KABLOSUZ ALGILAYICI AĞLARDA SINKHOLE ATAĞI İÇİN GELİŞTİRİLMİŞ BİR GÜVENLİK ALGORİTMASI

Majid MEGHDADI

DOKTORA TEZİ
ELEKTRONİK BİLGİSAYAR EĞİTİMİ BÖLÜMÜ

GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ

HAZİRAN 2010
ANKARA

Majid Meghdadi tarafından hazırlanan "KABLOSUZ ALGILAYICI AĞLARDA SINKHOLE ATAĞI İÇİN GELİŞTİRİLMİŞ BİR GÜVENLİK ALGORİTMASI" adlı bu tezin Doktora tezi olarak uygun olduğunu onaylarım.

Prof. Dr. İnan Güler	Yrd. Doç. Dr. Suat ÖZDEMİR
Danışman	İkinci Danışman

Bu çalışma jürimiz tarafından oy birliği ile Elektronik Bilgisayar Eğitimi Anabilim Dalında Doktora tezi olarak kabul edilmiştir.

Başkan : Prof. Dr. Ö. Faruk BAY

Üye : Prof. Dr. İnan GÜLER (Danışman)

Üye : Doç. Dr. Ayhan ERDEM

Üye : Doç. Dr. M. Ali AKCAYOL

Üye : Yrd. Doç. Dr. Mustafa SERT

Tarih : Haziran 2010

Bu tez, Gazi Bilişim Enstitüsü tez yazım kurallarına uygundur.

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, ayrıca tez yazım kurallarına uygun olarak hazırlanan bu çalışmada orijinal olmayan her türlü kaynağa eksiksiz atıf yapıldığını bildiririm.

Majid MEGHDADI

KABLOSUZ ALGILAYICI AĞLARDA SINKHOLE ATAĞI İÇİN GELİŞTİRİLMİŞ BİR GÜVENLİK ALGORİTMASI

(Doktora Tezi)

Majid MEGHDADI

GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ

Haziran 2010

ÖZET

Son yıllarda Kablosuz Algılayıcı Ağ (KAA)'ları çok geniş uygulama alanları bulmaktadır. KAA'lar genelde kötücül ortamlarda kullanılmaları nedeniyle ağ güvenliği açısından hassastırlar. Kaynakları kısıtlı algılayıcı düğümleri sebebiyle, araştırmalar geleneksel güvenlik mekanizmalarının bu ağlarda çözüm sağlayamadığını ortaya koymuştur. Özellikle askeri ve sağlık uygulamaları gibi güvenlik yönünden hassas veri iletiminin yapıldığı KAA'larda veri gizliliğini ve bütünlüğünü sağlayacak güvenlik mekanizmalarına fazlasıyla ihtiyaç vardır. Daha da önemlisi, KAA'ların kendilerine has özellikleri sebebiyle bu güvenlik mekanizmaları sistem tasarımı aşamasında geliştirilmelidir. KAA'larda geleneksel ağlarda görülmeyen birçok atak bulunmaktadır. Bu çalışmada, bu tür ataklardan birisi olan Sinkhole atağı ele alınmış ve bu atağa karşı iki savunma mekanizması önerilmiştir. Bu çalışmada bu mekanizmalardan birisi gerçekleştirilmektedir. Gerçekleştirilen savunma mekanizması ağ içindeki veri hareketlerinin analizine ve saldırıyı gerçekleştiren düğümlerin "oy çokluğu" metoduyla tespitine dayanmaktadır. KAA'larda Sinkhole saldırısı ve önerilen savunma mekanizması NS-2 simülatörü kullanılarak benzetimi yapılıp ve test edilmiştir. Benzetim sonuçları önerilen savunma mekanizmasının kaybolan paket sayısını %83,6 oranında azalttığını göstermiştir.

Bilim Kodu : **902.1.063**

Anahtar Kelimeler : **Kablosuz Algılayıcı Ağları, Saldırı Tespit Sistemleri, Solucan Deliği, Kötücül Düğüm, Ağ Simülatörleri.**

Sayfa Adedi : **109**

Tez Yöneticisi : **Prof. Dr. İnan Güler**

DESIGN AND IMPLEMENTATION OF AN ALGORITHM AGAINST SINKHOLE ATACK IN WIRELESS SENSOR NETWORKS

(Ph. D. Thesis)

Majid MEGHDADI

GAZİ UNIVERSITY

INFORMATICS INSTITUTE

June 2010

ABSTRACT

Recently, Wireless Sensor Networks (WSNs) have been found wide real world application areas. As they are usually deployed in hostile environments, WSNs are sensitive in terms of security. In particular, for military and medical WSNs that carry security sensitive data, it is more important to provide data privacy and integrity. Studies show that, due to limited resources of these networks, traditional security mechanisms cannot solve security problems. More importantly, because of the unique characteristics of WSNs, security mechanisms should be developed during system design phase. In WSNs, there are many attacks that do not exist in traditional networks, such as sinkhole. In this study two defense mechanisms have been proposed for sinkhole attack prevention and one of those methods has been implemented. Implemented security mechanism is based on the analysis of data flow within the network. Malicious sinkhole nodes are determined through the "votes" of neighboring sensor nodes. Sinkhole attack and proposed defense mechanism against the attack have been simulated and tested using NS-2 simulator. The simulation results show that the proposed defense mechanism decreases the number of lost packets due to sinkhole attack at a rate of 83.6%.

Science Code : 37N35, 93D20

Key Words : Wireless Sensor Networks, Malicious Node Detection, Sinkhole Attack, Malicious Node, Network Simulators.

Page Number : 109

Adviser : Prof. Dr. İnan Güler

TEŞEKKÜR

Ders dönemlerinde ve çalışmalarım boyunca değerli yardım ve katkılarıyla bana yol gösteren, yönlendiren önerilerde bulunarak değerli zamanını benden esirgemeyen tez danışmanlarım Prof. Dr. İnan GÜLER ve Yrd. Doç. Dr. Suat ÖZDEMİR'e, tez izleme komitesi üyelerim Doç. Dr. O.Ayhan ERDEM ve Doç. Dr. M. Ali AKCAYOL'a, teşekkür ederim. Ayrıca Elektronik Bilgisayar Eğitimi Bölümünde görevli tüm öğretim üyelerine ve manevi destekleriyle beni hiçbir zaman yalnız bırakmayan çok değerli ailem ve arkadaşlarıma teşekkürü bir borç bilirim.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER, KISALTMALAR VE TANIMLAR

Bu çalışmada kullanılmış bazı simgeler kısaltmalar ve tanımlar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklama**
Bit	Bilişimde kullanılan en küçük bilgi birimi (0 veya 1).
ID	Identification
KB	Kilobyte
KHz	Kilohertz (10^3 Hertz)
Mbps	Megabit (10^6 Bit) per second
MB	Mega Bayt
MHz	Megahertz (10^6 Hertz)
m	Metre
m/s	Meter per Second
s	Saniye
RC4, RC5	Simetrik Şifreleme Algoritmaları
TIK	Bir Kimlik Doğrulama Algoritması

Kısaltmalar	**Açıklama**
ACK	Acknowledgement
AODV	Ad-Hoc On-demand Distance Vector Routing
CBR	Constant Bit Rate
DLL	Data Link Layer
DoS	Denial of Service
DSDV	Destination-Sequenced Distance Vector Routing Protocol
DSRP	Dynamic Source Routing Protocol (bir yönlendirme protokolü)
GPS	Global Positioning System
HSRP	Hot Standby Router Protocol (bir yönlendirme protokolü)

IDS	Intrusion Detection System
IEEE	The Institute of Electrical and Electronic Engineers
KAA	Kablosuz Algılayıcı Ağ
KD	Kötücül Düğüm
KDB	Kötücül Düğümleri Bulma
LAN	Local Area Network
LEAP	Lightweight Extensible Authentication Protocol
LLC	Logical Link Control
MAC	Message Authentication Code
MAD	Mutual Authentication with Distance-bounding
MANET	Mobile Ad-Hoc Network
MDS-VOW	Multi Dimensional Scaling -Visualization of Wormhole
NAM	Network Animator
OTCL	Object Oriented Tool Command Language
PHY	Physical Layer
REWARD	Receive, Watch and Redirect (bir yönlendirme protokolü)
RF	Radio Frequency
RREQ	Route Request Packet
$\mathbf{R_{REP}}$	Route Request Packet (Yol bulma Paketi)
$\mathbf{R_{REQ}}$	Route Reply Packet (Yol Yanıt Paketi)
RREP	Route Replay Packet
RERR	Route Error Packet
RTT	Round Trip Time (paketin gediş dönüş zamanı)
SAM	Statistical Analysis Method
SAV	Saldırılmış Alandaki Veriler
SECTOR	Secure Tracking of Node Encounters in Multi-hop Wireless Networks
SPINS	Security Protocol for Sensor Networks
TCL	Tool Command Language
TCP	Transmission Control Protocol
TCP- SYN	TCP Synchronization (bir çeşit atak)
$\mathbf{T_{REP}}$	Time of Reply (paketin yanıt zamanı)
$\mathbf{T_{REQ}}$	Time of Request (paketin gönderme zamanı)

UDP	User Datagram Protocol
WLAN	Wireless Local Area Network
WODEM	Wormhole Attack Defense Mechanism in Wireless Sensor Networks
µTESLA	Macro version of the Time, Efficient, Streaming, Loss Tolerant Authentication

1.GİRİŞ

Kablosuz Algılayıcı Ağ (KAA) kavramı ilk kez 1980'lerin başlarında ortaya çıkmıştır [1]. Gelişen teknolojiyle birlikte küçük boyutlarda, az güç tüketen ve çok fonksiyonlu algılayıcı elemanlar tasarlanabilmektedir [2]. Bu elemanlardan yararlanarak, farklı mekânlardaki farklı nicelikleri, toplu bir şekilde izlemek için KAA'lar tasarlanmıştır. Bu ağlar çok sayıda sınırlı kapasiteli, kısa mesafeli vericiye sahip, düşük güçlü ve düşük maliyetli algılayıcının kolayca erişilemeyen ve çoğu zaman güvenilir olmayan bir ortama rastgele bırakılmasıyla oluşur. Her bir algılayıcı düğümü, çevresindeki sıcaklık, nem, basınç, ses, ışık ve nesne hareketleri gibi nicelikleri ölçebilme, basit hesaplama işlemleri yapabilme ve diğer algılayıcı düğümleri veya merkezi baz istasyonuyla haberleşme yapabilme özelliklerine sahiptir. KAA'larda planlanmış bir ağ omurgası yoktur ve algılayıcılar tarafından ortak gayret sarf ederek toplanan veriler, merkezi bir baz istasyonuna, diğer algılayıcılar üzerinden gönderilir. Kablosuz iletişim teknolojilerinin bir çeşidi olan KAA'lar günümüzde üzerine en çok araştırma ve geliştirme yapılan konulardan biri haline gelmiştir [2-6]. KAA'larda düğümlerin birbirleri ile iletişim halinde olabilmeleri için veri paketlerini ağdaki diğer düğümlere yönlendirerek iş birliği yapmaları gerekir. Bu sebeple düğümler yönlendirme protokollerini kullanarak baz istasyonuna bir yol bulurlar. Ancak kullanılan yönlendirme protokollerindeki güvenlik zafiyetlerinden dolayı kablosuz algılayıcı ağlar kötü niyetli kişilerin ataklarına açıktır.

Çalışmanın Amacı ve Kapsamı: Günümüzde KAA'ları üzerinde yapılan araştırmalar konum belirleme, enerji yönetimi, dinamiklik, güvenlik gibi başlıklar altında olmaktadır. KAA'larda geleneksel ağlarda görülmeyen birçok atak bulunmaktadır. Bu çalışmada, KAA güvenliğine karşı en önemli saldırılardan birisi olan Sinkhole atağı ele alınmıştır ve bu atağa karşı bir savunma mekanizması gerçekleştirilmiştir. Sinkhole atağı KAA'lara özgü bir ataktır. Bu atakta kötücül düğüm (KD), sahte yönlendirme paketlerini komşularına göndererek, kendisinin baz istasyonuna kısa ve yüksek kaliteli bir yol üzerinden ulaşabildiğini duyurur. Bu durumda o bölgedeki normal düğümler yönlendirme tablolarını değiştirip paketleri bu yeni bulunan kısa ve

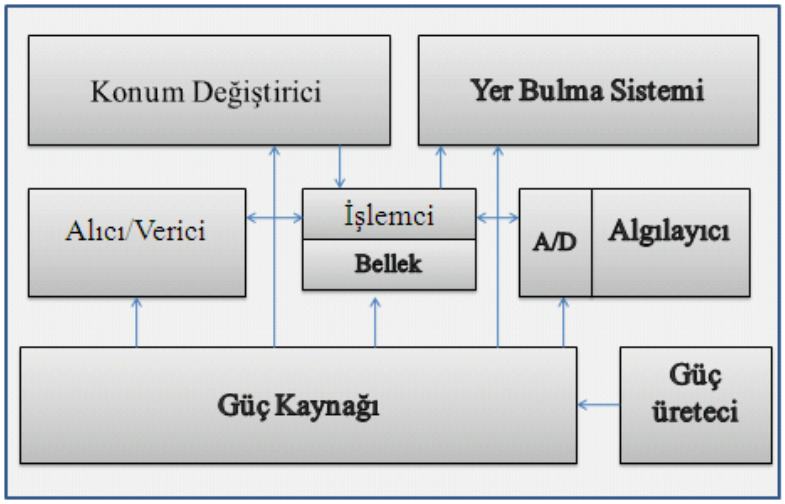
yüksek kaliteli yol üzerinden baz istasyonuna gönderirler. Kötücül düğüm, aldığı paketleri ya iptal eder ya da bir başka kötücül düğüme gönderir veya değiştirdikten sonra baz istasyonuna gönderebilir. Bu durumda baz istasyonunun doğru ve hatasız veri alabilmesi engellenmektedir.

Araştırmanın Yöntemi: Kablosuz algılayıcı ağlarda sinkhole saldırısını gerçekleştiren saldırganın etkilerini yok etmek için birçok tespit ve savunma mekanizmaları bulunmaktadır. Bu çalışmada, Sinkhole saldırısının çeşitli senaryolarla benzetimi yapılıp, benzetim ortamında bir savunma mekanizması bulunmaya çalışılmıştır. Önerilen savunma mekanizması, ağ içindeki veri hareketlerinin analizine ve saldırıyı gerçekleştiren düğümlerin "oy çokluğu" metoduyla tespitine dayanmaktadır.

Bu tezin ikinci bölümünde KAA'ların tanımı, özellikleri, zorlukları ve KAA'ları geleneksel ağlardan ayıran özellikler hakkında bilgi verilmiştir. Bu özelliklerden güvenlik üzerinde durulmuş ve KAA'lardaki güvenlik saldırılardan birisi olan Sinkhole atağı ele alınmış ve bu saldırıya karşı bir savunma mekanizması önerilmiştir. Üçüncü bölümde bu konu ile ilgili yapılan çalışmalara yer verilmiştir. Dördüncü bölümde, tezde önerilen ve kullanılan materyal, metotlar açıklanmıştır. Beşinci bölümde ise elde edilen bulgular ışığında yapılan yorumlara yer verilmiştir. Son bölümde tezde ulaşılan sonuçlar ve gelecek çalışmalar için öneriler yer almaktadır.

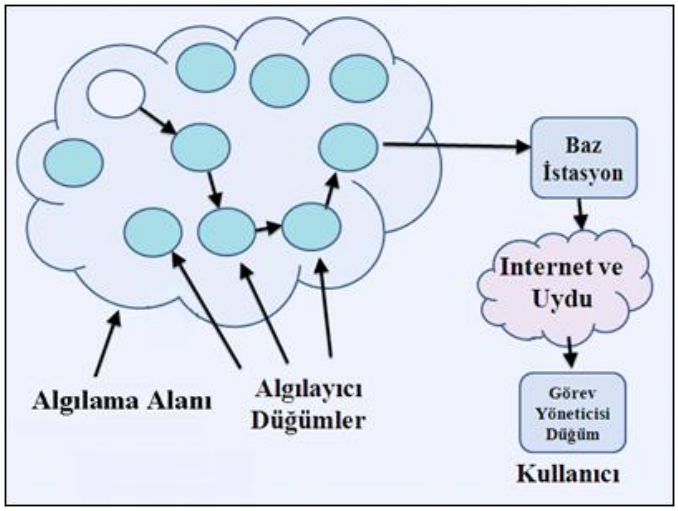
2. KABLOSUZ ALGILAYICI AĞLAR

KAA'lar yaygın bir şekilde algılayıcı, duyarga ya da sensör ağları olarak bilinmektedir. Algılayıcı ağları oluşturan düğümlerin (nodes) her biri ise algılayıcı düğümü veya küçük boyutları sebebiyle toz tanesi/zerre (mote) olarak adlandırılırlar. Algılayıcılar oldukça düşük kapasiteli cihazlardır. KAA'larda görev yapan düğümlerin işlemci, algılayıcı, alıcı/verici ve güç birimi olmak üzere dört ana elemanı vardır. Bunlara ilave olarak kullanım amacına göre bir düğümde, yer bulma sistemi, güç üretim birimi, konum değiştirici bulunabilir [2, 5, 7-10]. Şekil 2.1'de bir düğümün elemanları gösterilmektedir [2].



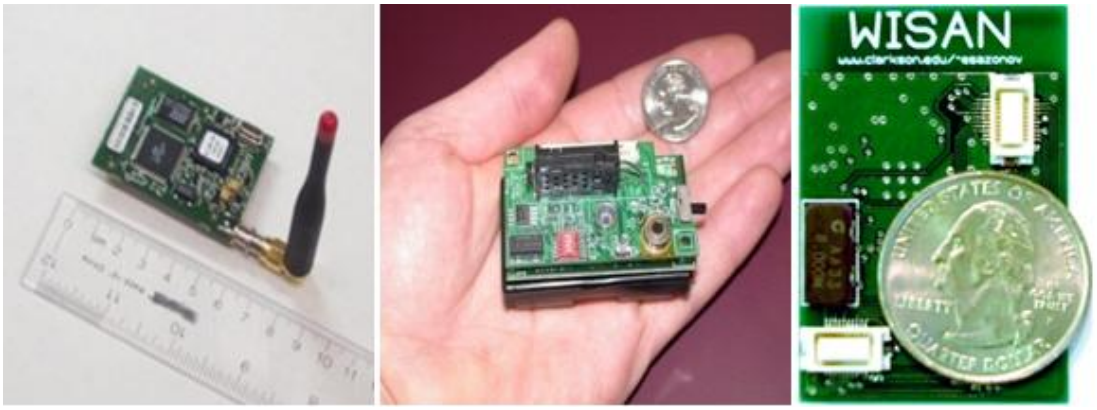
Şekil 2.1. Bir KAA düğümünü oluşturan elemanlar [2].

Düğümler, uygulama sahasında donanım ve iletişim gücü itibariyle baz istasyonu (sink) etrafında belirlenen protokoller çerçevesinde tamamen kendi kendilerine kısa sürede organize olurlar. Düğümler algılayıcıları vasıtasıyla tespit ettikleri veriyi baz istasyonuna birbirleri üzerinden ulaştırırlar. Baz istasyonu kendisine ulaşan veriyi erişim noktaları aracıyla ya da doğrudan kullanıcıya ulaştırır. Verinin iletimi sırasında internet, intranet gibi ağ erişimleri de kullanılabilir. Şekil 2.2'de gösterildiği gibi bu düğümler algılama alanı olarak adlandırılan fiziksel alana otonom bir şekilde iş birliği içerisine girerek, fiziksel dünyadan öğrendiklerini sanal dünya ortamına taşımaktadırlar.



Şekil 2.2. Algılayıcılardan kullanıcıya verinin ulaşması [2]

Bilgi işlem ağına olan geçit baz istasyonu olarak adlandırılmaktadır. Bu düğüm hem algılayıcı düğümleri ile hem de haberleşme ağı ile iletişim kurabilen özel bir düğümdür. Baz istasyonu, enerji problemi olmayan statik ve hesaplama kabiliyeti yüksek bir düğüm olarak kabul edilir. Algılayıcı düğümleri ise kablosuz ve genellikle radyo teknolojisi ile iletişim kuran, enerji ve hesaplama kabiliyetleri kısıtlı birimlerdir. Bu birimler gözlemlenmesi hedeflenen alandaki bazı durumları ve olayları algılamak ve takip etmek amacı ile otomatik olarak yerleştirilmektedirler. Küçük boyutlara sahip olmaları ise kullanılabilirlik açısından fiziksel bir gereksinimdir. Donanım teknolojisindeki ilerlemelere paralel olarak tüm bu birimler bir kibrit kutusu büyüklüğünde ya da bozuk para boyutlarında üretilebilmektedir. Şekil 2.3'te örnek birkaç algılayıcı düğümü gösterilmiştir.



Şekil 2.3. Algılayıcı düğümlerin örnek boyutları

2.1. Kablosuz Algılayıcı Ağların Özellikleri

Sınırlı enerji kapasiteleri, kısıtlı bellekleri, uygulamalarda karşılaşılan en önemli kısıtlamalar olarak karşımıza çıkmaktadır. Kısıtlı algılayıcılar ve KAA'ların boyutları, güvenlik çözümleri üreten araştırmacıların önüne yeni zorluklar olarak çıkarmaktadır. Bu bölümde açıklanan karakteristiklerin protokol tasarımı ve geliştirilmesi sırasında dikkate alınması protokolün kullanılabilirliğini artırmaktadır.

2.1.1. Büyük ölçek

KAA'ların genel uygulamaları (örneğin askeri gözetleme uygulamaları) coğrafi açıdan geniş bir alanın kapsanmasını gerektirir. KAA'lar genelde düğümlerin kısa ömür oranları, kısıtlı radyo kapasiteleri, güvenilirliği düşük, ucuz algılayıcılar sebebiyle çok geniş bir alana yayılabilir ve bir KAA'daki düğüm sayısı on binleri aşabilir [11].

2.1.2. Kısıtlı kaynak

KAA'ların düşük kurulum ve işletim maliyetli olma zorunluluğu algılayıcı düğümlerinin donanım açısından sade olmasını gerektirir. Bu nedenle KAA'larda işlem ve iletişim kaynakları kısıtlıdır. Örneğin genel bir algılayıcı türü olan (TelosB) 16 bitlik, 8 MHz işlemci, 48KB ana hafıza, 1024 KB anlık belleğe sahiptir. Ayrıca KAA'ların yaşam süresi algılayıcıların kullanım sürelerine bağlıdır fakat bataryaların değiştirmesi çok zor hatta çoğu zaman imkansızdır. İşlemci kapasitesinin düşüklüğü, hafıza ve radyo iletiminin kısıtlı olması, ağ ömrünün batarya ömrü ile sınırlı olması KAA'lar için tasarlanan her protokolü etkilemektedir [12].

2.1.3. Artıklılık

KAA'ları oluşturan algılayıcı düğümlerinin yaşam sürelerinin önceden kestirilememesi ve algılayıcıların kısa radyo iletişim aralıkları, KAA'larda düğüm artıklığını zorunlu kılmaktadır. Bu nedenle KAA'lar kurulurken algılayıcı düğümleri

yüksek dereceli bir artıklılıkla kullanılırlar. Bu yüksek dereceli düğüm artıklılığı sayesinde tek bir düğümün vaktinden önce kullanılmaz duruma gelmesi, sistemin kapasitesini çok fazla etkileyemez. Ancak düğüm artıklılığı nedeniyle her olay birden fazla algılayıcı düğümü tarafından algılanır ve dolayısı ile ağda taşınması gereken veri miktarı artar. Başka bir deyişle artıklık baz istasyonuna gönderilen verilerin miktarını artırmakta ve ağın yaşam süresini azaltmaktadır. Veri artıklılığından kurtulmak için veri kümeleme protokolleri kullanılmaktadır [13, 14].

2.1.4. Güvenlik

Askeri sistemler ve tıbbi takip sistemleri gibi KAA uygulamaları güvenlik açısından çok hassastırlar. Algılayıcı düğümlerinin kısıtlı kaynaklarından dolayı geleneksel güvenlik mekanizmaları KAA'larda kullanılamaz. Buna ek olarak KAA'larda geleneksel ağlarda görülmeyen algılayıcıların fiziksel güvenliklerinin olmaması sorunu vardır. Algılayıcıların fiziksel güvenlikleri sağlanamadığından, ağdaki algılayıcı düğümleri her an kötü niyetli kişilerce ele geçirilip, kötü amaçlar için kullanılabilirler. Bu nedenlerden dolayı KAA'ların güvenlik mekanizmaları algılayıcı düğümlerinin kaynak kısıtları ve kötücül algılayıcılar göz önüne tutularak tasarlanmalıdır [15, 16].

2.1.5. Veri merkezli işleme

Veri merkezli işleme KAA'ların en önemli özelliklerindendir. Algılayıcı düğümlerin ID'leri (Identifications) uygulamalar için çoğu zaman önemli değildir. Bu nedenle KAA uygulamalarında adlandırma düzeni çoğunlukla veriye yöneliktir (data oriented). Örneğin bir çevre gözetim sisteminde sıcaklık ölçümü yapmak için "X, Y ve Z düğümlerinden sıcaklık ölçüm değerlerini topla" şeklinde değil, "(X_1, Y_1 ,X_2, Y_2) koordinatları ile sınırlandırmış bölgeden sıcaklık ölçüm değerlerini topla" şeklinde olur. O bölgedeki algılayıcı düğümlerinin ID'lerinin uygulama için bir önemi yoktur [17-19].

2.1.6. Tahmin edilemezlik

Hava durumu ve zor çevre koşulları gibi nedenlerle algılayıcı düğümlerinde ölçüm hataları çok yaygındır. Çok sayıda dağıtılmış düğüm tarafından paylaşılan kablosuz iletişim ortamı istenmeyen tıkanıklık ve engellemelere neden olmaktadır. Yüksek bit hata oranı, düşük bant genişliği ve çok sayıda algılayıcı düğümünün aynı iletim ortamını kullanması nedeniyle, KAA'larda iletişim yüksek tahmin edilememezlik gösterir. Bu tahmin edilememezlik, genelde sistem parametrelerinin çevrim-dışı (off-line) tasarımını engellemektedir. Bu nedenle KAA tasarımında çevrim-içi (on-line) gözetim ve geri-beslemeli kontrol, yüksek servis kalitesini (quality-of-service) sağlamak için gereklidir [19].

2.1.7. Gerçek zaman kısıtlamaları

KAA'lar gerçek dünya işlemlerinde kullanıldıklarından çoğu zaman gerçek zaman kısıtlamalarına uymaları gerekmektedir. Gözetim sistemlerinde, örneğin iletişimdeki gecikme doğrudan doğruya uygulamanın hedef bulma niteliğini olumsuz yönde etkilemektedir. Kablosuz iletişimin tabiatından ve trafik yoğunluğunun önceden tahmin edilememesinden dolayı, KAA'ların katı-gerçek-zamanlı (hard-real-time) kısıtlamaları garanti etmesi beklenemez ancak olasılık temelli araştırmalar zaman kısıtlamalarının belli seviyelerde garanti edilmesini sağlayabilmektedir [20].

2.2. KAA'ları Geleneksel Ağlardan Ayıran Özellikler

Önceki bölümde bahsedilen özelliklerden dolayı, KAA'ları geleneksel kablosuz ağlardan ayıran özellikler şunlardır [20-22]:

- Algılayıcı ağlarındaki algılayıcı sayısı geleneksel kablosuz ağlardaki bilgisayar sayısından çok daha fazla olabilmektedir.
- Gerek ucuz donanım kullanımından gerekse de kullanıldıkları alanın özelliğinden bazı algılayıcı düğümlerinin düzenli çalışmama ihtimali yüksektir.

- Algılayıcı donanım, ucuz ve özellikleri kısıtlıdır (sınırlı batarya, işlemci, bellek, radyo)
- Her düğümün başında kullanıcısı yoktur, uygulama sahasına bırakıldıktan sonra kendi kendilerine organize olmak zorundadırlar.
- Algılayıcı ağları çok yoğun olarak konuşlandırılmaktadır. Normal koşullarda algılayıcılar arasındaki mesafe birkaç metre ile sınırlıdır.
- Algılayıcı ağlarındaki algılayıcılar, bilgisayarlar veya diğer telsiz haberleşme donanımı ile karşılaştırıldığında çalıştıkları ortamlar ve ucuz donanım sebebi ile çok daha yüksek oranda arıza yapabilmektedirler.
- Algılayıcı ağlarındaki haberleşme hatları çok daha sık değişebilmektedir.
- Algılayıcı ağlarındaki algılayıcıların özgün (unique) kimlikleri (adresleri) yoktur.
- Kullanıldıkları sahada algılayıcılar kötü niyetli kişiler tarafından ele geçirilebilir ve kötü amaçlar için kullanılabilirler.

Yukarıda bahsedilen özelliklerden dolayı, KAA'lar geleneksel kablosuz Ad-Hoc ağlardan farklıdırlar ve bu yüzden geleneksel kablosuz Ad-Hoc ağlar için geliştirilmiş olan güvenlik çözümleri KAA'larda kullanılamaz.

Çizelge 2.1. Geleneksel Ad-Hoc ağlar ve KAA'lar arasındaki farklar

Özellikler / **Ağlar**	**Ad-Hoc Ağlar**	**KAA**
Kaynak	Normal	Çok sınırlı
İş-birlik (Collaboration)	Az	Çok
Hareketlilik (Mobility)	Çoğunlukla	Bazen
Açık anahtar altyapısı	Kullanılır	Kullanılmaz
Gizli anahtar altyapısı	Kullanılır	Kullanılır
Haberleşme topolojisi	Bir noktadan bir noktaya	Birçok noktadan bir noktaya Bir noktadan birçok noktaya

Çizelge 2.1'de KAA'lar ve geleneksel kablosuz Ad-Hoc ağlar arasındaki farklar özetlenmiştir. Bu sebeplerden dolayı, KAA'larda kullanılacak olan güvenlik protokolleri bu ağların kendilerine has özellikleri ve "ele geçirilmiş algılayıcılar" göz önüne alınarak tasarlanmış olmalıdır.

