VE GÜVENLİĞİ

Nesnelerin internetinin önemli özelliği cihazların çevreyi algılama, iletişim kurma, bilgiyi toplama, elde ettiği bilgiyi işlenme gibi spesifik görevleri yerine getirebilmesi için birbirine ve dolayısıyla internete bağlı olması gerekmektedir. Nesnelerin interneti uç düğümlerden bilgiyi elde etme, iletme ve işleme özelliğine sahiptir [8]. Bu önemli özelliklere sahip olması sensör ağlar aracılığı ile taşınan bilginin hassasiyetine göre güvenlik problemini üst seviyeye çıkarmaktır. Özellikle askeri alanda ve sağlık alanında çok sıklıkla kullanılan bu sensör düğümlerin güvenliğinin sağlanmasını zorunlu hale getirmektedir. Nesnelerin internetinin gelişmesinde en büyük engel olarak görülen güvenlik problemi aşılmadan yaygınlaşması mümkün görünmemektedir.

Sensör düğümlerin doğasından bulunan sistem kısıtlarından kaynaklanan güvenlik zaafiyeti geleneksel internette var olan güvenlik mekanizmaları ile aşılamamaktadır. Güvenlik mekanizmalarından yoksun olan bu cihazlar aracılığı ile taşınan önemli bilgiler (kişisel, askeri, sağlık vb.) herhangi bir saldırı durumunda açığa çıkabilmektedir. Bu durum şifrelemeden ve kimlik doğrulamadan yoksun bu cihazların kolay bir şekilde başkalarının eline geçebileceğini göstermektedir. Şekil 2.1'de IoT'nin yapısı ve genel çalışma mantığı gösterilmektedir.



Şekil 2.1. Nesnelerin interneti sisteminin yapısı

Nesnelerin interneti yapısı, kaynağını çevreye yayılmış çeşitli sensörlerden aldığı veriye dayanır. Sağlık sektörünü gözönüne alırsak, ağ bağlantısına sahip akıllı cihazlar yavaş yavaş insanlar üzerinde kullanılmaya başlamıştır. Bu giyilebilir teknolojiler sayesinde insanlardan toplanan sağlık verileri bir ağ geçidi aracılığı ile bir bulut sistemine veya sunucuya aktarılmakta, daha sonra buralarda işlenerek anlamlı bilgilere dönüştürülmekte ve gerekli merciler tarafından değerlendirilmektedir. Bu şekilde elde edilen bilgiler sayesinde herhangi bir sağlık sorunu önceden tespit edilip müdahale edilebilmektedir. Bu uygulamalar çok çeşitli alanlara yayılmakta askeri sistemlerden akıllı şehirlere, ulaşımdan üretim sistemlerine kadar birçok alanda kulanılmaktadır. Bu kadar önemli ve hassas bilgilerin taşındığı bu sistemler tasarlanırken göz önüne alınması gereken en önemli etken güvenlik olmalıdır [8].

Çizelge 2.1. Kısıtlı cihazların sınıflandırılması

İsim	RAM	ROM
Sınıf-0	<<10 KB	<<100 KB
Sınıf-1	~ 10 KB	~100 KB
Sınıf-2	~ 50 KB	~250 KB

Fakat nesnelerin internetinde birçok cihaz güvenli veri iletişimini sağlayacak kaynaklardan yoksun durumdadır. Bu cihazlar şu an kullanılmakta ve güvenlik önlemi olmadan veri alışverişine devam etmektedir. Bornmann vd. 2014 yılında yaptıkları çalışmada Çizelge 2.1'deki bu cihazları sahip olduğu RAM, ROM miktarına göre sınıflandırmıştır [9]. Bu cihazların sınıflandırılması hangi sınıf cihazda hangi güvenlik mekanizmasını kullanacağımızı belirlemektedir. Şu anda kullanılan güvenlik yöntemleri Sınıf-2 cihazlara rahat bir şekilde entegre edilebilmektedir. Uygulama katmanında CoAP ve iletim katmanı güvenliği için kullanılan DTLS için yeterli kaynağa sahiptir. Ancak Sınıf-1 ve Sınıf-0 cihazlar bu protokolleri tam anlamıyla destekleyecek kaynaklara sahip durumda değildir. Özellikle iletim katmanında DTLS protokolünde gerçekleştirilen el sıkışma aşamasında (kimlik doğrulama ve anaktar yönetimi) ciddi miktarda kaynak

tüketilmekte ve bu nedenle Sınıf-0 ve Sınıf-1 cihazları bu güvenlik protokolünü kullanamamaktadır. Bu aşamada gerçekleştirilecek bir müdahale ile bu işlem yoğunluğu azaltılırsa kaynak kısıtına sahip bu cihazların birbirleri arasında ve internet ortamında iletişimi güvenli hale gelecektir.

2.1 Güvenlik

Güvenlik; verinin bir uçtan diğer uca iletilirken yetkisiz kişiler tarafından ele geçirilememesi veya geçirilse bile ele geçirilen verinin üçüncü şahıslar tarafından bir anlam ifade etmemesidir. Güvenliğin sağlandığının söylenebilmesi için Şekil 2.3'te belirtildiği gibi gizlilik, bütünlük ve erişilebilirlik dediğimiz üç ayağın sağlanması gerekmektedir.



Şekil 2.2. McCumber bilgi güvenliği modeli [10]

Gizlilik; bilginin yetkisiz kişiler tarafından ele geçirilmesinin engellenmesidir. Bilginin gizliliği hem depolanma aşamasında hem iletim aşamasında sağlanmalıdır. McCumber modelinde bilginin hem depolanma hem de iletim esnasında şifrelenmesi gerektiği belirtilmiştir [10].

Bütünlük; bilginin içeriğinin depolanma ve iletim esnasında değiştirilememesi ve bütünlüğünün korunmasıdır. Bunu sağlamak için hash fonksiyonu dediğimiz kriptografik yapı kullanılmaktadır. Bu fonksiyon verinin kısa bir parmak izini (özetini) oluşturmakta ve veri değiştirildiğinde bu parmak izi geçerli olmayacağından değiştirildiği anlaşılmaktadır. h bir hash fonksiyonu ve x bir veri olsun. $y = h(x)$ verinin özeti olarak

adlandırılmaktadır. Burada *y* güvenli bir yerde saklanmakta ancak *x* saklanmamaktadır. Eğer *x* mesajı değiştirilirse bu durumda x' mesajının hash fonksiyonu alınır, $y' = h(x')$'dir. Bu durumda $y \neq y'$ olacağından mesaj içeriğinin değiştirildiği anlaşılır. Verinin depolanma aşamasında değiştirilmediğini kontrol etmek için zaman zaman parmak izi hesaplanmakta ve değiştirilmediği kontrol edilmektedir. Hash fonksiyonlarına örnek olarak MD5, SHA-1 ve HAVAL örnek olarak verilebilir [11].

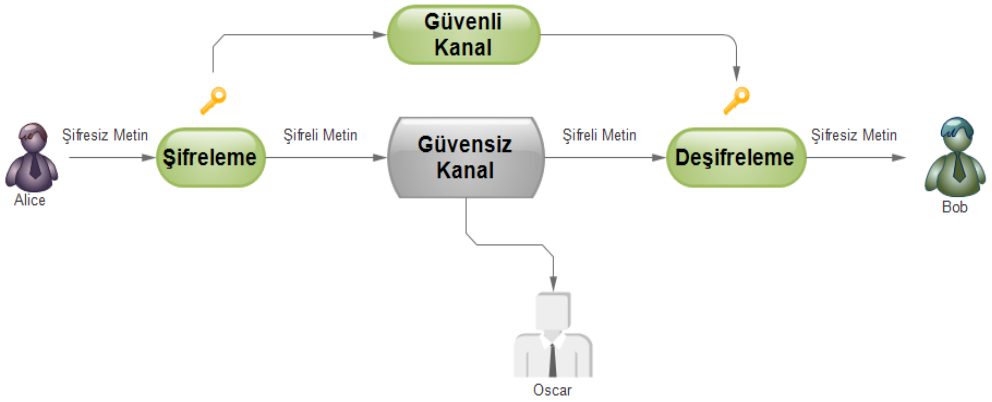
Erişilebilirlik; verinin ihtiyaç duyulduğu her an yetkili kişilerce ulaşılabilir olmasıdır. Bilgi, sürekli bir değişim halindedir ve bu nedenle sürekli ulaşılabilir olmak zorundadır. Sistem hasar görse dahi bir an önce onarılıp çalışır hale gelmesi gerekmektedir. Güvenlik duvarları ve proxy sunucuları gibi ekstra güvenlik ekipmanları ve yazılımlar, hizmet engelleme (DoS) saldırıları ve izinsiz ağ girişleri gibi zararlı eylemler nedeniyle kapalı kalma ve erişilemeyen verilere karşı koruma sağlayabilir.

Bu üç önemli ana unsurun yanında veri güncelliği, hizmet bütünlüğü, heterojenlik, anahtar yönetimi gibi gereksinimlerin sağlanması gerekmektedir. Veri güncelliği; ağda iletilen mesajların yeni olması, eski mesajların tekrarlanmamasıdır. Bunu sağlamak için ağdaki her mesaja zaman damgası eklenmektedir. Hizmet bütünlüğü; düğümlerden alınan verilerin bozulmadan güvenli bir şekilde toplanabilmesidir. Heterojenlik; nesnelerin internetinde üreticeler tarafından geliştirilen farklı özellikteki cihazların birbiriyle haberleşebilmesidir. Anahtar yönetimi ise; nesnelerin internetinde cihazlar verilerin güvenliği sağlamak için bazı parametreleri aralarında değiştirmeleri gerekmektedir. Bu güvenlik mekanizmaları için basitleştirilmiş anahtar yönetimi ve minimum enerji harcayan anahtar dağıtım yöntemler kullanılmalıdır [12].

2.2 Kriptolojik Kavramlar

Bu bölümde tez çalışmasında gerekli olan kriptolojik kavramlar ve özellikleri hakkında bilgi verilecektir.

Kriptoloji, Yunanca kökenli bir sözcük olup, gizli yazım anlamına gelmektedir. Kriptolojinin temeli iletişim kuran iki kişinin Şekil 2.3'teki gibi güvensiz bir kanal aracılığıyla haberleşirken üçüncü bir şahsın mesajı ele geçirse dahi ne olduğunu anlamamasıdır.



Şekil 2.3. Güvensiz kanal üzerinden simetrik şifreleme

Kriptosistem ise aşağıdaki şartları sağlayan (*P,C,K,E,D)* 5'li tanım grubudur;

1- *P;* mümkün olan şifresiz metinlerin sonlu kümesi,
2- *C* ; mümkün olan şifreli metinlerin sonlu kümesi,
3- *K;* mümkün olan tüm anahtarların sonlu kümesi,
4- $\forall k \in K$ olmak üzere şifreleme kuralına göre $e_k \in E$ *ve* $d_k \in D$ vardır. $e_k : P \rightarrow C$ ve $d_k : C \rightarrow P$ fonksiyonlardır öyleki $\forall x \in$ P için $d_k(e_k(x)) = x$'tir.

Burada şifresiz metin olan x, e_k fonksiyonu kullanılarak şifrelenir, d_k fonksiyonu kullanılarakta çözülür. Bu arada açıkça bellidir ki, e_k fonksiyonu birebirdir. Aksi taktirde fonksiyonun tersi olmaz ve çözümleme işlemi gerçekleştirilemez [11].

2.2.1 Simetrik Şifreleme

Simetrik şifreleme; haberleşmek isteyen iki birimin daha önce aralarında anlaşılan bir anahar aracılığı ile mesajın şifrelenerek güvensiz bir kanaldan gönderildiği ve yine karşı tarafta bulunan aynı anahtar ile mesajın çözüldüğü şifreleme türüdür [12]. Burada güvensiz kanaldan kasıt, mesajın üçüncü şahıslar tarafından ele geçirilse dahi anlamlandırılamamasıdır. Anahtar daha önce güvenli bir kanal yolu ile taraflar arasında paylaşılması gerekmektedir.

Simetrik şifrelemede şifreleme ve çözümleme işlemleri için aynı anahtar kullanılmaktadır. Ayrıca, Şifreleme ve çözümleme algoritmaları birbirinin tersidir. P;

şifresiz metin, C; şifreli metin ve K; anahtar olmak üzere, Şekil 2.4'te şifreleme ve çözümleme işlemi gösterilmiştir.

Şifreleme: $C = E_k(P)$ *ve* *Çözümleme:* $P = D_k(C)$

$D_k(E_k(x)) = x$'*tir.*

Şekil 2.4. Şifreleme ve çözümleme işlemi

Burada önemli olan nokta anahtarın güvenli bir kanal yoluyla iki tarafa ulaşmış olmasıdır. Bu güvenli kanal kişilerin görüşerek anahtarı arasında belirlemesi olabileceği gibi, güvenilir üçüncü şahıslar tarafından da gerçekleştirilebilir. Ayrıca Diffie Hellman anahtar değişim algoritması da buna çözüm olmaktadır [11,14].

Simetrik şifrelemede gönderici ve alıcı taraf tek bir şifre kullandığından veri şifreleme ve çözme için matematiksel açıdan az kaynak harcayan bir yaklaşımdır. Bu durumda nesnelerin interneti açısından oldukça kullanışlıdır. Çok hızlı bir şekilde ve az maliyetle şifreleme ve çözme işlemi gerçekleştirilebilir. Ayrıca kullanılan anahtar boyu ve bit sayısı da çok daha küçüktür. Bu da bizim konumuz açısından oldukça kullanışlı bir yaklaşımdır. Çizelde 2.2'de simetrik şifrelemenin olumlu ve olumsuz yönleri belirtilmiştir.

Simetrik şifreleme algoritmaları blok şifreleme ve dizi şifreleme şeklinde ikiye ayrılır. Blok şifreleme algoritmaları sabit uzunluktaki veriyi bit gruplarına ayırarak şifrelemektedir. Dizi şifreleme algoritmaları ise şifrelenecek açık metni bir bit dizisi olarak şifre basamak dizisiyle birleştirerek şifreleme işlemini gerçekleştirir [12].

Çizelge 2.2. Simetrik şifrelemenin olumlu ve olumsuz yönleri

Olumlu Yönleri	**Olumsuz Yönleri**
Algoritmalar hızlıdır ve az kaynak tüketir.	Anahtar dağıtımı ve kimlik doğrulama zordur.
Donanımla birlikte kullanılabilir.	Bütünlük sağlaması yoktur.
Güvenlidir.	Kimlik doğrulama ve anahtar değişimini gerçekleştirmek zordur.

Simetrik şifrelemeye örnek olarak DES (Veri Şifreleme Standartı) ve AES (Gelişmiş Şifreleme Standartı) örnek olarak verilebilir.

DES; blok şifreleme tekniğini kullanan bir algoritmadır. Daha çok büyük boyutlu verilerin şifrelenmesinde kullanılır. Şifreleme işlemi uzunlukları belli olan bloklar halinde gerçekleştirilir. DES'in güçlü yanı her kullanımda yeni bir anahtar üretmesi olup, günümüz teknolojisi için algoritmasının yavaş ve 56 bit'lik anahtar uzunluğunun yetersiz kalması DES'in zayıf yönleridir. Bu nedenle günümüzde DES'in üç defa uygulanmasıyla gerçekleştirilen yeni bir yöntem ile kullanılmaktadır. AES'e göre daha yavaş çalışmaktadır [12].

AES; DES'in güvenilirliğini yitirmesi ile onun yerini alan bir blok şifreleme algoritmasıdır. 128 bit, 192 bit ve 256 bit anahtar uzunlukları kullanılır. AES'in, DES'in aksine hızlı olması, esnek olması, daha kolay uygulanabilir olması ve daha az hafıza ile çalışması güçlü yönleri olarak söylenebilir. En yaygın olarak kullanılan simetrik şifreleme algoritmasıdır [12].

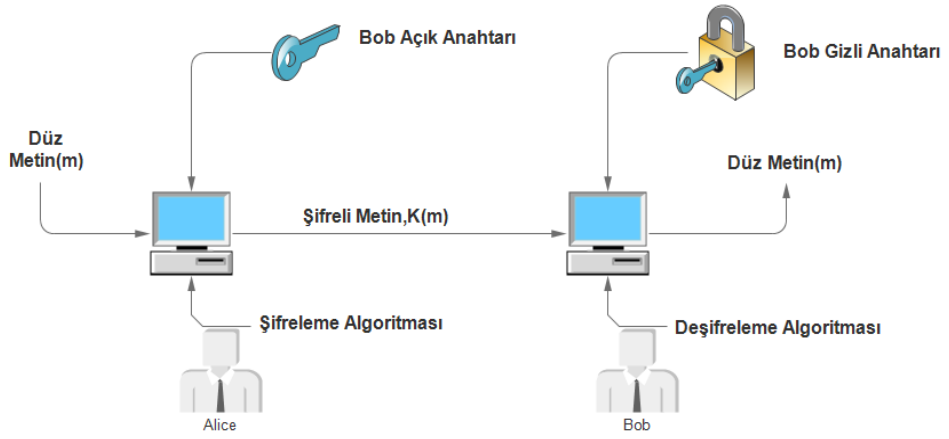
2.2.2 Asimetrik (Açık Anahtar) Şifreleme

Bu yöntemde haberleşmek isteyen birimlerin her birinde Şekil 2.5'te gösterildiği gibi iki anahtar bulunmaktadır. Bunlardan biri açık anahtar diğeri gizli anahtardır. Açık anahtar herkeste bulunabilir ancak hangi anahtarın kime dağıtıldığı bilinmelidir. Bu yüzden sertifika kullanılır. Sertifika, açık anahtar ile bu anahtarın sahibinin arasındaki bağlantıyı kuran yapıdır. Asimetrik şifreleme algoritması güvenlik açısından simetriğe göre çok daha başarılıdır. Az sayıda anahtar kullanarak simetrik şifreleme yapan çok kullanıcılı uygulamalarda ortaya çıkabilecek anahtar fazlalığı durumunu engeller.