2.3. KAA'larda Karşılaşılan Zorluklar

Algılayıcı ağlarında bir altyapının mevcut olmaması, işlemcilerin hesaplama gücünün sınırlı olması, bazı uygulamalarda haberleşen cihazların hareketli olması, bellek ve enerji kapasitesinin ve bant genişliğinin kısıtlı olması bu ağların en büyük problemleridir. Bunlara ilave yetersiz ağ topolojileri ve protokolleri gereğinden fazla veri transferine yol açabilmekte ve bu yüzden lüzumsuz paket kayıpları ve enerji tüketimi meydana gelmektedir. Bu da sistemin yaşam süresini ve performansını olumsuz yönde etkilemektedir.

Algılayıcılar pil ile çalışan ve kısıtlı ömre sahip olan kablosuz iletişim cihazlarıdır, bu yüzden enerji kaynakları kısıtlıdır. Algılayıcı ağlarda, her düğümün enerji kaynağının bitmesi ağın yaşam süresini azaltmaktadır. Yaşam süresinin bu kadar önemli olmasındaki temel sebep, çalışma ortamında bulunan düğümlerdeki enerji kaynaklarının doldurulmasının veya değiştirilmesinin çoğu zaman imkânsız veya aşırı maliyetli olmasıdır. Kablosuz iletişimin sebep olduğu pil enerjisi kaybını dengelemek için kablosuz iletişimin enerjiyi en iyi şekilde kullanacak bir yapıda kullanılması zorunludur.

Algılayıcı düğümleri potansiyel olarak güçlü ve etkili olmalarına rağmen, iletişim ve hesaplama konularında sınırlamalara sahiptirler. Kablosuz omurgalardan (backbone) elde edilen düşük sinyal-gürültü oranları (signal-to-noise ratio) algılayıcı düğümlerini paket kaybından çabuk etkilenir hale getirirler.

Güç ve bant genişliği kısıtlamaları göz önünde bulundurulduğu zaman, algılayıcı düğümleri arasında güvenilir şekilde iletilecek veri miktarında bir sınır bulunduğu gözlenmektedir. Büyük ölçekli pratik algılayıcı ağları kurulurken karşılaşılan en

önemli iki zorluk, verimli ağ topolojilerinin geliştirilmesi ve paylaşılan bilginin etkili biçimde işlenmesidir.

2.4. KAA'ların Uygulama Alanları

Bu aygıtların düşük maliyetleri ve gittikçe küçülen boyutları KAA'ların birçok yerde kullanılır hale getirmiştir. Büyük ölçekli dağıtık (distributed) algılama teknikleri sayesinde gözetleme sistemlerinin geliştirilmesi sağlanmış ve kuruluş şekilleri değişmiştir böylece bu sistemlerin uygulamaları savunma, güvenlik, üretim ve trafik gibi sektörlerde daha yaygın hale gelmiştir.

Düğümlerin ana birimlerinden olan ve uygulamalara temel teşkil edecek çok çeşitli algılayıcı tipleri vardır. Bunlar:

- Sıcaklık ölçümü
- Nem ölçümü
- Hareket algılama
- Aydınlık tespiti
- Basınç ölçümü
- Sismik değer ölçümü
- Görüntü tespiti
- Gürültü algılama/ölçümü
- Canlı/cansız varlık tespiti
- Mekanik gerginlik algılama/ölçümü
- Hız, yön, miktar tespiti/ölçümü
- Çevre izleme
- Ana yurt güvenliği

KAA'ların uygulama alanları, algılayıcı tiplerinin genişliği oranında çeşitlendirilebilmekle beraber, uygulamalar aşağıdaki başlıklar altında toplanabilir (Çizelge 2.2).

Çizelge 2.2. KAA'ların uygulama alanları

Uygulama	
Çevresel Uygulamalar	✓ Orman yangını, sel, deprem, gibi doğal afetlerin ölçümlendirilmiş olarak hızlı bir şekilde ihbar edilmesinde, ✓ Hava kirliliği tespiti ve ayrıntılı rapor alınmasında, ✓ Doğal yaşamın gözlenmesinde
Sağlık Uygulamaları	✓ İnsanların fizyolojik verilerinin uzaktan izlenmesi, ✓ Hastanede bulunan doktorların yerinin ve hastaların durumunun (kalp atışı, kan basıncı vb.) izlenmesi, ✓ Hastanedeki ilaç dağıtımının yönetimi
Ticari Uygulamalar	✓ Küçük çocukların konumlarının aileleri tarafından takip edilmesi, ✓ Güvenlik ihtiyaçları, hırsızlarının tespiti, ✓ Envanter yönetim yardımcı aracı, ✓ Araçların izlenmesi ve tespit edilmesi
Askeri Sistemler ve Uygulamalar	✓ Düşman İzleme, ✓ Alçak mesafe ses radarları, ✓ Denizaltı algılayıcılar, ✓ Personel ve taşıt izleme, ✓ Güvenlik. ✓ Dost kuvvetlerin teçhizat ve cephanesinin izlenmesi, ✓ Savaş alanının gözlenmesi, ✓ Arazi hakkında keşifte bulunma, ✓ Hedefin konumu, sürati gibi hedef bilgilerinin tespiti, ✓ Düşmana verdirilen hasar miktarının tespit edilmesi, ✓ Nükleer, biyolojik ve kimyasal saldırıları ihbarının alınması ya da keşfi
Endüstriyel Otomasyonu	✓ Süreç izleme ve kontrol, ✓ Varlıkların ve değerlerin korunması, ✓ Enerji hatlarının izlenmesi ve bütünlüğünün sağlanması, ✓ Benzin-Gaz üretimi ve taşımacılığı, ✓ Titreşim izleme
Üretim, Depolama ve Taşımacılık	✓ Depolarda ürün takibi, ✓ Isı kontrollü depo otomasyonu, ✓ Trafik İzleme, ✓ Ürün yer tayini, ✓ Güvenlik, ✓ Akıllı Taşıyıcılar, ✓ Otopark otomasyonu.
Yapı Otomasyonu	✓ İzleme ve Kayıt, ✓ Güvenlik, ✓ Yer tayini(çalışan, malzeme, araç vb), ✓ Işıklandırma kontrolü, ✓ Yangın alarmı, ✓ Deprem tahmini.
Çevresel Takip	✓ Tarım, Sulama, Seracılık, ✓ Doğal ortam izleme, ✓ Sel baskını ve yangın tespiti, ✓ Çevresel izleme, ✓ Gıda kalitesi, ✓ Hava durumu, ✓ Hayvancılık.
Sağlık	✓ Sağlık parametrelerini izleme, ✓ Yer tayini, ✓ Düşme tespiti, ✓ Yaşlı ve hasta kişileri izleme.
Güvenlik	✓ Çevre güvenliği, ✓ Sınır güvenliği, ✓ Boru hattı güvenliği, ✓ Sualtı güvenliği.

2.5. KAA'ların Avantajları ve Dezavantajları

KAA'ların en önemli avantajları aşağıda sıralanan başlıklar altında incelenebilir.

- Düşük maliyet,
- Hareketlilik (Mobility),
- Taşınabilirlik (Portability),
- Yeniden kullanılabilirlik (Reusability),
- Ölçeklenebilirlik(Scalability).

KAA'ların en önemli dezavantajları aşağıda sıralanan başlıklar altında incelenebilir.

- Kısıtlı kaynaklar,
- Yönetim ve izlenebilirlik zorluğu,
- Yüksek hata olasılığı,
- Servis kalitesinin düşük olması.

2.6. KAA'larda Güvenlik Atakları

KAA'lar da algılayıcılar bir işbirliği içerisinde çalışmaktadırlar. Dolayısıyla elemanlar kendi aralarında sürekli iletişim halinde olmaktadırlar. Dışarıdan dinlenilmeye son derece açık olan bu tür sistemler güvenlik problemini de beraberinde getirmektedir. Yayın yoluyla yapılan haberleşme dışarıdan dinlenilmeye açık olup saldırıya karşı son derce zayıf bir altyapı sunmaktadır. Ayrıca KAA'lardaki algılayıcı elemanlar sınırlı bant genişliğinde bir iletişim becerisine, sınırlı bir işlemci gücüne, düşük kapasiteli bir hafızaya ve düşük enerjili bir bataryaya sahiptir. Dolayısıyla sınırlı kaynaklara sahip olan KAA'ların yapılarında diğer ağ yapılarında kullanılan etkili güvenlik algoritmaları doğrudan kullanılamamaktadır.

KAA'larda komşu düğümler arasında yayınlanan verilerin gerçekten ilgili kaynak tarafından gönderilip gönderilmediğinin anlaşılması gerekmektedir. Çünkü bu veriler

uygulama dışından gönderilen saldırı amaçlı sahte veriler de olabilmektedir. Dolayısıyla veri doğruluğunun (data authentication) sağlanması gerekmektedir. Veri doğruluğu karşılıklı haberleşen elemanlar arasında bilinen gizli bir anahtar kodla gerçekleştirilmektedir. Bu kod zamanla değişen yapıda ve ağ dışındaki saldırı amaçlı diğer elemanlar tarafından çözümlenemeyecek yapıda olmalıdır. Haberleşme esnasında alınan bir verinin istenilmeyen unsurlarca değiştirilip değiştirilmediğinin de kontrolü gerekmektedir. Bu ise veri bütünlüğünün (data integrity) sağlanması anlamına gelmektedir. Diğer bir güvenlik gereksinimi ise veri tazeliğinin (data freshness) kontrolüdür. KAA'ların yapılarında algılayıcı elemanlar belirli zamanlarda bulundukları ortama ait ölçüm verileri göndermektedirler ve ölçüm değerlerinin ulaştırılma zamanları önemli olmaktadır. Saldırıcı bir unsur tarafından eski ölçüm değerlerinin kopyalarının yeniden yayınlanması söz konusu olabilmektedir. Dolayısıyla verinin güncel bir bilgi olduğunun denetimi önemli olmaktadır.

Kablosuz algılayıcı ağlarda ağ güvenliğine karşı saldırıla iki sınıfa toplanabilir:

- İç ataklar
- Dış ataklar

İç ataklarda saldırgan kişi bir ya da daha fazla algılayıcı düğümünü fiziksel olarak ele geçirir (node compromise). Dolayısı ile saldırgan bu algılayıcı düğümlerine ait tüm gizli anahtar bilgilere sahiptir ve ağın içinden saldırılar düzenleyebilir. Buna karşın dış ataklarda saldırgan ağdaki düğümlere ait gizli anahtar bilgisine sahip değildir ve sadece dışarıdan kendine ait algılayıcı düğümlerini kullanarak KAA'nın çalışmasını engellemeye çalışabilir. Atak tiplerinde olduğu gibi, KAA'larda saldırganları da işlem güçlerine göre iki gruba ayırmak mümkündür:

- Laptop sınıfı saldırganlar
- Algılayıcı düğüm sınıfı saldırganlar

Laptop sınıfı saldırganlar güçlü cihazlara, örneğin büyük batarya, güçlü işlemci, güçlü radyo veya daha hassas antene vb. sahiptirler. Ayrıca laptop sınıfı saldırgan yüksek bant genişliğine ve az gecikmeli iletişim yetisine sahiptirler. Bir laptop sınıfı saldırgan kaynaklarını kullanarak birden çok düğüm gibi davranabilir, iletim ortamını denetim altına alınabilir ve ağın her noktasına erişebilir. Buna karşın algılayıcı düğüm sınıfı saldırganlar sadece yakın çevresindeki düğümlere engel olabilir ve düşük işlem gücü ile birlikte düşük bant genişliğine sahiptir. Bu nedenle laptop sınıfı saldırganlar her zaman algılayıcı düğüm sınıfı saldırganlara göre daha tehlikelidir. Çizelge 2.3, atak ve saldırgan türlerini ile birlikte bunların zarar derecelerini özetlemektedir.

Çizelge 2.3. Atak ve saldırgan türlerinin karşılaştırılması

Saldırgan	**Ataklar**	
	Dış Ataklar	**İç Ataklar**
Laptop sınıfı	Orta Tehlikeli	Çok Tehlikeli
Algılayıcı düğüm sınıfı	Az Tehlikeli	Orta Tehlikeli

KAA'lar her biri farklı görevleri yerine getiren katmanlı bir protokol yapısına sahiptir. Bu katmanlara yapılabilecek saldırı türleri farklılık gösterir. Her katmanın güvenlik açıkları tanımlanarak bu açıklara dayalı olası bazı saldırılara karşı çözüm önerileri Çizelge 2.4'te verilmektedir. Bu bölümün devamında, önce KAA'lardaki önemli atakları özetle açıklandıktan sonra bu araştırmada seçilmiş olan Sinkhole atağı detaylı bir şekilde anlatılmaktadır.

KAA'larda laptop ve algılayıcı düğümü sınıfı saldırganlar birçok iç ve dış atak çeşidi gerçekleştirebilirler.

Çizelge 2.4. KAA'larda katmanlar, ataklar ve savunma teknikleri

katmanlar	Saldırı Türü	Savunma Önlemleri
Taşıma Katmanı	Sel (Flooding)	İşlemci bulmaca (Client puzzles)
	Eşzamanı bozma (De-synchronization)	Doğrulama (Authentication)
Ağ Katmanı	İhmal etme ve Açgözlülük (Neglect and greed)	Fazlalık (Redundancy), Araştırmak (Probing)
	Konumlama (Homing/traffic Analysis)	Şifreleme (Encryption)
	Yanlış Yönlendirme (Misdirection)	Çıkış filtreleme (Egress filtering), Yetkili, İzleme
	Sinkhole { Kara-delik / Solucan	Doğrulama İzleme (monitoring), Fazlalık (redundancy), Evrensel Durum Sistemi(GPS)
Veri Bağı Katmanı	Çarpışma (Collision)	Hata düzeltme kodu (Error correcting codes)
	Tüketme (Exhaustion)	Hız Sınırlaması (Rate limitation)
	Haksızlık (Unfairness)	Küçük Çerçeveler (Small frames)
Fiziksel Katman	Bozma (Jamming)	Yayılma spektrumu (Spread-spectrum), Öncelik mesajlar (Priority messages), Daha düşük görev çevrimi(Low duty cycle) Bölge haritalama(Region mapping) Mod değiştirme (Mode change)
	Kurcalama (Tampering)	Kurcalama-Kanıtlama(Temper-proofing) Saklama (Hiding)

2.7. KAA'larda En Önemli Ataklar

Fiziksel ataklar KAA'lara özgü bir ataktır ve geleneksel ağlarda görülmez. Bu ataklar sonucunda algılayıcılar tamamen kullanılmaz hale getirilebilir, yeniden programlanarak kötücül algılayıcı düğümü olarak kullanılabilir ya da yok edilen düğümler başka kötücül düğümlerle değiştirilebilirler.

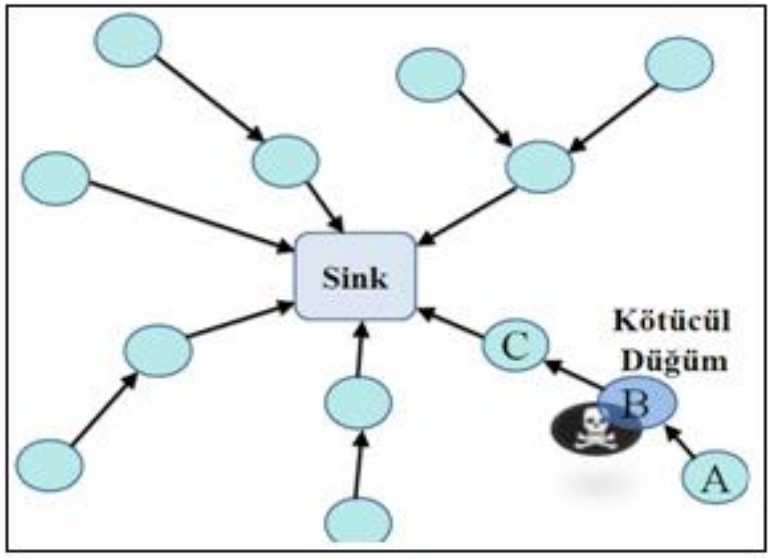
Veri bağı katmanının en önemli görevi iletişimden oluşan hataları düzeltme ve paketleri tekrar göndermedir. Bu nedenle KAA'lardaki ataklar, çarpışma gibi

olaylarla algılayıcıların pillerini erken bitirmesine çaba sarf etmektedirler. Bu katmandaki ataklara karşı hata düzeltme kodu gibi yöntemler kullanılır.

Taşıma katmanı uçtan uca iletimi denetler. KAA'lar bu katmanda iletim yükünü hafifletmek için çok basit protokoller kullanır. Bu sebepten dolayı kötücül algılayıcılar için taşıma katmanında DoS (Denial of Service) atakları gerçekleştirmek çok kolaydır. Bu katmandaki en kolay atak Internet'teki TCP SYN (TCP Synchronization) selinde olduğu gibi sel atağıdır. Sel atağında kötücül düğüm çevresindeki bir kurban düğüme, birçok iletişim başlatma isteği göndererek kurbanın iletişim kaynaklarını tüketmek ister.

2.7.1. Tekrarlama atağı

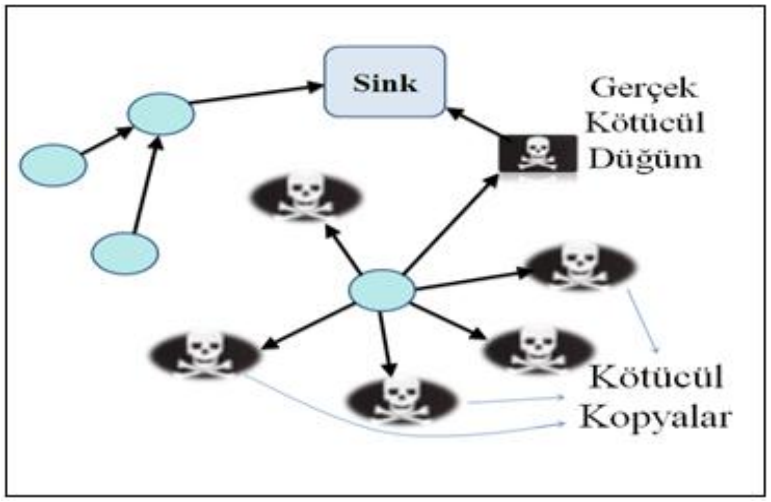
Tekrarlama ataklarında saldırgan düğüm iki düğümün arasında gönderilen mesajları tekrarlayarak ağdaki düğümlerin erken güç tüketmesine ve ağdaki trafiğin yoğunlaşmasına neden olmaktadır. Şekil 2.4'te görüldüğü gibi **B** kötücül düğümü, **A** düğümünden gelen mesajları **C** düğümüne aktardıktan sonra aynı mesajı defalarca tekrarlamaktadır. Bu şekilde **C** düğümünün kaynaklarını boşa harcayarak asıl görevini yerine getirmesini engellemektedir [22-25].



Şekil 2.4. Tekrarlama atağı

2.7.2. Sybil atağı

Sybil atakta bir kötücül algılayıcı düğümü kendisini ağdaki diğer düğümlere birden fazla kimlikle tanıtır. Bu durumda, kurban olarak seçilmiş olan düğüm bu kötücül düğümden gelen mesajları farklı düğümlerden geliyormuş gibi algılar. Kötücül düğüm bu şekilde kurban olarak seçtiği düğümlerin mesaj alıp vermesini engelleyebilir. Dahası, Sybil atak yolu ile bir kötücül düğüm sürekli yanlış bilgi göndererek ağda toplanan bilgiyi fazlasıyla değiştirebilir (Şekil 2.5). Böylece baz istasyonunda yanlış bilgi toplanmasına neden olarak karar verme mekanizmasını yanıltabilir [26-27].

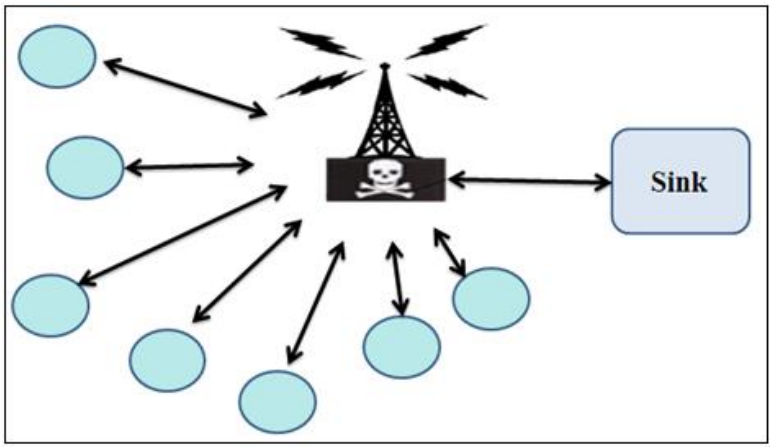


Şekil 2.5. Sybil atağı

2.7.3. Hello-Seli atağı

Düğümler kendilerini komşularına tanıtmak için "Hello" paketleri gönderiler. Taşıma katmanındaki iletişim istek seli gibi, bu atakta da bir kötücül düğüm 'Hello' paketini birçok düğüme göndererek komşularının iletişim yetilerini kısıtlamak istemektedir (Şekil 2.6). Bu tür atakta kötücül düğümün hem yüksek güçte sinyal gönderme kabiliyetine hem de güçlü işlemciye sahip olması gerekir. Ayrıca bu atak uzaktan laptop sınıfı bir saldırgan tarafından gerçekleşiyorsa, düğümler 'Hello' paketini aldıktan sonra uzaktaki düğümü komşu olarak kabul edip baz istasyonuna

gönderilecek mesajları bu düğüm aracılığıyla gönderirler. Mesajları toplayan saldırgan kara delik atağıyla bu mesajları yok edebilir [28].



Şekil 2.6. Hello-seli atağı

2.7.4. Trafik analizi atağı

KAA'ları bir baz istasyonu ile haberleşebilen birçok düşük güçlü algılayıcılardan oluşmaktadır. Düğümler tarafından toplanan veri son olarak çıkış düğümüne yönlendirilir. Saldırılar, ağı kullanılamaz duruma getirebilmek için çıkış düğümünü etkisizleştirir. "*Saldırı izleme oranı*" (rate monitoring attack), çıkış düğümüne yakın olan düğümlerin, uzak olan düğümlere nazaran daha çok paket iletme eğilimini gösterir. Saldırılarda, paket gönderen düğümler izlenilerek en çok paket gönderen düğüm belirlenir. "*Rastgele yürüme gönderme*" tekniği kullanılarak saldırı izleme oranı düşürülebilir. Bu yöntemde, algılayıcının ebeveyn (parent) düğümden başka bir düğüme zaman zaman paket göndererek saldırganın algılayıcıdan çıkış düğümüne açık bir yol belirlemesinin zorlaştırır [29].

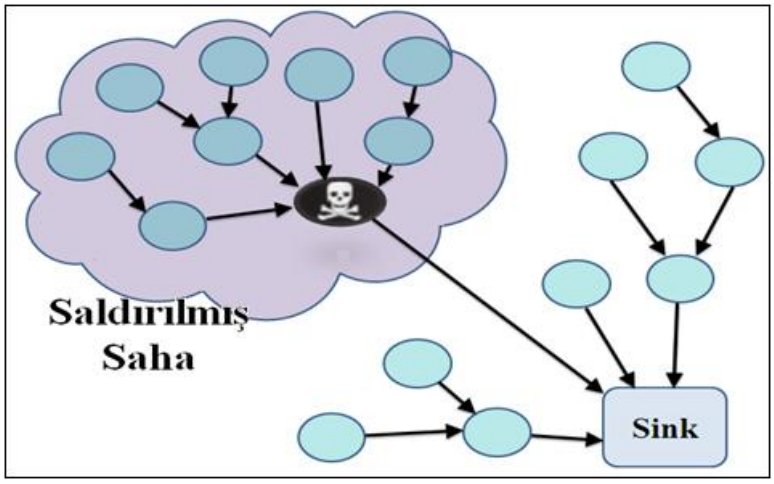
2.7.5. Sinkhole atağı

KAA'larda görülen en önemli atakların birisi Sinkhole atağıdır. Sinkhole atağı KAA'lara özgü bir ataktır ve iki şekilde gerçekleştirilebilir.

- Kara Delik (Blackhole) Atağı,
- Solucan Deliği (Wormhole) Atağı.

Eğer kötücül düğümlerin sayısı bir taneyse atağın adı kara delik atağı ve eğer bir taneden fazlaysa solucan deliği atağı adı verilir.

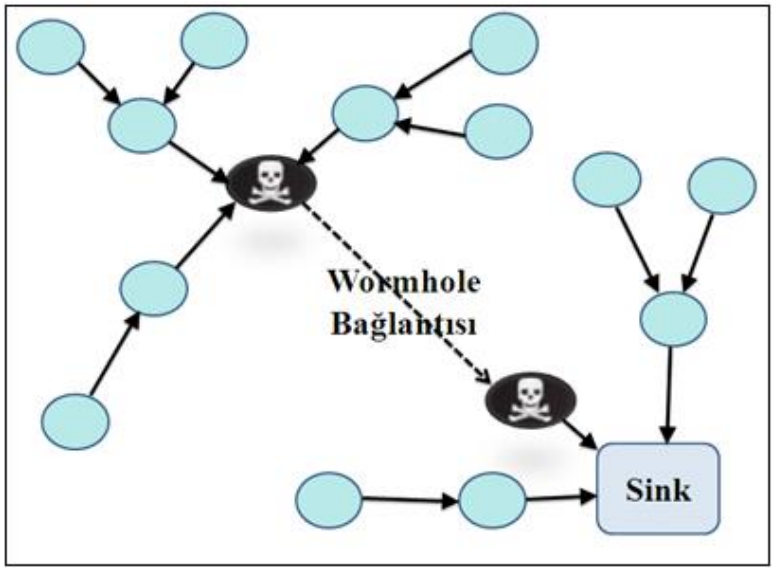
Kara delik atağında bir kötücül düğüm kendini baz istasyonuna en yakın düğüm olarak lanse eder ve çevredeki algılayıcılardan baz istasyonuna gönderilmek üzere veri toplarlar [23]. Gerçekte bu atakta kötücül düğüm baz istasyonundan uzak olabilir. Saldırının amacı toplanan verilerin baz istasyonuna ulaştırılmaması ya da toplanan verinin baz istasyonuna ulaştırılmadan değiştirilmesidir (Şekil 2.7). Bazı zamanlar saldırgan bulunmamak amacıyla paketleri rastgele çöpe atıp ve başka paketleri baz istasyonuna gönderir. Bu durumda bazı paketler baz istasyonuna yetişip bazı paketler yetişmeyecektir. Bazı araştırmalarda bu çeşit atağa, gri delik (Gray-hole) atağı adı verilmektedir [42, 43]. KAA'larda eğer bazı paketler kötücül düğümden dolayı kayıp olursa kötücül düğümler çok zor bulunur [30-33].



Şekil 2.7. Sinkhole atağı bir kötücül düğüm ile oluşturulabilir (kara delik atağı)

Solucan deliği atağında iki kötücül düğüm birbirleri arasında yüksek iletişim kalitesine sahip bir kanal oluştururlar. Bu kanal bir kablolu ya da kablosuz ortam olabilir. Daha sonra yönlendirme için bu kanalın reklamını yaparak çevredeki algılayıcılardan baz istasyonuna gönderilmek üzere veri toplarlar (Şekil 2.8). Ancak

baz istasyonu yakınındaki kötücül düğüm toplanan veriyi baz istasyonuna iletmeyebilir ya da verileri değiştirerek baz istasyonuna gönderebilir. Bu atak bazı zamanlar verileri ağın bir tarafından toplayıp, ağın bir başka tarafına enjekte eder. Böylece ağın etkinliği azalabilir. Ayrıca KAA'larda düğümler arasındaki iletişim güvenli olsa dahi yinede bu atağın olma ihtimali vardır. Bazı zamanlar iki kötücül düğümün arasındaki iletişimin hızını yükseltmek için gönderilen paketin sonunu beklemeden, birinci kötücül düğüm aldığı her bit'i ikinci düğüme gönderebilirler [34-42].



Şekil 2.8. Sinkhole atağı iki kötücül düğüm ile oluşturulabilir (solucan deliği)

3. LİTERATÜR TARAMASI

KAA'larda ki en önemli atakların birisi Sinkhole atağıdır. Sinkhole atağı iki çeşit olabilir. Eğer kötücül düğümlerin sayısı bir taneyse atağın adı kara delik (Blackhole) atağı ve eğer bir taneden fazlaysa solucan deliği (Wormhole) atağı adı verilir. Bu bölümde bu konu ile ilgili önemli çalışmalar özetle anlatılmaktadır. Çizelge 3.1'de bu konuda önemli araştırmalar ve o çalışmalarda kullanılan metotlar gösterilmiştir.

Çizelge 3.1. Sinkhole atakları ile ilgili çalışmalar.