Açık anahtar şifrelemenin altında yatan mantık verilen e_k fonksiyonu için $e_k(x)$ hesaplamak kolaydır ancak ters fonksiyonunu hesaplamak yani x'i bulmak uygulamada çok zordur (gizli anahtara sahip değilse). Bu tür fonksiyonlara tek yönlü fonksiyon (one-way function) denir.

Asimetrik algoritmaların güvenliğini sağlayabilmek için büyük asal sayılar (300 basamaklı) kullanılmaktadır. Bu da matematiksel işlem yoğunluğu demektir ve zaman açısından çok büyük problemler oluşturmaktadır. Asimetrik şifrelemeyi kullanan sistemler simetrik şifrelemeyi kullanan sistemlere göre çok daha yavaştır.

Ayrıca asimetrik şifreleme algoritmalarının çok büyük asal sayılar kullanması nedeniyle kaynak kısıtı fazla olan cihazlarda kullanımı oldukça zordur [11,13].



Şekil 2.5. Asimetrik şifreleme yöntemi

Asimetrik kriptosistemlerin, simetrik kriptosistemlere karşı bir diğer avantajı anahtar yönetimi ve kimlik doğrulamadır. Simetrik şifrelemede grup içerisinde n kişi olduğu düşünülürse n-1 kişiyle mesajlaşmak için n.(n-1) adet anahtara ihtiyaç vardır. Bu da anahtar dağıtım problemini ortaya çıkarmaktadır. n sayısı büyük rakamlara ulaştığında milyonlarca anahtar anlamına gelmektedir. Ayrıca gruba yeni kullanıcı ekleme işlemi de bir o kadar zordur. Bunun için n anahtar daha oluşturup bunu grup üyelerine dağıtmak gerekmektedir.

2.2.2.1 RSA Şifreleme

Bir açık anahtar şifreleme olan RSA; Rives, Shamir ve Aldeman adında 3 bilim insanı tarafından bulunmuştur. Anahtar dağıtımı, kimlik doğrulama, şifreleme-çözümleme, yüksek güvenlik gibi problemlere çözüm olmuştur. Sistemin temeli tam sayıların faktorizasyonuna dayanmaktadır. İki büyük asal sayını çarpımı çok kolay olmasına rağmen oluşan çarpımdan bu asal sayıları elde etmek çok zordur. Çalışma mantığı şu şekildedir;

1- p ve q gibi 2 adet büyük asal sayı seçilir.
2- n=pxq ve ϕ = (p-1)(q-1) değerleri hesaplanır.

3- $1 < e < \phi$ ve obeb $(e, \phi) = 1$ olacak şekilde bir e sayısı seçilir.

4- Öklid algoritması ile $1< d < \phi$ ve $ed = 1 \pmod{\phi}$ koşulunu sağlayan d sayısı hesaplanır.

n,e ~ açık anahtar ve d ~ gizli anahtardır. m şifrelenecek metin, (e,n) açık anahtardır. Şifreleme işlemi;

$$c= m^e \pmod n \qquad (1)$$

şeklinde gerçekleşir. Burada şifrelenmiş metin c ve gizli anahtar d'dir. Çözümleme işlemi

$$m = (c^e \bmod n) \qquad (2)$$

şeklinde gerçekleşir. Burada n bilinse dahi bundan p ve q'nun elde edilmesi imkansız veya çok güç olduğundan sistemin güvenliği garanti altındadır [14]. Şekil 2.6'da RSA şifrelemenin temel mantığı gösterilmiştir. Eğer yeterli anahtar uzunluğu sağlanmazsa RSA zayıf bir algoritma olur. Güvenli bir iletişim için en azından 1024-bit uzunluğunda bir anahtar kullanmak gerekmektedir.



Şekil 2.6. RSA şifreleme yöntemi

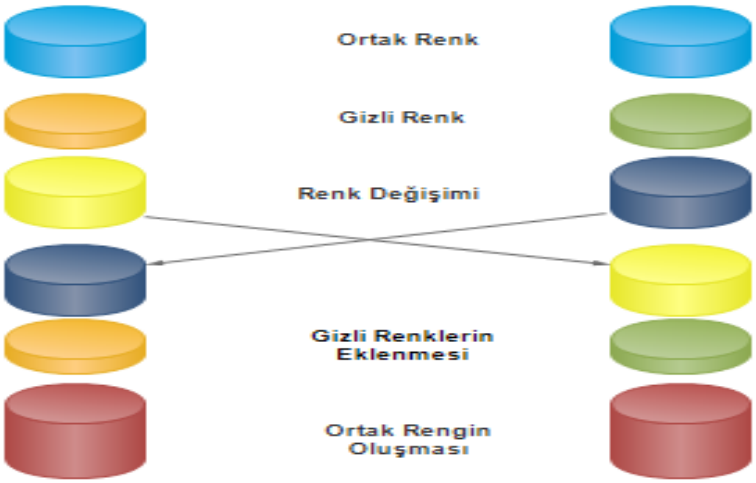
2.2.2.2 Diffie Hellman Anahtar Değişim Algoritması

Diffie Hellman anahtar değişim algoritması üs alarak anahtar değişme olarakta bilinir ve iki tarafın birbirlerine mesaj göndererek ortak bir anahtar oluşturma sürecidir. Şekil 2.7'deki gibi güvenli olmayan bir kanal üzerinden anahtar oluşturma işlemi gerçekleştirilir. Diffie–Hellman algoritması ile simetrik şifreleme algoritmalarının sorunu olan anahtar değişimi büyük ölçüde aşılmıştır. Ayrık logaritma problemine dayanmaktadır. Çalışma mantığı şu şekildedir;

p- yeterince büyük bir asal sayı ve g- bir primitif kök olmak üzere;

İki taraf aralarında p ve g parametrelerini belirler (Alice-Bob).

1- Alice A-random sayısını belirler ve $a = g^A \pmod p$ hesaplar.
2- Bob B-random sayısını belirler ve $b = g^B \pmod p$ hesaplar.
3- a ve b değerleri aralarında değiştirilir.
4- Alice $K = g^{BA} \pmod p$ ve Bob $g^{AB} \pmod p$ hesaplar.
5- $g^{BA} \pmod p = g^{AB} \pmod p$ olduğundan ortak anahtar oluşturulmuş olur.

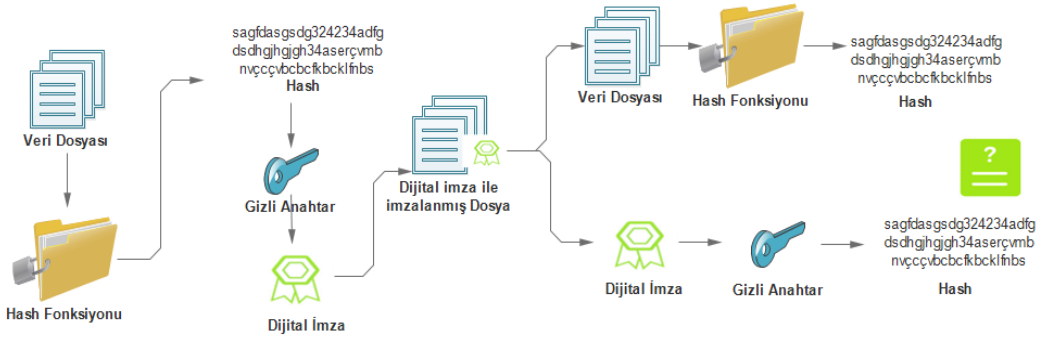


Şekil 2.7. Diffie Hellman anahtar değişim algoritması

2.2.2.3 Dijital İmza Algoritması

Dijital imza elektronik veriye eklenen ve kimlik doğrulama amacıyla kullanılan elektronik bir veridir. Dijital imza sürecinde; gönderici öncelikle mesajı hash fonksiyonuna sokar. Çünkü asimetrik key uzun mesajlarda verimsizdir. Bunun çözümü olarakta mesajın özetini çıkaran etkili bir hash fonksiyonu kullanmaktır. Şekil 2.8'deki gibi Alice kendi gizli anahtarı ile mesajı imzalar ve imzalanmış mesaj ile birlikte mesajın orjinalini Bob'a gönderir. Bob gelen imzalanmış mesajı Alice'in açık anahtarı ile açar. Gelen orijinal mesajıda hash fonksiyonuna sokarak bununla karşılaştırır. Eğer eşleşme doğru ise onaylanır değilse reddedilir. Daha önceki bölümlerde geçen kimlik doğrulama, mesaj bütünlüğü ve reddedilememe dijital imza ile sağlanmaktadır. Ancak gizlilik için

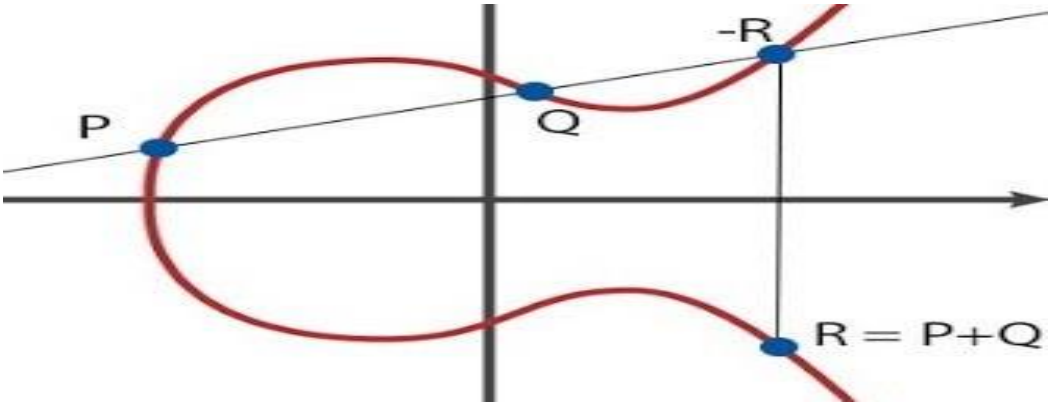
hala ek bir şifreleme sistemine ihtiyaç vardır. Normal şartlarda dijital imzada gizliliğe gerek olmasada, ihtiyaç halinde ek açık anahtar şifreleme uygulanarak bu durum aşılabilir [14].



Şekil 2.8. Dijital imza algoritması

2.2.2.4 Eliptik Eğri Şifreleme

Eliptik eğri şifreleme; sonlu cisimler üzerinde eliptik eğrilerin cebirsel topolojisine dayanan bir asimetrik şifreleme yöntemdir. ECC'nin güvenilirliği, eliptik egri ayrık logaritma probleminin çözümünün zorluguna dayanmaktadır. Anahtar anlaşması, dijital imza gibi bir çok alanda kullanılabilmektedir. Eliptik eğri, gerçek sayılar kümesi üzerinde tanımlanan ve $y^2 = x^3 + ax + b$, genel denklemini x ve y gerçek sayıları için sağlayan eğrinin ismidir. Bu genel denklem için her a ve b değeri farklı bir eğri verir [15]. Bu denklemdeki a ve b sayıları gerçel sayılardır ve $x^3 + ax + b$ denkleminin çoklu kökü olmaması için $4a^3 + 27b^2 \neq 0$ olmalıdır. Eliptik eğri şifreleme, açık anahtar kriptosistemi ile karsılastırıldıgında aynı güvenlik seviyesi için daha kısa anahtar uzunluğu kullanır. Örnegin; 1536-bit anahtar boyutlu RSA ile sağlanan güvenlik, 160-bit anahtarlı eliptik egri şifreleme ile saglanabilmektedir. Bu da gömülü sistemler, akıllı telefonlar için oldukça kullanışlıdır. Eliptik eğri algoritmasının çalışması şu şekildedir;



Şekil 2.9. Eliptik eğri üzerinde iki noktanın toplanması

p asal üstel bir sayı Fp, p eleman içeren sonlu bir alanı ifade eder. Bu kümede tanımlı olan herhangi bir işlemin sonucu yine bu alanın elemanıdır (mod p). Fp alanında tanımlanmış bir E eliptik eğrisi ve bu eğri üzerindeki P ve Q noktaları verilmişken $Q=kP$ denkleminin biricik çözümü olan k, $0 \leq k \leq p-1$ değerinin bulunması eliptik eğri ayrık logaritma problemi olarak tanımlanmaktadır. P, Q $\in$ EF (a,b) ve k<p iken, $Q=kP$ olsun. k ve P verildiğinde Q değerini hesaplamak nispeten kolay olduğu halde, Q ve P verildiğinde k değerini hesaplamak gerçekten çok zordur [15,16].

Burada P ve Q eğri üzerinde iki nokta ve bu noktalardan çizilen bir doğru yine eğriyi 3. bir nokta -R'de kesmek zorundadır. -R noktasının x-eksenine göre simetriği yine eğri üzerinde bir R noktasıdır. Bu durumda Şekil 2.9'daki gibi $P+Q = R$ elde edilir. Bu işlem k defa tekrar edildiğinde yine eğri üzerinde bir nokta elde edilir. Bu güzel bir trapdoor fonksiyonudur çünkü başladığın P noktasını ve kaç defa bu işlemi yürüttüğü bilindiğinde Q noktasını elde etmek kolaydır. Ancak P ve Q bilindiğinde, k'yı elde etmek neredeyse imkansızdır.

2.2.2.4.1 Eliptik Eğri Dijital İmza Algoritması

Dijital imza daha önce bahsettiğimiz gibi asimetrik şifreleme tabanlı bir kimlik doğrulama yöntemidir. Eliptik eğri dijital imza yöntemi (EEDİA) ise çok daha kısa anahtar, hızlı hesaplama süresi, daha az saklama alanı ve bant genişliği kullanması nedeniyle klasik dijital imzaya göre kullanımı çok daha avantajlıdır. EEDİA sayesinde veri bütünlüğü, kimlik doğrulama ve reddedeme başarılmış olmaktadır. Çalışma mantığı şu şekildedir;

Gönderici ve alıcı p, a, b, G, n, h parametreleri üzerinde anlaşır.

Göndericinin gizli anahtarı [1, n-1] aralığında seçilen rastgele tamsayı d_s'dir.

Göndericinin açık anahtarı ise $e_s = d_sG$'dir.

Dijital İmza Hesaplama Algoritması

X = HASH (m) hesapla.

do {

[1,n−1] aralığından rastgele bir k tamsayısı seç.

(x1, y1) = kG olsun.

$r \equiv x_1 \pmod n$ hesapla.

$s \equiv k^{-1} (X + d_S r) \pmod n$ hesapla.

} while (r == 0 || s == 0)

(r, s) alıcıya gönderilen imzadır.

Dijital İmzayı Doğrulama Algoritması

Alıcı, göndericinin açık anahtarı e_s yi ve (r, s) yi ve m'yi alır.

r ve s değerlerinin [1, n-1] aralığında olduğundan emin olunmalıdır.

X = HASH (m) hesapla.

$w \equiv s^1 \pmod n$ hesapla.

$u_1 \equiv Xw \pmod n$ hesapla.

$u_2 \equiv rw \pmod n$ hesapla.

$(x_1 , y_1) = u_1G + u_2 e_S$ hesapla.

$x_1 \equiv r \pmod n$ ise imzayı doğrula [18].

2.2.2.4.2 Eliprik Eğri Diffie Hellman Anahtar Değişim Algoritması

Eliptik Eğri Diffie-Hellman Anahtar Değişim Algoritması güvensiz bir kanal üzerinden iki tarafın eliptik eğri şifreleme yöntemi ile bir anahtar üzerinde anlaşmasını sağlayan anahtar değişim protokolüdür. Çalışması şu şekildedir;

Anahtar Üretim Aşaması

Gönderici ve alıcı p,a,b,G,n,h parametreleri üzerinde anlaşır.

1-d_s ve d_r gizli anahtarları [1,n-1] aralığında rastgele seçilir.

2-Açık anaharlar $e_s = d_sG$ ve $e_R = d_RG$'dir.

Gizli Anahtarın hesaplanması

Gönderici $K = (x_K,y_k) = d_Se_R$'yi hesaplar.

Alıcı $L=(x_L,y_L) = d_R e_S$ hesaplar.

$d_s e_R = d_s d_R G = d_R d_S G = d_R e_s$

Böylece $K=L$ ve $x_K = x_L$ elde edilir.

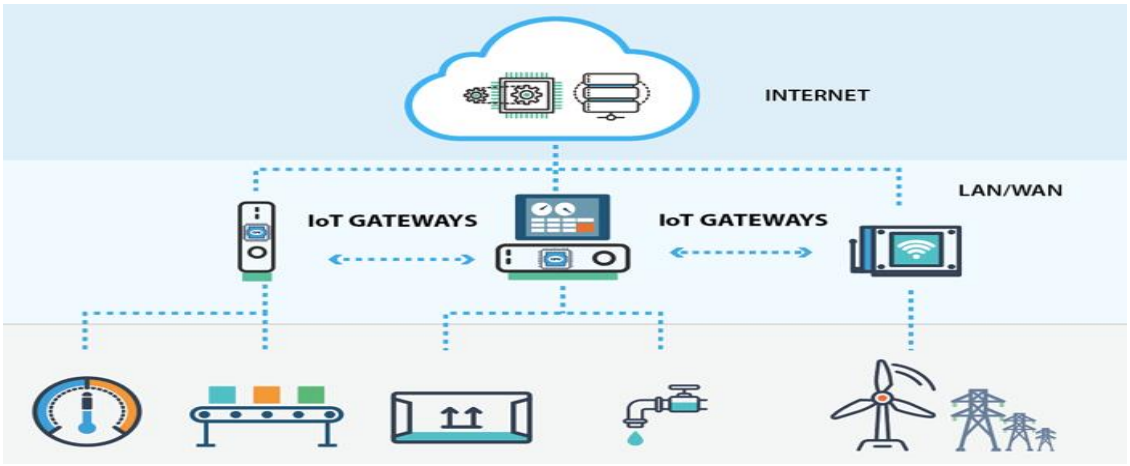
Gizli anahtar x_k olur [18].