	Yıl	Atak		Modelin Adı	Kullanılan Metot
		Kara delik	Solucan Deliği		
Hu et al.[35, 36]	2001	-	+	Packet Leashes	Paketlerin en çok iletimini sınırlamak
Perrig et al. [16]	2002	+	-	SPINS	Veri gizliliği, Veri Doğrulama, Veri Tazeliği
Deng et al. [44]	2002	+	-	AODV	Orta düğümlerin doğruluğu
Zhu et al. [45]	2003	-	+	LEAP	Dört çeşit anahtar her düğüm için
Ramaswamy et al. [32]	2003	+	-	---	Veri Yönlendirme bilgi tablosu
Capkun et al.[46]	2003	-	+	SECTOR	Her düğüm bir özel donanım ile teçhiz edilmek.
Deng et al. [47]	2003	+	-	---	Destek vektör Makinesi (Support Vector Machine)
Wang et al.[34]	2004	-	+	MDS_VOW	Multi Dimentional Scaling algoritması, Dijkstra algoritması
Karakehayov et al.[30]	2005	+	_	REWARD	Yönlendirme protokolünde iki çeşit mesaj kullanmak
Song et al.[38]	2005	-	+	SAM	Çoklu yol Yönlendirme
Pirzada et al. [41]	2005	+	+	Trust Base	Güven puanı, Komşuları izlemek
Yin et al. [33]	2006	+	-	HSRP	Simetrik Anahtar Şifreleme
Ngai et al[42]	2007	+	+	---	Bilgi Tutarlığı, Şebekenin Akım Bilgisi Analizi
Yun et al. [37]	2007	-	+	WODEM	Sınırlı sayıda düğümleri yer bulma aygıtlara ve uzun ömürlü pil ile teçhiz etmek
Stafrace ve Antonopoulos [49]	2010	+	+	WAHN	Scout Patrol Squad

Hu ve arkadaşları, solucan ve kara delik saldırılarına karşı *Packet Leashes* adlı mekanizma yardımıyla bir çözüm sunmuşlardır [35, 36]. Bu metotta bir paket kaynak düğümden gönderildiğinde, düğümün konum bilgileri ve paketin gönderildiği zaman, paketin başına eklenmektedir. Paket kaynak düğüme vardığında bu zamanlar ve kaynak düğüme varış zamanı araştırılıp ve uyumsuzluk varsa kaynak düğüme bildirilir. Bu metotta tüm düğümler arasında sıkıca saat senkronizasyonu (tightly synchronized clocks) gerekmektedir. Sıkıca saat senkronizasyonu, bu metodun en önemli dezavantajlarından sayılmaktadır. Çünkü KAA'larda düğümlerin atıldığı alan çok geniş olduğundan normalde sıkıca saat senkronizasyonu yapılamaz ve bu işlemi yapabilmek için düğümler bir donanım ile teçhiz edilmesi gerekmektedir.

KAA'lardaki kara delik atağına karşı, Perrig ve arkadaşları veri gizliliği (data confidentiality), veri doğrulama (data authentication), veri tazeliği (data freshness) metotlarından yararlanarak bir çözüm önermişlerdir [16]. Önerilen metotta her pakete 8 bayt eklenmektedir ve önerilen savunma mekanizması sadece kara delik atağına karşı kullanılabilir. Bizim çalışmada her pakete sadece 4 bayt eklenmektedir. Ayrıca kara delik ve solucan deliğine karşı savunma mekanizması kullanılmaktadır.

Deng ve arkadaşları, 2002, 2003 yıllarında kara delik atağına karşı iki farklı metot önermişlerdi [44, 47]. Birinci çalışma, MANET üzerinde yapılmıştır. Bu çalışmada bir güvenli yönlendirme protokolü önerilmiştir. Önerilen metotta kaynak düğüm RREQ paketini birden fazla komşularına gönderip ve birden fazla yoldan paketleri hedefe göndermeye çalışmaktadır. Böylece kötücül düğüm paketleri çöpe atsa bile başka yoldan paket hedefe ulaşabilir. Bu metotta ağdaki düğümlerden sadece bir tane kötücül düğüm olması farz edilmiştir ama gerçek hayatta kötücül düğümler işbirliği yapabilirler. Bu durumda önerilen metot iyi bir performansa sahip değildir. Deng ve arkadaşları tarafından 2003 yılında kara delik atağına karşı Destek Vektör Makinesi tabanlı bir saldırı tespit sistemi önerilmişti [47]. Geliştirilen sistem sinkhole atağını algılamak için kullanılmaktadır. Saldırıyı tespit edebilmek için düğümler arasında bilgi paylaşımından yararlanmışlardır. Bu tez çalışmasında gerçekleştirilen metotta bilgi paylaşımına hiçbir şekilde gerek yoktur. Önerilen mekanizmada kötücül düğümü bulmak için dağıtılmış hiyerarşik (distributed hierarchical) ve tam dağıtılmış

(completely distributed) bulma metodu kullanılmıştır. Bu metodun dezavantajlarından birisi sadece kara delik atağı bulmaya tasarlanması ve diğer dezavantajından, önerilen algoritma tüm normal düğümler üzerinde çalışmasıdır. Ayrıca sistemi hiyerarşik olarak tasarladığımız zaman baş düğümler (head nodes) normal düğümlere göre daha güçlü ve daha iyi kaynaklara sahip olmaları gerekmektedir.

Zhu ve arkadaşları, LEAM (Localized Encryption and Authentication Protocol) adlı mekanizma yardımıyla KAA'larda birkaç çeşit atağa karşı bir mekanizma önermişler [45]. Önerilen mekanizmada, KAA'larda gizlilik ve kimlik doğrulamak için dört çeşit farklı anahtar kullanılmıştır. Birinci anahtar baz istasyon ile düğümler arasında, bir ikili anahtar tüm düğümler arasında, bir küme anahtarı komşu düğümler arasında ve en son bir grup düğümler arasında tüm ağda paylaşılmaktadır. Bu anahtarlar ağın yaşam süresinde sürekli güncelleşmektedirler. Bu metodun avantajı birkaç çeşit atağı önlemek ama çeşitli anahtarların kullanması bu mekanizmanın en önemli dezavantajlarından sayılmaktadır.

Ramaswamy ve arkadaşları, MANET ağlarda kara deliği önlemek için veri yönlendirme bilgi tablosu (data routing information table) oluşturarak ve çapraz kontrolü (Cross Checking) yaparak bir mekanizma önermişlerdir [32]. Bu mekanizmada atağı bulmak için ara düğümler üzerinde sürekli çapraz kontrolü yapılmaktadır. Bu yöntem sadece kara delik atağını önlemektedir. Ayrıca sürekli kontrol yapılma bu metodun dezavantajlarından sayılmaktadır. KAA'larda normal düğümlerin kısıtlı güç kaynağına ve zayıf işlemciye sahiptirler bu yüzden sürekli kontrol yapma bu ağlarda iyi bir yöntem değildir.

Solucan deliği atağına karşı, Capkun ve arkadaşları tarafından önerilen *The Secure Tracking of Node Encounters in Multi-hop Wireless Networks* (SECTOR) protokolü bir kaç metodun bir araya gelmesinden oluşmuştur [46]. SECTOR hiçbir saat senkronizasyonu veya konum bilgisine gerek duymaz ve düğümlerin uzaklığını bulmak için MAD (Mutual Authentication with Distance-bounding) protokolünü kullanır. MAD protokolünde her düğüm bir özel donanım ile teçhiz edilmiştir. MAD

protokolü paketlerin bütünlüğünden emin olmak için Hash-Tree ve One-Way-Hash zincirinden yararlanır. Mesajları doğrulamak için MAC (Message Authentication Code) 'den yararlanır. KAA'larda düğüm sayısı çok fazla olduğundan, düğümlerde kullanılan donanım bu metot için bir dezavantajdır. Çünkü her düğümün özel bir donanım ile teçhiz edilmesi düğümün maliyetini yükseltecektir. Son yıllarda araştırmacılar, savunma mekanizmaları tasarlarken donanım yerine yazılımı tercih etmektedirler.

Solucan deliği atağına karşı, Wang ve arkadaşları 2004 yılında Multi Dimentional Scaling Visualization of Wormhole (MDS_VOW) adlı mekanizma yardımıyla bir çözüm sunmuşlardır [34]. Bu çalışmada her düğüm komşularından gelen sinyal gücünü ölçüp, uzaklığını tahmin ederken bu uzaklıkları baz istasyonuna gönderir. Daha sonra baz istasyonunu da Dijkstra algoritması yardımı ile tüm düğümlerin komşularından uzaklıklarını hesaplar[48]. Baz istasyonu MDS algoritmasını kullanarak düğümlerin gerçek konumlarını bulmaya çalışıp bir düzleme (Smoothing) fonksiyonu kullanarak hataları hesaplar. Eğer elde edilen harita düz olursa şebekede Solucan deliği atağı bulunmamaktadır ama eğer iki düğüm arasındaki yüzey (surface) eğri olursa Solucan deliği atağının olduğunu gösterir. Bu durumda baz istasyonu, Solucan deliği atağının uzaklığını başka düğümlerden hesaplar ve böylece kötücül düğümün konumunu belirttikten sonra normal düğümler kötücül düğümle iletişim kurmazlar. Bu çözümün dezavantajlarından birsi, komşu düğümlerden gelen sinyalin gücünü ölçmeye yarayan donanım kullanılmasıdır. Ayrıca bu çalışmada sadece solucan deliği saldırısına karşı çözüm sunulmuştur.

Kara delik saldırısına karşı, Karakehayov ve arkadaşları, REWARD adlı yönlendirme protokolü yardımıyla bir metot önermişlerdir[30]. Önerilen metotta iki MISS ve SAMBA yayın mesajları (broadcast messages) kullanılmıştır. MISS mesajını kötücül düğümlerin ID'lerini belirtmek için, SAMBA mesajını ise kötücül düğümlerin fiziksel konumlarını tespit etmek için kullanmışlardır. Bu metotta, ağdaki düğümlerde şüpheli davranış bulunduğunda bir dağıtılmış veritabanı oluşturulur ve sözü geçen iki mesajdan yararlanılarak ağdaki kötücül düğümlerin ID'leri ve konumları belirtilir. Bu metodun dezavantajı, normal yönlendirme protokollerine ek

bir paketin eklenilmesidir. Ayrıca önerilen metot, sadece kara delik atağının KAA'larda bulunmasına yardımcı olabilir.

Song ve arkadaşları, Ad-hoc ağlarda solucan deliği saldırısını önlemek için bir istatistiksel analiz yaklaşımı (Statistical Analysis Approach) önerilmişler [38]. Önerilen mekanizmanın ana fikri, bu metodun ağdaki istatistiksel bulunan yolların, solucan deliği ile değiştirilmesidir. Bu metodun dezavantajı paketleri çeşitli yollardan baz istasyona gönderilmesi ve fazla enerji harcamasıdır. Ayrıca bu yöntem KAA değil Ad-hoc ağlarda uygulanmıştır.

Kara delik ve Solucan deliği ataklarına karşı, Pirzada ve arkadaşları, Trust-Based adlı metodunu önermişlerdir [41]. Bu metotta her düğümün benzersiz bir numarası (ID) bulunmaktadır ve her düğümün sekme sayısı (hop-count) belirlenmiştir. Ayrıca tüm paketlere özel bir başlık eklenmiştir. Her hangi bir paket düğümden geçtiğinde, o düğümün ID'si paketin başlığına eklenir. Böylece her bir paketin geldiği yol sürekli belirlenebilir. Her düğüm bir paketi aldığında o paketi, baz istasyonuna yakın olan komşularına gönderir. Böylece baz istasyonu her paketi bir kaç kez alabilir. Her düğüm tüm komşularına bir güven puanı verilmektedir. Eğer bir düğümün komşusu aldığı paketi belli bir zaman aralığında başka bir düğüme gönderirse bu güven puanı bir sayı artacak aksi halde güven puanını bir sayı azalacaktır. Bu durumu anlamak için her düğüm, paketi komşularına gönderdikten sonra komşuların telsizlerini izlemeye başlar ve böylece hangi komşuları paketi gönderip, hangi komşuları paketi göndermediğini anlar. Böylece bir algılayıcı ağda kötücül düğüm bulunur. Bu metodun dezavantajlarından her düğümün ID'sinin pakete eklenmesidir. Ağdaki düğüm sayısı ve kaynak düğümün adım sayısı baz istasyonundan arttıkça paket boyutu artmaktadır. Bu durum enerji harcamayı ağda artmaktadır. Ayrıca komşu düğümleri izlemek de enerji bakımından KAA'larda iyi bir yöntem değildir.

Yin ve Mardia, KAA'larda kara delik saldırısına karşı bir güvenli yönlendirme protokolü önermişlerdir [42]. Önerilen savunma mekanizmasında sadece simetrik anahtar şifrelemesi (symmetric key cryptography) kullanılmıştır. Bu yöntemde (bu çalışmada önerilen yöntem gibi) saldırı tespit sistemi, saldırıyı bulmak için yerel

olarak çalışmaktadır. Bu nedenle başka yöntemlere göre daha hızlı kötücül düğümler bulunmaktadırlar. Önerilen mekanizmada algılayıcılar iki A, B gruba ayrılmışlardır. Grup A düğümler (lider düğümler) grup B (Normal) düğümlere göre daha iyi kaynaklara sahiptirler ve sayıda tüm düğümlerin %10 - %20 oranında düğümlerden oluşmuşlardır. Grup A düğümler birbirinden uzak olmalarına rağmen birbirleriyle iletişim kurma yeteneğine sahip oldukları farz edilmiştir. Ayrıca grup A düğümler Grup B düğümlere göre fazla bilgi içermektedirler. Bu bilgiler, komşu gruplardaki düğümlerin ID'leri ve o gruplardaki lider düğümlerin ID'lerini içermektedir. Böylece bir grupta bir tutarsızlık bulunduğunda o grubun lider düğümü komşu lider düğümler yardımıyla bu tutarsızlığı baz istasyona gönderir. Baz istasyonda durum incelendikten sonra kötücül düğüm ağdan dışlanır. Bu yöntemin en önemli dezavantajı %10 - %20 oranında farklı ve daha yüksek kaliteli düğüm farz edilmesidir. Ayrıca bu yöntem sadece kara delik atağına karşı kullanılmaktadır.

Ngai ve arkadaşları, 2007 yılında Sinkhole atağına karsı bir algoritma geliştirmişlerdir [42]. Bu algoritmada her bir düğümün farklı bir tanımlayıcı ID numarasına sahip olduğu farz edilmiştir. Algoritma iki aşamadan oluşur. Birinci aşamada veri tutarlığı (Data Consistency) yardımıyla şüpheli düğümlerin listesi bulunur hale getirilir. İkinci aşamada, ağ akım bilgisinin (Network Flow Information) analizi yardımıyla bu listeden kötüsel kişiler bulunur. Algoritma, kötücül düğümlerin sayısının birden fazla olduğu ya da işbirliği yapan kötücül düğümlerin bulunduğu durumlarda da başarılı olarak çalışabilmektedir. Bu metodun en önemli dezavantajı saldırı tespit sistemi algoritmasının ağın tamamına uygulanmasıdır. KAA'larda düğüm sayısı çok fazla olduğundan bu tür algoritmalar kısa bir sürede sonuca varamaz. Bu metodun diğer dezavantajı, kötücül düğümü bulmak için algoritmanın ağda sürekli çalışmasıdır. Normalde KAA'larada enerji tasarrufu için algoritmalar sürekli çalışmamaktadırlar.

Yun ve arkadaşları solucan deliği saldırısına karşı bir algoritma geliştirmişlerdir [37]. Bu çalışmada sınırlı sayıda özel algılayıcı düğümünün yer bulma aygıtlarına sahip olduğu kabul edilmiştir. Ayrıca bu düğümlerdeki pillerin normal düğümlere göre daha uzun ömürlü oldukları kabul edilmiştir. Simülasyon sonuçları, yaklaşık 400

düğümlü bir algılayıcı ağda 10 tane yer bulma aygıtına sahip düğüm bulunduğunda solucan deliği saldırılarının %90 oranında tespit edilebildiğini göstermiştir. Bu yöntemin en önemli dezavantajı özel algılayıcı düğümlerden yararlanmak ve bu düğümlerde yer bulma cihazlarının kullanılmasıdır. Ayrıca bu çalışmada sadece solucan deliğine karşı bir savunma mekanizması önerilmektedir.

Stafrace ve Antonopoulos, 2010 yılında Sinkhole atağına karşı bir algoritma geliştirmişlerdir [49]. Bu algoritma iki aşamadan (Keşif ve Devriye gezmek) oluşmaktadır. Keşif aşamasında düğümler arasındaki mesafe bulunur. o mesafeler baz istasyona gönderilir ve bu bilgiler baz istasyonunda araştırıldıktan sonra ikinci aşama (Devreye gezmek) çalıştırılır. Bu aşamada Scout Patrol Squard adlı bir yöntem kullanarak kötücül düğümler bulunmaya çalışılmıştır. Bu metot askeri uygulamalara özel olarak tasarlanmıştır ve J-Sim simülatörü kullanılarak benzetimi yapılıp test edilmiştir. Ama J-Sim simülatörü KAA uygulamalarında iyi performans sağlamamaktadır.

Bu bölümde sinkhole atağı ile ilgili çalışmalar incelenmiştir. Bir sonraki bölümde bu yöntemlerin iyi ve kötü yönlerini göz önüne alarak, KAA'larda bu atağı önlemek için iki savunma mekanizması önerilip mekanizmalardan birisi NS-2 simülatöründe gerçekleştirilmektedir.

4. MATERYAL VE METOT

Bu bölümde önerilen metodun açıklanması, kullanılan benzetim ortamı seçilmesi, bu benzetim ortamında yeni yönlendirme protokolü geliştirmek ve önerilen savunma mekanizmasını AODV protokolünde geliştirmek açıklanmaktadır.

4.1. Önerilen Metot

Bu bölümde Sinkhole saldırısına karşı bu çalışmada önerilen metot açıklanmaktadır. Sinkhole saldırısında kötücül düğümler, ağın işleyişini düzensiz hale getirip performansını düşürebilmek için aşağıdaki işlemleri yapabilirler:

- Gelen tüm paketlerin aktarımına engel olup paketlerin hepsini çöpe atabilirler.
- Gelen bazı paketleri çöpe atıp diğer paketleri normal düğümler gibi aktarabilirler.
- Gelen paketleri değiştirip, daha sonra değiştirilmiş paketleri baz istasyonuna gönderebilirler.
- Ağın normal işleyişine engel olarak performansını düşürmek için sahte paket üretip baz istasyonuna gönderebilirler.

Bu çalışmada önerilen çözüm iki aşamadan oluşmaktadır:

- Atağın olup olmadığını incelemek,
- Kötücül düğümü bulma algoritması.

Bu bölümün devamında bu iki aşama anlatılmaktadır.

4.1.1. Atağın olup olmadığını incelemek

Bu aşamada ağın fiziksel ortamını bir kaç küçük alana ayırıp bu alanlardan gelen verilerin baz istasyonunda birbiriyle karşılaştırdıktan sonra bu verilerin uyumlu olup olmadığı incelenecektir. Bu aşamada yukarıda verilen kötücül düğümlerin gerçekleştirebilecekleri olumsuzluklar göz önüne alınmalıdır. Bunlara:

- Eğer kötücül düğüm, tüm paketleri iptal ederse o zaman paket bir bölgeden asla, baz istasyonuna gelmez.

- Eğer gelen bazı paketleri çöpe atıp başka paketleri, normal düğümler gibi aktarırsa, paketlere bir sıra numarası eklenerek baz istasyonu bu durumu anlayabilir.

- Eğer gelen paketleri değiştirip daha sonra değiştirilmiş paketleri baz istasyonuna gönderirse, mesaj doğrulama teknikleri (Message Authentication Code) kullanarak bu problem çözülebilir. KAA'larda enerji tasarrufu için şifreleme teknikleri pek çok kullanılmaz ama bazı teknikler, örneğin RC4 ya da RC5, KAA'larda kullanabilir [50].

- Eğer kötücül düğüm sahte paket üretip baz istasyonuna gönderirse uygun bir eşik (Threshold) kullanılarak bu durum anlaşılabilir.

Verilerin uyumlu olmaması şebekede Sinkhole atağının olma ihtimalini göstermektedir. Bu durumda Kötücül Düğümleri Bulma (KDB) algoritması kullanılarak değişik bölgelerdeki kötücül düğüm ya da düğümler bulunur.

4.1.2. Kötücül düğümü bulma (KDB) algoritması

Önerilen çözümün birinci aşaması ağın bir bölgesinde problem olduğunu göstermektedir. Bu bölgede tüm şüpheli düğümlerin arasından kötücül düğümü bulmak için aşağıdaki yöntem kullanılmaktadır.

Örneğin saldırılmış alanda, tüm düğümlerden oluşan yönlendirilmiş çizginin adı **G** olsun ve bu düğümlerin sayısı **K** tane olsun. Bu çizginin düğümlerinin adı $\mathbf{N_1}$, $\mathbf{N_2}$, …, $\mathbf{N_k}$ ve bu düğümlerin baz istasyonuna olan uzaklıkları adım sayısı olarak $\mathbf{N_{1h}}$, $\mathbf{N_{2h}}$, …, $\mathbf{N_{kh}}$ notasyonları ile ifade edilsin. Ayrıca bu düğümlerden baz istasyonuna

giden yol üzerindeki ilk düğümler **N_{1n}, N_{2n} , … , N_{kn}** adlandırılmıştır. Bu algoritmada bu düğümler şüpheli düğümler olarak adlandırılmaktadır. Çünkü bu düğümlerden birisi kötücül düğüm olmaktadır.

Bu algoritmada kötücül düğümü bulmak için Saldırılmış Alandaki Veriler (SAV) adlı tablosu, baz istasyonu tarafından oluşturulmaktadır (Çizelge 4.1). Bu çizelgede kötücül düğüm N_x olarak kabul edilmiştir. Kötücül düğümden sonra baz istasyonu yer almaktadır, ($N_{xh} = 0$) bu nedenle kötücül düğümün adım sayısı baz istasyonundan **1** olmalıdır ($N_{xn}= 1$). Kötücül düğüm, algoritmayı aldatmak için kendi verisini baz istasyonuna yanlış gönderebilir. Bu nedenle N_{xn} , N_{xh} değerleri bu çizelgede yanlış olabilir ve önerilen algoritma bu değerlerden bağımsız çalışmak zorunda kalabilir. Başka bir deyişle bu değerler ne olursa olsun algoritmanın bulduğu cevap değişmemelidir.

Çizelge 4.1. Saldırılmış alandaki veriler tablosu (SAV tablosu)

Düğüm	Baz istasyondan uzaklık	Sonraki düğüm
N_1	N_{1h}	N_{1n}
N_2	N_{2h}	N_{2n}
…	…	…
N_x (kötücül düğüm)	N_{xh} (bu veri yanlış olabilir)	N_{xn} (bu veri yanlış olabilir)
…	…	…
N_k	N_{kh}	N_{kn}

Aşağıda bu adımlar, bir örnek ile açıklanmaktadır.

Örnek 4.1. Bir KAA'da saldırılmış sahada 15 normal düğüm ve bir tane kötücül düğüm (12.düğüm) farz edilmiştir (Şekil 4.1). Bu düğüm kendini baz istasyonuna en yakın düğüm olarak bildirip çevredeki algılayıcılardan baz istasyonuna gönderilmek üzere veri toplar. Saldırının amacı, toplanan verilerin baz istasyonuna ulaştırılmaması ya da toplanan verinin baz istasyonuna ulaştırılmadan değiştirilmesidir. Bu örnekte

kötücül düğüm (gizli kalmak için) 6.düğümü, (baz istasyonun yerine) kendinden sonraki düğümü göstermektedir. Bu nedenle kötücül düğümün adımı da baz istasyonunda **1** yerine **4** gösterilmektedir.

KDB Algoritması:

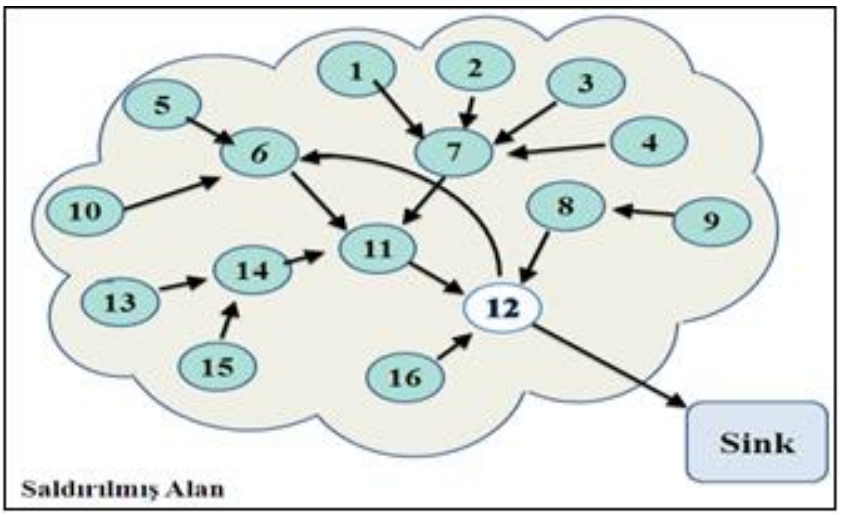
Adım1: Saldırılan sahadaki düğümler için SAV tablosu oluşturulur.

Adım2: Eğer SAV tablosunun 2. sütununda 1sayısı olduğunda, kötücül düğüm bulunmuş sayılır ve 1sayısı olduğu satırdaki düğüm kötücül düğümü göstermektedir ve algoritma bitmiş sayılır.
Aksi halde kötücül düğümü bulmak için algoritma devam etmektedir.

Adım3: SAV tablosunun 2. Sütununun yardımıyla baz istasyonundan 2 adımdaki düğümler Belirlenir. (D_1, D_2, … , D_L)

Adım4: Adım3'deki belirtilen düğümlerden sonraki düğümleri (şüpheli düğümler) belirlenir. Bu işlem SAV tablosunun 3.sütununun yardımıyla yapılır.

Adım5: Şüpheli düğümlerin arasından bir düğüm kötücül düğüm olarak sunulmaktadır. Kötücül düğüm "Oy Çokluğu" metodu kullanarak belirlenir.



Şekil 4.1. Kötücül düğüm bir başka düğümü kendi yerine kötücül göstermektedir.

Bu durumda *örnek 4.1* için SAV tablosu Çizelge 4.2'de gösterilmektedir.

Çizelge 4.2. Örnek 4.1. için SAV tablosu

Düğüm	Sink'ten uzaklık	Sonraki düğüm_ID
N_1	4	7
N_2	4	7
N_3	4	7
N_4	4	7
N_5	4	6
N_6	3	11
N_7	3	11
N_8	2	12
N_9	3	8
N_{10}	4	6
N_{11}	2	12
N_{12}	1→4	0→6
N_{13}	4	14
N_{14}	3	11
N_{15}	4	14
N_{16}	2	12

SAV tablosunun 2. sütununda, 1 sayısı olmaması nedeniyle algoritma 2. adımda bitmez ve 2. adımdan devam eder. Algoritmanın 3. adımında şüpheli düğümlerin listeleri düzenlenmektedir (Çizelge 4.3).

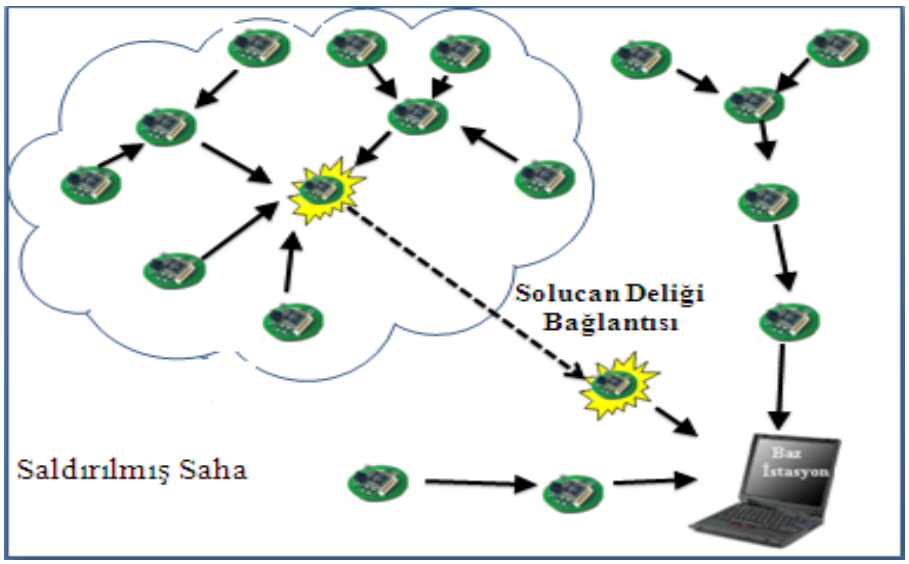
Çizelge 4.3. Örnek 4.1. için bulunan şüpheli düğüm

2.Adımdaki Düğümler	Sink'ten Uzaklık	Şüpheli Düğüm
N_8	2	12
N_{11}	2	12
N_{16}	2	12

Çizelgede görüldüğü gibi 2. adımdaki düğümlerin hepsi 12. düğümü kötücül düğüm olarak göstermektedirler. Eğer kötücül düğüm kendini 2. adımda gösterse de algoritma yine doğru çalışacaktır. Bu durumda 3 tane düğüm 12. düğümü kötücül olarak gösterecek ve sadece bir tane düğüm (kötücül düğüm) bir başka düğümü kötücül gösterecektir.

4.2. Önerilen İkinci Metot

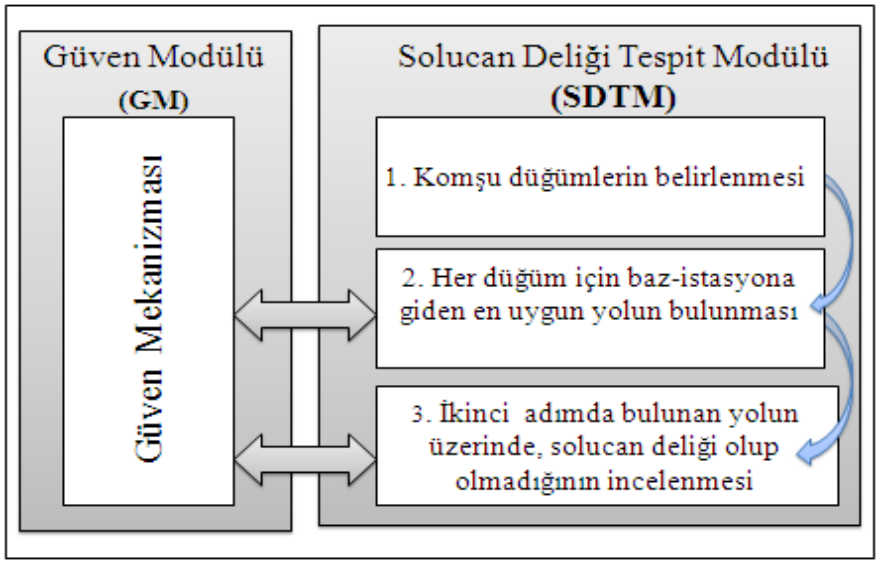
Solucan deliği saldırısında iki kötücül düğüm birbirleri arasında yüksek iletişim kalitesine sahip bir kanal oluştururlar. Bu kanal bir kablolu ya da kablosuz ortam olabilir. Daha sonra yönlendirme için bu yüksek kaliteli kanalın reklamını yaparak çevredeki algılayıcılardan baz istasyonuna gönderilmek üzere veri toplarlar (Şekil 4.2). Amaç, toplanan bu verinin baz istasyonuna erişiminin engellenmesi ya da bozularak iletilmesidir. Bu sebeple baz istasyonu yakınındaki kötücül düğüm toplanan veriyi baz istasyonuna iletmeyebilir ya da verileri değiştirerek baz istasyonuna gönderebilir. Saldırı sonucunda ağın veri toplama performansı ve toplanan verinin doğruluğu/hassasiyeti azalır [54].



Şekil 4.2. Solucan deliği (wormhole) saldırısı

Bu çalışmada solucan deliği saldırısına karşı önerilen çözüm tabanlı bir sistem olup, aynı zamanda yanlış bilgi veren kötücül düğümleri tespit edebilmek için bir güven mekanizması da içermektedir. Literatürdeki zaman tabanlı solucan deliği bulma mekanizmaları kötücül düğümlerin yanlış bilgi aktarma ihtimallerini göz önüne almadıklarından saldırı tespit etkinlikleri düşüktür. Bu çalışmada zaman tabanlı bir sistem ile düğümler arasındaki güven ölçümünü yapabilen bir güven mekanizması ile birleştirilerek bu problem ortadan kaldırılmıştır [55].

Önerilen çözüm paralel çalışan iki modülden oluşmaktadır: "Güven Modülü (GM)" ve "Solucan Deliği Tespit Modülü (SDTM)". Sistem yapısı Şekil 4.3'te gösterilmiştir. SDTM üç aşamadan oluşmaktadır.



Şekil 4.3. Önerilen çözümün sistem yapısı

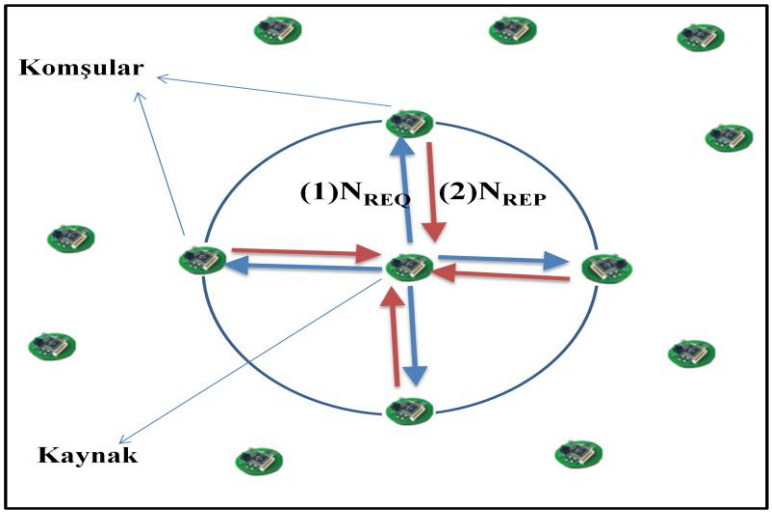
Solucan Deliği Tespit Modülü (SDTM):

Bu modülün birinci adımında, her düğüm için komşu düğümler belirtilmektedir. İkinci adımda ise her düğüm için baz istasyonuna giden en uygun yol bulunmaktadır. Üçüncü adımda, baz istasyona giden yollar üzerinde solucan deliği olup olmadığı incelenmektedir. Ancak, yol üzerindeki kötücül düğümler yanlış bilgi vererek

SDTM'yi yanıltıp solucan deliği saldırısının tespitini önleyebilirler. Bunu engellemek için GM sürekli SDTM'nin ikinci ve üçüncü aşamasını gözlemleyerek, baz-istasyona giden yollar üzerindeki düğümler için güven/itibar değerleri hesaplar. Bu güven/itibar değerleri kullanılarak solucan deliği saldırısı ihtimali olan yolların değiştirilmesi sağlanır [56].

Komşu düğümlerin belirlenmesi:

Bu aşamada her düğüm N_{REQ} - Neighbor Request adlı özel bir mesajı tüm komşularına gönderir. N_{REQ} mesajını alan bütün komşu düğümler, bu mesajın cevap olarak N_{REP} - Neighbor Reply mesajını gönderir. N_{REQ} mesajını gönderen düğüm N_{REP} mesajları aldıktan sonra komşularını tanır ve bunlarla bir yol oluşturur ve komşuların sayısı belirtilir. Örneğin Şekil 4.4'teki düğüm, 4 tane N_{REP} aldıktan dolayı 4 tane komşusu olduğunu fark etmektedir. Bu çalışmada bu sayı N ile gösterilmiştir.



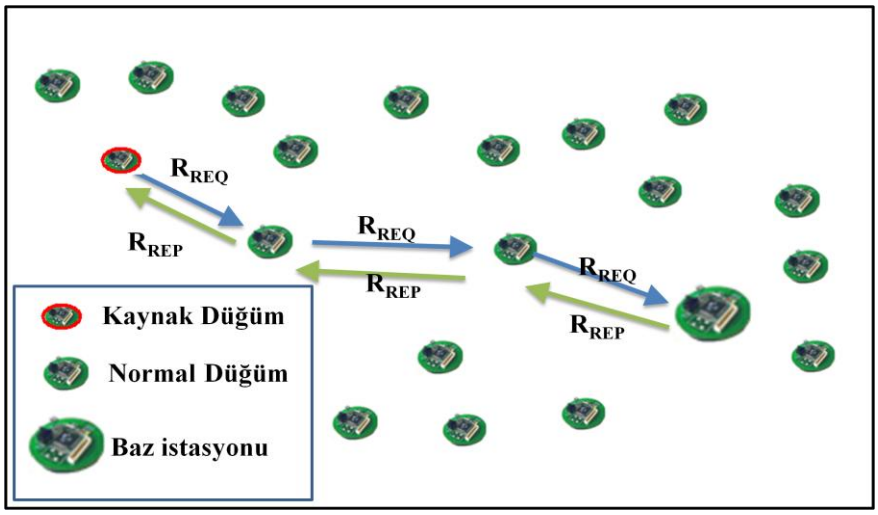
Şekil 4.4. Komşu düğümlerin belirlenmesi

Baz-istasyonuna giden en uygun yolun bulunması:

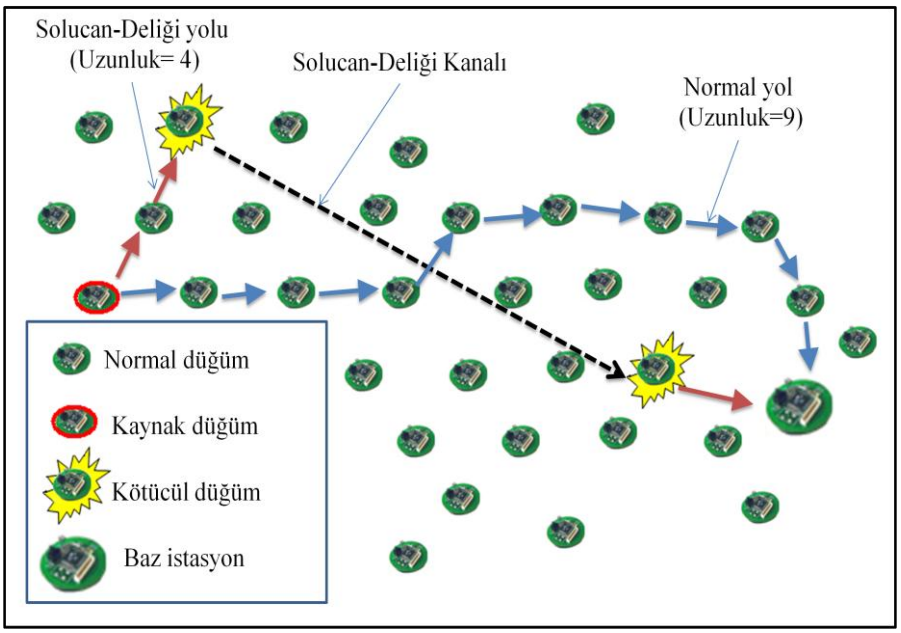
Bu aşamada her düğüm kendisi ile baz istasyonun arasındaki en kısa yolu bulmaktadır. Bu işlemi yapabilmek için her düğüm Yol Bulma (R_{REQ} – Route Request) paketi tüm komşularına gönderir ve bu mesajın gönderilme zamanını kayıt eder (T_{REQ}-Time of Request). Bu paketi alan komşular bu paketi sonraki düğüme gönderir ve onlarda bu paketi gönderme zamanını kayıt ederler. Böylece bu paket ağda yayılır ve sonunda bu paket baz istasyonuna ulaşır. Baz istasyonu bu paketi aldıktan sonra Yol Yanıtı (R_{REP} – Route Reply) paketi ile R_{REQ} paketini gönderen düğümü yanıtlar. Bu paket, kaynaktan baz istasyonuna kadar tüm bulunan düğümleri içermektedir. Bu paket her düğüme vardığında o paketin yanıt zamanını (T_{REP} -Time of Reply) kayıt eder. Her düğümde bu paketin gidiş dönüş zamanlarının farkı (RTT-Round Trip Time) kayıt edilmektedir (Eş. 4.1) [57].

$$RTT=T_{REP}-T_{REQ} \tag{4.1}$$

Tüm aradaki düğümler RTT bilgisini hesapladıktan sonra baz istasyonuna göndermektedir. R_{REP} paketinin kaynağa ulaşması baz istasyonu ile kaynak arasında bir yol oluşturduğu anlamına gelir (Şekil 4.5). Baz istasyonuna gelen her R_{REQ} mesajını yanıtladığı için, kaynak gelen R_{REP} mesajlarının RTT değerlerini inceledikten sonra baz istasyonu ile arasındaki en kısa yolu seçmektedir. Şekil 4.6'da gösterildiği gibi yol seçimi yapılırken kötücül daha kısa bir yolun reklamını yaparak solucan deliği saldırısı gerçekleştirebilir ve paket iletimini engelleyebilir ya da içeriklerini değiştirebilirler. SDTM'nin üçüncü basamağı bu saldırıları tespit eder.



Şekil 4.5. Yol bulma paketleri



Şekil 4.6. KAA'larda bir solucan deliği saldırısı

Solucan deliği saldırılarının tespiti:

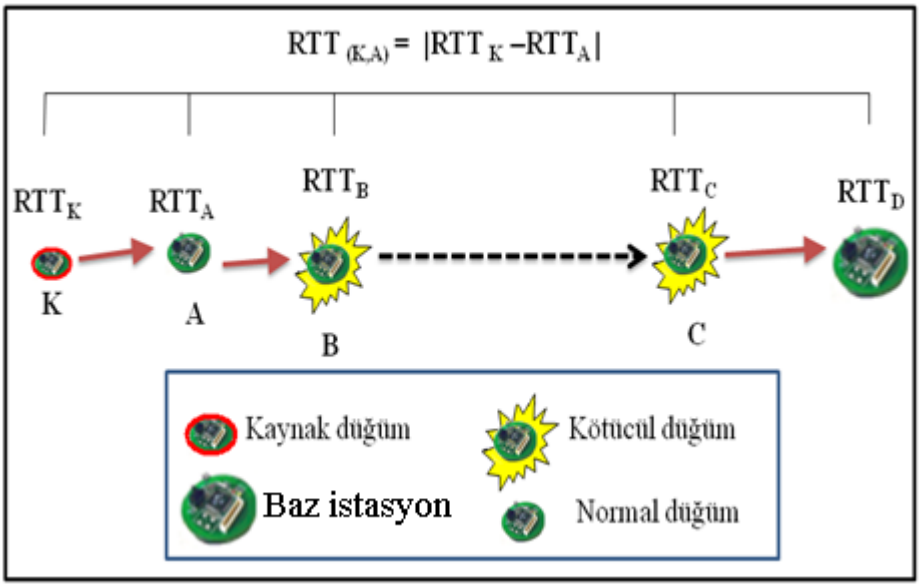
Bu aşamada kaynak düğüm yoldaki bütün düğümlerin gönderdiği RTT değerlerini inceleyip bu yolda solucan deliği saldırısı olup olmadığını anlamaktadır. Bu işlemi yapabilmek için yol üzerindeki her iki ardışık düğümün arasındaki RTT farkından faydalanılır. Normal düğüm için bu fark bir eşik (threshold) değerden fazla olmaması

gerekir. İki ardışık düğüm A ve B arasındaki RTT farkı ($RTT_{(A,B)}$) ile tanımlanmaktadır (Eş. 4.2).

$RTT_{(A,B)} = |RTT_A - RTT_B|$ (A, B ardışık düğümler) (4.2)

$RTT_{(A,B)}$= Paketlerin A düğümünden B düğümüne ulaşma zamanı

Eğer iki ardışık düğümün RTT farkları başka düğümler arasındaki RTT farkından çok fazlaysa bu solucan deliği saldırısının bir göstergesidir. Kötücül düğümler arasındaki mesafe uzun olduğundan kötücül düğümler arasındaki RTT farkı diğer düğümlere göre fazla olacaktır. Şekil 4.7'deki örnekte görüldüğü gibi $RTT_{(B,C)}$, yol üzerindeki başka RTT'lere göre çok daha büyüktür. Çünkü B, C düğümleri arasındaki mesafe diğer düğümlerin arasındaki mesafelerden daha uzundur [57, 58].



Şekil 4.7. RTT değerlerinin farkı iki normal düğüm ve iki kötücül düğüm arasında

Önerilen zaman tabanlı solucan deliği bulma mekanizması, kötücül düğümlerin arasındaki paket iletim zamanından yararlanmaktadır. Eğer iki düğüm arasındaki paket iletim zamanı bir belli eşikten fazlaysa bu bir aykırılık (anomaly) demektir. Bu durumda kötücül düğümler eğer kaynağa yanlış bilgi iletilirse zaman tabanlı algoritma her zaman doğru sonuca varamaz. Bu problemi ortadan kaldırabilmek için

bu çalışmada zaman tabanlı bir modül ile düğümler arasındaki güven (itibar) tahmin edebilen güven mekanizması ile birleştirilmiş ve böylece solucan deliği saldırıları daha yüksek performansta bulunabilmiştir. Bu işlemi yapabilmek için bir izleme modülü tasarlanmıştır. Bu modül bir güven mekanizmasını içermektedir. Bu bölümün devamında bu mekanizma anlatılmaktadır.

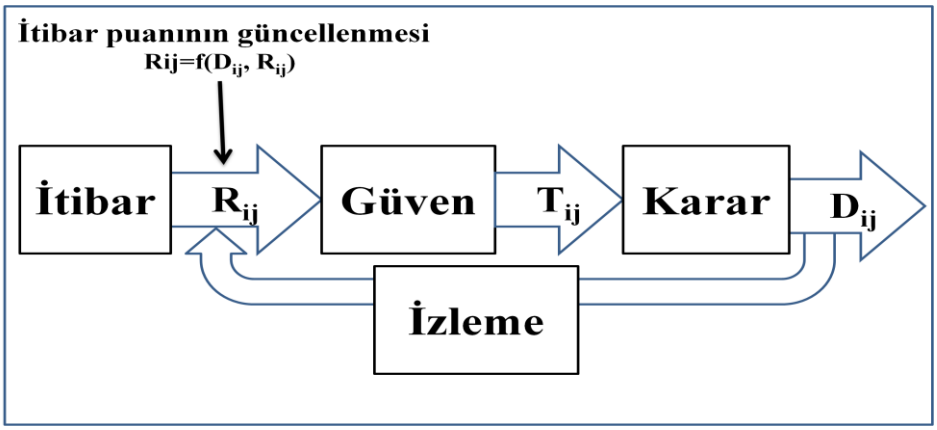
Güven Mekanizması:

Çizelge 4.4. Kullanılan simgeler ve açıklamaları

Simge	Açıklama
RTT	Paketin gediş dönüş zamanı (Round Trip Time)
T_{REP}	Paketin yanıt zamanı (Time of Reply)
T_{REQ}	Paketin gönderme zamanı (Time of Request)
R_{REQ}	Yol bulma Paketi(Route Request Packet)
R_{REP}	Yol Yanıt Paketi (Route Reply Packet)
R_{ij}	j'inci komşuya i'inci düğümün verdiği itibar Puanı $0 \leq R_{ij} \leq 1$
D_{ij}	j'inci komşuya i'inci düğümün verdiği en son Karar $K_{ij}=0$ yada $K_{ij}=1$
T_{ij}	j'inci komşuya i'inci Düğümün güven puanı $0 \leq T_{ij} \leq 1$
f()	f fonksiyonu R_{ij} değerini güncellemektedir
θ	Gelecek iletişimin başarı olma ihtimali
α_{ij}	j'inci komşuya i'inci düğüm gönderen paketlerin başarılı olan paketlerin sayısı
β_{ij}	j'inci komşuya i'inci düğüm gönderen paketlerin başarısız olan paketlerin sayısı

Eğer kötücül düğümler yanlış bilgi göndermediği takdirde, SDTM'nin üçüncü aşamasında ağda solucan deliğinin olup olmadığı kolayca anlaşılmaktadır. Ancak kötücül düğümler tarafından paketlerdeki RTT değerleri değiştirilirse ve yanlış değerler kaynağa ulaşırsa SDTM'nin performansı azalmaktadır. Bu bölümde kullanılan simgeler ve açıklamalar Çizelge 4.4'te verilmiştir. GM her düğümde sürekli çalışmaktadır. GM çalışırken komşulardan alınan RTT değerleri, beklenen eşik değeri ile karşılaştırılıp her komşuya bir itibar (Reputation) puanı (R_{ij} = j'inci komşuya i'inci düğüm tarafından verilenen itibar puanı) verilmektedir. Güven puanı zaman içerisinde değiştirilmektedir. Başlangıçta tüm düğümler için 0.5 değeri alan güven puanı kötücül düğümlerin gönderdikleri her eşik değere göre yanlış RTT için

beta dağılımına göre düşürülür (beta dağılımı birazdan açıklanacaktır). Güven puanının 1 değerini alması "güvenin tam olması" anlamındadır. Başka söyleyişle bir paketi bu komşu aracı ile göndermek güvenilirdir ve bu yolda solucan deliği saldırısı olma ihtimali azdır. Güven puanının sıfır olması "asla bu komşu güvenilir değil" anlamındadır. Başka bir değişle bir paketi bu komşu aracı ile göndermek güvenilir değildir ve bu yolda saldırı olma ihtimali yüksektir. GM ile komşular sürekli izlenir ve itibar puanları sürekli güncellenir (Şekil 4.8).



Şekil 4.8. Güven modülünde itibar puanı güncellenmesi

Şekil 4.8'de görüldüğü gibi R_{ij} izlemenin sonuçlarına göre sürekli güncellenmektedir (Eş. 4.3). Güven puanı (T_{ij}), itibar puanın (R_{ij}) beklenen değeridir (Eş. 4.4).

$$R_{ij}=f(D_{ij},R_{ij}) \tag{4.3}$$

$$T_{ij}=E(R_{ij}) \tag{4.4}$$

Güven modellerini ifade etmek için birçok matematiksel dağılımdan yararlanılabilir. Örneğin, Beta, Gaussian, Poisson, Binomial dağılımları vb. Bu çalışmada sözü edilen dağılımlardan esneklik ve kolaylık göz önüne alınarak Beta dağılımı seçilmiştir. Beta dağılımı iki parametre ile (α, β) ifade edilir (Eş. 4.5) [59].

$$P(x)=\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\times\Gamma(\beta)}x^{\alpha-1}\times(1-x)^{\beta-1}\quad \forall 0\le x\le 1,\alpha\ge 0,\beta\ge 0 \tag{4.5}$$

Bu çalışmada beta dağılımından yararlanmak için α başarılı olan iletişimleri ve β başarısız olan iletişimleri ifade etmektedir. Şimdiye kadar n+m paket farz edilmiştir. Başarılı olan iletişimlerin sayısı **n** ve başarısız olan iletişimlerin sayısı **m** olsun. Sonraki iletişimde başarılı bir iletişim olma ihtimali(θ) ne kadardır? Geçmişten bir bilgi olmazsa θ bir üniforma dağılım olmalıdır ($0 \le \theta \le 1$). Bu nedenle P(θ) = uni (0,1) = beta(1,1) denilebilir. Buna göre gelecek seferdeki iletişimde başarılı olma ihtimali Eş 4.6'daki gibi yazabilir [59].

$$P(\theta) = \frac{Bin(m+n,m) * Beta(1,1)}{Normalization} = Beta(m+1,n+1) \tag{4.6}$$

Formül 6 gösteriyor ki θ'nin gelecekteki dağılımı beta dağılımı olabilir. R_{ij} (i'inci düğümün j'inci düğüm için hesapladığı itibar değeri) beta dağılımı yardımıyla Eş 4.7 deki gibi yazılmaktadır.

$$R_{ij} = Beta(\alpha_{ij} + 1, \beta_{ij} + 1) \tag{4.7}$$

Eğer daha önceden hiçbir bilgi olmazsa $\alpha_{ij} = \beta_{ij} = 0$ olmalıdır. Bu nedenle görüldüğü gibi $R_{ij} = uni(0,1)$ olmalıdır (Eş. 4.8).

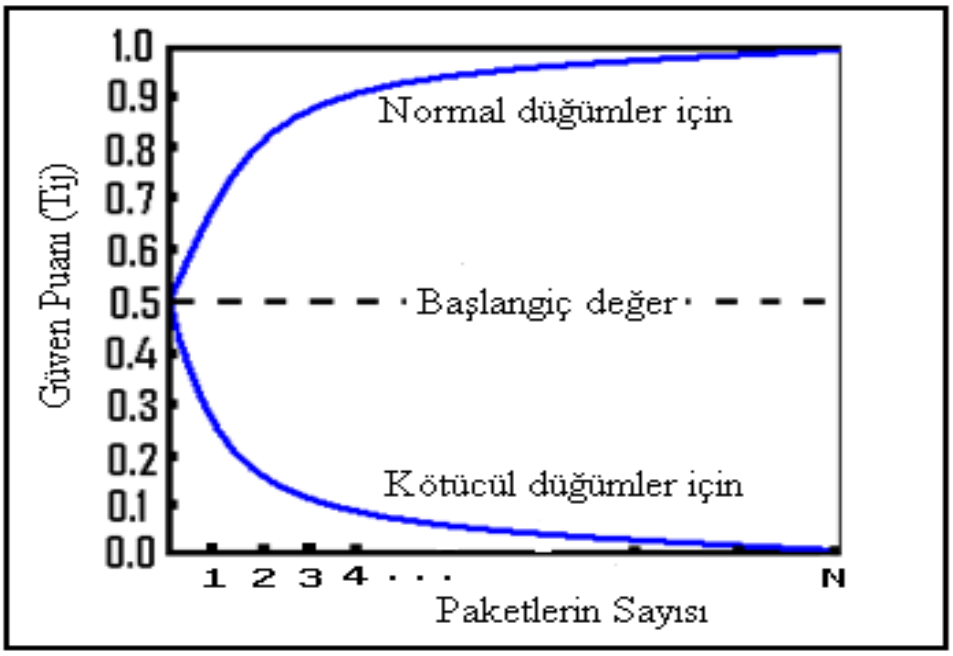
$$R_{ij} = Beta(\alpha_{ij} + 1, \beta_{ij} + 1) = Beta(1,1) = uni(0,1) \tag{4.8}$$

Güven (T_{ij}), itibarın (R_{ij}) istatistiksel olarak beklenen değeridir ve Eş. 4.9 gibi ifade edilebilir [59].

$$T_{ij} = E(R_{ij}) = E(Beta(\alpha_{ij} + 1, \beta_{ij} + 1)) = \frac{\alpha_{ij} + 1}{\alpha_{ij} + \beta_{ij} + 2} \tag{4.9}$$

Eğer bir düğümün komşusu tüm paketlerin iletmesine tam yardım ederse (yolda saldırı olmama durumunda) ve RTT değerlerini belirlenen eşik değerine yakın

verirse bu komşu düğümün güven puanı en sonunda bir olmaktadır. Başka söyleyişle paketlerin gönderildiği yolda kötücül düğüm bulunmamaktadır. Aksi takdirde eğer bir komşu asla paketlerin iletimi için yardım etmezse, güven puanı sıfır olmaktadır. Şekil 4.9'da görüldüğü gibi güven puanının başlangıç değeri tüm komşulara 0.5 farz edilmiştir. Sistem çalıştıktan sonra bu puan her komşu için değiştirilmektedir.



Şekil 4.9. Güven puanının değiştirilmesi

GM modülü tarafından üretilen güven puanlarından faydalanılarak, SDTM'nin üçüncü adımında kötücül düğümlerin yanlış bilgi verdiği durumlarda da solucan deliği saldırıları tespit edilmektedir. Paket gönderiminde rol alan bütün düğümler periyodik olarak komşularını gözlemler ve onlara ait güven puanlarını hesaplar. Her düğüm kendi komşularının güven puanlarını ve RTT değerlerini inceler, eğer RTT değerlerinde herhangi bir anormallik varsa o komşuya ait güven puanını düşürür. Böylece her düğüm komşularının solucan deliği saldırısında yer alıp almadığını tahmin etmeye çalışır. Eğer bir düğümün güven puanı 1'e yaklaşıyorsa solucan deliği saldırısında yer alma ihtimali 0'a yaklaşmaktadır. Güven puanlarına göre solucan deliği saldırısında bulunduğu düşünülen düğümler paket gönderiminde kullanılmaz ve böylece paketlerin baz istasyona ulaşma olasılığı artırılır.

4.3. Benzetim Ortamları

Bu bölümün amacı KAA'larda kullanılan simülatörleri incelemek, onların iyi ve kötü özelliklerini belirtmek ve çalışmada en uygun simülatörü seçmektir. Bu amaca varmak için KAA'larda en çok kullanılan simülatörler incelenmiş ayrıca bu simülatörlerin avantajları ve dezavantajları belirtilmiştir. Günümüzde kablosuz ağlara yönelik çeşitli simülatörler bulunmaktadır. Bu simülatörler farklı ağ türlerine odaklanmıştır ve bu yüzden farklı mimarilere sahiptirler. Ağ uygulamalarında kullanılan simülatörlerin maalesef bazıları ticari amaç için tasarlanmış, bazıları iyi bir dokümana sahip değildirler ve bazılarının ise kullanılışı zordur. Ancak bu simülatörlerde bazı ortak özellikler de bulunmaktadır.

Bu çalışmada geliştirilecek algoritmaların uygulanacağı simülatörün seçilebilmesi için, KAA simülatörlerinin incelenmesi yapılmıştır. KAA'larda gerçek uygulama yapma pahalı, karmaşık ve hatta imkansız olabilir. Ayrıca çoğu denemeler tekrarlanabilir değildir. Özellikle KAA'ların gerçek uygulamalarında, düğüm sayısı çok fazla olduğundan simülatörleri kullanmak kaçınılmaz bir gereksinimdir.

Günümüzde var olan simülatörlerin hiçbirisi tam olarak KAA'ların tüm uygulamalarına uygun bulunmamaktadır. Örneğin KAA'larda algılayıcılar için uygun model bulunmamaktadır. Ayrıca bazı simülatörler ticari amaçla tasarlanmıştır ve bu simülatörleri satın almak gerekir. Bu nedenle bu simülatörler bilimsel araştırmalarda kolayca kullanılmamaktadır.

Yaygın kullanılan simülatörlerden olan Ns-2 KAA'ları desteklememektedir ancak bazı uzantıları (extension) KAA'ları desteklemektedir. Ns-2 kullanılması zor olan simülatörlerden biri sayılmaktadır. Glomosim KAA'lar için tasarlanmıştır ama düğüm sayısı fazla olduğunda iyi performansa sahip değildir. OPNet ölçeklenebilir bir simülatördür ancak KAA'ları desteklememektedir. Ayrıca ticari bir simülatördür. SensorSim NS-2'nin bir uzantısıdır ve KAA'ları desteklemektedir. Ancak iyi bir dokümana sahip değildir ve ticari amaçla tasarlanmıştır.

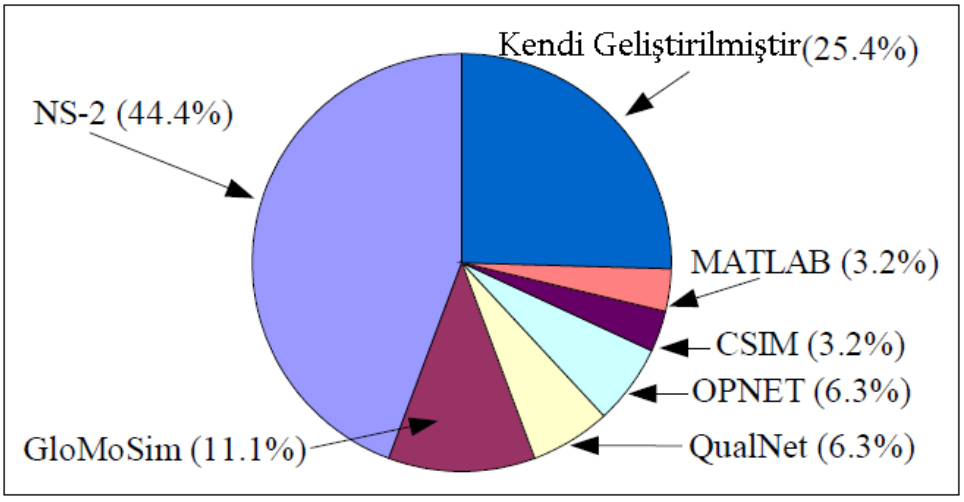
Çizelge 4.5. Önemli kablosuz ağ simülatörlerin özellikleri

Simülatör	Ticari olmama	Ölçeklenirlik	Yazılım Dili	KAA'ları Destekleme	Kablosuz ve Kablolu ortam	Grafiksel Aracı olma	Yeni Protokol Eklemek	Başka Özellikler
AdventNet	_	+	C++	_	+	+	_	Kullanım kolay, Bit tabanı
Avrora	_	+	Java	_	_	_	_	Desteklenmemektedir(2005)
Atemu	_	_	XML/Assembly/nesC	_	+	+	_	Düğüm sayısı≤120, Desteklenmemektedir(2004)
CNet	+	_	C	_	+	_	_	Linux ortamında, Kullanımı sade, kütüphane zayıf
EmStar	_	+	nesC / C++	+	+	_	_	Simülatör/Emulator ama kütüphane zayıf, konfigürasyon dosyasına çok zor düzen verilemektedir
GLOMOSIM/ Qualnet	+/-	+	C++/ Parsec	_	+	+	+	2000 yıldan sonra ticari ürün olarak sunulmuştur
JSN	+	_	Java	+	+	-/+	+	Ns-2 nun Java Sürümü, kütüphane zayıf, Desteklenmemektedir (2000).
J-Sim	+	_	Java/Jacl	+	+	+	+	Ns-2'dan sonra en çok kullanılan simülatör, KAA uygulamalarında iyi performans sağlamamaktadır.
NS-2	+	+	C++ / oTcl	-/+	+	+	+	Bilimsel çalışmalarda en çok kullanılan simülatördür, JNS bu simülatör üzerinden ama Java dilinde tasarlanmıştır.
OMNeT++	+	+	C++/ NED	-/+	+	+	+	Windows işletim sisteminde çalışmaktadır, SensorSim bu simülatörün yeni versiyonudur, KAA'lara uygun tüm protokoller tasarlanmamış, iyi bir dokümana sahip değildir.
OPNet	_	+	C++	_	+	+	_	Açık kaynak ama uygulamaların kodu açık değildir, kütüphanesine çok az protokol bulunmaktadır. Ticaridir.
SENS	_	+	C++	+	+	+	_	Simülatör/Emulator, bazı protokolleri
Sense	_	_	CompC++	+	+	+	+	KAA'lar için tasarlanmıştır, iyi bir performansa sahip değildir.
SensorSim	+	+	C++	+	+	+	_	Ns-2nun genişletmesi sayılmaktadır. Ticaridir.
Sidh	_	+	Java	+	+	+	+	Kütüphanesine çok az protokol bulunmaktadır, iyi bir performansa sahip değildir.
Tossim	+	_	C++ /nesC/oTcl	+	+	+	+	Bit düzeyinde bir simülatör, Simülatör/ Emulator, Düğüm sayısı≤100.