Eliptik eğri şifreleme yöntemiyle Diffie-Hellman anahtar değişimindeki asimetrik şifrelemeden kaynaklanan yük ciddi oranda düşürülmüş olur. Özellikle hibrit sistem dediğimiz simetrik ve asimetrik şifre yönteminin beraber kullanımında simetrik şifrelemeden kaynaklanan anahtar değişim problemi eliptik eğri Diffie-Hellman anahtar değişim protokolü ile çok az bir yükle aşılmış olacaktır ve özellikle kaynak kısıtlı cihazlarda var olan hesaplama, hafıza, batarya gibi problemler aşılmış olacaktır. Bu da nesnelerin internetinde var olan güvenli haberleşme sorununa bu açıdan çözüm üretebilmektedir.

2.3 Nesnelerin İnternetinin Yapısı

Bu bölümde nesnelerin internetinin yapısı, var olan protokoller, güvenlik problemleri ve bununla alakalı çözüm yolları anlatılacaktır.

Gelecekte IoT akıllı cihazların ve fiziksel objelerin iletişim ve ağ kapasitelerinin genişlemesi ile insanların günlük yaşantısının bir parçası haline gelecektir. Bu cihazlar, insanların faydası için bilgiyi toplama işleme ve yeri geldiğinde karar verme yeteneğine sahip, birbirleriyle işbirliği içerisinde çalışan cihazlardır. IoT ağ yapısı Şekil 2.10'daki gösterildiği gibi çok farklı alanlarda bulunan sensör cihazları farklı ağ geçitleri aracılığı ile kablolu/kablosuz şekilde internete açılmaktadır.



Şekil 2.10. Nesnelerin interneti ağ yapısı

Nesnelerin İnterneti teknolojisini kullanmak; bütün kişisel mahremiyetimizi internete açmamız, hakkımızda eşi görülmemiş erişim ve bilgi toplanmasına izin vermemiz anlamına gelmektedir. Bu da güvenliğin ne kadar önemli bir konu olduğunu göstermektedir. Güvenlik sorunu çok boyutlu bir sorundur. Cihazın fiziksel olarak korunmasından, cihaz içindeki bilginin şifrelenmesine, farklı firmalara ait cihazların uyum içinde çalışmasına, sistemin her zaman çalışır olmasına kadar pek çok yönü kapsamaktadır. Bizim bu tez kapsamında ilgilendiğimiz kısmı ise, verinin iletim esnasında şifreli bir şekilde karşı tarafa ulaştırılmasıdır.

Güvenli bir yapı dizayn edilirken aşağıdaki durumlar göz önünde bulundurulmalıdır:

1-Teknik faktörler; algılama tekniği, iletişim metodu, ağ teknolojisi

2-Güvenlik koruması; bilgi gizliliği, iletim gübenliği, gizlilik koruması [8].

Nesnelerinin internetinin halen daha kabul görmüş bir mimari yapısı bulunmamaktadır. Çizelge 2.3'te belirtildiği gibi IETF tarafından kaynak kısıtlı cihazlar için 6LoWPAN, RPL, CoAP gibi protokoller geliştirilmiştir.

Çizelge 2.3. Nesnelerin interneti katman ve protokol yapısı

Iot Katmanı	IoT Protokol Yığını	Güvenlik Protokolü
Uygulama Katmanı	CoAP	-
İletim Katmanı	UDP	DTLS
Ağ Katmanı	RPL, IPv6	IPsec,RPL security
Adaptasyon Katmanı	6LoWPAN	-
Fiziksel Katman	IEEE 802.15.4	802.15.4 security

2.2.1 Kısıtlı Uygulama Protokolü (CoAP)

Kısıtlı Uygulama Protokolü, IETF tarafından kaynak kısıtlı cihazlarda M2M haberleşme için geliştirilmiş bir uygulama katmanı protokolüdür. Çok kullanılan senkron internet protokolü HTTP'nin bir çeşidir. HTTP'ye benzer şekilde REST arayüzü sunsa da daha çok sistem kısıtına yönelmiştir. Nesnelerin internetinde düğümler genel olarak 6LoWPAN (IPv6 üzerinde düşük güçlü kablosuz kişisel alan ağları) gibi kısıtlı ağlarda, 8-bit mikrodenetleyici ve kısıtlı RAM, ROM miktarına sahip, paket kayıp oranı yüksek yapılardır. CoAP, kaynak kısıtlı çevrelerde, düşük güç, basitlik ve çoklu haberleşme gibi

özel gereksinimleri karşılarken, web ile entegrasyon için HTTP ile kolay bir şekilde arayüz oluşturacak şekilde geliştirilmiştir.

6LoWPAN gibi kısıtlı ağlar, IPv6 paketlerini küçük bağlantı katmanı çevçevelerine parçalamayı desteklemektedir fakat bu durum paketlerin iletilmesinde ciddi sorunlara neden olmaktadır. CoAP dizayn edilmesindeki temel sebeplerden biri de mesaj boyutunu küçük tutarak parçalanma ihtiyacını azaltmaktır. CoAP'ın sahip olduğu ana özellikler şunlardır;

1-Web protokolü kısıtlı çevrelerde M2M gereksinimini tam anlamıyla karşılamaktadır.

2-Tek yönlü ve çoklu iletişimi destekler.

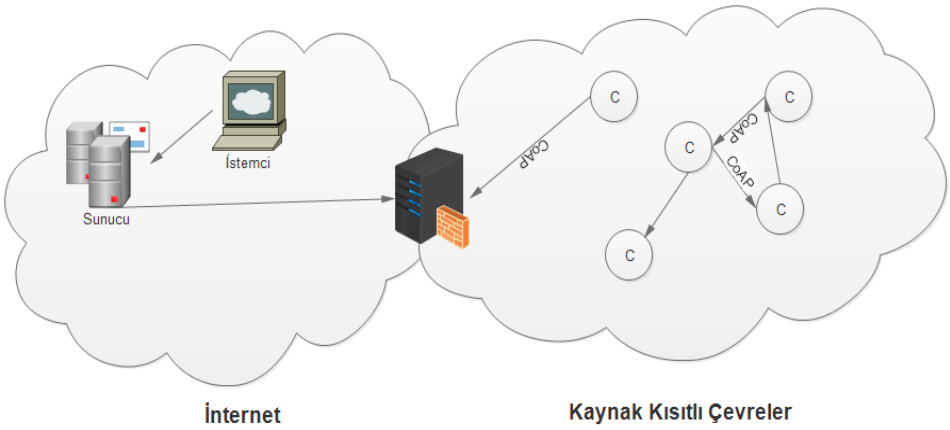
3-Asenkron mesaj değişimi ile iletişim kurar.

4-Düşük başlık yükü ve kompleksliği azaltır.

5-Basit proxy ve caching kapasitesi sağlar.

6-DTLS sayesinde güvenilir bağlantı yapısı sunar [19].

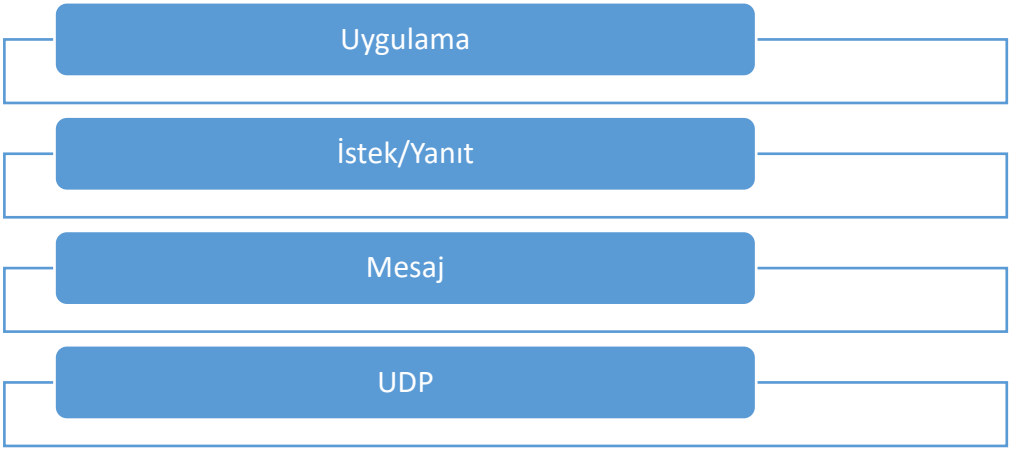
CoAP'ın etkileşimli modeli, HTTP'nin istemci/sunucu yapısıyla benzerlik gösterir. Ancak CoAP uygulamasındaki M2M etkileşim yapısı cihazlara hem sunucu hem istemci rolünde görev yapmasına olanak verir. Bunun sebebi ise bu yeni protokol kısıtlı ağlar için geliştirildiğinden, uygulamadaki karmaşıklığı (kod boyutunu) ve bant genişliğini ciddi oranda azaltmasıdır. Bu kaynak kullanımındaki azalma güvenilirliğin artmasına (bağlantı katmanındaki parçalanmayı azaltarak) ve kayıplı ağlardaki gecikmenin azalmasına yardımcı olur [20]. CoAP mimari yapısı Şekil 2.11'deki gibidir.



Şekil 2.11. CoAP mimarisi

CoAP, HTTP'ye benzer şekilde REST servis yapısını kullanır. Çünkü data boyutu ve uygulama hızı önem arzettiği için bu yapıyı kullanmak ciddi avantaj sağlamaktadır. Ayrıca kısıtlı cihazlardaki dezavantajın üstesinden gelmek için veri boyutunu optimize etmeli ve güvenilir iletişimi sağlamalıdır. Diğer taraftan CoAP; GET, POST, PUT, DELETE gibi URI, REST metodlarını desteklemektedir. UDP (Kullanıcı Veribirimi Protokolü) protokolünün güvensiz yapısındaki zaafiyeti gidermek için ise CoAP tekrar iletim mekanizması ve kaynak keşif mekanizması tanımlamıştır [21].

HTTP'den farklı olarak, CoAP mesaj değişimleri UDP üzerinden asenkron (eş zamansız) olarak gerçekleşmektedir. Bunu da isteğe bağlı güvenilirliği destekleyen mesaj katmanı kullanarak gerçekleştirir.

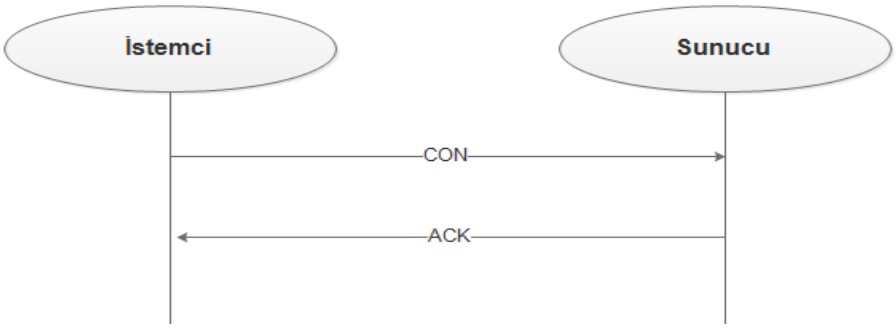


Şekil 2.12. CoAP katman yapısı

CoAP'ın etkileşimli modeli HTTP'nin istemci/sunucu modeli ile benzerlik göstermektedir. Şekil 2.12'de görüldüğü gibi CoAP, iki katman yapısı ile çalışmaktadır. Alt katman olan mesaj katmanı UDP ile ilgilenir. İstek/yanıt katmanı ise iletişim yöntemiyle ilgilidir ve istek/yanıt mesajları ile ilgilenir. Mesaj katmanı dört tip mesajı desteklemektedir: Onaylanabilir (CON), Onaylanamaz (NON), Kabul edilebilir (ACK) ve Sıfırlama (RST).

2.2.1.1 Mesaj Katmanı Modeli

Güvenilir mesaj iletimi: Şekil 2.13'te görüldüğü gibi aynı mesaj ID'sine sahip ACK yanıtını alana kadar tekrar mesaj iletimine devam eder.



Şekil 2.13. Güvenilir mesaj iletimi

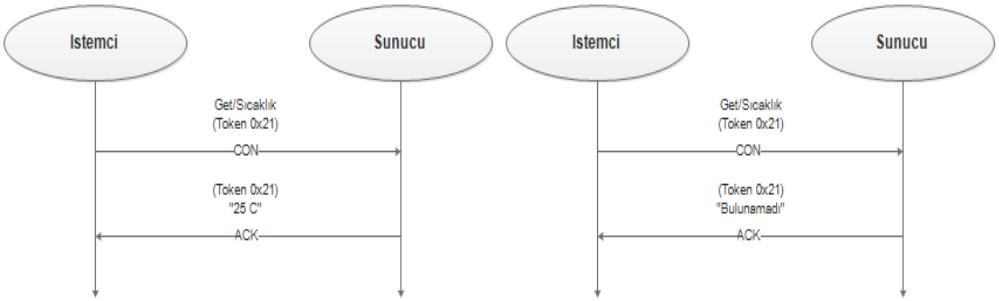
Güvensiz mesaj iletimi: Şekil 2.14'te görüldüğü gibi NON mesaj tipi ile iletilir. ACK mesajına ihtiyaç duymaz fakat tekrar iletim durumunda denetleme için mesaj ID bulundurmak zorundadır. Eğer alıcı mesaj işlemekte başarısız olursa sunucu RST mesajı ile cevaplar.



Şekil 2.14. Güvenilir olmayan mesaj iletimi

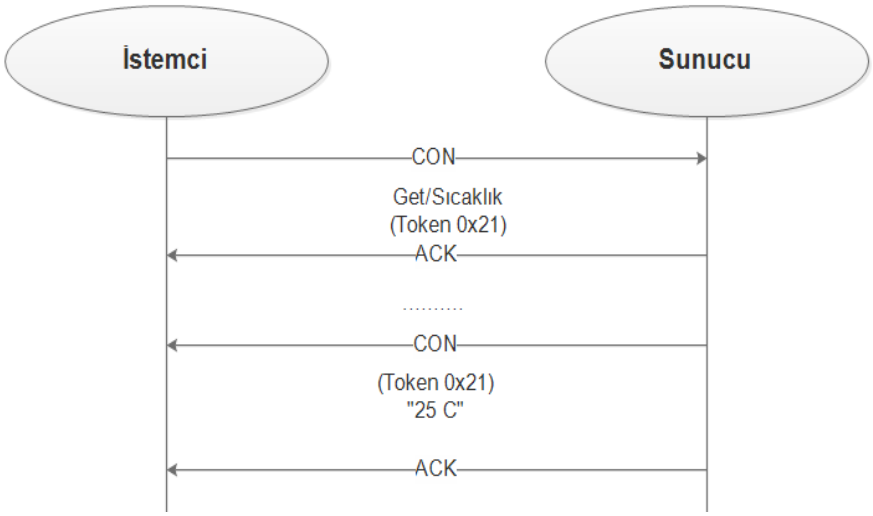
2.2.1.2 İstek/Yanıt Katman Modeli

Arka plan: İstemci, CON veya NON mesaj tipi ile bir istek gönderir ve anında ACK onay mesajı ile cevap alır. Başarılı yanıtı için ACK bir yanıt mesajı içerir (token kullanılarak tanımlanmış), başarısız yanıt için ise, ACK başarısız yanıt kodu içermektedir. Token, isteklere verilen yanıtları temel mesajlardan bağımsız olarak eşleştirmek için kullanılır (mesaj ID'den ayrı bir yapıdadır). Yani istek ve yanıt mesajları arasında bir bağıntı kurar. Şekil 2.15'te GET metodu ile sıcaklık parametresi isteğinin başarılı ve başarısız sonuçları gösterilmiştir.



Şekil 2.15. GET metodunun başarılı ve başarısız yanıt sonuçları

Ayrı Yanıt: Şekil 2.15'te görüldüğü gibi sunucu bir CON tipinde mesaj alır ve bunu hemen cevaplamak istemezse, istemcinin tekrar bu mesajı göndermesi durumunda boş bir ACK mesajı gönderir. Ne zaman sunucu bu mesajı cevaplamak isterse, istemciye yeni bir CON mesajı gönderir ve istemci ACK ile onaylanabilir mesajı yanıtlar. ACK yalnızca CON mesajını onaylar.



Şekil 2.16. Ayrı yanıt ile GET isteği

Onaylanmayan istek ve yanıt: İstemci NON tipinde mesaj gönderir. Bu durum sunucunun bu mesajı onaylamasına gerek olmadığını bildirir. Sunucu tekrar NON tipinde mesajla cevap verir [19].

2.2.1.3 CoAP Mesaj Formatı

CoAP, varsayılan olarak UDP üzerinden aktarılan mesaj alışverişine dayanmaktadır. Bu mesajlar basit ikili format kullanmaktadır. CoAP mesaj yapısı şekil 2.17'deki gibidir;



Şekil 2.17. CoAP mesaj yapısı

Versiyon (V): 2-bit işaretsiz tamsayıdır. CoAP versiyonunu belirtir.

Mesaj Tipi (T): 2-bit işaretsiz tamsayıdır. Mesajın; Onaylanabilir (0), Onaylanamaz (1), Kabul Edilebilir (2) ve Reset (3) tiplerinden oluşması durumunda kullanılır.

Token Uzunluğu (TKL): 4-bit işaretsiz tamsayıdır. Değişken uzunluktaki token alanının uzunluğunu belirtir.

Kod (Code): 8-bit işaretsiz tamsayıdır. Bu bit sınıfı, bir isteği (0), başarılı bir yanıtı (2), istemcinin hata mesajını (4) veya sunucunun hata mesajını (5) belirtir. Özel olarak (0.00) kodu boş mesajı belirtmektedir.

Mesaj No: 16-bit işaretsiz tamsayıdır. Mesaj kopyasını tespit etmek ve Kabul edilebilir/Sıfırlama tip mesajlar ile Onaylanabilir/Onaylanamaz mesaj tiplerini eşleştirmek için kullanılır [19].

Özelliklerinden bahsedilen CoAP protokolü kaynak kısıtlı cihazlar ile birlikte kablosuz sensör ağlarında kullanılmaktadır. Geleneksel protokoller, sınırlı kaynağa sahip cihazlar için uygun olmadığından CoAP kullanımı bir gerekliliktir. Düşük güçlü cihazlar yıllarca pillerle veya enerji toplama yoluyla çalışması gerektiğinden az enerji harcamak zorundadır. UDP destekli cihazlarda çalışan CoAP, küçük mesaj boyutları, mesaj yönetimi ve düşük güç tüketen, düşük bellekli cihazlar için ideal olan hafif mesaj yükü özelliklerine sahip bir uygulama katmanı protokolüdür. Ayrıca CoAP taşınan paket başlığını 1000 byte'tan 100 byte'a kadar indirebilir. 6LoWPAN gibi diğer protokoller kullanıldığında bu boyut 10 byte'a kadar düşürülebilir.

Çizelge 2.4. İnternet ve IoT ile taşınan paket başlığı karşılaştırması

İnternet yığını 1000-byte	IoT yığını 10-byte
XML	JSON
HTTP/FTP/SMTP	COAP
TLS	DTLS
TCP/UDP	UDP
IPv4/IPv6	6LOWPAN

CoAP mesajlarının şifrelenmesi için iletim katmanındaki güvenlik mekanizması olan DTLS kullanılmaktadır. Geleneksel internette kullanılan TLS'in UDP tabanlı versiyonudur. DTLS'in güvenliğinin sağlanması CoAP'ın da güvenliğinin sağlanması demektir. CoAP için dört tip güvenlik mekanizması tanımlanmıştır. Bunlar Çizelge 2.5'te belirtildiği gibi NoSec, PreSharedKey, RawPublicKey, Certificate modudur [19].

Çizelge 2.5. CoAP güvenlik mekanizmaları

NoSec	DTLS kullanılmaz.
PreSharedKey	DTLS aktiftir, haberleşme için kullanılacak simetrik anahtarlar düğümlere önceden dağıtılmıştır.
RawPublicKey	DTLS aktiftir, haberleşme için asimetrik anahtar çiftleri kullanılır.
Certificate	DTLS aktiftir, asimetrik key çiftleri sertifika ile kullanılır.

2.2.2 Veri Birimi İletim Katmanı Güvenliği (DTLS)

TLS protokolü, ağ trafiğini güvenli hale getirmek için oluşturulmuş, iletişim kuran iki uç arasında veri bütünlüğü ve gizliliği sağlamaktadır. TLS protokolü, TLS Kayıt Protokolü ve TLS El Sıkışma olmak üzere iki katmandan oluşmaktadır. Fakat TLS, TCP gibi güvenilir iletim kanalı üzerinde çalışmalıdır. İletim katmanı olarak UDP kullanan uygulamalar için ise iletişim güvenliğinin sağlanması amacıyla DTLS protokolü dizayn edilmiştir.

Nesnelerin interneti uygulamalarında güvenlik protokolü genel itibariyle küçük, düşük güçlü, uzaktan kontrol edilebilen sensörleri ve komponentleri hedef almaktadır. DTLS protokolü TLS'i temel alarak geliştirilmiş; gizlilik, doğrulama ve bütünlük koruma gibi eşit güvenlik servisi sağlayan bir iletim katmanı protokolüdür. TLS, TCP üzerinden çalıştığı için paket kaybı ve paket sıralaması gibi problemlerle uğraşmak durumunda kalmaz. DTLS'te ise el sıkışma mekanizması paket kaybı, yeniden sıralama ve tekrar iletim gibi problemlerle uğraşmak zorundadır. DTLS'te düğümlerin başlangıçta doğrulanması, anahtar üzerine anlaşması ve veri koruması güvenli bir kanal üzerinden gerçekleşir.

Veri birimi iletimi, verinin sırayla iletimi ve güvenilirliğini garanti etmemektedir. DTLS protokolü ise uygulama verisi için bu özellikleri korumaktadır. İnternet üzerinden bir medya, internet telefonu ya da oyunlar gibi uygulamalarda iletişim için iletilen verinin gecikme hassasiyetinden dolayı UDP kullanılmaktadır [22].

DTLS protokolü, UDP'den kaynaklı paket kayıplarını (packet loss) yöneten, alıcı tarafta paketin yeniden düzenlenmesini sağlayan (reordering) ve daha düşük çerçeveler üzerinde işlem yapan bir protokoldür.

2.2.2.1 Yeniden Sıralama

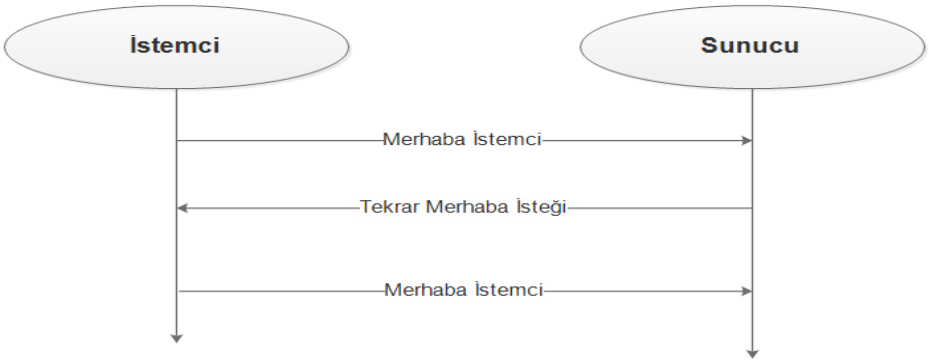
DTLS'te her el sıkışma (handshake) mesajı özel bir dizi numarasına atanmıştır. Bir uç el sıkışma mesajı aldığında, bu mesajın beklediği bir sonraki mesaj olup olmadığını hızlıca belirleyebilir. Eğer o ise, onu işler; değilse bütün mesajlar alınana kadar ileride ele almak üzere onu sıraya koyar.

2.2.2.2 Mesaj Boyutu

TLS ve DTLS'in el sıkışma mesajları oldukça büyük miktarda olabilmektedir. Tersine, UDP veribirimleri genellikle 1500 byte'tan daha az olmaktadır. Bu kısıtı halletmek için, her DTLS el sıkışma mesajı, her biri tek IP paketine sığacak şekilde, birkaç DTLS kayıt katmanına ayrılabilir. Her DTLS el sıkışma mesajı parça pozisyonu ve parça uzunluğunu içermektedir. Bu şekilde alıcı gelen el sıkışma mesajlarını tekrar toplayarak gönderildiği haline dönüştürebilir.

2.2.2.3 Paket kaybı

DTLS, paket kayıplarının üstesinden gelmek için basit bir tekrar iletim zamanlayıcısı kullanmaktadır. Şekil 2.18'de DTLS el sıkışma aşamasının ilk safhası kullanarak basit bir gösterimi sunulmuştur.



Şekil 2.18. DTLS tekrar iletim örneği

Öncelikle istemci "Merhaba İstemci" mesajını gönderir, daha sonra "Merhaba Yeniden Dene" veya "Merhaba Sunucu" mesajlarının gelmesini bekler. Fakat sunucunun mesajı kaybolursa, istemci bilir ki ya "Merhaba isteci" ya da sunucudan gelen cevap kaybolmuştur ve bu durumda mesajı yeniden iletir. Sunucu tekrar iletim mesajını aldığında, onun tekrar iletim mesajı olduğunu bilir. Sunucu ayrıca yeniden iletim zamanlayıcısını korur ve bu zamanlayıcı sona erdiğinde tekrar iletir [22].

DTLS'in üst katmanı olan Kayıt Katmanı, dört protokol tipinden oluşmaktadır. Bunlar El Sıkışma Protokolü, Uyarı Protokolü, Şifre Değişimi Protokolü ve Uygulama protokolüdür.

2.2.2.4 DTLS El Sıkışma Protokolü

El sıkışma protokolü, istemci ve sunucu arasında güvenli bir iletişim kanalı için anahtar oluşturma ve uç noktaların birbirini doğrulaması için kullanılmaktadır. Bu protokol iletişim kuran uçlarda güvenliği sağlamada DTLS'in özüdür diyebiliriz. DTLS'te açık anahtar ve sertifika modu gibi ağır kriptografik işlemler içeren çeşitli doğrulama ve anahtar değişim kod blokları bulunmaktadır. DTLS'te daha önce kurulan

bağlantının devam etmesini sağlayan bir mekanizma bulunmaktadır, buna RESUME protokolü denir. Daha önce üzerinde anlaşılan kriptografik parametreleri geri getirmek için oturum ID'si kullanılabilir ve bu şekilde güvenli kanal tekrar kurulmuş olur. Bu sayede pahalı sertifika doğrulama operasyonların kurtulmuş oluruz. Günümüzde kullandığımız gömülü sistemlerin büyük çoğunluğu temel el sıkışma protokollerini uygular ve oturum devam ettirme özelliğini desteklemez. Ayrıca yapılan çalışmalarda DTLS sadece önceden paylaşımlı anahtar ile kullanılmaktadır ve açık anahtar şifreleme ve Sertifika modlarının gelecekte kullanılabileceği hala net değildir [23].

DTLS Protokolü, UDP protokolü üzerinden çalıştığından el sıkışma mesajlarının iletileceği garanti edilmez, yani bu mesajlar iletim sırasında kaybolabilir veya farklı sırada iletilebilir. Bu da demektir ki kaybolan mesajlar el sıkışma sürecinin başarıyla tamamlanabilmesi için tekrar iletilmesi gerekmektedir. Bu sebeple DTLS mesajları gruplamak için akış (flight) yöntemini kullanmaktadır. Bu sayede mesajların yeniden sıralama ihtiyacı azalmaktadır. Ayrıca DoS ataklarını önlemek için de çerez mekanizmasını kullanmaktadır. Ancak bu durum hem değişim yapılan mesaj sayısını arttırmakta hem de durum makinesinin karmaşıklığını arttırmaktadır. Çerez mekanizması opsiyonel olsada, DoS koruması önemli bir durum olduğu için IoT uygulamalarında kullanımı bir gereklilik olacaktır [23].

DTLS kullanarak doğrulama işlemini gerçekleştirmenin bir çok yolu bulunmaktadır. Ön paylaşımlı anahtar metodu oldukça basit bir işlemdir. Sertifika doğrulama işlemi kompleks bir yapıdır çünkü değiştirilen sertifikaları, güvenilir üçüncü şahsı doğrulamayı ve sertifikanın zaman geçerliliğini kontrol etmeyi gerektirmektedir. Cihazda güvenilir bir zaman geçerliliği bulunmuyorsa, sertifikanın geçerlilik süresi kontrol edilemeyecektir.

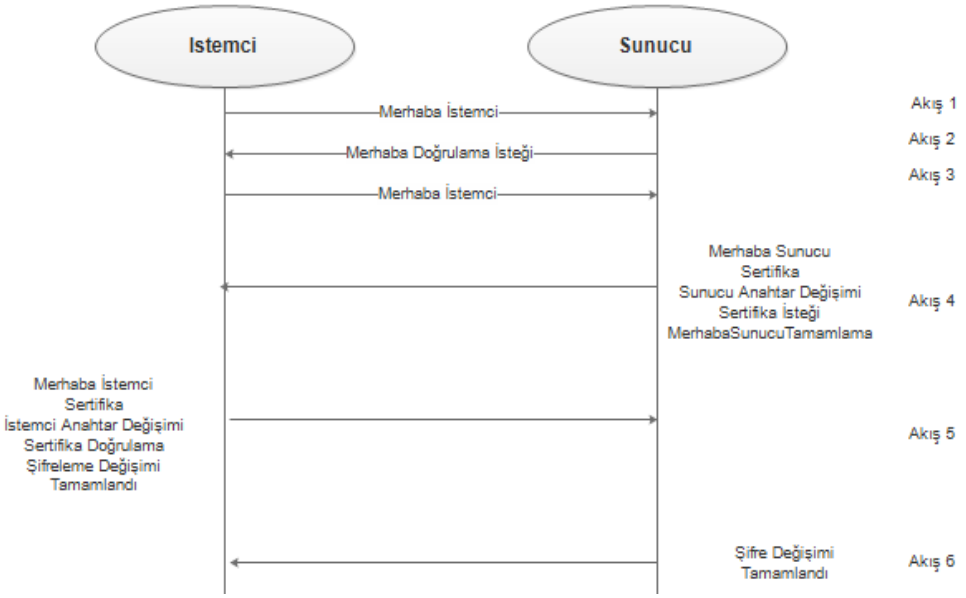
DTLS protokolüyle ilgili yapılan çalışmalar genel olarak kod boyutunun azaltılması yönelik olmuştur. Çünkü 6LoWPAN, CoAP, RPL gibi diğer protokollerin düzgün çalışabilmesi ve cihazın üretilme amacındaki fonksiyonelliği yerine getirebilmesi için DTLS'in kullandığı kod boyutunun azaltılması gerekmektedir. Uygulamaların büyük çoğunluğu önceden paylaşımlı anahtar yöntemini kullanmaktadır ve bu yöntem bile ortalama 16 KB ROM ve 4 KB RAM kullanmaktadır. Açık anahtar şifreleme ve sertifika şeması kullanıldığını düşünürsek kod boyutu ciddi şekilde artacaktır. Bu da cihazın fonksiyonelliğini yerine getirememesi demektir.

Simetrik şifreleme tabanlı gizlilik ve kimlik doğrulama işlemleri AES-CCM şifreleme kullanılarak rahat bir şekilde sağlanabilir. Dahası AES-CCM çoğu 802.15.4 cihazında yerleşik olarak bulunduğundan kod ihtiyacını azaltarak, hesaplama hızını ciddi oranda artırmaktadır. Asimetrik şifreleme tabanlı el sıkışma aşaması için, eliptik eğri şifreleme yöntemi ise kısıtlı cihazlar için oldukça kullanışlıdır. Eliptik eğride ise bir şifre paketi için seçilebilecek alan türleri (F_p, F_2^m) ve eğriler açısından bir çok seçenek vardır. Sertifika tabanlı kod blokları için ise sertifika imzalama algoritması eliptik eğri tabanlı seçilebilir. Bu seçimler ile kaynak kısıtlı çevrelerde el sıkışma aşamasında gerekli olan hesaplama miktarını ciddi manada azaltabiliriz.

TLS'te bulunan bazı uzantıları kısıtlı çevrelerde DTLS için kullanabiliriz. Örneğin, "Maksimum Parça Uzunluğu" uzantısı alt katmanlarda parçalanma miktarını azaltacak daha küçük parça boyutlarına olanak sağlar. "İstemci Sertifika URLs" uzantısı ise el sıkışma mesajlarında sertifika gönderme ihtiyacını azaltır, bant genişliği ve büyük sertifikalar nedeniyle parçalanmayı azaltmaktadır. Bu uzantılardan uygun olanlar seçildiğinde DTLS kısıtlı çevrelerde çok daha etkili şekilde kullanılacaktır.

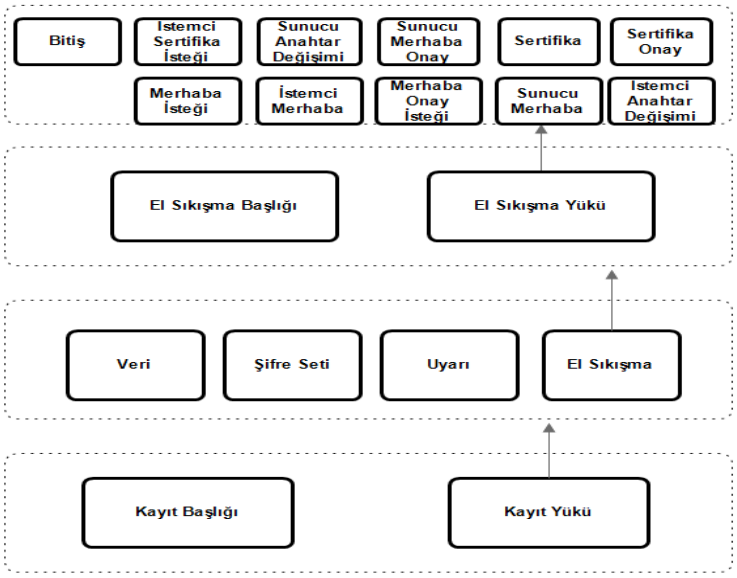
IoT uygulamalarının çoğunda kullanılması beklenmeyen bazı DTLS özelliklerini uygulamayarakta kaynaktan tasarruf edilebilir. Örneğin, DTLS'te daha önce kurulan bağlantının devam etmesini sağlayan RESUME protokolü ve Alert protokolünde hata sayısının azaltılması gibi özellikler kullanılmayabilir. Bunlar DTLS güvenilirliğine hiçbir etki yapmamaktadır [23].