Tossim bit düzeyinde bir simülatördür ancak 100 düğümden fazla olduğunda çok yavaş çalışmaktadır. QualNet ve Opnet simülatörleri araştırmalarda çok nadir kullanılmıştır ve araştırmanın sonuçlarını başka araştırmaların sonuçlarıyla karşılaştırmak imkansızdır. Çizelge 4.5'te KAA'larda kullanılan simülatörlerin önemli özellikleri gösterilmektedir.

Sözü geçen simülatörlerden başka, birçok simülatörlerde ağ uygulamalarında bulunmaktadır. LSU, Mannasim, MoteLab, Parsec, Prowler, QualNet, Real, VisualSense, Shawn, WISENET, GTNetS, Tossf bunlardan bazılarıdır. Bu simülatörlerin daha çok eksiklikleri bulunduğundan bu çalışmada yer verilmemiştir. Önemli kablosuz ağ simülatörlerin avantajları ve dezavantajları EK-1'de bulunmaktadır.

KAA'larda kullanılan simülatörlerin incelenmesinin sonucunda, NS-2 simülatörü bu çalışmaya en uygun simülatör olarak bulunmuştur. NS-2 simülatörü, kullanılması zor olan simülatörlerden biri olarak sayılmaktadır. Ayrıca son yıllarda kablosuz ağlar uygulamalarında en çok kullanılan simülatördür (Şekil 4.10) [51].



Şekil 4.10. NS-2 KAA'larda en sık kullanılan simülatör olarak tanımlanmıştır.

4.4. NS-2 Benzetim Ortamı

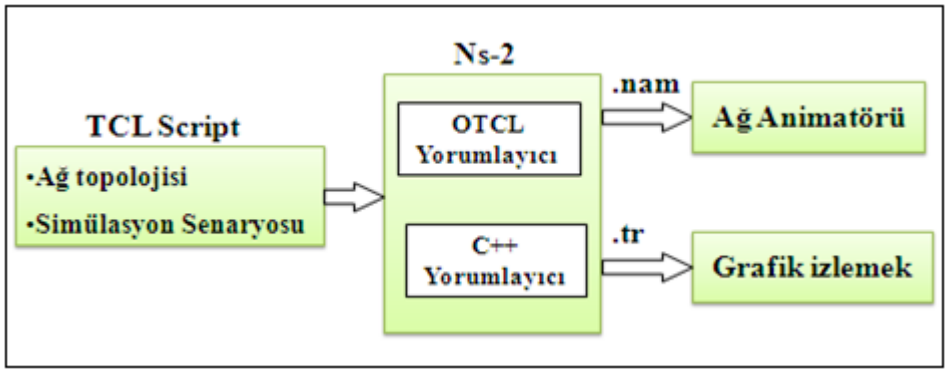
Bir önceki bölümde NS-2 simülatörünün bu çalışmaya uygun olup olmadığından ve niçin bu çalışmaya seçildiğinden bahsedilmiştir. Bu bölümde ise NS-2 simülatörünün kurulum talimatı ve bu simülatörde kullanılan programlama dilleri özetle anlatılmaktadır.

4.4.1. NS-2 simülatörünün kurulması

NS-2 simülatörü Mac, Linux ve Windows işletim sistemleri üzerinde kullanılabilmektedir. Windows işletim sistemi üzerinde kullanılabilmesi için Cygwin programının kurulması gerekir. Yükleme talimatları, "http://nsnam.isi.edu/nsnam/index.php/Main_Page" adresinde bulunmaktadır. Bu çalışmada NS-2 simülatörü Linux ve Windows XP işletim sistemleri üzerinde kurulmuştur. NS-2 simülatörünün kurulma talimatı EK-2'de bulunmaktadır.

4.4.2. NS-2 simülatörünün tanımı

NS-2 Simülatöründe, kullanıcıların yazmış olduğu senaryoları yorumlamak için OTCL (Object oriented Tool Command Language) programlama dili kullanılır. Bir sonraki bölümde TCL dilinin kullanımı ayrıntılı olarak anlatılmıştır. Bu çalışmanın uygulama aşamasında, atağın davranışı AODV protokolüne eklenerek, C++ dilinde yazılmıştır.



Şekil 4.11. NS-2 simülatörünün şeması [52].

Şekil 4.11'de gösterildiği gibi bir kullanıcı tarafından yazılmış olan OTCL senaryosu NS-2 tarafından yorumlandıktan sonra iki analiz raporu oluşturulur. Bunlardan biri simülasyonun görsel animasyon gösteri nesnesidir. Diğeri simülasyonda tüm nesnelerin davranışından oluşan iz nesnesidir[51, 52]. Bu nesnelerin her ikisi, NS-2 tarafından farklı dosya olarak oluşturulur. Birinci dosya NAM programı tarafından kullanılan " _.nam" dosyası ikincisi ise bütün simülasyonu kapsayan "_.tr" dosyasıdır. Bu dosya ("_.tr" dosyası) metin biçiminde tüm simülasyon izlerini içerir ve Xgraph programı tarafından yorumlanabilir. Bir inceleme dosyası simülasyondaki tüm olayları içermektedir. Örneğin paket gönderildiği zaman paketi gönderen düğüm, paketi alacak düğüm, paketin türü ve eğer bir paket düşürülürse onun nedeni bu olaylara örnek verilebilir. Şekil 4.12'de bir örnek standart izleme dosyası ve bu tür dosyalardaki alanların açıklaması Şekil 4.13'te gösterilmektedir.

```
+ 1.84375 0 2 cbr 210 ------- 0 0.0 3.1 225 610
- 1.84375 0 2 cbr 210 ------- 0 0.0 3.1 225 610
r 1.84471 2 1 cbr 210 ------- 1 3.0 1.0 195 600
r 1.84566 2 0 ack 40 ------- 2 3.2 0.1 82 602
+ 1.84566 0 2 tcp 1000 ------- 2 0.1 3.2 102 611
- 1.84566 0 2 tcp 1000 ------- 2 0.1 3.2 102 611
r 1.84609 0 2 cbr 210 ------- 0 0.0 3.1 225 610
+ 1.84609 2 3 cbr 210 ------- 0 0.0 3.1 225 610
d 1.84609 2 3 cbr 210 ------- 0 0.0 3.1 225 610
```

Şekil 4.12. NS-2 simülatöründe metin biçimindeki örnek bir izleme dosyası

Bu çalışmada "*yeni-izleme*" dosya biçiminden yararlanılmıştır (Şekil 4.14). Bu yeni biçim dosya özellikle KAA'lar için tasarlanmıştır. Yeni izleme dosyası, olaylarla ilgili daha detaylı bilgiler içermektedir.

Olay	Zaman	Düğümden	Düğüme	Paket türü	Paket boyutu	Bayraklar	paket akış	Kaynak adresi	Hedefin adresi	Sıra no.	Paket no.
d	0.52	0	1	cbr	500	---	82	0:0	1:0	0	4

d: **s:** send **r:** receive **d:** drop **f:** forward **+:** enqueue **-:** dequeue

0.52: Zaman

0: Düğüm0 = paketi gönderen düğüm

1: Düğüm 1= paketi alacak düğüm

cbr: Paket tipi **cbr**: sabit bit hızı (constant bit rate)

500: Paketin boyutu

---- : Bayraklar (gelecek için ayrılmış)

82: IP akış tanımlayıcı (IP flow identifier)

0:0: Gönderen (düğüm adresi: bağlantı noktası numarası)

1:0: Alıcı (düğüm adresi: bağlantı noktası numarası)

0 Sıra numarası(Sequence number)

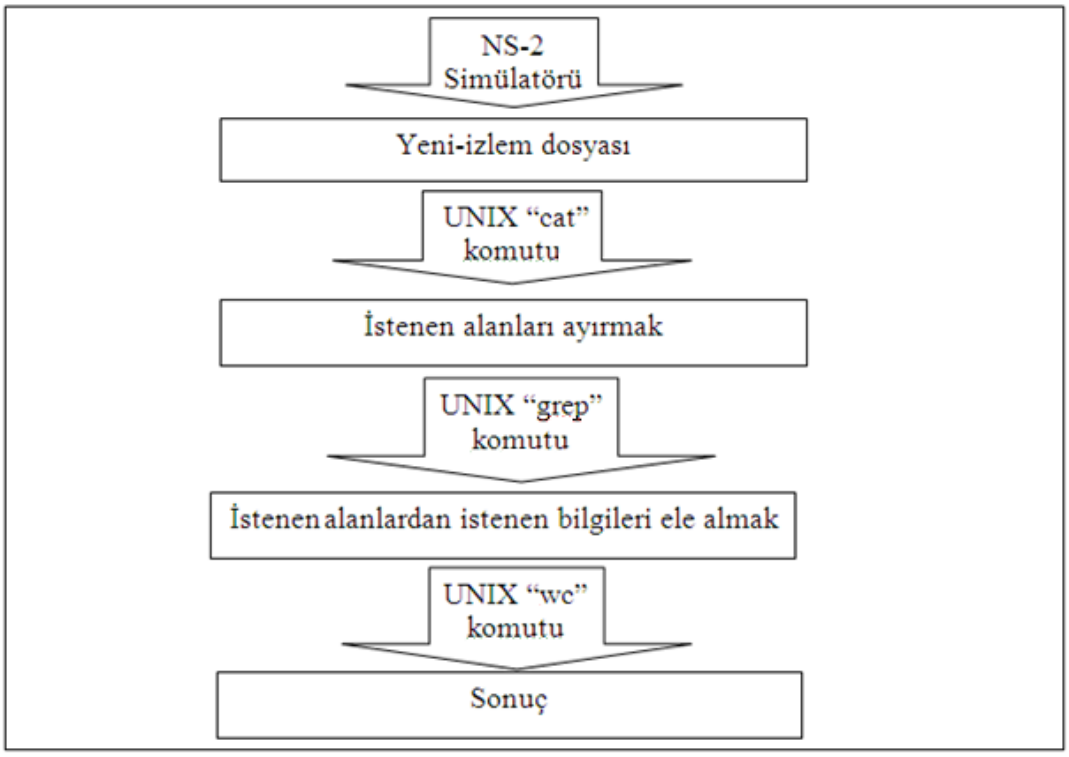
4: Tekil paket tanımlayıcı (Unique packet identifier)

Şekil 4.13. NS-2 simülatöründe standart izleme dosyasındaki alanların açıklaması

```
M 0.00000 0 (100.00, 100.00, 0.00), (100.00, 100.00), 30000.00
M 0.00000 1 (200.00, 100.00, 0.00), (200.00, 100.00), 30000.00
                    ...
M 0.00000 29 (600.00, 500.00, 0.00), (600.00, 500.00), 30000.00
r 0.050000000 _0_ RTR  --- 0 cbr 1000 [0 0 0 0] ------- [0:2 29:0 32 0] [0] 0 0
s 0.050000000 _0_ RTR  --- 0 AODV 48 [0 0 0 0] ------- [0:255 -1:255 30 0] [0x2 1 1 [29 0] [0 4]] (REQUEST)
s 0.050435000 _0_ MAC  --- 0 AODV 100 [0 ffffffff 0 800] ------- [0:255 -1:255 30 0] [0x2 1 1 [29 0] [0 4]]
(REQUEST)
r 0.051235333 _6_ MAC  --- 0 AODV 48 [0 ffffffff 0 800] ------- [0:255 -1:255 30 0] [0x2 1 1 [29 0] [0 4]] (REQUEST)
r 0.051235333 _1_ MAC  --- 0 AODV 48 [0 ffffffff 0 800] ------- [0:255 -1:255 30 0] [0x2 1 1 [29 0] [0 4]] (REQUEST)
r 0.051235471 _7_ MAC  --- 0 AODV 48 [0 ffffffff 0 800] ------- [0:255 -1:255 30 0] [0x2 1 1 [29 0] [0 4]] (REQUEST)
r 0.051260333 _6_ RTR  --- 0 AODV 48 [0 ffffffff 0 800] ------- [0:255 -1:255 30 0] [0x2 1 1 [29 0] [0 4]] (REQUEST)
r 0.051260333 _1_ RTR  --- 0 AODV 48 [0 ffffffff 0 800] ------- [0:255 -1:255 30 0] [0x2 1 1 [29 0] [0 4]] (REQUEST)
r 0.051260471 _7_ RTR  --- 0 AODV 48 [0 ffffffff 0 800] ------- [0:255 -1:255 30 0] [0x2 1 1 [29 0] [0 4]] (REQUEST)
s 0.058033823 _6_ RTR  --- 0 AODV 48 [0 ffffffff 0 800] ------- [6:255 -1:255 29 0] [0x2 2 1 [29 0] [0 4]] (REQUEST)
s 0.058188823 _6_ MAC  --- 0 AODV 100 [0 ffffffff 6 800] ------- [6:255 -1:255 29 0] [0x2 2 1 [29 0] [0 4]]
(REQUEST)
s 0.058882652 _1_ RTR  --- 0 AODV 48 [0 ffffffff 0 800] ------- [1:255 -1:255 29 0] [0x2 2 1 [29 0] [0 4]] (REQUEST)
r 0.058989156 _0_ MAC  --- 0 AODV 48 [0 ffffffff 6 800] ------- [6:255 -1:255 29 0] [0x2 2 1 [29 0] [0 4]] (REQUEST)
r 0.058989156 _12_ MAC  --- 0 AODV 48 [0 ffffffff 6 800] ------- [6:255 -1:255 29 0] [0x2 2 1 [29 0] [0 4]]
(REQUEST)
r 0.058989156 _7_ MAC  --- 0 AODV 48 [0 ffffffff 6 800] ------- [6:255 -1:255 29 0] [0x2 2 1 [29 0] [0 4]] (REQUEST)
r 0.058989294 _1_ MAC  --- 0 AODV 48 [0 ffffffff 6 800] ------- [6:255 -1:255 29 0] [0x2 2 1 [29 0] [0 4]] (REQUEST)

 [event type] -t [time] -s [src node] -d [dst node] -p [pkt type] -e [pkt size] -c [color] -i [pkt id] -a [flow id] -x {[src.port] [dst.port] [seqno]
```

Şekil 4.14. NS-2 simülatöründeki metin biçiminde örnek bir yeni-izleme dosyası

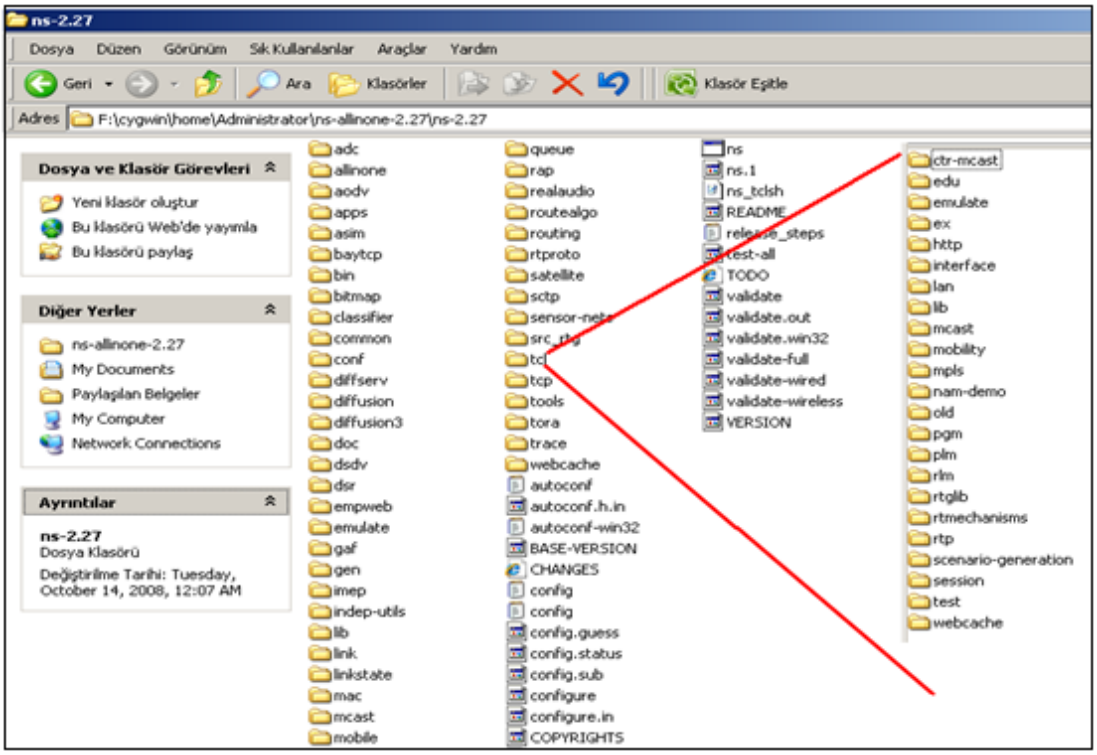
İzleme dosyasındaki sonuçları ele almak için izleme dosyasının alanlarından sadece olay türü, gönderen düğüm, hedef düğüm ve paket türü gereklidir. Bu çalışmada yukarıdaki bilgileri belirtmek için UNIX "*cat*" komutunu kullanmakta ve çıkışlar, izleme dosyasına yazılmaktadır. UNIX "*cat*" komutunu kullandıktan sonra bu alandaki bilgileri filtreden geçirmek için UNIX "*gerp*" komutu kullanılmakta ve bu komutun çıkışları istenen bilgileri sayabilmek için UNIX "*wc*" komutuna gönderilmektedir. UNIX "*wc*" komutunun çıkışı bir yeni dosyaya yazılmaktadır. ".tcl" dosyaları bir metin editöründe yazılıp ve "cat", "awk", "wc" ve "grep" Unix işletim sistemi komutları kullanılarak sonuçlar analiz edilebilir. Bu işlemler Şekil 4.15'te gösterilmektedir. Bu komutlar tüm simülasyonlarda tüm düğümler için bir toplu iş dosyası olarak (batch file) kullanılmaktadır.



Şekil 4.15. Simülasyonda istenen bilgileri ele almak için yapılan işlemler

4.4.3. OTCL programlama dili

OTCL dili aslında TCL (Tool Command Language) dilinin bir nesne uzantısı sayılmaktadır. TCL, California'da Berkeley Üniversitesi'nde John Ousterhout tarafından geliştirilen güçlü ve yorumlanabilir bir dinamik programlama dilidir. TCL yüksek ölçüde genişletilebilir bir programlama dilidir ve geniş bir kullanım alanına sahiptir. TCL dili tamamen C++ programlama dili ile uyumludur. NS-2 simülatöründe, kullanıcıların yazmış olduğu senaryoları yorumlamak için OTCL programlama dili kullanılır. NS-2 simülatöründe genellikle ağ topolojisini ve simülasyon senaryosunu belirtmek için OTCL kullanılmaktadır. Şekil 4.16'da NS-2 simülatörünün klasöründe TCL dosyalarının yer aldığı klasörler gösterilmektedir.

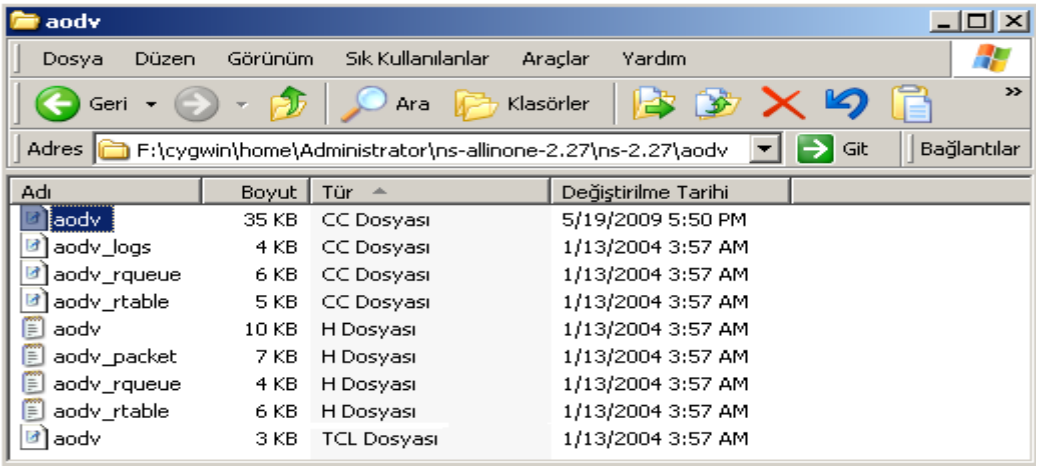


Şekil 4.16. NS-2 simülatöründe TCL dosyalarının yer aldığı klasörler

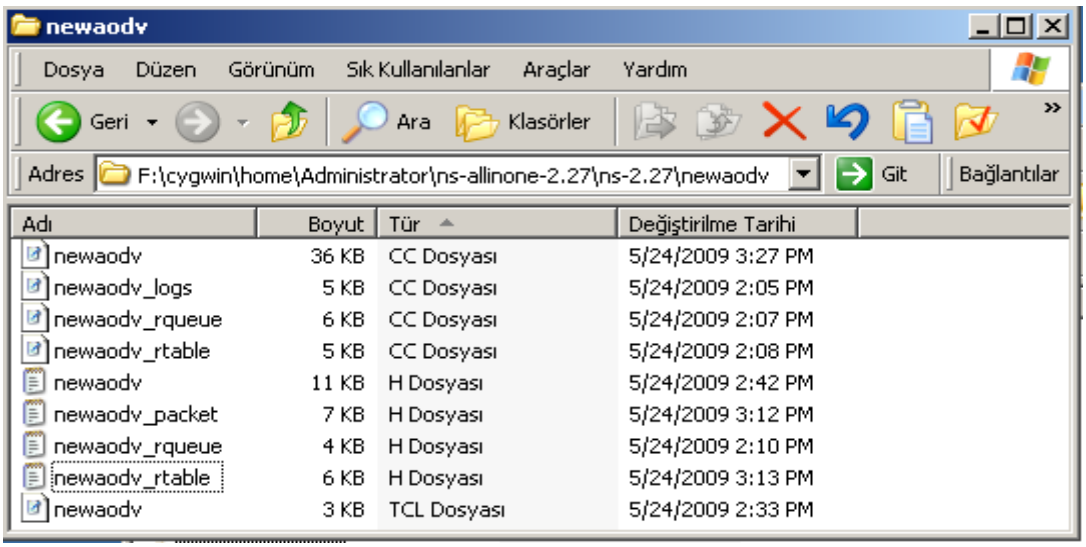
Bir sonraki bölümde NS-2 simülatöründe yeni bir yönlendirme protokolü geliştirmenin adımları ve daha sonraki bölümde bu yeni protokolü test etmek için yazılan TCL dosyaları açıklanmaktadır.

4.5. NS-2'de Yeni Yönlendirme Protokolü Geliştirmek

Bu çalışma KAA'larda en sık kullanılan yönlendirme protokolü AODV protokolü üzerinde yapılmıştır. AODV üzerinde yeni bir yönlendirme protokolü geliştirme aşamaları bu bölümde anlatılmıştır. AODV protokolünün tüm dosyaları *.../ns-2.27/aodv/* klasöründe bulunmaktadır. Bu klasörde değiştirilmesi gereken dosyalar Şekil 4.17'de gösterilmektedir.



Şekil 4.17. NS-2 simülatöründe AODV yönlendirme protokolünün dosyaları



Şekil 4.18. NS-2 simülatöründe *newAODV* yönlendirme protokolünün dosyaları

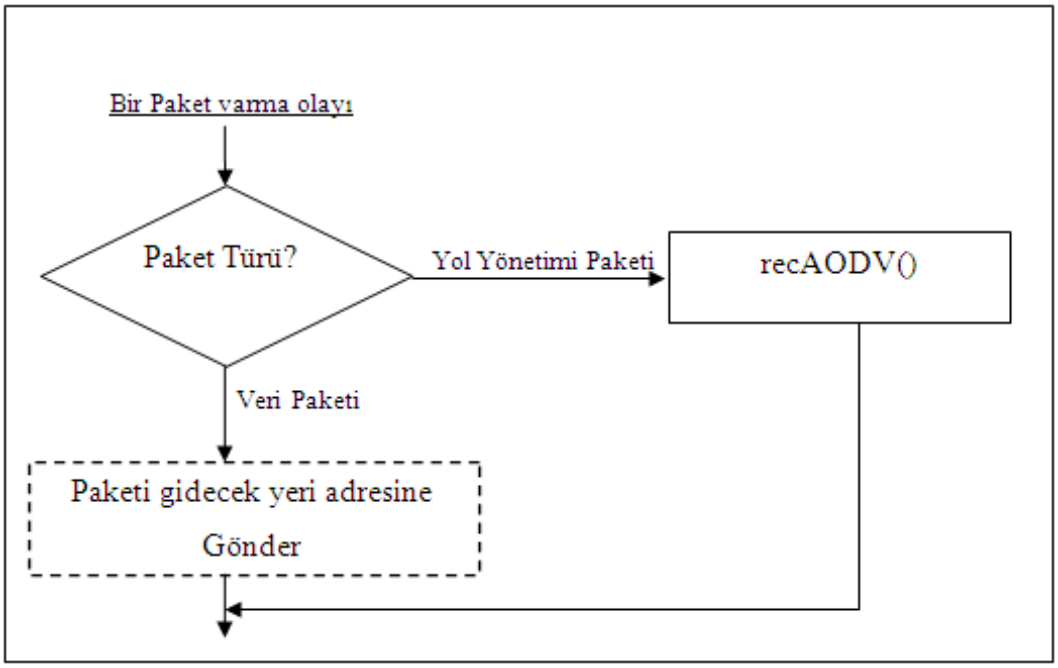
Başlangıçta, *aodv* klasörünü aynı yere (yeni bir isim ile) kopyalanması gerekir. Yeni protokolle *newAODV* ismi verilmiştir bu nedenle bu protokolün dosyaları *.../ns-2.27/ newaodv/* klasöründe yer almaktadır. Şekil 4.18'de gösterildiği gibi yeni klasörde *aodv* olarak etiketlenen bütün dosyaların isimleri *newaodv*'ye değiştirilmiştir. Ayrıca bu dosyaların içindeki tüm sınıflar (classes), fonksiyonlar (functions), yapılar (structs), değişkenler (variables) sabit isimlerin (constant names) de değiştirilmesi gerekir (Bkz EK-3).

Sözü edilen değişimlerden başka */tcl/lib/ns-lib.tcl* dosyasının da değiştirilmesi gerekir. Bu dosyanın içinde protokol ajanları bir prosedür (procedure) olarak kodlanılmıştır. Bu prosedürlere yeni bir ajan (yada yeni bir prosedür) eklenmesi gerekir. Düğümlerin *newaodv* protokolü kullanıldığı zaman, bu ajan simülasyonun başlangıcında programlanır (scheduled), ve *newaodv* protokolünü kullanacak olan düğümlere tahsis edilir. Bu prosedür Şekil 4.19'da gösterilmektedir.