Her DTLS mesajı tek bir iletim katmanı veri birimine yerleştirilmedir. Fakat el sıkışma mesajları maksimum kayıt boyutundan büyüktür. Bu nedenle DTLS, kayıtların numaraları üzerinden el sıkışma mesajlarının parçalanması için bir mekanizma tanımlamıştır ve bu sayede her mesaj ayrı olarak iletilebilmektedir. Bir el sıkışma mesajı iletilirken, gönderici mesajı N adet sıralı veri aralığına böler. Bu aralık el sıkışma mesajları için iletilecek en küçük çerçeve boyutundan küçük olmalıdır ve N mesajın toplamı el sıkışma mesajını oluşturmalıdır. Ayrıca bu aralıklar çakışmamalıdır. Aynı mesaj dizi değerine sahip N mesaj birleşerek orijinal el sıkışma mesajını oluşturur. Her yeni mesaj parça sırası ve parça uzunluğu ile etiketlendirilir. Bir DTLS uygulamasında el sıkışma mesajı alındığında bütün mesajlar gelene kadar o mesajı korumalıdır.



Şekil 2.19. DTLS el sıkışma protokolü

Uygulamaların korunması amacıyla güvenli bir DTLS bağlantısı kurmak için istemci ve sunucu arasında 15 mesajdan oluşan bir bağlantı süreci Şekil 2.19'daki gibi gerçekleşir. Bu el sıkışma mesajları 6 mesaj akışı içerisinde gerçekleşir. Birinci ve ikinci mesajlar DoS koruması için geri dönüş testi niteliğinde mesajlardır. El sıkışma mesajlarının ana bölümü üç, dört ve beşinci mesajlardır. Burada taraflar şifreleme operasyonunda kullanacakları kod blokları üzerinde anlaşırlar. Daha sonra taraflar birbirini doğrular ve uygulama verisi ve el sıkışma mesajlarının korunması için anahtar anlaşma sürecini gerçekleştirirler. Beşinci ve altıncı mesajlar süreci bitirme mesajıdır ve bu mesajlar iki tarafa da el sıkışma sürecinin başarıyla gerçekleştiğini bildirir. Taraflar tüm el sıkışma sürecini için bir hash hesaplar ve bunu şifreli formda tüm iletişim kurduğu birimlere gönderir. Başarılı bir çözümleme ve hash doğrulamadan sonra üretilen oturum anahtarı ve tüm el sıkışma sürecinin bütünlüğü onaylanmış olur. Daha sonra taraflar üretilmiş olan oturum anahtarını kullanarak uygulama verilerini birbirlerine iletebilirler [24].



Şekil 2.20. Güvenli DTLS'in paket düzeni

Şekil 2.20'de görüldüğü gibi DTLS protokolü alt katman ve üst katman olmak üzere iki katmandan oluşmaktadır. Alt katman kayıt protokolden ve üst katman ise el sıkışma, uyarı, şifre bloğu değişimi ve uygulama verisinden oluşmaktadır. Şifre bloğu değişimi, el sıkışma sürecinde kullanılır ve sonraki mesajların yeni anlaşılan güvenlik anahtarı ve şifre bloğu ile koruması gerektiğini belirtir. Uyarı protokolü, DTLS uçları arasında hata mesajları için kullanılır. Kayıt protokolü, üst katman protokolleri için taşıyıcı konumundadır. Kayıt başlığı, diğerlerinin içerik türünü ve parça alanlarını içermektedir. İçerik türündeki değere göre el sıkışma protokolü, uyarı protokolü, şifre bloğu değişim prokolü ve uygulama protokolünden birini içerir. Kayıt başlığı ise el sıkışma işlemi tamamlandıktan sonra üst katman protokollerini veya uygulama verilerini şifreli bir biçimde korumakla görevlidir [22].

2.2.3 Düşük Güçlü Kablosuz Kişisel Alan Ağlarında IPv6 (6LoWPAN)

6LowPAN protokolü, IEEE 802.15.4 fiziksel katmanına sahip cihazların internete dahil olması için IETF tarafından oluşturulmuş bir adaptasyon katmanıdır. IEEE 802.15.4 katmanı ile IPv6 katmanları arasında bir uyum katmanı olarak çalışmaktadır. IEEE fiziksel katmanının gönderebileceği maksimum transfer boyutu (MTU) 127 byte ve IPv6 paketlerinin gönderebileceği minimum transfer boyutu (MTU) 1280 byte olması bir

adaptasyon katmanının kullanımını zorunlu hale getirmiştir. Bu sebeple geliştirilen 6LoWPAN katmanı IP paketlerini küçük parçalara ayırıp sıkıştırarak iletmektedir. 6LoWPAN protokolünün sahip olduğu özellikler şu şekildedir;

1-Nesnelerinin internetinin ihtiyaç duyduğu kod ve bellek kullanımının minimum olması,

2-Uzun batarya ömrü,

3-Verimli IP ve TCP/UDP başlık sıkıştırma,

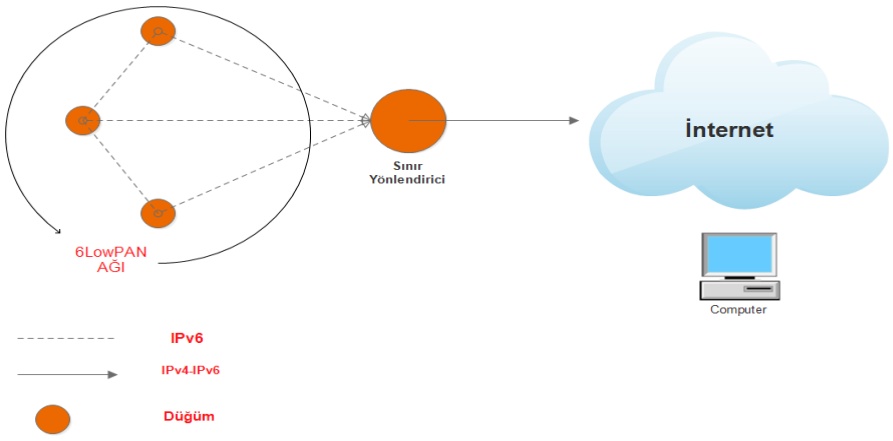
4-16 ve 64 bitlik 802.15.4 adreslemeyi destekleme,

5-Komşu keşif metodu ile otomatik ağ konfigürasyonu,

6-Tekli, çoklu iletişim ve yayın desteği,

7-Düşük güç ve maliyet gereksinimi,

8-Başlık sıkıştırma, IPv6 paketlerini parçalama ve birleştirme [8].



Şekil 2.21. 6LoWPAN ağlarının yapısı

2.2.4 Düşük Güçlü ve Kayıplı Ağlar İçin IPv6 Yönlendirme Protokolü (RPL)

RPL protokolü, nesnelerin interneti cihazları için geliştirilmiş bir yönlendirme protokolüdür. Düşük kontrol trafiği gerektiren bir yönlendirme protokolüdür. Çoklu atlamalı (multi hop) ağ topolojilerinde sıklıkla oluşan hareketlilikler nedeniyle yönlendirme işlemi yapmak zorlaşmaktadır. Nesnelerin interneti ağına dışarıdan erişmek ya da düğümlerin birbiriyle iletişim kurması ve ağın yönetimi RPL tarafından gerçekleştirilmektedir [8].

2.4 Contiki İşletim Sistemi ve Cooja Simülasyon Programı

Bu bölümde Contiki işletim sisteminin ve Cooja simülasyon programının özelliklerinden bahsedilecek ve temel kavramlar açıklanarak örnek simülasyonlar gösterilecektir.

Contiki, algılayıcı cihazlar için geliştirilmiş açık kaynak kodlu bir işletim sistemidir. Düşük güç ve düşük maliyetli mikrodenetleyicilerin (Skymote, Wismote) internete açılmasını sağlamaktadır.

Contiki işletim sistemi, internet iletişimini düşük maliyetle gerçekleştirmesi nesnelerin internetinde bulunan kaynak kısıtlı cihazlar için önemli bir özelliktir. IPv6 ve IPv4 desteği vardır. Ayrıca nesnelerin interneti için geliştirilen 6LoWPAN, RPL, CoAP protokollerini desteklemektedir. Contiki uygulamaları standart C dilinde yazılmıştır. Cooja simülatörü sayesinde de yazılan kod donanıma yazılmadan önce simülasyonu gerçekleştirilebilmektedir.

2.4.1 Contiki İşletim Sisteminin Yüklenmesi

Contiki ortamında çalışmak için gerekli olan araçlar;

1- Contiki kaynak kodu http://www.contiki-os.org/ adresinden indirilebilir.

2- Sanal Makine (VMware) https://www.vmware.com adresinden indirilebilir.

Contiki'nin son geliştirilen sürümü 2015 yılında çıkarılmış olan contiki-3.0 sürümüdür. Bu çalışma kapsamında da bu sürümü kullanılacaktır.

Yükleme işlemi tamamlandıktan sonra sanal makine VMware Workstation programı açılarak "Open a Virtual Machine" seçilir ve indirilen Contiki kaynak kodu gösterilir. Daha sonra "Play Virtual Machine" seçeneğine tıklanarak simülasyon başlatılır. Açılırken bir şifre istenecektir. Standart şifre olarak "user" girilir ve devam edilir. Bu basamaklar tamamlandığında Contiki işletim sistemine giriş yapılmış olur. Ekran görüntüsü Şekil 2.22'deki gibidir.



Şekil 2.22. Contiki işletim sistemi

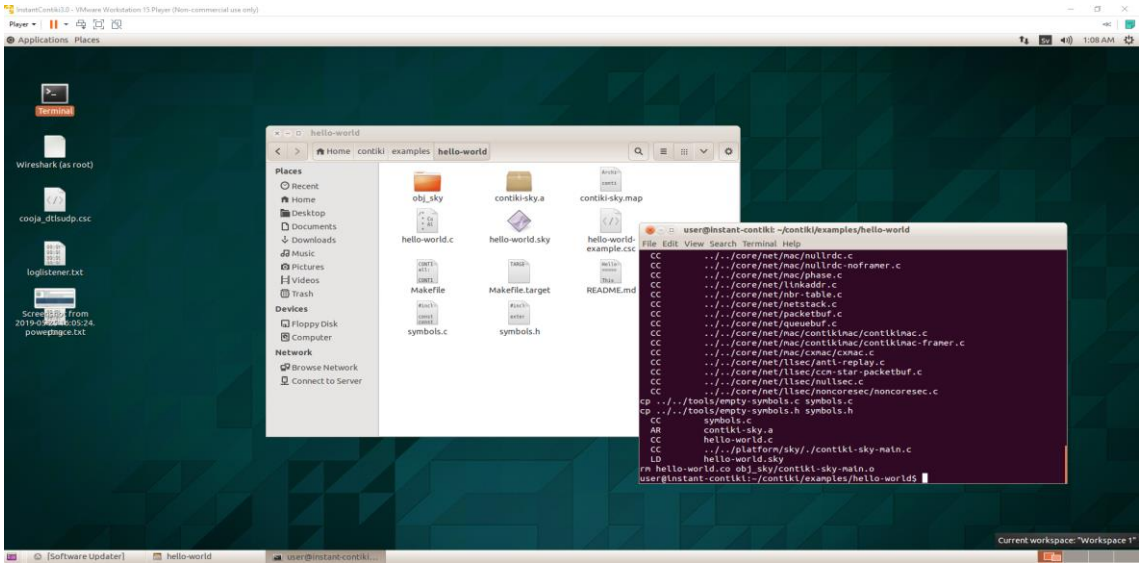
2.4.2 Terminal Üzerinden “hello world” Uygulaması

Masaüstünde bulunan terminal açılır ve aşağıdaki komutları sırasıyla uygulanır.

-cd contiki/examples/hello-world komutuyla hello-world

-make TARGET=sky savetarget komutuyla hedef düğüm olarak sky düğümünü seçilir ve bu hedef kaydedilir.

-make komutuyla kod çalıştırılır ve sonuç Şekil 2.23’teki gibidir.



Şekil 2.23. Contiki hello world örneği

-gedit hello-world.c komutuyla kod üzerinde istenilen değişiklik gerçekleştirilebilir.

Bu uygulamanın ve diğer tüm uygulamaların standart kod yapısı Şekil 2.24'teki gibidir.

```
PROCESS(hello_world_process, "Hello world process");
// gerçekleştirilen işlemin adı
AUTOSTART_PROCESSES(&hello_world_process); //Önyükleme adımı tamamlandığında kod başlatılacağını belirtir.
   PROCESS_THREAD(hello_world_process, ev, data)  //işlemin içeriğinin belirtildiği bölüm
   {
    PROCESS_BEGIN(); //işlemin başlagıcı
    printf("Hello, world\n");  //yapılacak işlemler
    PROCESS_END();    //işlemin sonu
   }

   Uygulamaların derlenmesi için ayrıca bir makefile dosyasına ihtiyaç duyulur. Onunda yapısı aşağıdaki gibidir.
   CONTIKI_PROJECT = hello-world //derlenecek olan uygulamayı belirtir.
   all:  $(CONTIKI_PROJECT)// all kullanıldığında tüm kod derlenecek anlamına gelir.
   CONTIKI = ../..   //contiki ana dosyasının adresi belirtir.
   include   $(CONTIKI)/Makefile.include  //contiki sisteminin makefile dosyasını ekler.
```

Şekil 2.24. Contiki standart kod yapısı

2.4.3 Cooja Simülasyonu

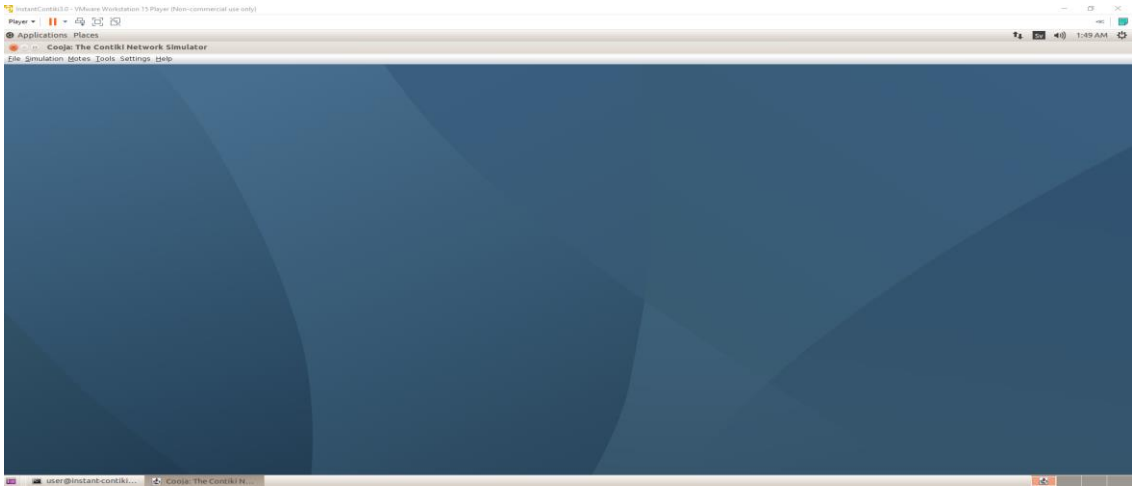
Cooja, sensör düğümlerin bilgisayar ortamında simülasyonun yapılabildiği bir ağ simülatörüdür. Düğüm tarafından yürütülen kod, fiziksel düğümlere yüklenebilecek aynı ürün yazılımıdır.

Cooja simülasyonunu çalıştırmak için sırasıyla aşağıdaki komutlar terminale yazılır.

-cd contiki/tools/cooja

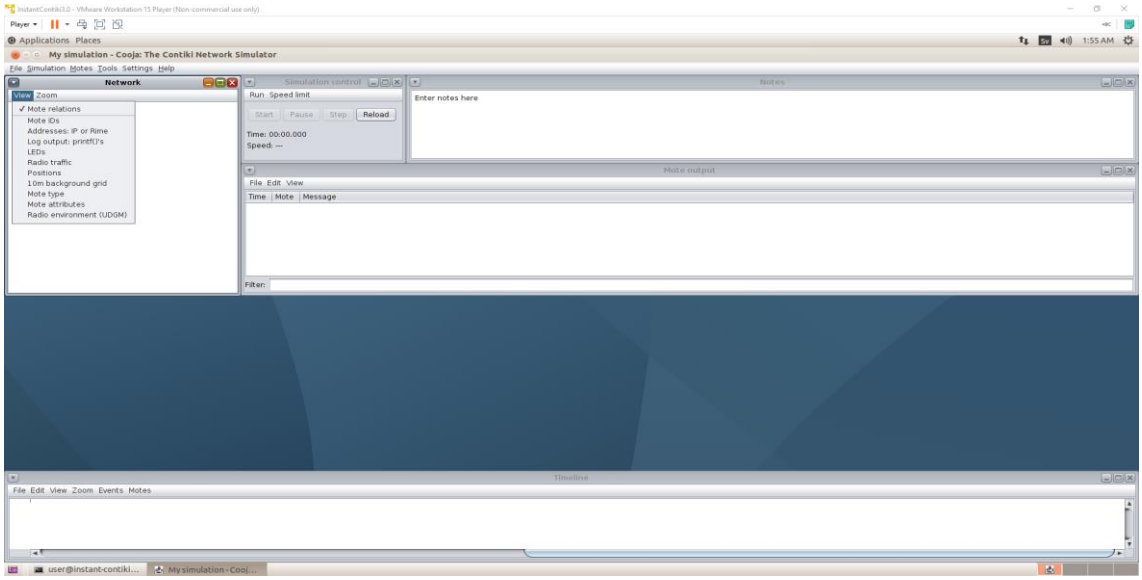
-ant run

Cooja yazılımı başlayacak ve karşınıza Şekil 2.25'teki ekran gelecektir.



Şekil 2.25. Cooja simülasyon programı

File > *New Simulation* adımları ile yeni bir simülasyon başlatılır. Simülasyonun ismi girildikten sonra *Create* seçeneğine tıklanarak simülasyon başlatılır. İşlem sonucunda Şekil 2.26'daki ekran ile karşılaşılacaktır.