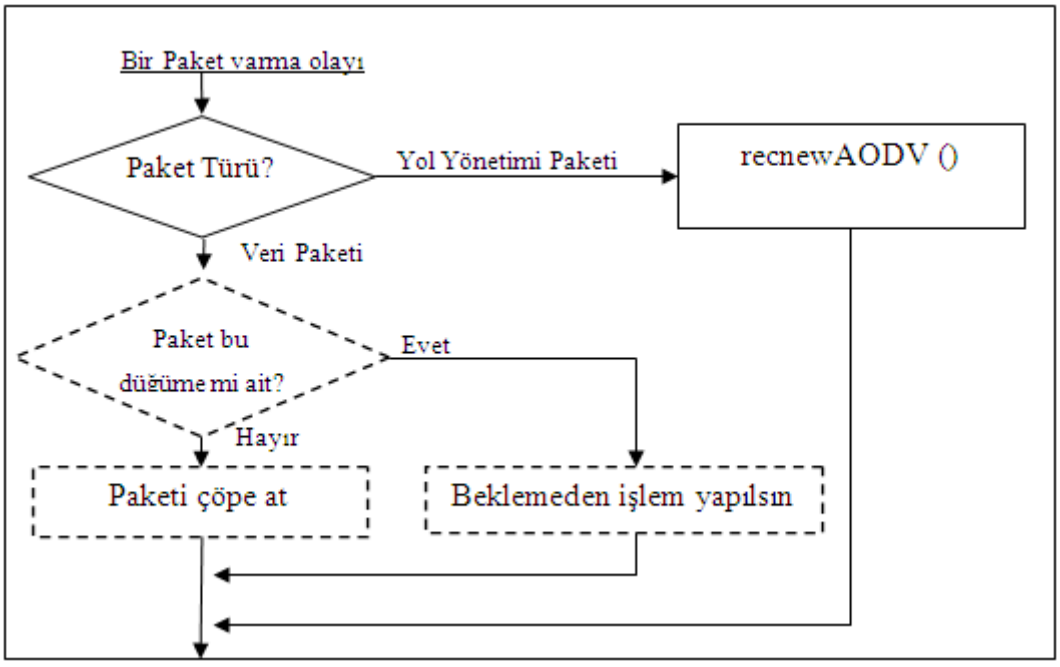
```
newAODV {
   set ragent [$self create-newaodv-agent $node]
}

Simulator instproc create-newaodv-agent { node } {
   set ragent [new Agent/newAODV [$node node-addr]]
   $self at 0.0 "$ragent start"
   $node set ragent_ $ragent
   return $ragent
}
```

Şekil 4.19. *newAODV* protokol ajanı */tcl/lib/ns-lib.tcl* dosyasına eklenmektedir

Şimdiye kadar, *newaodv* etiketlenen yeni bir protokol yapılmıştır. Ama henüz bu protokolden yararlanan düğümlerin anormal davranışlarını yeni protokolde yerine getirilmemiştir. Yeni protokolde anormal davranışları gerçekleştirmek için */ns2.27/newaodv/newaodv.cc* dosyasında bazı değişikliklerin yapılması gerekir.



Şekil 4.20. AODV protokolünde, *recv()* fonksiyonunda yapılan işlemler



Şekil 4.21. *newAODV* protokolünde, *recv()* fonksiyonunda yapılan işlemler

AODV protokolünde bir paketin, *recv()* fonksiyonu ile alındığı zaman, o paketin türüne göre farklı işlemler yapılmaktadır. Şekil 4.20'de gösterildiği gibi, eğer kabul

edilen paketin tipi AODV yol idare paketlerinin (route management packets) herhangi birisiyse alınan paket *RecvAODV()* fonksiyonuna gönderilir, ama eğer kabul edilen paket bir veri (data) paketiyse normal koşullarda AODV protokolü, paketi gideceği yerin adresine (destination address) yollar. Bu durumda eğer bahis edilen düğüm, bir kötücül düğüm ise kendine gelmeyen bütün veri paketlerini çöpe atar. Bu işlemleri yapabilmek için, *newAODV* protokolünde yapılan değişimler Şekil 4.20 ve 4.21'de "- - -" çizgisi ile gösterilmektedir.

AODV protokolünde yapılan değişimlerin kodu Şekil 4.22'de gösterilmektedir. Aşağıdaki kodda ikinci "*Eğer-İse-Değilse*" komutunda "*Eğer*" kısmında, veri paketinin gideceği yerin bu düğüm olduğunu söylemektedir. Veri paketinin gideceği yer başka düğümse "*Değilse*" kısmında bu paket çöpe atılmaktadır.

```
void newAODV::recv(Packet *p, Handler*) {
  ...
if(ch->ptype() == PT_AODV) {               // Eğer yol Yönetimi
Paketi ise
      ih->ttl_ -= 1;
      recvnewAODV(p);
      return;

}else if((u_int32_t)ih->saddr() == index)//eğer gidecek yer bu düğümse
         forward((newaodv_rt_entry*) 0, p, NO_DELAY);
      else
         drop(p, DROP_RTR_ROUTE_LOOP);     // eğer veri paketinin
gidecek yeri
   ...                                     //  başka düğümse bu paketi
çöpe at
 }
```

Şekil 4.22. *newAODV* protokolünde veri paketinin çöpe atılması

AODV protokolünde eğer kabul edilen paket, bir *newAODV* yönetimi paketiyse *recv*() fonksiyonu, o paketi *recAODV()* fonksiyonuna yollar. *recAODV()* fonksiyonu

bu paketin türünü kontrol eder ve paketin türüne göre *case* komutu ifadesiyle alınan paket uygun fonksiyona gönderilir. Örneğin; RREQ paketleri, *recvRequest()*, RREP paketleri *recvReply()* fonksiyona vb.

newAODV protokolünde *recvAODV()* fonksiyonunun ismi, *recvnewAODV()* fonksiyonuna değiştirilmiştir. Şekil 4.23'te bu fonksiyonda yapılan işlemler gösterilmektedir. Bu paket türlerinden RREQ paketi bizim için önemlidir, çünkü kötü niyetli düğüm komşu düğümlerden gelen RREQ paketini (gideceği yere yolu olup olmadığını incelemeden) derhal RREP paketi ile cevaplamaktadır. Böylece komşu düğümleri aldatır ve baz istasyonuna en yakın düğümmüş gibi tanıtır.

```
void  newAODV::recvnewAODV(Packet *p) {

 ...
 switch(ah->ah_type) {

    case AODVTYPE_RREQ:     // Route Request Paketi
      recvRequest(p);
      break;

    case AODVTYPE_RREP:     // Route Reply Paketi
      recvReply(p);
      break;

    case AODVTYPE_RERR:     // Route Error Paketi
      recvError(p);
      break;

    case AODVTYPE_HELLO:    // Hello Paketi
      recvHello(p);
      break;

    default:                // Yanlış  paket
      fprintf(stderr, " Error:: Bilinmeyen Paket türü(%x)\n", ah->ah_type);
      exit(1);

 }//swıtch
...
}//recnewAODV
```

Şekil 4.23. *recvnewAODV()* fonksiyonda yapılan işlemler.

AODV protokolünde bir halka oluşturulmasını önlemek için paketlere bir sıra numarası eklenmektedir. Bu sıra numarası 4294967295 (2^{32} -1) sayısını aştığında o paket çöpe atılmaktadır [53]. Şekil 4.24'de görüldüğü gibi bu yanlış pakette adım sayısının değeri de " **1**" ayarlanmaktadır.

```
sendReply(rq->rq_src,          // kaynak
          1,                   // Adım sayısı
          index,               // Gönderilecek yer
          4294967295,          // sıra numaranın ( en yüksek değeri)
          MY_ROUTE_TIMEOUT,    // yaşam süresi
          rq->rq_timestamp);   // Zaman bilgisi
```

Şekil 4.24. Kötücül düğümün gönderdiği yanlış RREP paketin yapısı

NS-2 simülatörünün dosyalarında yapılan değiştirmelerden sonra NS-2 simülatörünün kaynak dosyalarının hepsini derlemek (compile) gerekmektedir. Bu işlem esnasında */ns-2.27/makefile.in*, */ns-2.27/makefile.vc* ve */ns-2.27/makefile* dosyalarında isimleri geçen tüm dosyaların nesne dosyaları (object files) yeniden yaratılır. Bu nedenle yeni eklenen **.CC* dosyalarının isimlerini bu dosyalarda eklenmesi gerekir. Bu dosyalara eklenen yeni satırlar Şekil 4.25'te gösterilmektedir.

```
newaodv/newaodv_logs.o newaodv/newaodv.o \
newaodv/newaodv_rtable.o newaodv/newaodv_rqueue.o \
```

Şekil 4.25. "/ns2.27/*makefile* " de eklenen gerekli değiştirmeler

Bu çalışmanın devamında bu aşamaya kadar yapılan değişiklikler birkaç senaryo ile test edilmektedir.

4.6. Yeni Yönlendirme Protokolünün Test Edilmesi

Yeni protokolün doğru çalışıp çalışmadığını test edebilmek için NAM (Network Animator) aracından yararlanılmıştır. Bu araç yardımıyla, TCL dilinde yazılan

senaryoların sonuçları görsel olarak görünebilir. Çalışmanın bu aşamasında protokolü test edebilmek için birçok senaryo yazıldı. Bu senaryolardan ikisi bu bölümde anlatılmaktadır. Birinci senaryo herhangi bir kötücül düğümü kullanmadan yazılmıştır. İkinci senaryoda ise senaryoya bir kötücül düğüm eklenmiştir. En son olarak bu iki senaryodan elde edilen sonuçlar kıyaslanmıştır.

4.6.1. Simülasyon parametreleri

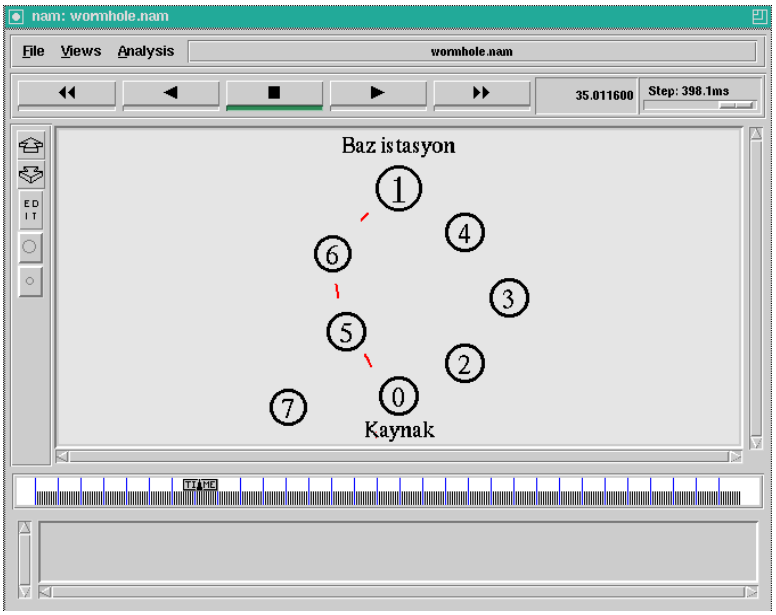
Bu çalışmada, simülasyonlardan doğru sonuçları almak için UDP kullanılmıştır. Kaynak düğümde UDP protokolü kullanıldığında, kötü niyetli düğüm, paketleri düşürse bile kaynak düğüm ACK paketini beklemeden paket göndermeye devam edecektir, bunun yerine TCP protokolünü kullanılır olsaydık ACK paketini almadığında önceki paketi tekrar gönderecektir ve bu durumda belli bir sonuç elde edilemeyecektir. Kullanılan diğer parametreler Çizelge 4.6'da gösterilmektedir.

Çizelge 4.6. Simülasyon parametreleri

Simülatör	NS-2
Simülasyon Zamanı	150(s)
Sabit Düğümlerin Sayısı	8
Mobil Düğümlerin Sayısı	1
Topoloji	600m × 600m
Yönlendirme Protokolü	AODV
En çok Bant Genişliği	2 Mbps
Akış (Sabit bit hızı)	1 Mbps
Mobil Düğümlerin En Çok Hız	5(m/s)

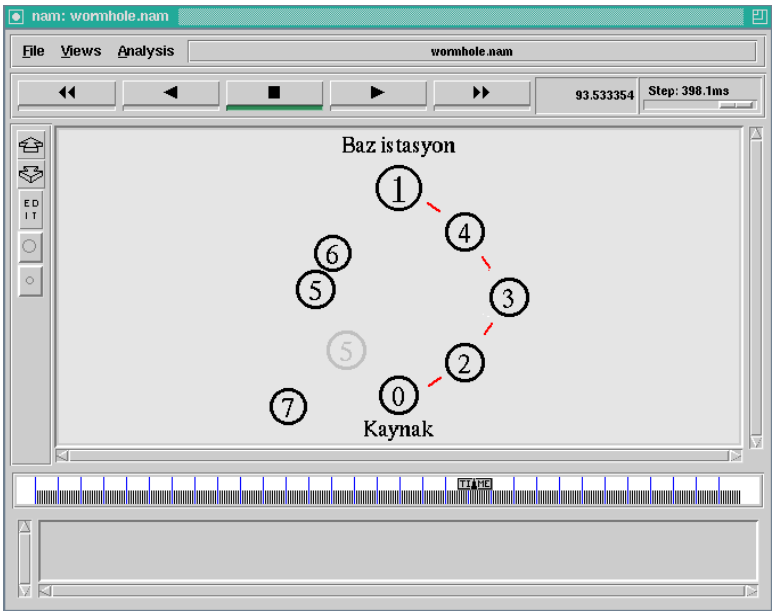
4.6.2. Simülasyon değerlendirmesi

İlk senaryo, 8 düğüm kullanarak canlandırılmaktadır. Bu senaryoda kötücül düğüm bulunmamaktadır ve düğüm 0 ve düğüm 1'in arasında, AODV yönlendirme protokolü kullanılarak bağlantı kurulmaktadır. Şekil 4.26'da gösterildiği gibi düğüm 0'dan düğüm 1'e veri akışı en yakın yoldan 3 adımla gerçekleştirilmektedir.



Şekil 4.26. Düğüm 0 ve 1 arasındaki veri akışı, 5 ve 6 yoluyla gerçekleştirilir.

Düğüm 5 düğüm 0'dan uzaklaştırıldığında düğüm 0 ile düğüm 5 iletişim kurmaz, bu durumda düğüm 0 paketleri düğüm 1'e göndermek için başka bir yol bulur. Bu yeni yol 4 adımdan oluşmaktadır ve düğüm 2, 3 ve 4 yoluyla gerçekleşmektedir (Şekil 4.27).



Şekil 4.27. Veri akışı düğüm 0 ve 5 bağlantısı kesildikten sonra düğüm 2, 3 ve 4 yoluyla gerçekleştirilmektedir.

AODV protokolünde paketler gideceği yere en yakın yoldan ulaşmaktadır ve bu protokol, isteğe bağlı (On Demand) protokollerden sayılmaktadır. Bu nedenle yönlendirme tablosunu oluşturmadan yönlendirme görevini yapmaktadır ve KAA'lar için uygun bir yönlendirme protokolü sayılmaktadır.

İkinci senaryoda, kötücül düğümün kullandığı protokol değiştirilmiştir. AODV protokolünün yerine *newAODV* protokolü yazılmıştır. Bu senaryoda en son düğüm kötücül düğüm olarak etiketi 8 ile belirlenmiştir. Şekil 4.28'de yapılan değişimler gösterilmektedir. Bu düğüm komşu düğümlerden paketleri toplayıp onları çöpe atmaktadır.

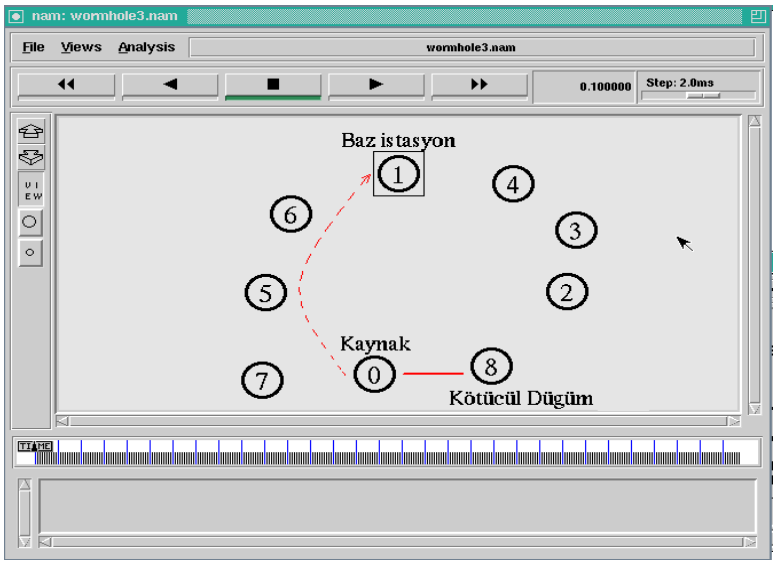
```
...

set val(nn)     9                          ;# Düğümlerin sayısı

$ns node-confıg -adhocRouting AODV     ;# Tüm Düğümler normal
çalışmaktadır
for {set i 0} {$i < $val(nn) } { incr i } {
            set node_($i) [$ns node]
}

...
```

Birinci senaryodaki TCL dosyası

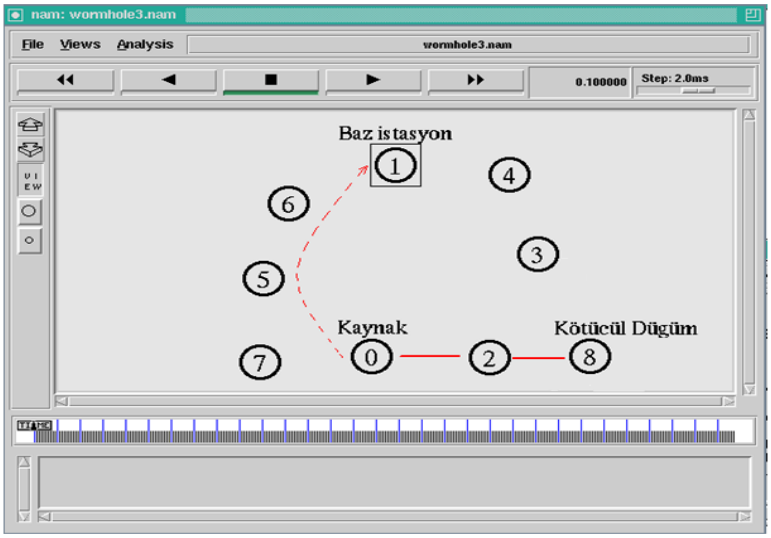
```
...

set val(nn)     9                      ;# Düğümlerin sayısı

$ns node-confıg -adhocRouting AODV  ;# Normal Düğümler
for {set i 0} {$i < ($val(nn)-1) } { incr i } {
            set node_($i) [$ns node]
}

$ns node-config -adhocRouting newAODV
set node_($i) [$ns node]           ;# En son düğüm =Kötücül düğüm

...
```

İkinci senaryodaki TCL dosyası

Şekil 4.28. İkinci senaryodaki yapılan değişiklikler



Şekil 4.29. Kaynak düğüm paketleri sink yerine kötücül düğümüne gönderir.



Şekil 4.30. KD kaynaktan uzak olsa bile yine paketleri alma ihtimali vardır.

Sözü edilen değişimleri yaptıktan sonra kötücül düğüm kendini baz istasyonuna en yakın düğüm olarak anons edecektir. Bu durumda şebekedeki veri akımı, baz istasyonu yerine bu düğüme gönderilir. Eğer kötücül düğüm kaynak düğüme en yakın düğümse kaynak düğüm gönderilecek paketleri baz istasyonu yerine bu kötücül düğüme gönderecektir (Şekil 4.30). Ama kaynak düğümden uzak olsa bile başka düğümler aracılığıyla paketleri kendine gönderilmesini sağlayabilir. Şekil 4.30'da görüldüğü gibi kötücül düğüm, düğüm 2'ye kendini baz istasyonun 1.

adımındaki düğüm gibi anons edip ve düğüm 2'de kaynak düğüme 2. adımda olduğunu belirtir. Bu durumda kaynak düğüm, baz istasyonuna en yakın düğüm, düğüm 2'yi bulup ve tüm paketleri bu düğüme göndermektedir. Böylece tüm paketler kötücül düğüme gönderilir.

4.6.3. Yazılan TCL kodunun açıklaması

Senaryo dosyası EK-4'te bulunmaktadır. Şekil 4.31'de görüldüğü gibi kodda öncelikle simülasyonun konfigürasyonu belirlenmiştir. Her satır için açıklamalar kodun içinde verilmiştir.

```
# Seçeneklerin tanımlaması

set val(chan)           Channel/Wireless Channel ;# Kanal  tipi
set val(prop)           Propagation/TwoRayGround ;# Radyo-propagation
                                                 ;# modeli
set val(netif)          Phy/WirelessPhy          ;# Şebeke arayüz tipi
set val(mac)            Mac/802_11               ;# MAC tipi
set val(ifq)            Queue/DropTail           ;# Arayüz kuyruk tipi
set val(ll)             LL                       ;# bağlantı katmanı tipi
set val(ant)            Antenna/OmniAntenna      ;# anten modeli
set val(ifqlen)         50                       ;# kuyrukta en fazla
                                                 ;# paket sayısı
set val(nnaodv)         8                        ;# normal Düğümlerin
                                                 ;# sayısı
set val(nn)             1                        ;# kötücül Düğümlerin
                                                 ;# sayısı
set val(rp1)            AODV                     ;# yönlendirme
                                                 ;# porotokollü
set val(rp2)            newAODV                  ;# yeni yönlendirme
                                                 ;# protokolü
set val(x)              600                  ;# topografının X boyutu
set val(y)              600                  ;# topografının Y boyutu
set val(stop)           150                  ;# Simülasyonun son zamanı

set val(cstop) 145                               ;# bağlantıların süresü

# ./setdest -n 9 -p 1.0 -M 1.0 -t 150 -x 600 -y 600 > senaryoAODV-n9-
t500-x750-y750
set val(cp) "scenarios/sce-n9-t150-x600-y600"    ;#Bağlantıların modeli

set val(cc) "scenarios/cbr"                       ;#Trafik Seçenekleri
```

Şekil 4.31. Yazılan TCL kodunda seçeneklerin tanımlaması

Kodun sonunda "./setdest" komutu ile oluşturulan senaryo dosyası bulunmaktadır. Bu komut ile oluşturulan dosyada düğüm sayısı simülasyonun süresi ve topografının boyutları belirlenmektedir. Ayrıca kodun sonunda bağlantıların türü belirlenmiştir.

Bağlantıların türü "./cbrgen" komutu ile oluşturulmaktadır. Paketlerin boyutu (packets size) ve veri oranı (data rate) bu komut ile belirlenmektedir. Yazılan kodda oluşturulan dosyaların "scenarios" klasörüne kaydedilmesi belirtilmiştir.

Şekil 4.32'deki kodda düğümlerin yapıları belirlenmiştir. Her satır için açıklamalar kodun içinde verilmiştir.

```
# Düğümlerin yapısını oluşturmak

$ns_ node-config -adhocRouting $val(rp) \
   -llType $val(ll)                     \ ;# ll     = Link Layer
   -macType $val(mac)                   \ ;# mac    = MAC Layer
   -ifqType $val(ifq)                   \ ;# ifq    = interface Queue
   -ifqLen $val(ifqlen)                 \ ;# ifqlen = Interface Queue Length
   -antType $val(ant)                   \ ;# ant    = Antenna Model
   -propType $val(prop)                 \ ;# prop   = Radio Propagation model
   -phyType $val(netif)                 \ ;# netif  = Network Interface Type
   -topoInstance $topo                  \ ;# topo   = Topography
   -agentTrace ON                       \ ;# Agent Trace is enable
   -routerTrace ON                      \ ;# Router Trace is enable
   -macTrace ON                         \ ;# MAC Trace is Enable
   -movementTrace ON                    \ ;# Movement Trace is Enable
   -channel $chan_1_                      ;# Chanel Type = Channel 1
```

Şekil 4.32. Düğümlerin yapıları

Şekil 4.33'te AODV yönlendirme Protokolünden yararlanan düğümler ve *newAODV* protokolünden yararlanan kötücül düğümlerin oluşturulması gösterilmiştir.

TCL dosyasının sonunda izleme dosyalarını kapatıp ve canlandırmayı görebilmek için NAM programı çalıştırılmaktadır (Şekil 4.34). Senaryo dosyası EK-2'de bulunmaktadır.

```
# Normal düğümlerin oluşturulması...

$ns_ node-config -adhocRouting AODV
for {set i 0} {$i < $val(nnaodv)} {incr i} {
   set node_($i) [$ns_ node]
   $node_($i) random-motion 0 ;# Rastgele Hareketleri
Etkisizleştirmek
}

# Kötücül düğümlerin oluşturulması...

$ns_ node-config -adhocRouting newAODV
for {set i $val(nnnewaodv)} {$i < $val(nn)} {incr i} {
   set node_($i) [$ns_ node]
   $node_($i) random-motion 0 ;# Rastgele Hareketleri Etkisizleştirmek
   $ns_ at 0.01 "$node_($i) label \"kötücül Düğüm\""
}
```

Şekil 4.33. Düğümlerin oluşturulması

```
proc finish {} {
   global ns_ tracefd namtrace
   $ns_ flush-trace
   close $tracefd
   close $namtrace
   exec nam senaryo.nam &
exit 0
}
puts "Starting Simulation..."
$ns_ run
```

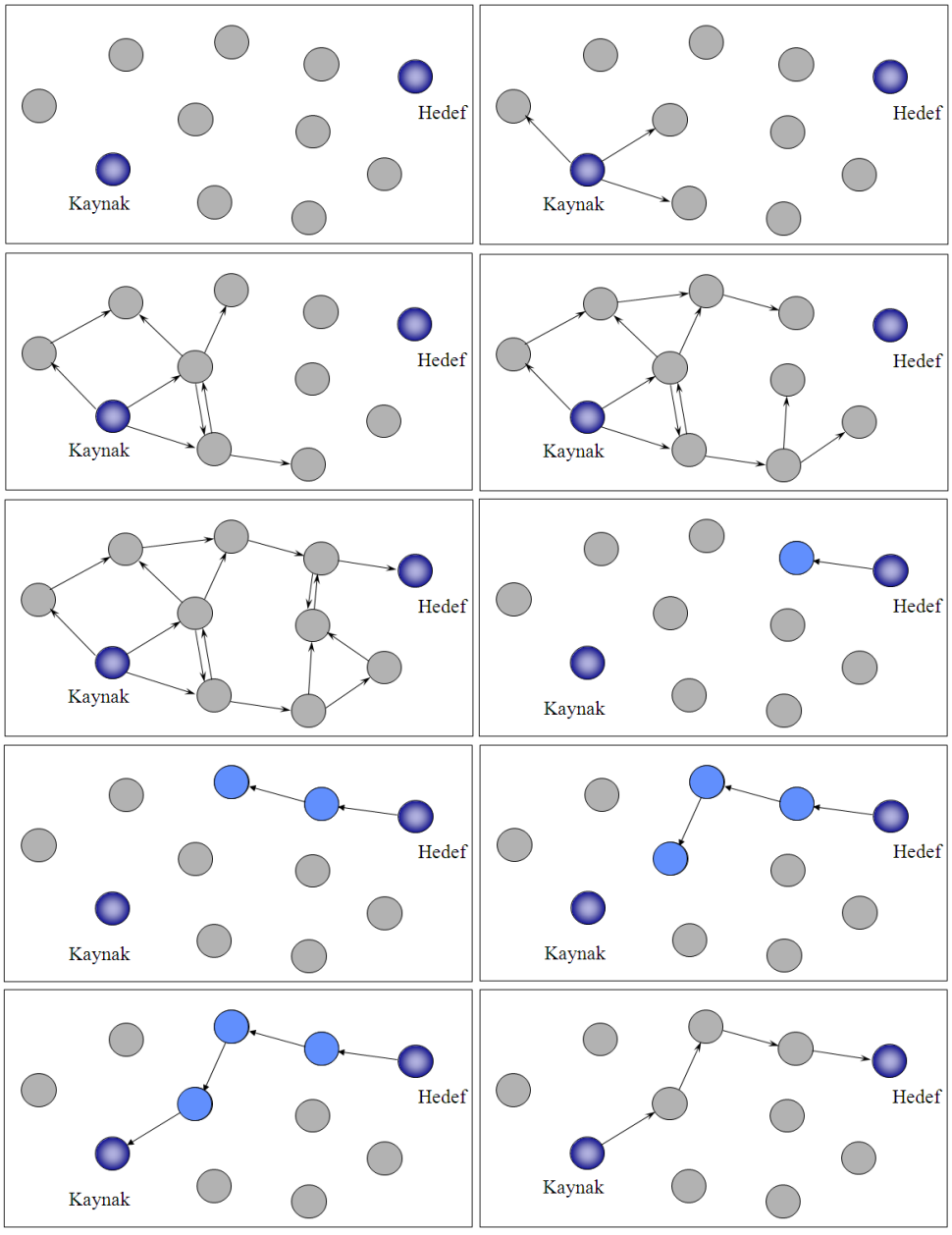
Şekil 4.34. TCL dosyanın sonunda izleme dosyalarını kapatıp NAM çalıştırılır

4.7. AODV Protokolünde Hedefi Bulma Mekanizması

Önerilen savunma mekanizması AODV yönlendirme protokolün üzerinde gerçekleştirilmektedir. Bu nedenle bu bölümde AODV protokolünde kısaca yönlendirme mekanizması açıklanmıştır.