Şekil 2.26. Cooja simülatör ana ekranı

Network: Ağın fiziksel düzenini gösterir. View penceresi ile düğüm ve ağ ile ilgili ayrıntıların gösterilmesi sağlanır (düğüm id, düğüm çeşidi, düğüm çıktısı, ağ trafiği vb). Buraya düğüm fiziksel olarak konularak ağın yapısına göre düzenlenebilir.

Simulation Control: Simülasyonun başlatılıp durdurulması ve yüklenmesinin gerçekleştirildiği bölümdür. Ayrıca simülasyon hızı da kontrol edilebilir.

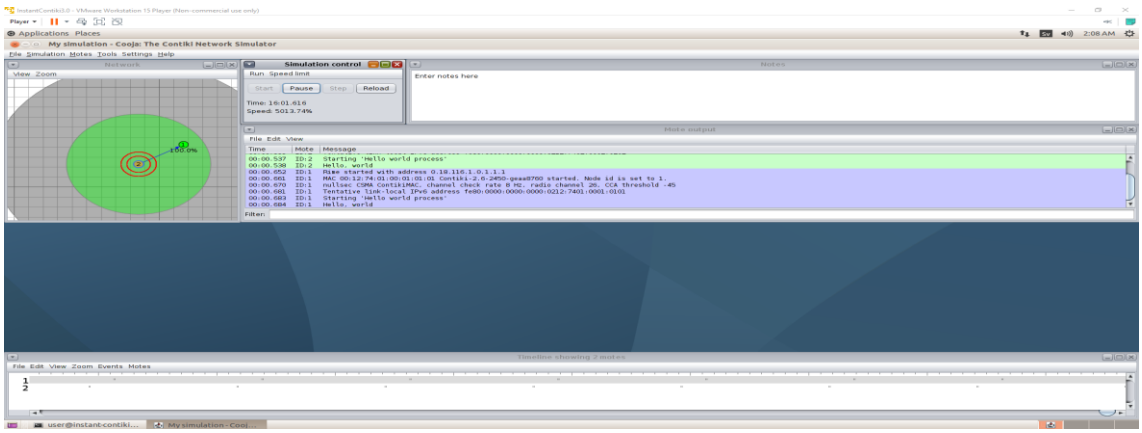
Mote Output: Tüm düğümlerden elde edilen ekran çıktılarını gösteren bölümdür. Ancak Filter alanından elde edilen çıktılar ayıklanabilir. Örneğin sadece 1. düğümle ilgili çıktılar isteniyorsa ona göre filtreleme yapılabilir.

Timeline: Zaman çizelgelerini, her karede simülasyonun zaman çizelgesi üzerinde gerçekleşen olayları gösterir. Bu çıktılar *file>save to file* ile kaydedilebilir.

Notes: Simülasyon ile ilgili notların alınabildiği bölümdür.

2.4.3.1 Hello World Simülasyonu

Örnek bir simülasyon gerçekleştirmek için öncellikle *Motes* > *Add Motes* > *Create new mote type* > *Sky Mote* adımları izlenir. Açılan pencerede *Browse* sekmesinde istenilen kod düğüme yüklenerek derlenebilir. *Contiki / examples / hello-world* klasöründe bulunan *hello-world.c* dosyası düğüme yüklenerek kod derlenmiştir. Burada kodda herhangi bir hata olması durumunda *Create* butonu aktif olmayacaktır. *Create* butonuna tıklanarak düğüm eklenir. Daha sonra *Start* butonuna basıldığında simülasyon Şekil 2.27'deki gibi başlayacaktır.

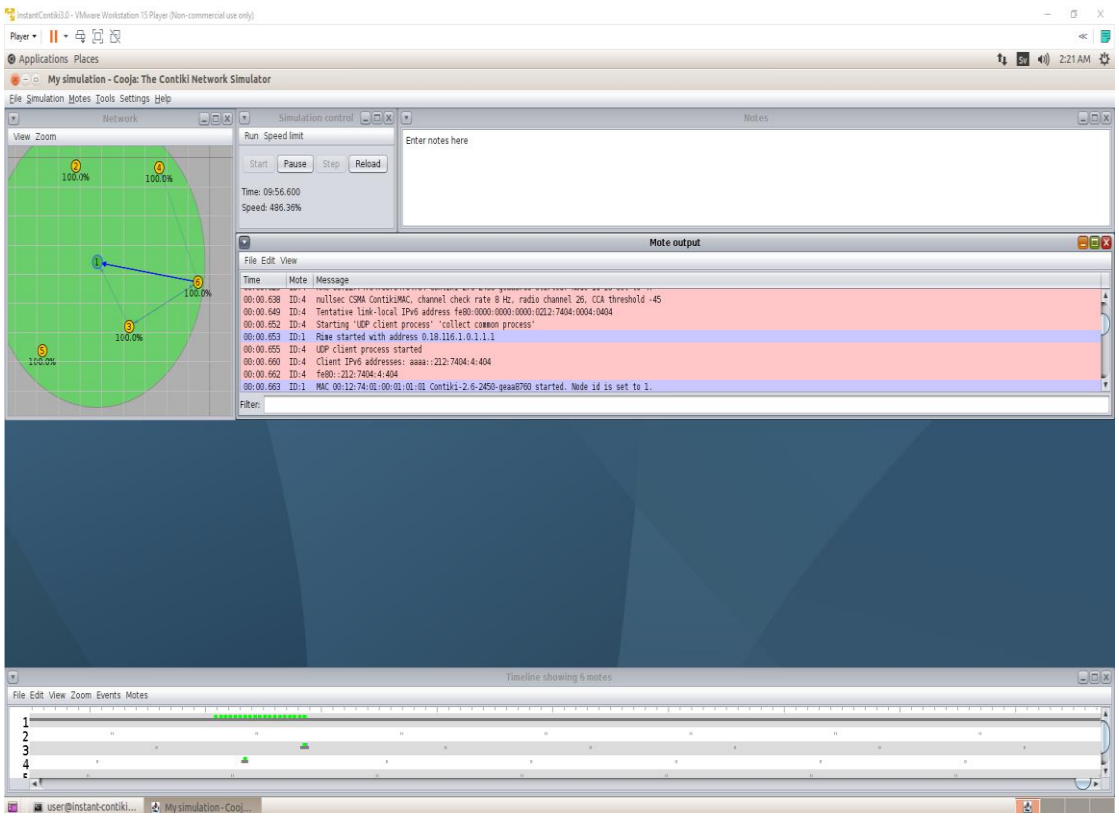


Şekil 2.27. Cooja hello world simülasyonu

2.4.3.2 IPv6-RPL-UDP Örneği

Nesnelerin internetinin temel protokolleri olan IPv6, RPL ve UDP'yi kullanarak düğümler arası iletişimi gösteren örnek bir simülasyon için aşağıdaki adımlar izlenir;

Motes > *Add Motes* > *Create new mote type* > *Sky Mote* adımları izlenir. Browse seçeneğine tıklanarak *user/ contiki / examples / ipv6 / rpl-collect* klasöründeki *udp-sink.c* kodu düğüme eklenir. Daha sonra *Clean* seçeneğine tıklanarak önceki derlemelerden kalan kod düğümden temizlenerek *Compile* seçeneğine tıklanır ve kod düğüme yüklenir. Düğüm sayısı olarak 1 seçilir. Aynı adımlar tekrarlanarak yeni bir düğüm eklenir ve klasör içerisinde bulunan udp-sender.c kodunu düğüme yüklenir. Düğüm sayısı olarak 4 seçilir. *Network* alanından düğümlerin topolojisi istenilen şekilde ayarlanabilir. Aynı şekilde *View* sekmesinden de düğüm ve ağ ile ilgili istenilen ayarlar eklenebilir. Oluşturulan ağ yapısı şekil 2.28'deki gibidir.

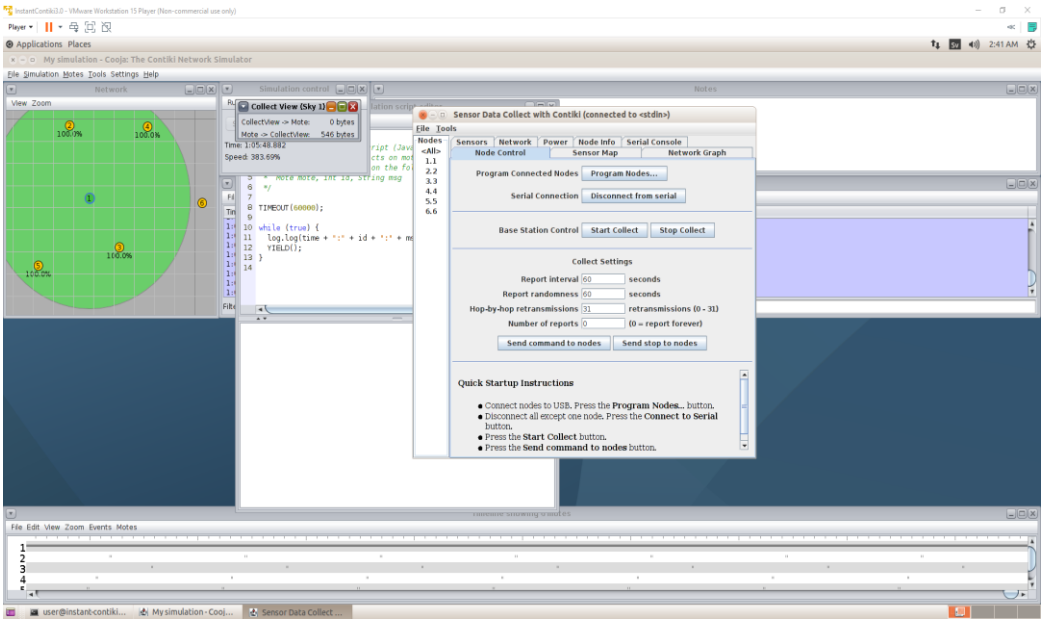


Şekil 2.28. IPv6-RPL-UDP protokolleri ile düğümler arası iletişim

2.3.4.3 Algılayıcı Veri Toplama

Cooja, düğümlerden veri toplamak için farklı araçlara sahiptir. Yukarıda gerçekleştirilen örneklerde gönderilen verileri, döngü sayıları, düğümlerin tükettiği ortalama güç değerleri mW cinsinden görüntülenebilir.

Bunun için düğüme sağ tıklanarak *Mote Tools for Sky1* > *Collect View* seçeneğine tıklanır. Açılan pencerede farklı ayrıntılar görüntülenmektedir. Şekil 2.29'daki görüldüğü gibi *Start Collect* seçeneğine tıklanarak veriler elde edilmeye başlanır. Örneğin, *Power* seçeneğine tıklanıldığında düğümlerin tükettiği ortalama enerji miktarı elde edilebilir, *Sensors* seçeneğinden düğümlerin anlık sıcaklık verisi alınabilir, *Radio Duty Cycle* seçeneğinden ise düğümlerin döngü sayısını elde edilebilir.



Şekil 2.29. Düğümlerden veri toplama

2.3.4.4 Sınır Yönlendirici ile CoAP Protokolü Örneği

Motes > *Add Motes* > *Create new mote type* > *Sky Mote* adımları izlenir. Browse seçeneğine tıklayarak *contiki* / *examples* / *IPv6* / *rpl-border-router* klasöründe bulunan *border-router.c* kodunu düğüme yüklenir. Tekrar aynı adımları izleyerek bu sefer düğüme *examples* / *er-rest-example* klasöründeki *er-example-server.c* kodu yüklenir. Simülasyonu başlatmadan önce sınır yönlendirici ile sunucu arasında bağlantı kurulması

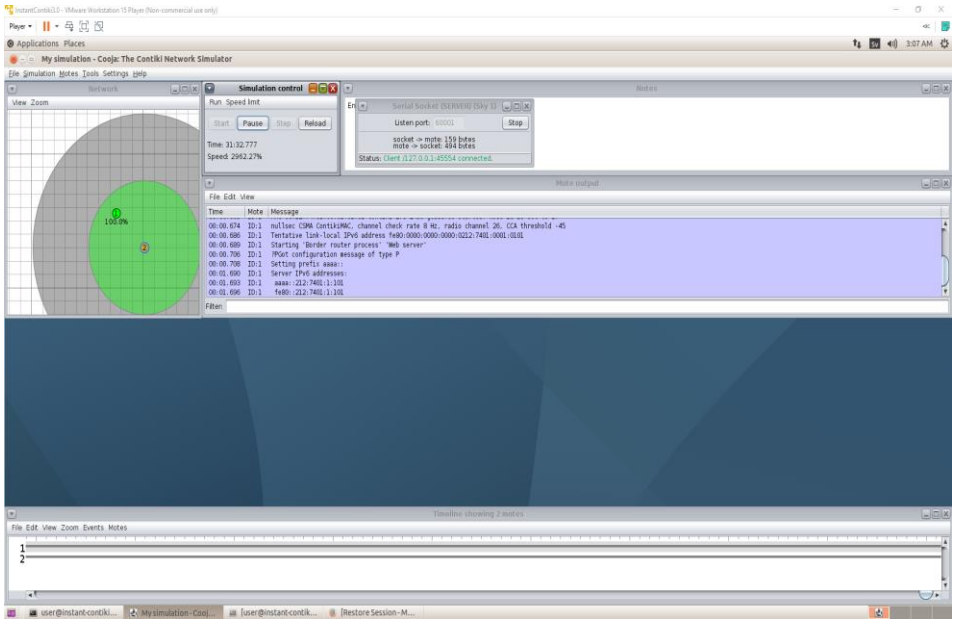
gerekmektedir. Bunun için öncelikle sınır yönlendirici (*border-router.c*) kodu eklenen düğüme sağ tıklayarak *Mote Tools for Sky* > *Serial Socket (server)* seçilip açılan pencereden *Start*'a tıklanır. Daha sonra terminal açılarak sırasıyla aşağıda yazılan komutları girilir.

cd contiki/ examples / ipv6/ rpl-border-router

pwd

make connect-router-cooja

Komutlarını sırasıyla çalıştırdıktan sonra şekil 2.30'da görüldüğü gibi bağlantı sağlanmış olur.



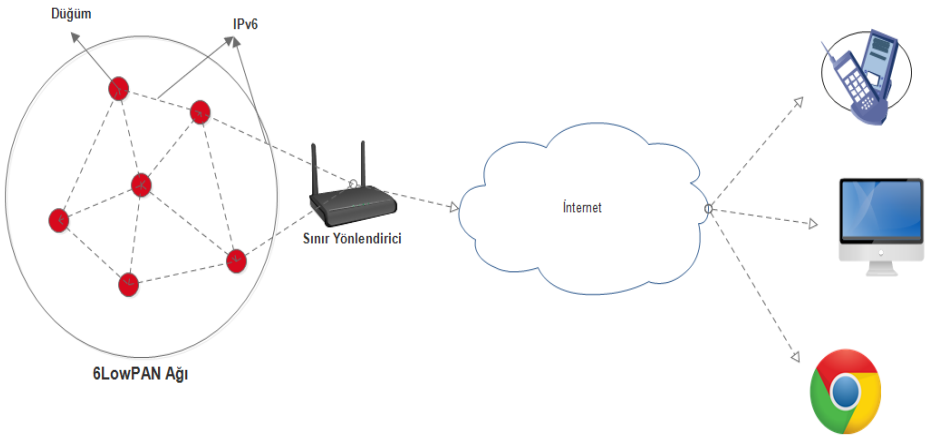
Şekil 2.30 CoAP protokolü aracılığı ile sınır yönlendiri ve sunucu iletişimi

Sonrasında Application > Internet > Firefox tıklanarak tarayıcı açılır ve arama kısmına coap://[ipv6 adresi] yazılır. Açılan pencereden CoAP protokolü aracılığı ile GET ve POST metodlarını kullanarak düğüme veri gönderip alınabilir. Düğümlerin IPv6 adresi düğüm üzerinden View sekmesi aracılığı ile veya terminalden elde edilebilir.

BÖLÜM 3. ELİPTİK EĞRİ KULLANARAK DTLS KATMANINDA EL SIKIŞMA AŞAMSINDA KİMLİK DOĞRULAMANIN GERÇEKLEŞTİRİLMESİ

3.1 Sistemin Tanıtımı

Bu tez kapsamında nesnelerin internetinde var olan güvenlik probleminin çözümü için eliptik eğri şifrelemeye dayalı kimlik doğrulama ve anahtar değişimi gerçekleştirilmiştir. İletim protokolü olarak asenkron olarak çalışan UDP seçilmiştir çünkü TCP protokolü sisteme ağır yük bindirmekte ve bu da kaynak kısıtına sahip cihazlar için büyük problem oluşturmaktadır. Güvenli iletişimin sağlanması için DTLS protokolü kullanılmıştır. DTLS için ise açık kaynak kodlu TinyDTLS kütüphanesi tercih edilmiştir. Kullanılan eliptik eğri dijital imza algoritması ve Diffie-Hellman anahtar değişim algoritması el sıkışma aşamasında devreye girmekte ve asimetrik şifrelemeden kaynaklanan yükü azaltmaktadır. Bu çalışma ile eliptik eğri şifrelemenin nesnelerin internetinde kullanılan kaynak kısıtlı cihazlar için kullanılabileceğini ve düğümlerin uçtan uca haberleşebileceğini göstermek amacıyla gerçekleştirilmiştir. Çalışmanın genel yapısı Şekil 3.1'deki gibidir.



Şekil 3.1. DTLS ile uçtan uca şifreleme

3.2 Düğüm Seçimi

Eliptik eğri şifreleme ve DTLS protokolünün büyük miktarda RAM ve ROM tüketmesi, işlemciye yoğun yük getirmesi nedeniyle kullanılacak düğümün seçilmesi önem arz etmektedir. Kod boyutu dikkate alındığında Skymote düğümü ve Z1 düğümüyle gerçekleştirilen çalışmalarda RAM ve ROM miktarları düşük olması sebebiyle çalışmamızda Wismote düğümünü kullanılmıştır ve Cooja simülasyon programı kullanılarak TinyDTLS kütüphanesi ile simülasyonunu gerçekleştirilmiştir. Çizelge 3.1'de düğümlerin sahip olduğu hafıza miktarları yer almaktadır.

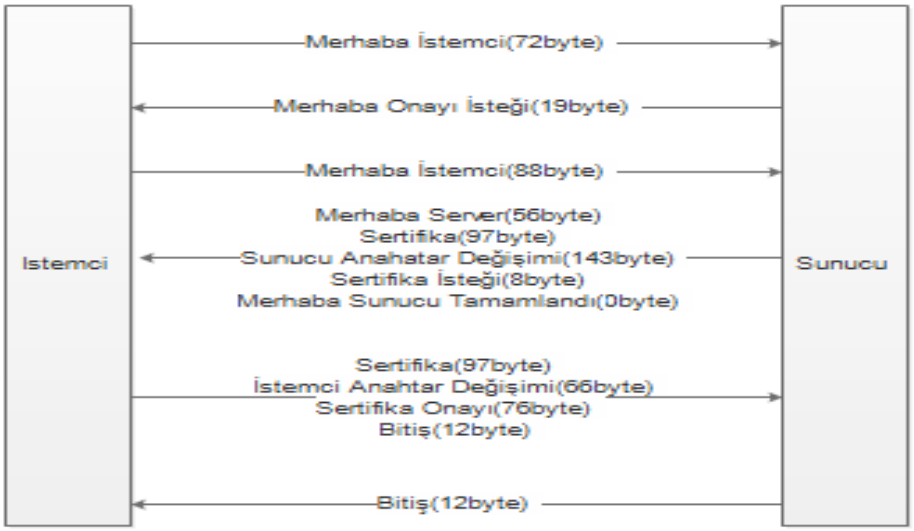
Çizelge 3.1. Sensör düğümlerin hafıza değerleri

	Skymote	Z1mote	Wismote
Ram	10KB	8KB	16KB
Rom	48KB	92KB	128KB
Mikrodenetleyici	MSP430	MSP430	MSP430

3.3 DTLS Eliptik Eğri Şifreleme ile İstemci Sunucu Haberleşmesi

Contiki işletim sisteminde Cooja simülasyon programında Wismote düğümü ile TinyDTLS kütüphanesi kullanılarak gerçekleştirilen el sıkışma aşaması ve anahtar değişimi simülasyonunun dizaynı Şekil 3.2'deki gibidir. Düğümlerin haberleşme aşaması 1. Akış olan istemcinin Merhaba İstemci mesajı ile başlamaktadır. Bu mesajın içerisinde oturum ID'si, başlık sıkıştırma yöntemi, şifre blokları yer almaktadır. Bu mesaja karşılık olarak 2. Akış sunucu tarafından Merhaba Onay İsteği mesajı gönderilir. Bu mesajın içeriğinde çerez denilen ve DoS saldırılarına karşılık güvenlik için kullanılan oturum bilgisi mesajı gönderilir. 3. Akış olarak istemci ilk gönderdiği mesajı çerez dosyasını ekleyerek tekrar gönderir ve iletişim başlamış olur. 4 Akışta sunucu tarafından Merhaba Server mesajı ile kendi bilgilerini, Sertifika mesajı ile kendi kimliği doğrulamak amacıyla gerekli olan bilgileri, Sunucu Anahtar Değişimi mesajı ile kendi açık anahtar bilgisini ve Sertifika İsteği mesajı ile de karşı tarafın kimliğini doğrulamak amacıyla gerekli bilgileri

istemektedir. 5. Akış mesajında istemci Sertifika mesajı ile kendi kimliğinin doğrulanması için gerekli olan bilgileri, İstemci anahtar değişimi mesajı ile kendi açık anahtarını ve Sertifika Onayı mesajı ile karşı tarafın kimliğinin doğrulanma bilgisini sunucuya iletir. Bitiş mesajı ile de kimlik doğrulama ve anahtar değişim işlemleri gerçekleşmiş olur.



Şekil 3.2. DTLS el sıkışma aşamasındaki mimari

```
File  Edit  View  Search  Terminal  Help
ramprof
TARGET not defined, using target 'native'
  CC        dtls-server.c
  LD        dtls-server.native
contiki-native.a(dtls.o): In function `dtls_send_client_hello':
/home/user/contiki/apps/tinydtls/examples/contiki/../../../../apps/tinydtls/dtls
.c:2449: undefined reference to `assert'
contiki-native.a(dtls.o): In function `clear_hs_hash':
/home/user/contiki/apps/tinydtls/examples/contiki/../../../../apps/tinydtls/dtls
.c:1088: undefined reference to `assert'
contiki-native.a(dtls.o): In function `dtls_send_certificate_ecdsa':
/home/user/contiki/apps/tinydtls/examples/contiki/../../../../apps/tinydtls/dtls
.c:1860: undefined reference to `assert'
contiki-native.a(dtls.o): In function `dtls_send_finished':
/home/user/contiki/apps/tinydtls/examples/contiki/../../../../apps/tinydtls/dtls
.c:2301: undefined reference to `assert'
contiki-native.a(dtls.o): In function `dtls_send_certificate_verify_ecdh':
/home/user/contiki/apps/tinydtls/examples/contiki/../../../../apps/tinydtls/dtls
.c:2269: undefined reference to `assert'
contiki-native.a(dtls.o):/home/user/contiki/apps/tinydtls/examples/contiki/../..
/../../apps/tinydtls/dtls.c:2561: more undefined references to `assert' follow
contiki-native.a(dtls_time.o): In function `dtls_clock_init':
/home/user/contiki/apps/tinydtls/examples/contiki/../../../../apps/tinydtls/dtls
_time.c:40: undefined reference to `clock_init'
```

Şekil 3.3. Eliptik eğri ile DTLS el sıkışma aşamasının gerçekleştirilmesi

3.4 RAM ROM Kullanımı

Nesnelerin interneti, sensör cihazlar ciddi oranda kaynak kısıtına sahiptir. Bu sebeple yapılan çalışmalarda göz önünde bulundurulması gereken en önemli kriterlerden biri budur. Özellikle asimetrik şifreleme algoritmaları ciddi oranda sisteme yük bindirmekte RAM ve ROM miktarını, işlemci gücünü önemli oranda tüketmekte ve cihazlar üretilme amacını yerine getirememektedir. Bu nedenle yapılan çalışmada RAM, ROM kullanım miktarı ve sensör cihazın tükettiği enerji miktarı ölçülmüştür. Contiki işletim sisteminde yapılan bir çalışmanın RAM ve ROM miktarını ölçme işlemi şu şekilde yapılmaktadır.

size derlenen-kod.düğüm çeşidi

Kodun kaplamış olduğu hafıza miktarı kullanacağımız düğüm çeşidini doğrudan etkilemektedir. Buna göre çalışmamızda kullandığımız TinyDTLS kodunun kaplamış olduğu hafıza miktarı Çizelge 3.2'deki gibidir.

Çizelge 3.2. Sistemin RAM ve ROM gereksinimi

Text	Data	Bss	Dec	File
83885 byte	422B	10140B	94447B	Client
83227 byte	346B	9628B	93201B	Server

Tabloda gösterilen verilerden kullanılan hafıza miktarlarını görebiliriz. Verilen değerler byte cinsindendir. Uygulamanın kullandığı ROM miktarı "Text+Data" verisinden elde edilir. Kullanılan RAM miktarı ise "Data+Bss" değerinden elde edilir. Bu durumda istemcinin kullandığı RAM miktarı 84307 byte ve ROM miktarı ise 10562 byte'tır. Sunucunun kullandığı ROM miktarı 83573 byte ve RAM miktarı ise 9974 byte'tır.

Yine bazı önemli fonksiyonların RAM ve ROM değerlerini elde etmek için aşağıda belirtilen arac kullanılabilir. Bu şekilde elde edilen değerler ile hangi aşamada ne kadar kaynak kullanıldığı öğrenibilir. Daha detaylı inceleme yapılmak istenirse

“msp430-objdump” aracını kullanılabilir. Çizelge 3.3’te DTLS el sıkışma aşamasında kullanılan hafıza değerleri gösterilmiştir.

Make TARGET = wismote derlenen-kod.ramprof

Make TARGET = wismote derlenen-kod.flashprof

Çizelge 3.3. İstemcinin harcadığı ROM değerleri

Aşama	ROM Miktarı-İstemci
handle_handshake_msg	5,204B
powertrace_print	2,058B
dtls_handle_message	1,818B
dtls_prepare_record	704B
dtls_send_client_hello	596B

Çizelge 3.4. İstemcinin harcadığı RAM değerleri

Aşama	RAM Miktarı-İstemci
ecc_order	32B
ecdsa_priv_key	32B
ecc_order	36B
handshake_storage	342B

DTLS’in el sıkışma aşaması en çok kaynak harcanan aşamadır. Özellikle kimlik doğrulama ve anahtar değişimi aşamalarında ciddi kaynak tüketilmektedir. Simetrik ve

asimetrik şifrelemede, el sıkışma aşamalarında tüketilen hafıza miktarlarında ciddi fark bulunmaktadır. Çizelge 3.5'te el sıkışma aşamasında tüketilen hafıza miktarları gösterilmiştir.

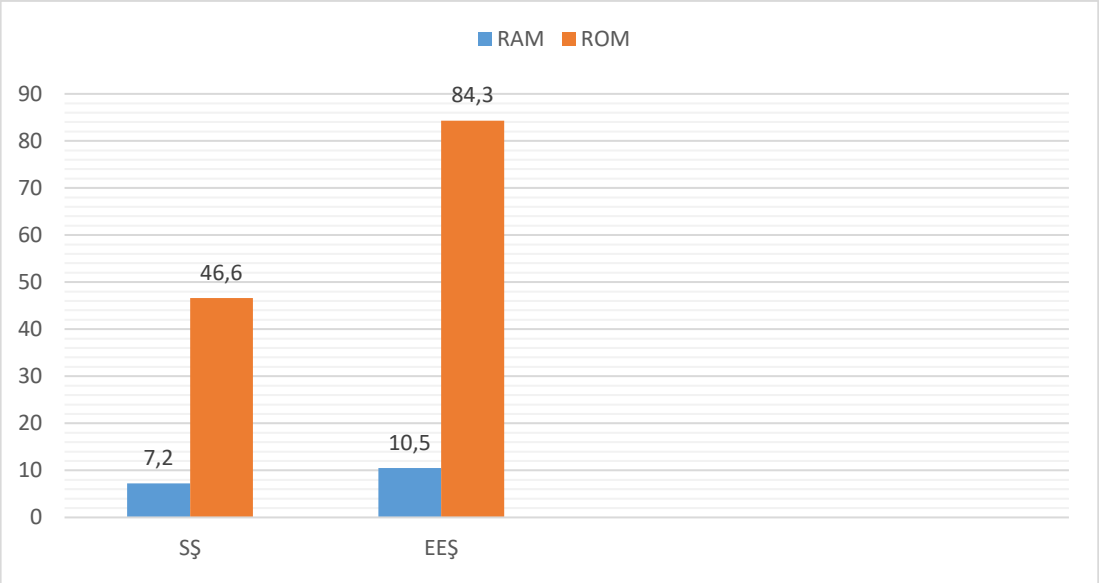
Çizelge 3.5. DTLS el sıkışma aşamasında ECC ve simetrik şifreleme modlarında tüketilen hafıza miktarı

Aşama	Sunucu	İstemci	Toplam
DTLS-SŞ	69B	129B	198B
DTLS-EEŞ	408B	441B	849B

Simetrik şifreleme modu bize gösteriyor ki asimetrik şifreleme yöntemi eliptik eğri şifreleme kullanılsa dahi ciddi hafıza tüketmekte ve işlemciye yük bindirmektedir. Özellikle Sınıf-0 ve Sınıf-1 cihazlar için önemli problem oluşturmaktadır. Çizelge 3.6'da ise Wismote simetrik şifreleme modunda tükettiği RAM ve ROM miktarları gösterilmektedir.

Çizelge 3.6. Wismote simetrik şifreleme modunda hafıza tüketimi

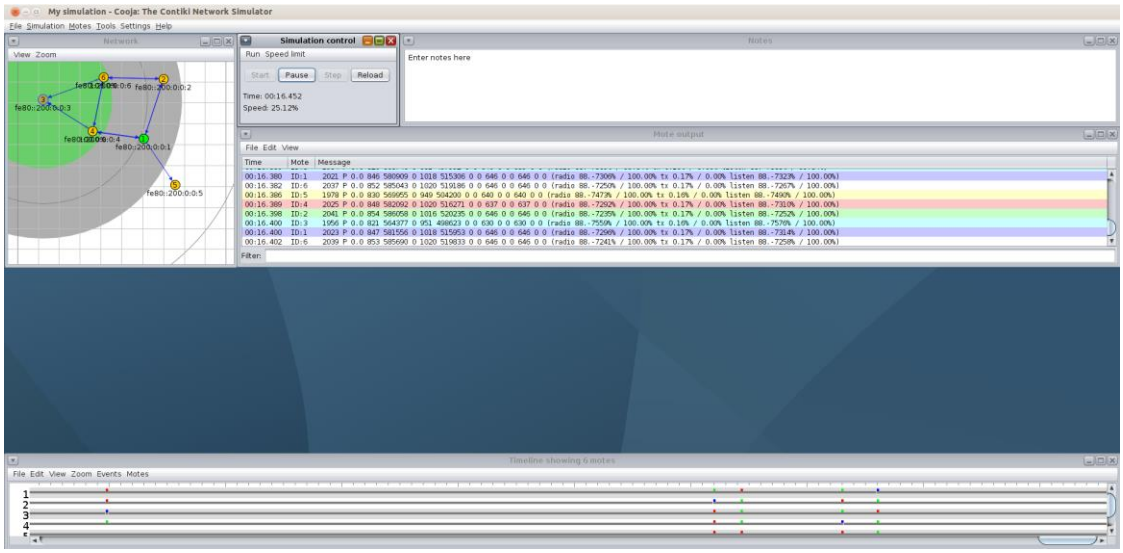
Rom Tüketimi	Ram Tüketimi
46,6 Kb	7,2 Kb



Şekil 3.4. DTLS'in EEŞ ve Simetrik Şifreleme modlarının kullandığı toplam hafıza miktarının KB cinsinden karşılaştırılması

3.5 Enerji Tüketimi

Cooja benzetim ortamında gerçeklenen uygulamanın güç tüketimi hesaplamak için Powertrace aracı kullanılmıştır. Makefile dosyasına eklenen APP+ = powertrace kodu ve ana koda eklenen #include "powertrace.h" kütüphanesi ve powertrace_start (CLOCK_SECOND * 10); metodu ile 10 saniye arayla güç tüketimi elde edilmiştir ve Şekil 3.5'te görüldüğü gibidir. El edilen verilen Çizelge 3.7'deki gibidir.



Şekil 3.5. Cooja simülasyon ortamında Powertrace aracı ile güç tüketiminin ölçülmesi

Çizelge 3.7. PowerTrace aracı ile 10 sn ara ile düğümünden elde edilen döngü sayıları

CPU	LPM	TX	RX
223	196.212	270	126.677
1.208	818.580	1.425	752.565
2.046	1.375.614	2.375	1.308.646
2.998	2.013.722	3.394	1.865.354
4.203	2.812.932	4.755	2.743.367
5.357	3.603.987	6.112	3.533.275
6.713	4.521.397	7.607	4.449.186

Çizelge 3.7'de verilen değerler wismote düğümü üzerinden test edilmiştir. CPU değeri toplam CPU döngüsünü, LPM; düşük güç modunda toplam döngü sayısı, TX; iletim durumunda toplam döngü sayısını ve RX; alış durumunda toplam döngü sayısını gösterir. Güç tüketim değeri mW cinsinden aşağıda verilen formül aracılığı ile hesaplanmıştır;

$$Enerji\ Değeri = \frac{Energest\ Değeri * Voltaj\ Değeri}{RTIMER_{SECOND} * Çalışma\ Süresi} \quad (3)$$

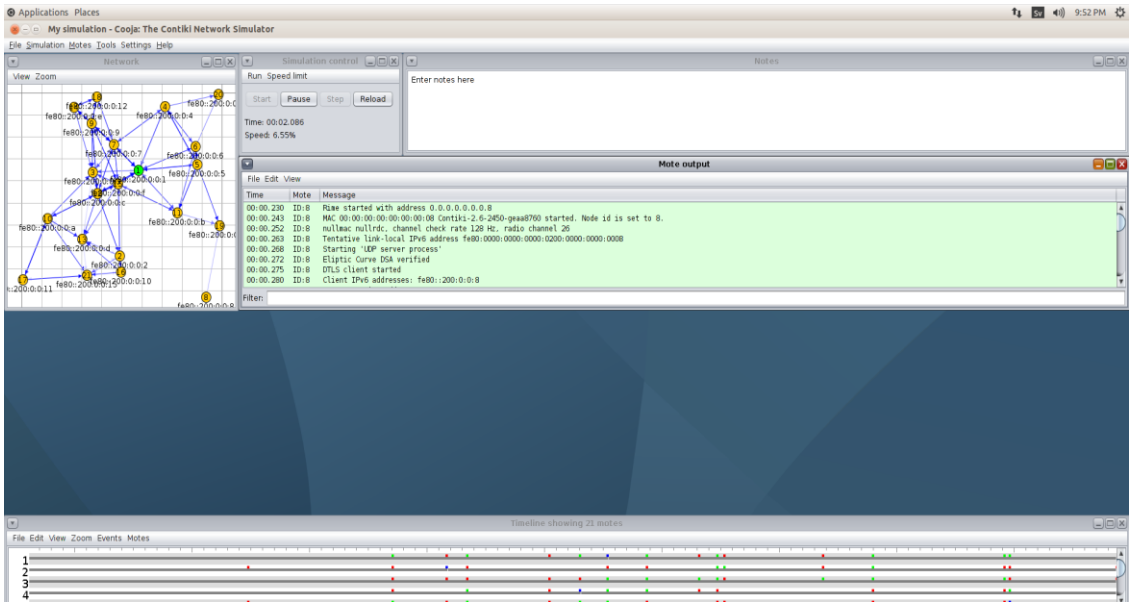
Çizelge 3.8. Wismote düğümünün mW cinsinden tükettiği enerji miktarı

CPU	LPM	TX	RX	Toplam
0,00128069551	0,80920193816	0,00147572525	0,81377863687	1,6257369957
0,00108956634	0,72425476956	0,00123518857	0,72301568039	1,449565912
0,00123778896	0,8296670625	0,00132490226	0,72383090048	1,5560606542
0,00272521604	0,8285285222	0,00176957015	1,14159118292	1,9745824267

0,00156673918	0,8391316407	0,00176436936	1,02703719435	1,8694999435
0,00150042906	1,19281510301	0,00194379673	1,19088610556	1,9871454343

Çizelge 3.8'de Wismote düğümünün harcadığı enerjinin mW cinsinden değeri verilmiştir. Formülde verilen Energest değeri iki çalıştırma arasındaki döngü farkı, Voltaj değeri düğümün çalışıtırıldığı voltaj (3V)'dır. RTIMER değeri koda eklenen metodla elde edilmiştir ve 34962'dir. Wismote düğümünde ortalama tüketilen enerji değeri 1.6 mW'tır. Bu değer sunucunun tükettiği enerji değerdir.