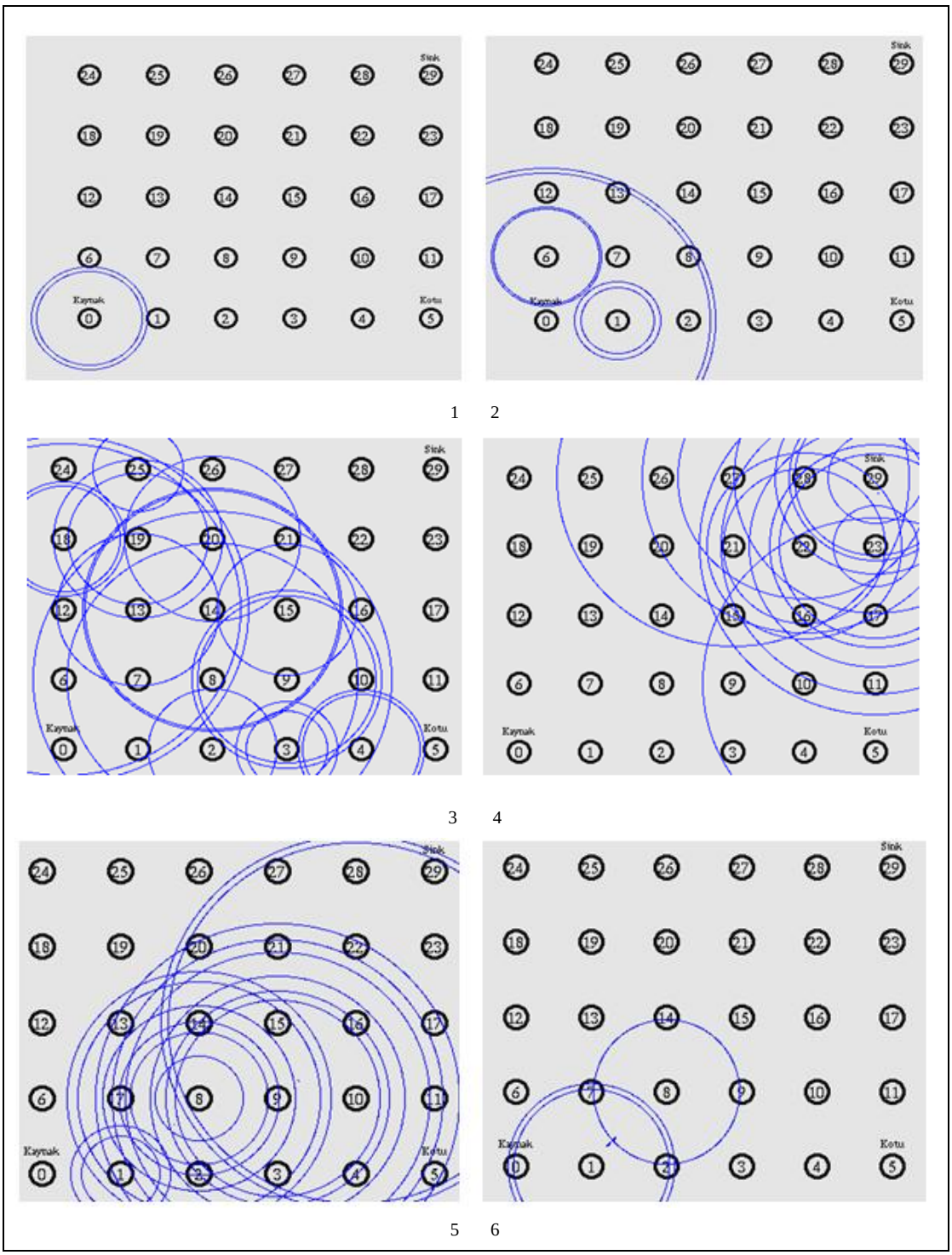
AODV protokolünde kaynak düğüm bir paketi hedefe ulaştırmak için RREQ paketini "*Multicast*" olarak tüm komşu düğümlerine gönderir. Komşu düğümler de kendi

komşu düğümlerine bu isteği (daha önce iletmediği durumda) iletirler. Böylece bu RREQ paketi ağda dağılır ve hedef düğüme ulaşır. Hedef düğüme RREQ paketi ulaştığında en uygun (yakın) yol seçildikten sonra RREP paketi ile cevabı gönderir. RREP paketi kaynak düğüme ulaştığında, kaynak düğüm bu bulunan yoldan paketleri hedef düğüme gönderir. Bu işlemler Şekil 4.35'te gösterilmektedir.



Şekil 4.35. AODV protokolünde paketin gönderilen yolunu bulma mekanizması

NS-2 simülatörü AODV protokolünü desteklemektedir. Bir örnek senaryoda AODV protokolünde düğüm 0 baz hedefe yol bulma mekanizması Şekil 4.36'da gösterilmektedir.



Şekil 4.36. Ns2 simülatöründe AODV protokolünün yol bulma mekanizması

4.8. Önerilen Savunma Mekanizmasının Gerçekleştirilmesi

Bu çalışmada KAA'larda Sinkhole atağını önlemek için geliştirilen güvenlik mekanizması bölüm 4.1'de kısaca açıklanmıştır. Geliştirilen güvenlik mekanizması ağ içindeki veri hareketlerinin analizine dayanmaktadır. Önerilen çözüm yapısı iki aşamadan oluşmaktadır. Bu bölümde bu iki aşamanın geliştirilmeleri açıklanmıştır. Önerilen algoritma sözde kod (pseudo code) biçiminde Şekil 4.37'de gösterilmektedir.

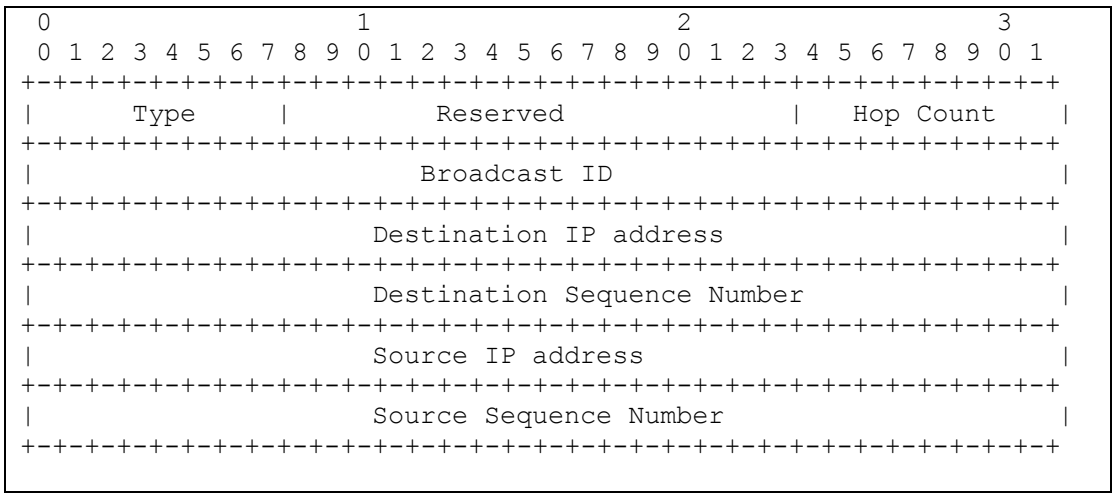
```
/*Saldırılmış alandaki düğümler için baz istasyonunda SAV tablosunu oluşturulur. Bu tablo düğümlerden alınan
    bilgiler üzerinden oluşturulmaktadır ve kötücül düğüm tarafından gönderilen bazı bilgiler yanlış olabilir. */

1. Adım:        /* Eğer kötücül düğüm yanlış bilgi göndermemişse 1.adımda kötücül düğüm bulunmuş sayılır  */
M= ϕ                                                                    /* M: Kötücül düğümler kümesi */
For i=1 to K Do                          /* {v( 1), …, v(K)} = Saldırılmış alandaki düğümler kümesi  */
  If ( SAV[i , 2] = 1 and v(i) ∉{Sink'in komşuları} )                          /* Tutarsızlık */
M = M ∪ {v(i) }                                                      /* v(i) = Kötücül düğüm */
                    /* Birinci adımda kötücül düğüm bulunmadığı halde ikinci adın çalıştırılmaktadır */

 2. Adım:
S=ϕ                                                                  /* S: Şüpheli düğümler kümesi  */
For i=1 to k Do                                                      /* S = Şüpheli düğümler kümesi  */
  If SAV[ i ,2] = 2  then                                    /* Adım sayısı baz istasyonundan = 2  */
    S = S ∪{ Next(v(i))}                          /* (v(i)) den sonraki düğüm = kötücül düğüm */

3. Adım
   /* Algoritmanın son adımında SAV tablosunda OY ÇOĞUNLUĞU metodu kullanarak şüpheli düğümler arasından
                                                                  kötücül düğümleri tespit edilmektedir */
```

Şekil 4.37. MND algoritmasının sözde kodu

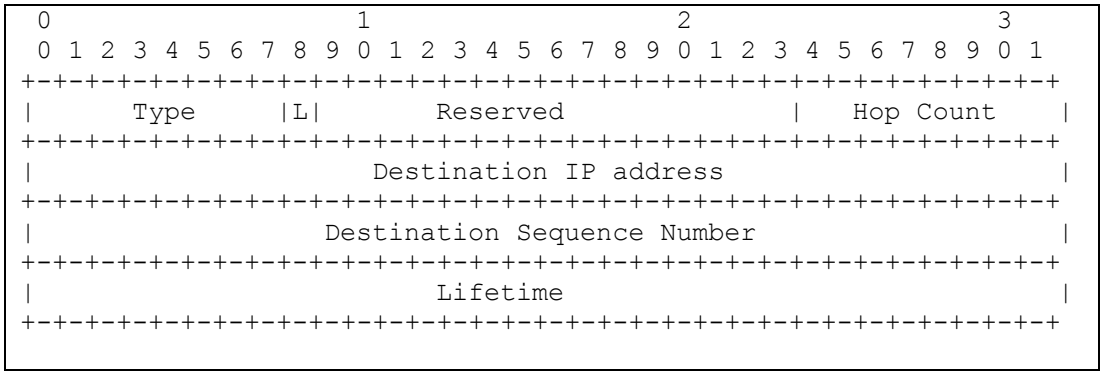
4.8.1. Saldırının olup olmadığını incelemek

Bu bölümde, Sinkhole atağına karşı önerilen algoritmayı nasıl NS-2 simülatöründe gerçekleştirildiği açıklanmıştır. AODV protokolündeki en önemli standart kontrol paket formatları RREQ ve RREP Şekil 4.38 ve Şekil 4.39'da gösterilmektedir. Bu çalışmada bu iki paketten, birçok yerde yararlanılmaktadır. RREQ paketlerin boyutu

192 bit(24 bayt) ve RREP paketinin boyutu 128 bit(16 bayt) olarak belirtilmektedir ama data paketlerinin boyutu daha fazla ve değişken olabilir.



Şekil 4.38. AODV'de RREQ paket formatı [3]

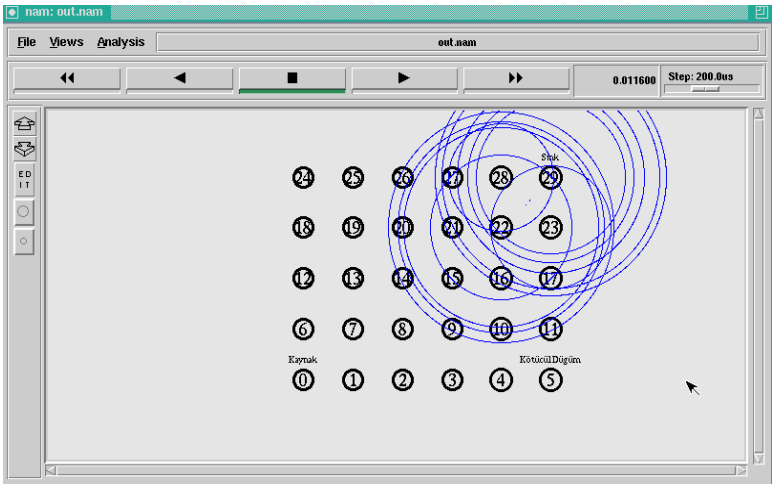


Şekil 4.39. AODV'de RREP paket formatı [3]

NS-2 simülatöründe AODV protokolünde kullanılan paket tanımları "*aodv_packet.h*" dosyasında bulunmaktadır. Şekil 4.40'da RREQ paketinin tanımı gösterilmektedir. "*mndNodeX*" ve "*mndNodeY*" alanları bu çalışmada kötücül düğümü bulma algoritmasındaki düğümlerin konumlarını belirtmek için kullanılmaktadır.

```
struct hdr_aodv_request {
        u_int8_t        rq_type;        // Packet Type
        u_int8_t        reserved[2];
        u_int8_t        rq_hop_count;   // Hop Count
        u_int32_t       rq_bcast_id;    // Broadcast ID
        nsaddr_t        rq_dst;         // Destination IP Address
        u_int32_t       rq_dst_seqno;   // Destination Sequence Number
        nsaddr_t        rq_src;         // Source IP Address
        u_int32_t       rq_src_seqno;   // Source Sequence Number
        double          rq_timestamp;   // REQUEST gönderilen zamant;
        u_int_32_t      mndNodeX;       // 1. eklenen alan
        u_ınt_32_t      mnxNodeY        // 2.eklenen alan
        inline int size() {
           int sz = 0;
           sz = 8*sizeof(u_int32_t);
          return sz;
        }
}
```

Şekil 4.40. AODV'de RREP paket yapısı



Şekil 4.41. Baz istasyonu tüm düğümlerden adım sayılarını talep etmektedir.

Algoritmanın birinci adımında saldırının olup olmadığı incelenmektedir. Şekil 4.41'de gösterildiği gibi bu işlemi yapabilmek için baz istasyonu "broadcast" olarak bir paket, tüm ağdaki düğümlerden adım sayılarını baz istasyonundan talep etmektedir. Bu işlemi yapabilmek için "*AODV.cc*" dosyasında "*sendRequest()*" ve "rrep_insert()" fonksiyonlarında yapılan değiştirilmeler Şekil 4.42'de gösterilmektedir. Bu çalışmada tasarlanan senaryolarda 30 düğüm (düğüm 0, düğüm 1,… , düğüm 29) farz edilmektedir. En son düğüm (düğüm 29) baz istasyonu olarak "*here._add*" komutu ile bu fonksiyonlarla belirtilmektedir. RREQ paketinin

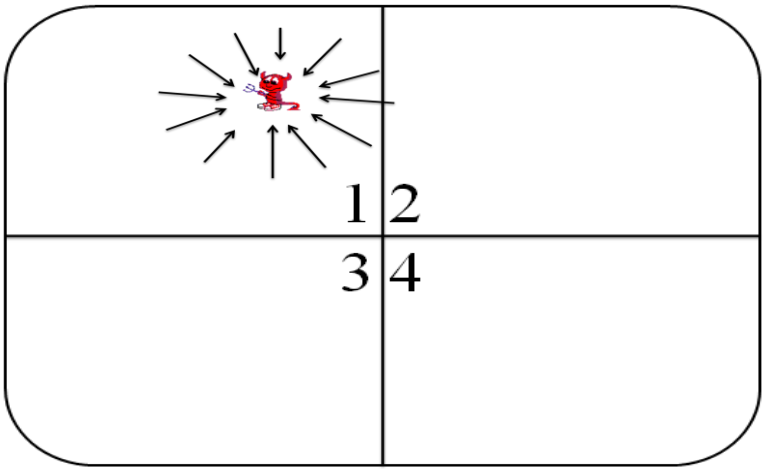
alanları "*sendRequest()*" fonksiyonunda belirtilmektedir. Bu fonksiyonlarda tüm "*AODV*" yerine yeni tasarlanan protokol "*mndAODV"* (Malicious Node Detection AODV) kullanılmaktadır.

```
Void
mndAODV::rrep_insert(nsaddr_t id) {
   if  (here_.add== Sink) {
    mndBroadcastRREP *r = new mndBroadcastRREP(id);
    assert(r);
    r->expire = CURRENT_TIME + BCAST_ID_SAVE;
    r->count ++;
    LIST_INSERT_HEAD(&rrephead, r, link);
   }
}
/*************************************************/
void
mndAODV::sendRequest(nsaddr_t dst) {
    if  ((here_.add== Sink)&&(firstT)) {
      Packet *p = Packet::alloc();// Allocate a RREQ packet
      struct hdr_cmn *ch = HDR_CMN(p);
      struct hdr_ip *ih = HDR_IP(p);
      struct hdr_mndaodv_request *rq = HDR_mndAODV_REQUEST(p);
      mndaodv_rt_entry *rt = rtable.rt_lookup(dst);
      assert(rt);
      firstT=false;
      ...
      rq->rq_type = mndAODVTYPE_RREQ;
      rq->rq_hop_count = 1;
      rq->rq_bcast_id = bid++;
      rq->rq_dst = dst;
      rq->rq_dst_seqno = (rt ? rt->rt_seqno : 0);
      rq->rq_src = index;
      seqno += 2;
      assert ((seqno%2) == 0);
      rq->rq_src_seqno = seqno;
      rq->rq_timestamp = CURRENT_TIME;
      rq->mndNodeX =indexX;
      rq->mndNodeY =indexY;
      Scheduler::instance().schedule(target_, p, 0.);
   }
    else{
   ...
   }
   ...
}
```

Şekil 4.42. *sendRequest()* ve *rrep_insert()* fonksiyonlarında yapılan değiştirilmeler

Bu aşamada baz istasyonu tarafından tüm düğümlere bir paket gönderilip düğümlerin konum bilgileri talep edilmektedir. Ayrıca ağın fiziksel ortamı 4 küçük alana ayrılmakta ve baz istasyonu bu alanlardan gelen verileri birbiriyle karşılaştırdıktan

sonra verilerin uyumlu olup olmadığı incelemektedir. Verilerin uyumlu olmaması ağda solucan deliği saldırısının olma ihtimalini göstermektedir. Şekil 4.43'te gösterildiği gibi eğer saldırı bir alandaysa o alandan gelen paket sayısı başka alanlardan çok az olmalıdır.



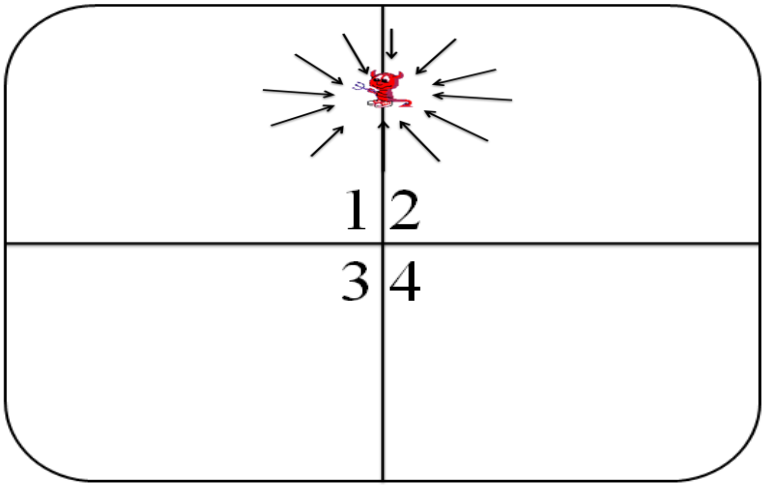
Şekil 4.43. Algoritmanın 1. aşamasında ağın fiziksel ortamını 4 küçük alana bölünmektedir ve saldırı 1. alanda bulunmaktadır

```
void
mndAODV::recvRequest(Packet *p) {

  struct hdr_ip *ih = HDR_IP(p);
  struct hdr_mndaodv_request *rq = HDR_mndAODV_REQUEST(p);
  mndaodv_rt_entry *rt;

  ...

  if(rq->rq_dst == index){

     if((here_.add== Sink)&&(mnd) {

      if( rq->mndNodeX <MaxX/2)) {
          if (rq->mndNodeY <MaxY<2)
             rqueue.enque(p,1)
          else
             rqueue.enque(p,3)
      } else{
           if (rq->mndNodeY <MaxY<2)
             rqueue.enque(p,2)
          else
             rqueue.enque(p,4)
      }
     } else{
     ...
     }
  }else{
    ...
}
```

Şekil 4.44. "mndAODV" algoritmasında SAV tablosu oluşturmak.

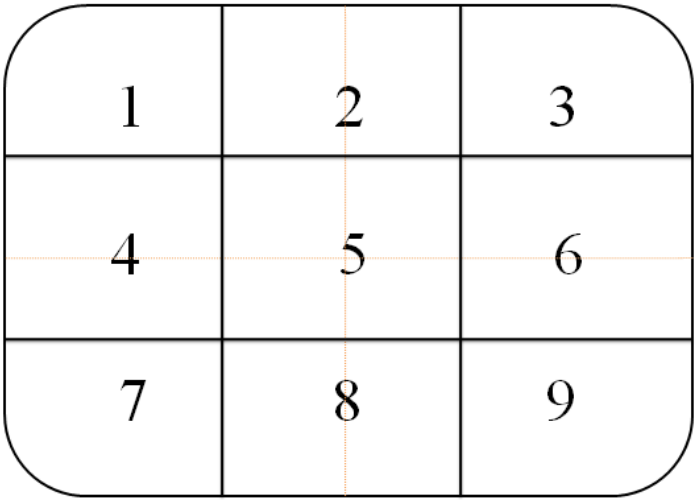
Baz istasyonundan gönderilen istek paketinin cevabı tüm düğümlerden baz istasyonuna gelmektedir. Baz istasyonunda bu gelen paketler analiz edilip bir anormallik olup olmadığı belirtilmektedir. Şekil 4.44'te gösterildiği gibi bu aşamada kötücül düğümü bulma algoritmasında kullanılan SAV tablosunu oluşturmak için her alanın paketleri ayrı ayrı kuyrukta kayıt edilmektedir. SAV tablosu Bölüm 4.1'de açıklanmıştır. Bu paketlerin sayısı, kötücül düğümün konumunun belirtilmesinde önemli yere sahiptir. Paket sayısı az olan alan, kötücül düğümün alanını belirtmektedir.

Eğer saldırı bu alanların sınırındaysa, iki alandan baz istasyonuna gelen paket sayısı diğer iki alana göre az olmalıdır. Örneğin Şekil 4.45'te saldırı 1. ve 2. alanların sınırlarındaysa, 1. ve 2. alanlardan gelen tüm paketlerin sayısı, 3. ve 4. alanlardan gelen paketlerin sayısından çok az olmalıdır.



Şekil 4.45. Saldırı 1. ve 2. alanların sınırlarında bulunmaktadır.

Kötücül düğüm iki alanın arasında bulunduğu zaman iki alandan gelen paket sayısı azalmaktadır. Bu durumda saldırılmış alanı bulabilmek için ağın fiziksel alanı 9 küçük alana ayrılmaktadır. Şekil 4.46'da gösterildiği gibi sınırdaki alanlar 2, 4, 5, 6 ve 8. alanlara ayrılmaktadır. Eğer ağın fiziksel alanı çok büyükse bu ayırma işlemi tekrarlama (özyineleme) yaparak daha küçük bir alana dönüşebilir.



Şekil 4.46. Algoritmanın 1. aşamasında ağın fiziksel ortamı 4 yerine 9 küçük alana bölünmektedir.

Sözü edilen işlemleri yapabilmek için "*mndaodv_packet.h*" ve "*mndaoldv.cc*" adlı dosyaların değiştirilmeleri gerekir. Bu aşamada baz istasyonuna gelen paketlerin sayısı analizi yapılarak önemli ölçüde fark olduğunda kötücül düğümün bulunma ihtimali vardır. Başka bir deyişle kötücül düğümün ağda bulunması bu adımda belirtilmektedir. İkinci adımda kötücül düğümü bulma algoritması kullanarak bu kötücül düğümün bulunulmasına çalışılmaktadır.

4.8.2. Kötücül düğümü bulma algoritmasi

Bu durumda Kötücül Düğümleri Bulma (KDB) algoritması kullanarak şüpheli alandaki kötücül düğüm ya da düğümler bulunmaktadır. Baz istasyonu ilk adımda kendi komşularını belirlemektedir. Şekil 4.47'de gösterildiği gibi bu işlem aodv.cc dosyasındaki *reccHello()* fonksiyonunda yapılmaktadır.

"*aodv.cc"* dosyasında "*local_rt_repair()*" ve *"rt_resolve()"* fonksiyonları ağa bir link eklendiğinde veya kaldırıldığında çalıştırılıp yeni yollar belirtilmektedir. Şekil 4.48 ve Şekil 4.49'da gösterildiği gibi yapılan çalışmada bu fonksiyonlardan yararlanarak, kötücül düğüm normal düğümlerden ayrılıp ağdan dışlanmaktadır.

```
void
mndAODV::recvHello(Packet *p) {
  struct hdr_ip *ih = HDR_IP(p);
  struct hdr_aodv_reply *rp = HDR_AODV_REPLY(p);
  mndAODV_Neighbor *nb;
  if  (here_.add== Sink) {
    nb = nb_lookup(rp->rp_dst);
    if(nb == 0) {
      nb_insert(rp->rp_dst);
    }
    else {
      nb->nb_expire = CURRENT_TIME +
                  (1.5 * ALLOWED_HELLO_LOSS *
      HELLO_INTERVAL);
    }
    Packet::free(p);
  }
}
```

Şekil 4.47. Baz istasyonun komşuları *recvHello()* fonksiyonunda belirtilmektedir.

```
void
AODV::local_rt_repair(aodv_rt_entry *rt, Packet *p) {

...
   rqueue.enque(p);
   rt->rt_flags = RTF_IN_REPAIR;
   sendRequest(rt->rt_dst);
   Scheduler::instance().schedule(&lrtimer, p->copy(), rt-
                                          >rt_req_timeout);
...
}
```

Şekil 4.48. AODV protokolünde değiştirilmiş *local_rt_repair()* fonksiyonu

```
void
AODV::rt_resolve(Packet *p) {
   struct hdr_cmn *ch = HDR_CMN(p);
   struct hdr_ip *ih = HDR_IP(p);
   aodv_rt_entry *rt;
   ...
   ch->xmit_failure_ = aodv_rt_failed_callback;
   ch->xmit_failure_data_ = (void*) this;
   rt = rtable.rt_lookup(ih->daddr());
   if(rt == 0) {
        rt = rtable.rt_add(ih->daddr());
   }
   if(rt->rt_flags == RTF_UP) {
      assert(rt->rt_hops != INFINITY2);
      forward(rt, p, NO_DELAY);
   }
   else if(ih->saddr() == index) {
      rqueue.enque(p);
      sendRequest(rt->rt_dst);
   }
   else if (rt->rt_flags == RTF_IN_REPAIR) {
      rqueue.enque(p);
   }
   else {
      Packet *rerr = Packet::alloc();
      struct hdr_aodv_error *re = HDR_AODV_ERROR(rerr);
      assert (rt->rt_flags == RTF_DOWN);
      re->DestCount = 0;
      re->unreachable_dst[re->DestCount] = rt->rt_dst;
      re->unreachable_dst_seqno[re->DestCount] = rt->rt_seqno;
      re->DestCount += 1;
      sendError(rerr, false);
      drop(p, DROP_RTR_NO_ROUTE);
   }
    ...
}
```

Şekil 4.49. AODV protokolünde değiştirilmiş *rt_resolve ()* fonksiyonu

Bir sonraki bölümde çeşitli senaryolar tasarlanarak algoritmanın test edilmesi ve performans ölçümü gösterilmiştir.

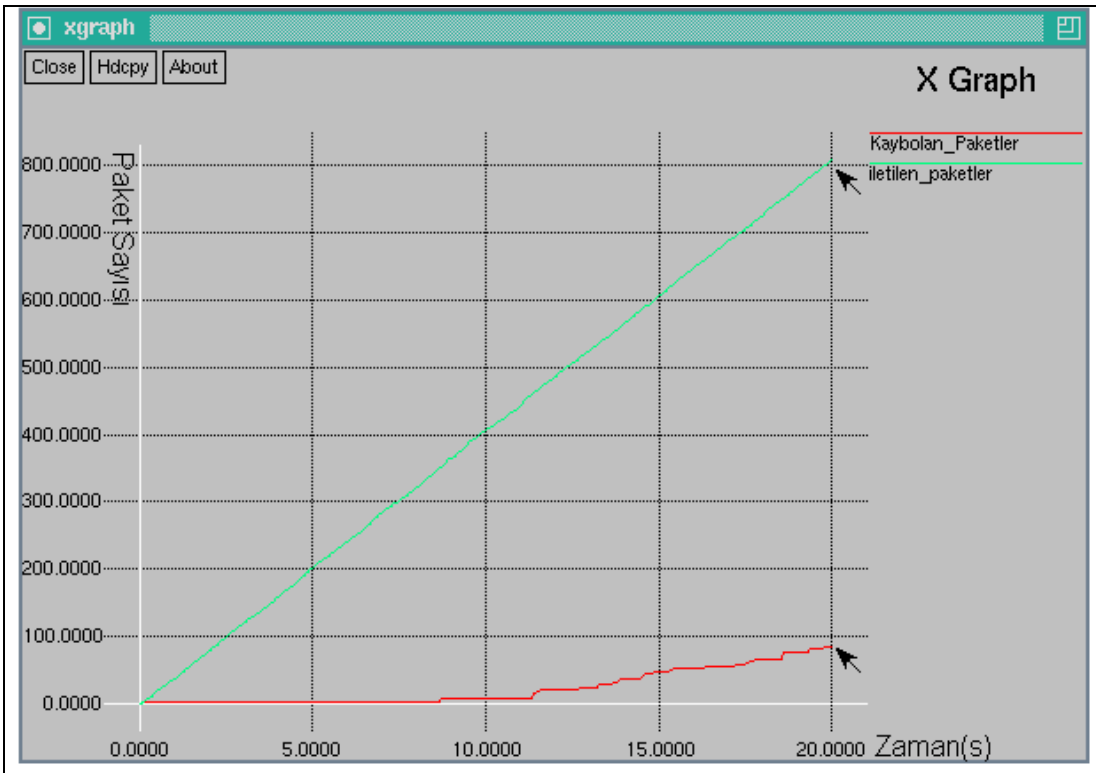
5. DENEYSEL BULGULAR

AODV protokolünde (kötücül düğümler hariç) iki önemli parametre paket kaybına yol açmaktadır. Tasarlanan algoritmanın performansında bu iki parametre göz önüne alınması gerekmektedir. Bu bölümde tasarlanan algoritmayı test etmeden önce bu iki önemli parametre ve paket kaybına etkileri açıklanmaktadır.