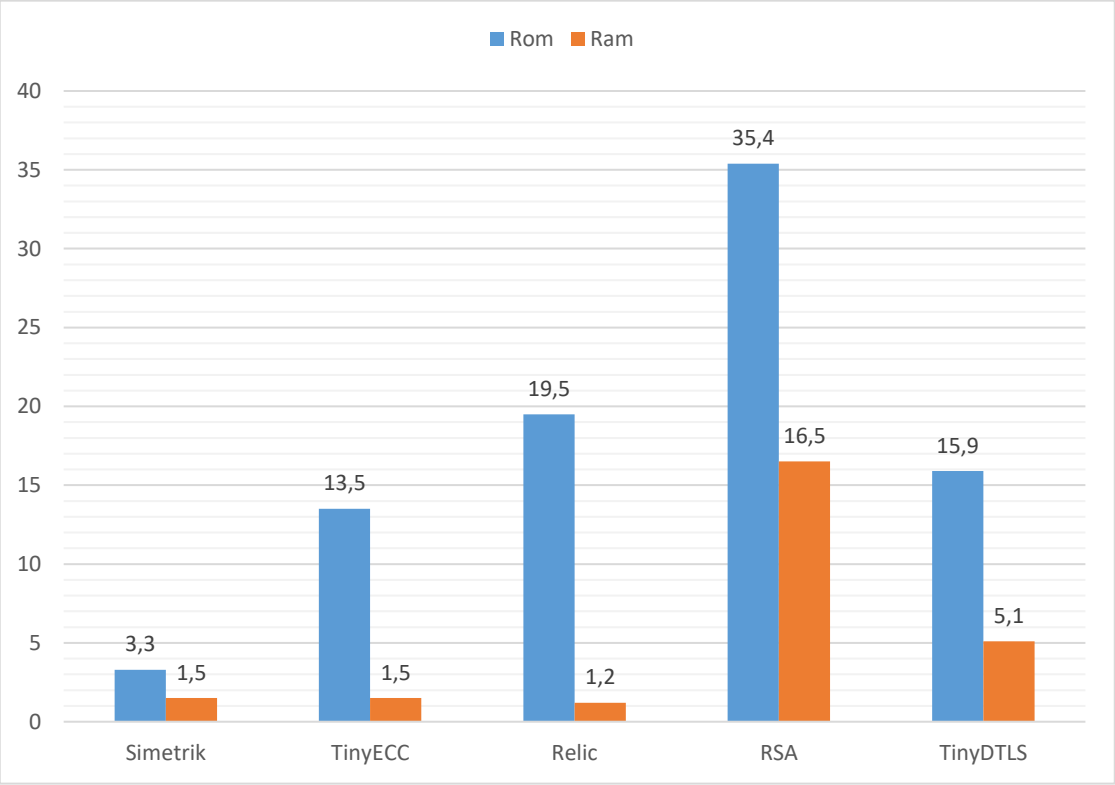
3.6 Benzetim Yöntemiyle Elde Edilen Değerler



Şekil 3.6. Eliptik eğri şifreleme kullanılarak düğümlerin haberleşmesinin Cooja simülasyon programı ile gerçeklenmesi

Görüldüğü üzere asimetrik şifreleme yalnızca kod boyutunu değil DTLS aşamasında gerçekleştirilen mesajlaşma sayısını da ciddi oranda arttırmaktadır. Ancak eliptik eğri şifreleme kod boyutunu diğer asimetrik şifreleme yöntemlerine göre ciddi oranda düşürdüğünden kaynak kısıtlı düğümlerde kullanımı fikri ortaya çıkmış, yapılan çalışmalar bunun üzerine yoğumlaşmıştır. Fakat hala görülmektedir ki eliptik eğri şifreleme kullanılsa dahi Sınıf-0 ve Sınıf-1 cihazlarda etkili şekilde çalışmamaktadır. Bu

konuda yapılan en önemli çalışmalardan biri olan TinyECC kodu dahi Sky ve Z1 düğümlerinde değil yine Wismote düğümünde çeşitli kaynakları kısırak çalıştırılmıştır. Diğer çalışmalara kıyasla TinyDTLS kodunun etkili çalıştığı görülmektedir. Ancak Contiki işletim sisteminin ve diğer protokollerin kullandığı kod miktarları düşünüldüğünde çok daha etkili çalışan bir eliptik eğri metoduna ihtiyaç duyulduğu görülmektedir. Özellikle Sınıf-0 cihazlar için uçtan uca şifreli haberleşmeyi sağlayacak şekilde kod optimize edilmesi gerektiği görülmektedir. Şekil 3.7'de yapılan çalışmaların kullandıkları Ram ve Rom miktarlarına göre kıyaslanması gösterilmiştir.



Şekil 3.7. Yapılan ECC çalışmalarının SŞ ve RSA ile RAM ve ROM kullanımı açısından karşılaştırılması

Şekil 3.7'de görüldüğü üzere eliptik eğri şifreleme kaynak tüketimini diğer asimetrik şifreleme yöntemlerine (RSA) göre azaltsa da simetrik şifrelemenin çok daha az kaynak tükettiği görülmektedir. Özellikle RSA ile kıyaslandığı büyük miktarda kazanç sağlanmaktadır. Ancak nesnelerinin inernetinin doğasında bulunan her zaman her yerde internete bağlı olan cihazların birbirleriyle haberleşmesi için asimetrik şifrelemenin kullanılması bir gerekliliktir. Ayrıca yapı büyüdükçe simetrik şifrelemenin anahtar

sayısından kaynaklanan dezavantajıda ortaya çıkmaktadır. N düğümlü bir ortamda simetrik şifreleme ile haberleşme için her düğümün diğer düğümlerle haberleşme için toplam N.(N-1) adet anahtara ihtiyaç duyulur. Asimetrik şifrelemede ise her düğümün bir açık bir de gizli anahtarı vardır ve toplam anahtar sayısı 2N'dir. Bu da düğüm sayısı milyonlara ulaştığında simetrik şifreleme kullanmanın ne kadar zor oluduğunu göstermektedir. Ayrıca kullanılan eliptik eğri şifreleme yöntemlerinden TinyECC kütüphanesinin en etkili çalışan eliptik eğri şifreleme yöntemi olduğu görülmüş. Diğer asimetrik şifrelemelere göre oldukça hızlı ve daha az kaynak tüketen eliptik eğri şifreleme yönteminin üzerinde durulduğu takdirde kaynak kısıtlı cihazlara uygulanabileceği gösterilmiştir.

BÖLÜM 4. SONUÇLAR

Gerçekleştirilen bu tez çalışmasında nesnelerin internetinde bulunan kaynak kısıtlı cihazlarda uçtan uca haberleşmeyi sağlamak için düğümlerde DTLS katmanında eliptik eğri şifreleme yöntemi kullanılarak güvenli haberleşmesi gerçekleştirilmiştir. Tezin birinci bölümünde nesnelerin internetinin yapısından ve gerçekleştirilen çalışmalardan bahsedilmiştir. İkinci bölümünde ise nesnelerin internetinde var olan protokoller ve şifreleme yöntemlerinden bahsedilmiştir. Üçüncü bölümde kablosuz sensör cihazlar için geliştirilen Contiki işletim sisteminin yapısı ve Cooja simülasyon programından bahsedilmiştir. Dördüncü bölümde açık kaynak kodu olan TinyDTLS kullanılarak düğümler arası güvenli haberleşmenin gerçeklemesi yapılmıştır. Bunun için Contiki işletim sistemi ve Cooja simülasyon programı kullanılmış ve kimlik doğrulama ve anahtar değişimi işlemleri eliptik eğri şifreleme yöntemi kullanılarak gerçekleştirilmiştir. Veri bütünlüğünü sağlamak için ise SHA2 hash algoritması kullanılmıştır. Bu sayede veri ele geçirilse dahi herhangi bir anahtar olmadığı için bir anlam ifade etmeyecektir. Ayrıca hash algoritması sayesinde verinin değiştirilmesi ve tekrarlanması ataklarına karşıda koruma sağlanmıştır. Bu sayede eliptik eğri şifrelemenin kaynak kısıtlı cihazlarda kullanılabileceği gösterilmiştir. Benzetim yöntemiyle elde edilen sonuçlar sunulmuş ve asimetrik şifreleme yönteminin kaynak kısıtlı cihazlarda özellikle Sınıf-0 cihazlarında etkili şekilde çalışmadığı görülmüştür. Bu çalışmada, belirtilen sebeplerden dolayı ciddi kaynak kısıtına sahip Sky düğümü Z1 düğümünü kullanılamamış ve Wismote düğümü tercih edilmiştir. Ancak nesnelerin internetinde cihazların her an internete bağlı olması diğer cihazlarla uçtan uca haberleşebilmesi için asimetrik şifreleme yönteminin entegre edilmesi bir gereklilik olarak görülmektedir. Bu nedenle yapılan çalışmalar eliptik eğri şifrelemenin en uygun asimetrik şifreleme yöntemi olduğunu, ancak herhangi bir optimizasyon yapılmadığı sürece kablosuz sensör cihazlar için ciddi yük getirdiği görülmüştür.

Eliptik eğri şifreleme yönteminin diğer asimetrik şifreleme yöntemlerine göre çok daha düşük anahtar boyutuyla yüksek seviye güvenliği sağladığı görülmüştür. Geliştirilen bazı kütüphanelerin kullandıkları hafıza miktarları karşılaştırılmıştır. TinyECC kütüphanesi en etkili çalışan eliptik eğri şifreleme yöntemi olduğu ve dinamik bir yapıya

sahip olması nedeniyle istenilen optimizasyonların yapılmasına olanak verdiği görülmüştür.

KAYNAKÇA

[1] F. A. Aloul, I. A. Zualkernan, R. Mahmoud, Internet of things (IoT) security: Current status, challenges and prospective measures. In 2015 10th International Conference for Internet Technology and Secured Transactions(ICITST), London, UK, December, 336-341.

[2] A. Liu, P. Ning, TinyECC: A configurable library for elliptic curve cryptography in wireless sensor networks. In 7th International Conference on Information Processing in Sensor Networks (IPSN'08), Washington, DC, USA, 2008.

[3] P. Szczechowiak, L.B. Oliviera, M. Scott, M. Collier, R. Dahab, NanoECC: Testing the Limits of Elliptic Curve Cryptography in Sensor Networks. Jan 2008, pp. 305-320.

[4] K. Zhao, L. Ge, A Survey on the "Internet of Things Security". 9th International Conference on Computational Intelligence and Security, Leshan, 14-15 December 2013, 663-667.

[5] G. Santos, V. T. Guimaraes, G. C. Rodrigues L. Z. Granville, L. M. R. Tarouco, A DTLS-based Security Architecture for the Internet of Things, 20th IEEE Symposium on Computers and Communication (ISCC), pp. 809-815, 2015.

[6] S. Raza, H. Shafagh, K. Hewage, R. Hummen, Thiemo Voigt, Lithe: Lightweight Secure CoAP for the Internet of Things". IEEE Sensors Journal, 13(10), 3711-3720, October 2013.

[7] S. Görmüş, H. Aydın, G. Ulutaş, Nesnelerin interneti teknolojisi için güveenlik: var olan mekanizmalar, protokoller ve yaşanılan zorlukların araştırılması, Gazi Üniversitesi Mühendislik-Mimarlık Fakültesi Dergisi, 5000211237, Ekim 2017.

[8] S. Li, L. D. Xu, Securing the internet of things, 2017.

[9] C. Bormann, M. Ersue, and A. Keranen, 2014, Terminology for Constrained-Node Networks, RFC 7228, IETF.

[10] J. McCumber, "Assessing and Managing Security Risk in IT Systems" in A Structured Methodology. New York: AUERBACH PUBLICATIONS CRC Press LLC. , 2005.

[11] D. R. Stinson, Cryptography Theory and Practice Third Edition, 2006.

[12] Şifreleme yöntemleri, http://bidb.itu.edu.tr/seyir-defteri/blog [Ziyaret Tarihi: Mart 2019].

[13] Açık Anahtarlı Şifreleme, http://bilgisayarkavramlari.sadievrenseker.com [Ziyaret Tarihi: Mart 2019].

[14] B. A. Forouzan, Cryprography and Network Security, 2008

[15] W. Trappe, L. C. Washington, Introduction to Cryptography with Coding Theory, Pearson Prentice Hall. ISBN 0-13-186239-1.

[16] Ö.Ö. Bozkurt, Eliptik Eğri Şifreleme Kullanarak Güvenli Soket Katmanı Protokolünün Gerçeklemesi ve Performansının Değerlendirilmesi, 2005 İstanbul.

[17] A. Khalique, K. Singh, S. Sood, Implementation of Elliptic Curve Digital Signature Algorithm, International Journal of Computer Applications (0975 – 8887) Volume 2 – No.2, May 2010.

[18] Elliptic Curve Public Cryptography, gauss.ececs.uc.edu/Courses/c6053/lectures/PDF/elliptic.pdf [Ziyaret Tarihi: Nisan 2019].

[19] Z. Shelby, K. Hartke, C. Bormann, The Constrained Application Protocol (CoAP), Internet Engineering Task Force (IETF) Request for Comments: 7252 Category: Standards Track ISSN: 2070-1721, June 2014.

[20] A.A. Chavan, M.K. Nighot, Secure CoAP Using Enhanced DTLS for Internet of Things, International Journal of Innovative Research in Computer and Communication Engineering (An ISO 3297: 2007 Certified Organization) Vol. 2, Issue 12, December 2014.

[21] Xi Chen, Constrained Application Protocol for Internet of Things, URL:https://www.cse.wustl.edu/~jain/cse574- 14/ftp/coap/

[22] E. Rescorla, H. Tschofenig, N. Modadugu, The Datagram Transport Layer Security (DTLS) Protocol Version 1.3 draft-ietf-tls-dtls13-31, TLS Internet-Draft Obsoletes: 6347 (if approved) Intended status: Standards Track Expires: March 25, 2019.

[23] S. Keoh, S. Kumar, Z. Shelby, Profiling of DTLS for CoAP-based IoT Applications draft-keogh-dtls-profile-iot-00, Internet Draft Intended Status: Informational Expires: December 14, 2013.

[24] H. Shafagh, S. Raza, T. Voig, K. Wehrle, Delegation-based Authentication and Authorization for the IP-based Internet of Things, Eleventh Annual IEEE International Conference on Sensing, Communication, and Networking (SECON), June 30, 2014 - July 3, 2014.

[25] O. Bergmann. (2013, Feb. 15). tinyDTLS [Online]. Available:http://tinydtls.sourceforge.net/

[26] Z. Shelby, K. Hartke, C. Bormann, and B. Frank. (2013, May). Constrained Application Protocol (CoAP). Internet-Draft draft-ietf-corecoap-16 [Online]. Available:http://datatracker.ietf.org/drafts/current/

[27] S. Keoh, S. Kumar, O. Garcia-Morchon, Securing the ip-based internet of things with DTLS draft-keoh-lwig-dtls-iot-02, LWIG Working Group Internet-Draft Intended Status: Informational Expires: February 2014.

[28] A. Capossele, V. Cervo, G. De Cicco, C. Petrioli, Computer Science Department, University of Rome "La Sapienza" WSENSE S.r.l , Rome, Italy, June 2015.

[29] Y. Maleh, A. Ezzati, Towards an Efficient Datagram Transport Layer Security for Constrained Applications in Internet of Things , International Review on Computers and Software (I.RE.CO.S.), Vol. 11, N. 7 ISSN 1828-6003, July 2016.

[30] H. Suo, J. Wan, Security in the Internet of Things: A Review, College of Information Engineering Guangdong Jidian Polytechnic Guangzhou, China, International Conference on Computer Science and Electronics Engineering, 2012.

[31] A. Alhaj, Constrained Application Protocol (CoAP) for the IoT, Iot Semınar, Hıgh Integrıty System, Frankfurt Unıversıty Of Applıed Scıence, 2018.

[32] J. King, A Distributed Security Scheme to Secure Data Communication between Class-0 IoT Devices and the Internet, 2015.

[33] A. L. Colina, A. Vives, M. Zennaro, A. Bagula, E. Pietrosemoli, Internet of Things in Five Days, 24th June 2016