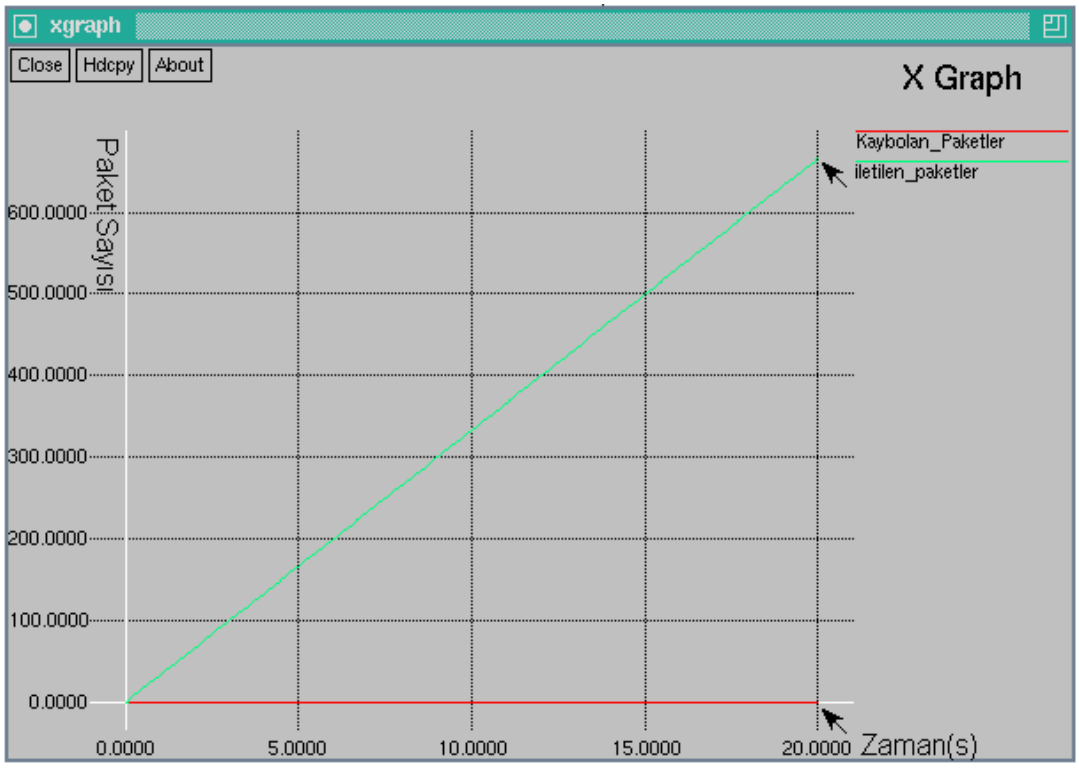
5.1. Paket Kaybında Gecikmenin Etkisi

Gecikme, gönderilen paketlerin arasındaki zaman farkıdır. Şekil 5.1'de gösterildiği gibi bu zaman arttıkça kaybolan paket sayısı azalmaktadır. Simülasyon parametreleri her şeklin altında gösterilmektedir. Yapılan simülasyonlarda paket boyutu 512 ve simülasyon süresi 20 saniye farz edilmiştir. Şekil 5.1'in (a) kısmında gecikme 0.02 saniye farz edilmektedir ve simülasyonun sonunda 83 paket, kayıp olmaktadır. Gecikme arttıkça bir belli değerden sonra paket kaybı sıfır olmaktadır. Şekil 5.1 (b) kısmında gecikme 0,03 saniye olduğunda hiçbir paket kaybı bulunmamaktadır. Bu zamanı arttırma sadece paket sayısının azalmasına neden olmaktadır. KAA'larda veri miktarı çok fazla değildir. Bu nedenle tasarlanan algoritmanın test edilmesinde bu değeri sıfıra yaklaştırmak gerekmektedir. Algoritmanın performans analizinde eğer gecikme nedeniyle paket kaybı olursa onun göz önüne alınması gerekmektedir. Bu paket kayıpları AODV paket kaybından sayılmaktadır.

Çizelge 5.1'de gecikme parametresinin paket kaybına etkisi çeşitli değerlerle ölçülmüş olup, Şekil 5.2'de bu değerlerin karşılaştırılması gösterilmiştir. Şekil 5.2'de görüldüğü gibi gecikme süresi arttıkça paket kaybı azalmaktadır. Örneğin gecikme değeri 0,0005 saniye olduğunda gönderilen paketlerin %97,8 oranı hedefe ulaşamamaktadır ve sadece %2,2 oranında gönderilen paketlerden hedefe iletilmektedir. Tam tersi eğer gecikmenin değeri 0,03 saniye olduğunda tüm gönderilen paketler hedefe ulaşmaktadır.



(a) Gecikme= 0,02s (Paket Boyutu= 500 bit, İletilen =808 Kayıp=83)

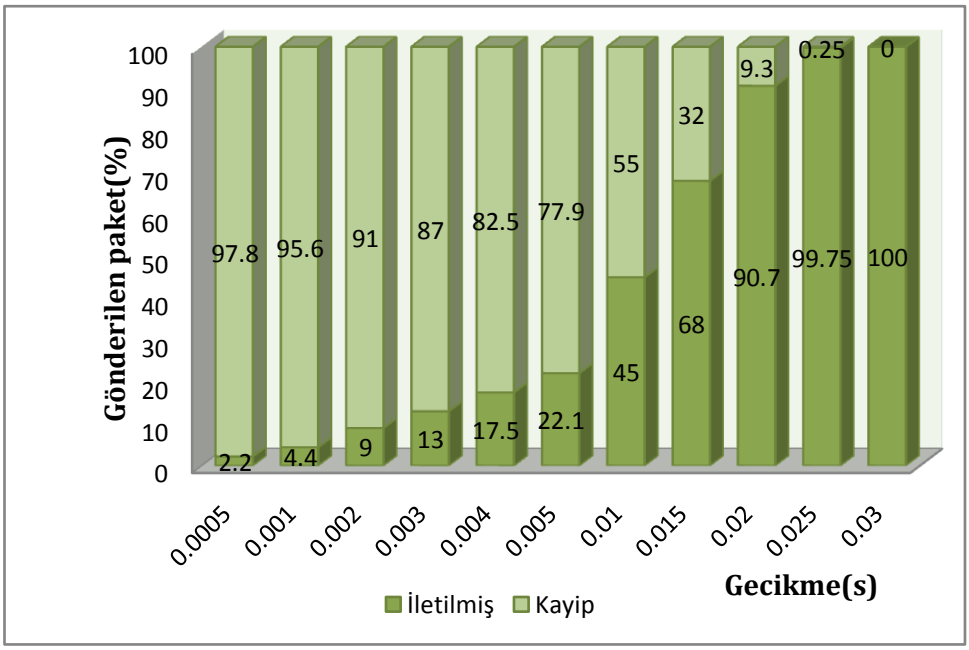


(b) Gecikme= 0,03s (Paket Boyutu= 512 bit, İletilen =666, Kayıp=0)

Şekil 5.1. Gecikme arttıkça kaybolan paketlerin sayısı azalmaktadır.

Çizelge 5.1. Kaybolan paketler ile gecikme parametresinin ilişkisi

Gecikme (Saniye)	Toplam gönderilen paket sayısı	Hedefe iletilen paket sayısı		Kayıp olan paket sayısı	
0.0005	36424	808	%2,2	35616	%97,8
0.001	18277	808	%4,4	7469	%95,6
0.002	9299	808	%9	8491	%91,0
0.003	6254	808	%13	5446	%87,0
0.004	4615	808	%17,5	3807	%82,5
0.005	3648	808	%22,1	2840	%77,9
0.010	1791	808	%45	983	%55,0
0.015	1187	808	%68	379	%32,0
0.020	891	808	%90,7	83	%9,3
0.025	802	800	%99,75	2	%0,25
0.030	666	666	%100	0	%0,00



Şekil 5.2. Gecikme arttıkça kaybolan paketlerin sayısı azalmaktadır.

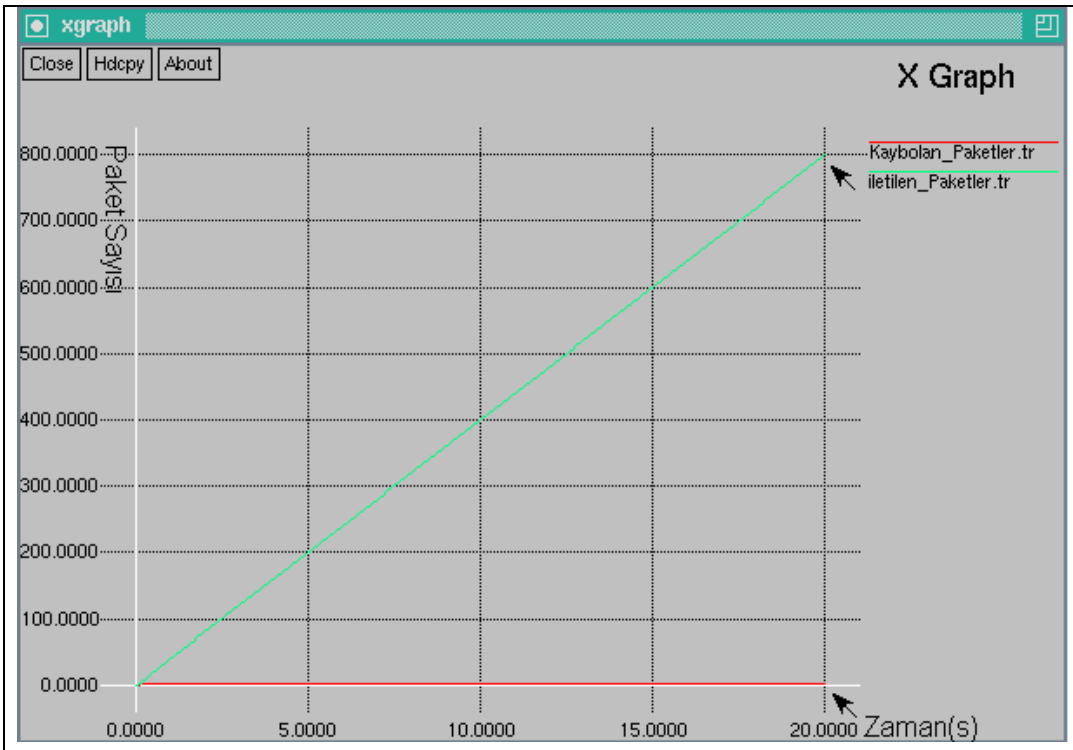
5.2. Paket Kaybına Paket Boyutunun Etkisi

Paketlerin boyutu ikinci parametre olarak incelenmişti. Şekil 5.3'te gösterildiği gibi bu değer arttıkça kaybolan paket sayısı artmaktadır. KAA'larda 512 bit (64 bayt) uygun bir boyut sayılmaktadır[4]. Çeşitli uygulamalarda bu değer değiştirilmektedir. Bu çalışmada senaryolar 512 bit ile tasarlanmıştır.

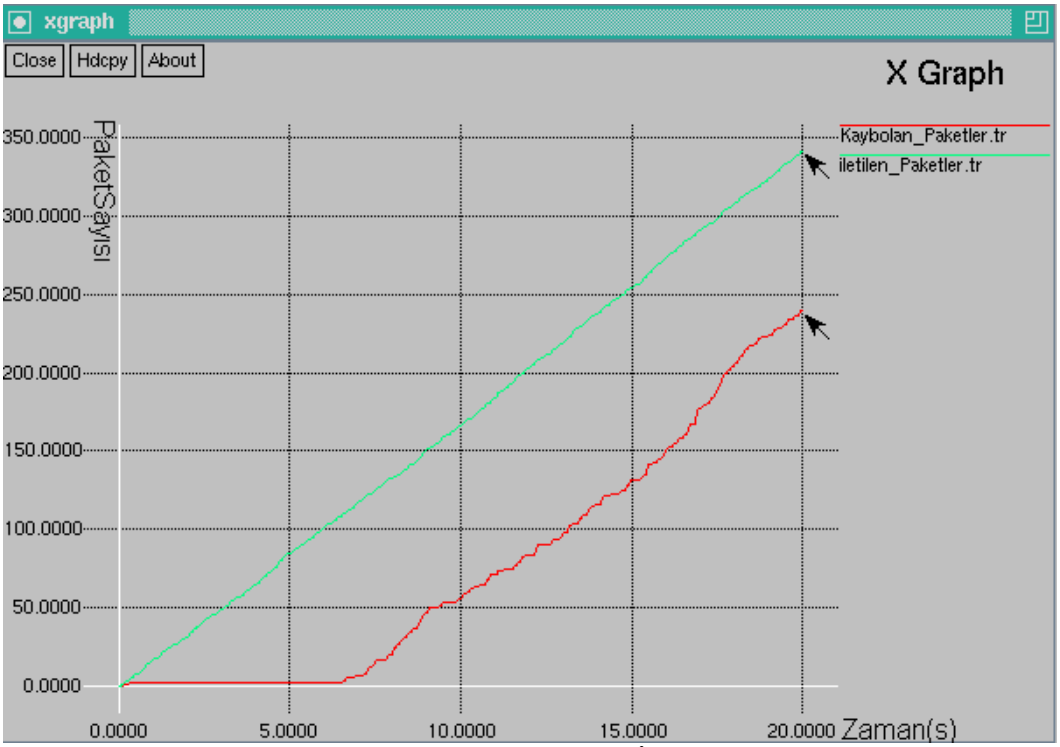
Çizelge 5.2'de paket boyutu parametresinin, paket kaybındaki etkisi çeşitli değerlerle, çeşitli senaryolarda ölçülüp Şekil 5.4'te bu değerlerin karşılaştırılması gösterilmiştir. Bu şelilde gösterildiği gibi paket boyutu arttıkça, paket kaybı artmaktadır. Örneğin paket boyutu 500 bit olduğunda gönderilen paketlerin %99.8 oranında hedefe iletilmiştir ama bu parametrenin değeri 1500 bit olduğu halde kaybolan paketlerin sayısı iletilen paketlerin %80 oranından da fazladır. Performans analizinde eğer paket boyutu nedeniyle paket kaybı olursa onun da göz önüne alınması gerekmektedir. Gecikme parametresi gibi bu paket kayıpları da AODV paket kaybından sayılmaktadır.

Çizelge 5.2. Kaybolan paket sayısı ile paket boyutunun ilişkisi

Paket boyutu (Bit)	Toplam gönderilen paket sayısı	Hedefe iletilen paket sayısı		Kaybolan paket sayısı	
500	802	800	%99,8	2	%0.2
600	742	738	%99	4	%1
700	694	641	%92	53	%8
800	680	580	%85	100	%15
900	660	527	%80	133	%20
1000	652	476	%75	163	%25
1100	635	452	%71	183	%29
1200	627	414	%66	213	%34
1300	607	376	%62	231	%38
1400	603	366	%60	237	%40
1500	580	340	%58	240	%42

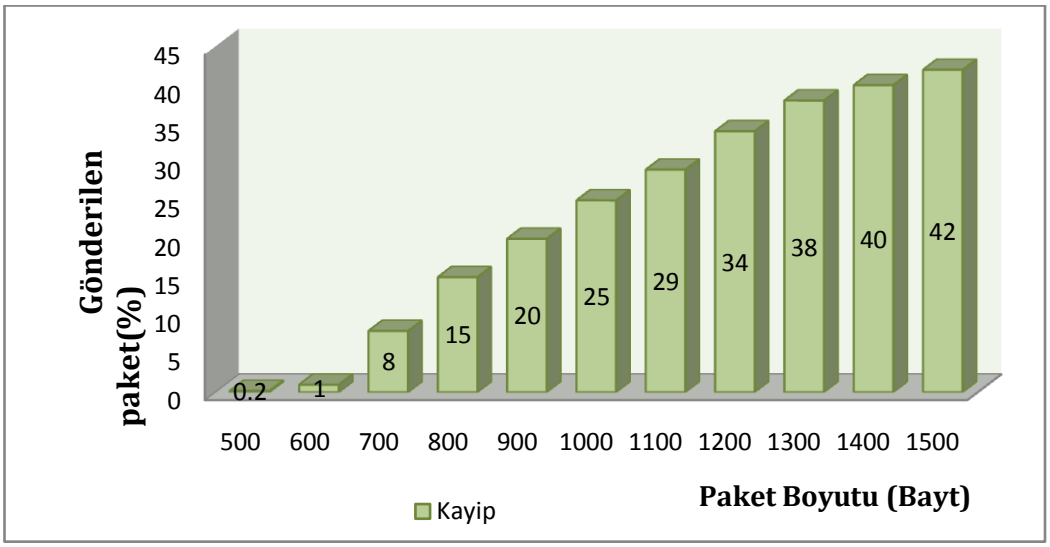


(a) Paket Boyutu= 500 bit (gecikme=0.025s, İletilen=800, Kayıp=2)



Paket Boyutu= 1500 bit (gecikme=0.025s, İletilen=340, Kayıp=240)

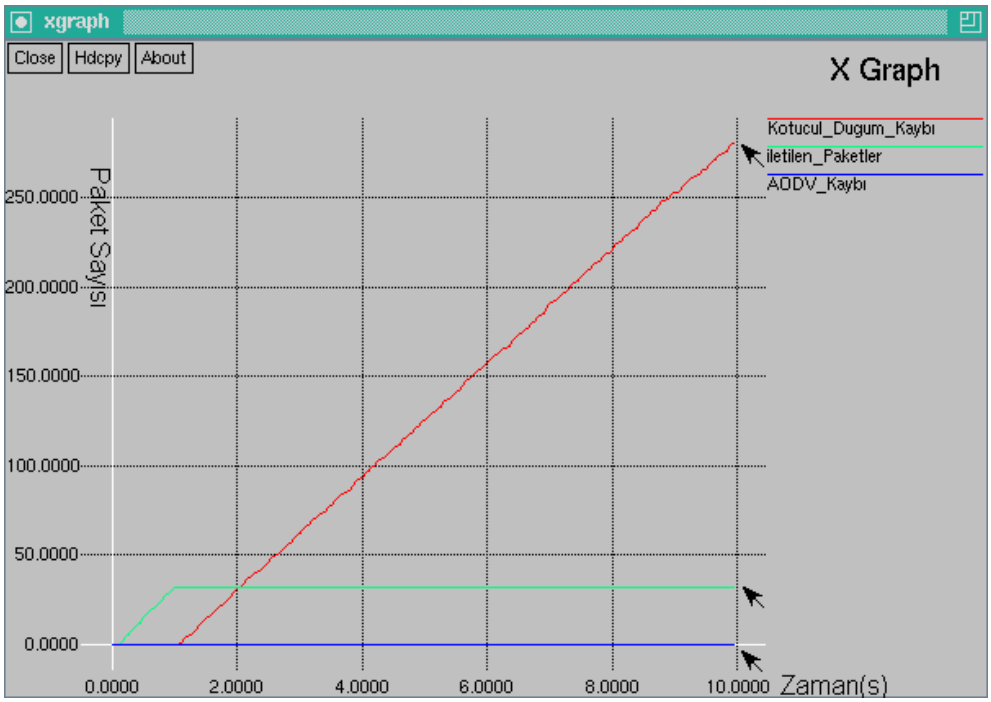
Şekil 5.3. Paket boyutu artması ile kaybolan paket sayısı arasındaki ilişki.



Şekil 5.4. Kaybolan paket sayısı ile paket boyutunun ilişkisi

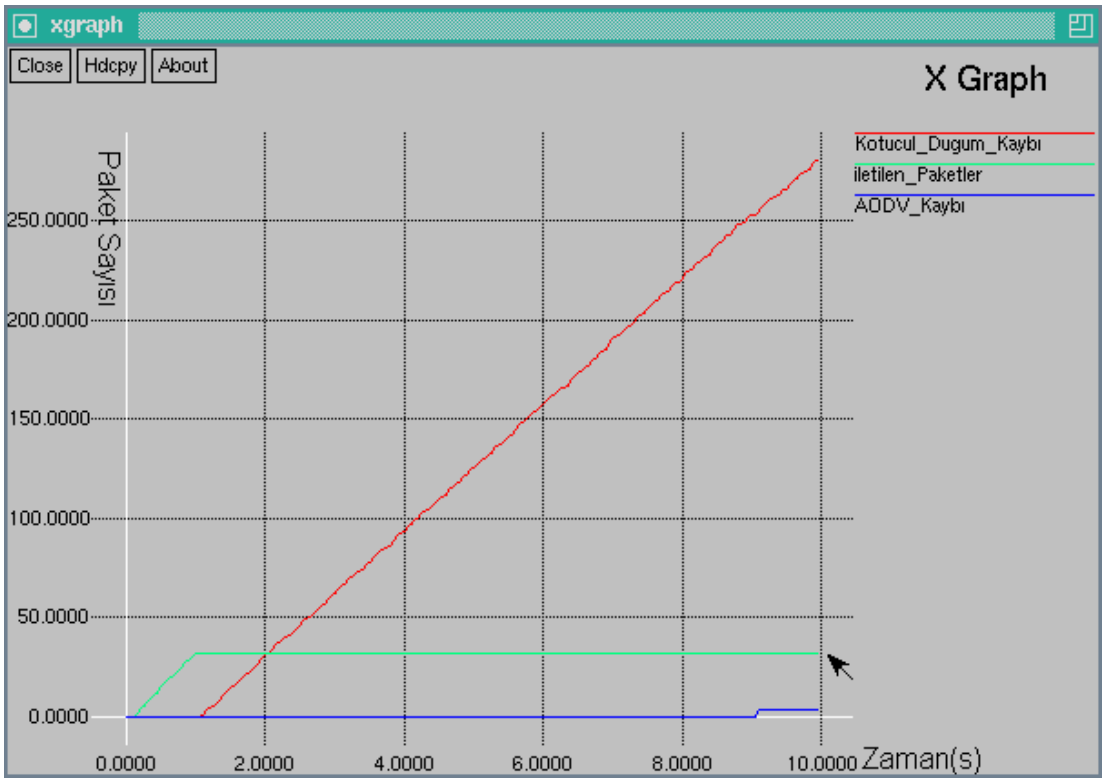
5.3. Paket Kaybında Kötücül Düğümün Etkisi

Şekil 5.5'te bir KAA da 29 normal düğüm ve bir kötücül düğüm olarak senaryonun sonucu gösterilmiştir.



Şekil 5.5. Paketlerin kaybolmasında kötücül düğümün etkisi, (AODV kaybı yok)

Bu senaryoda bir kötücül düğüm aktif hale gelir kendini komşularına baz istasyonuna en yakın düğüm olarak belirtmektedir, baz istasyonuna giden tüm paketleri toplayıp çöpe atmaktadır. Bu nedenle baz istasyonuna giden paketler kötücül düğüme gönderilip baz istasyonuna hiçbir paket iletilmemektedir. Bu senaryo AODV yönlendirme protokolü üzerinden yapılmaktadır ve AODV paket kayıpları 0 gözükmektedir. Şekil 5.6'daki bir başka senaryoda AODV paket kayıpları 2 paket gözükmektedir. Gecikme parametresinin değiştirilmesi bu iki senaryonun farkına neden olmaktadır.

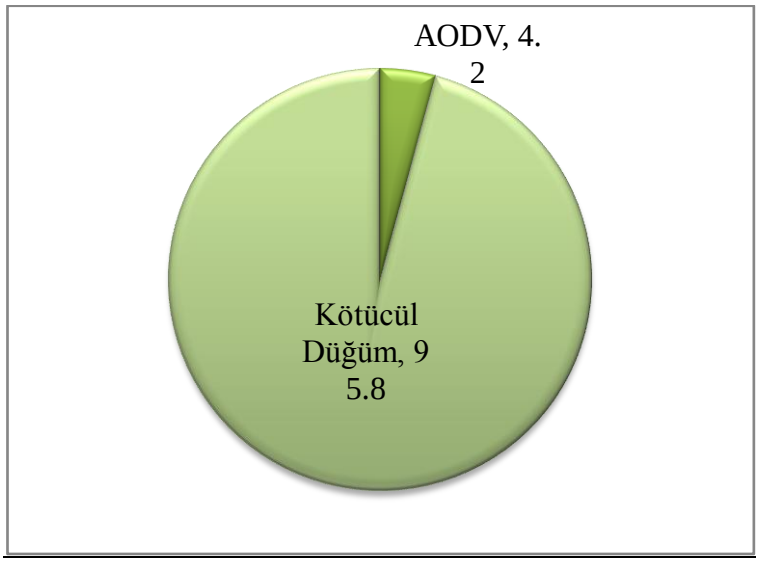


Şekil 5.6. Paketlerin kaybolmasında kötücül düğümün etkisi, (AODV kaybı=2)

Çizelge 5.3. Paketlerin kaybolmasında saldırı ve AODV'den kaynaklanan etkisi

Senaryolar	Toplam gönderilen paket sayısı	Hedefe iletilen paket sayısı		Kaybolan paket sayısı					
				Toplam		AODV		Saldırı	
Senaryo 1	144	14	%9,8	130	%90,2	5	%3,8	125	%96,2
Senaryo 2	150	16	%10,7	134	%89,6	7	%5,2	127	%94,8
Senaryo 3	135	17	%12,6	118	%89,5	8	%6,8	110	%93,2
Senaryo 4	140	10	%7,1	130	%91,8	4	%3,0	126	%97,0
Senaryo 5	138	10	%7,3	128	%91,6	8	%6,3	120	%93,7
Senaryo 6	150	13	%8,6	137	%91,1	9	%6,6	128	%93,4
Senaryo 7	145	18	%12,4	127	%89,4	0	%0,0	127	%100
Senaryo 8	137	19	%13,9	118	%88,5	7	%5,9	111	%94,1
Senaryo 9	139	19	%13,7	120	%89,6	0	%0,0	120	%100
Senaryo 10	141	8	%5,7	133	%92,5	5	%3,8	128	%96,2
Toplam/ Ortalama	1419	144	%10,1	1275	%89,9	53	%4,2	1222	%95,8

Doğru bir sonuca varabilmek için birçok senaryo hazırlanmıştır. Çizelge 5.3'te bu senaryolarda toplam gönderilen paketlerin sayısı, hedefe iletilen paket sayısı ve kaybolan paket sayısı gösterilmiştir. Ayrıca çizelgede kötücül düğümden kaybolan paket sayısı ve AODV protokolünden kaybolan paket sayısı ayrı ayrı gösterilmiştir. Bu çizelgenin özeti Şekil 5.7'de gösterilmiştir.



Şekil 5.7. Paketlerin kayıp nedenleri (Tüm senaryoların ortalaması)

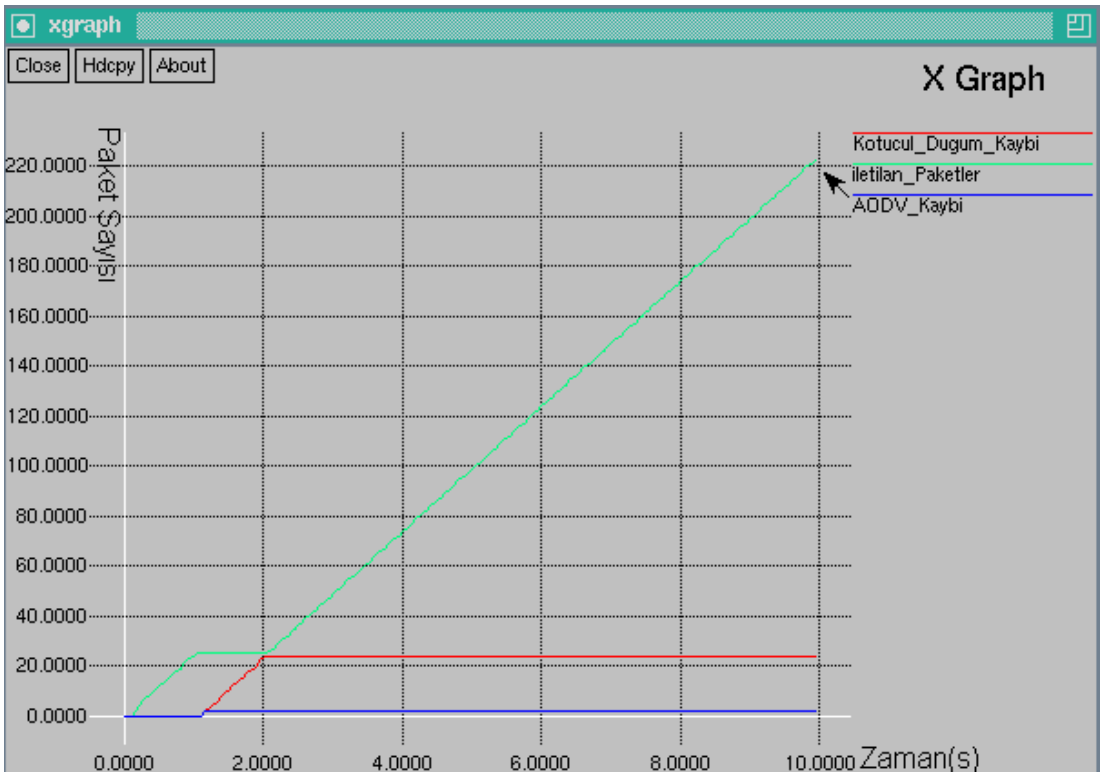
5.4. Paket Kaybının Azaltmasında KDB Algoritmasının Etkisi

KDB algoritmasının performansını test edebilmek için çeşitli senaryolar hazırlanmıştır. Bu bölümde öncelikle ağ normal çalıştırılmıştır. Daha sonra bir kötücül düğüm aktif hale getirilip bu düğüm kendini komşularına baz istasyonuna en yakın düğüm olarak ifade etmiştir. Baz istasyonuna giden tüm paketleri toplayıp çöpe atmaktadır. Daha sonra da kötücül düğümün etkisini yok etmek için KDB algoritması çalıştırılmıştır.

Paket kaybında KDB algoritmasının etkisi, algoritma çalıştırılmadan önce ve çalıştırdıktan sonra çeşitli senaryolarda çizelge 5.4 bulunmaktadır. Bu senariyoların ortalama sonuçları çizelgenin son satırında gösterilmektedir. Son satırda gosterildiği gibi KBD algoritması çalıştıktan sonra %83,6 ornında (2015 paket) kaybolan paketlerin sayısı azalmaktadır. Şekil 5.8'de sözü edilen senaryolardan birisi gösterilmiştir.

Çizelge 5.4. Çeşitli senaryolarda paketlerin kaybolmasındaki KDB algoritmasının etkisi

Senaryolar	Kötücül düğüm etkisinden kaybolan paketler				Fark	
	KDB Algoritmasından önce		KDB Algoritmasından Sonra			
Senaryo 1	222	90,1	22	%9,9	-220	- %80,2
Senaryo 2	225	90,0	25	%10	-200	- %80
Senaryo 3	230	91,3	22	%9,7	-208	- %81,6
Senaryo 4	218	91,6	20	%9,4	-198	- %82,2
Senaryo 5	220	92,4	18	%7,6	-202	- %84,8
Senaryo 6	235	90,4	25	%9,6	-210	- %80,8
Senaryo 7	210	93,3	15	%6,7	-195	- %86,6
Senaryo 8	204	93,2	15	%6,8	-189	- %86,4
Senaryo 9	235	90,4	25	%9,6	-210	- %80,8
Senaryo 10	213	95,5	10	%4,5	-203	- %91
Toplam/ Ortalama	2212	91,8	197	%8,2	2015	-%83,6



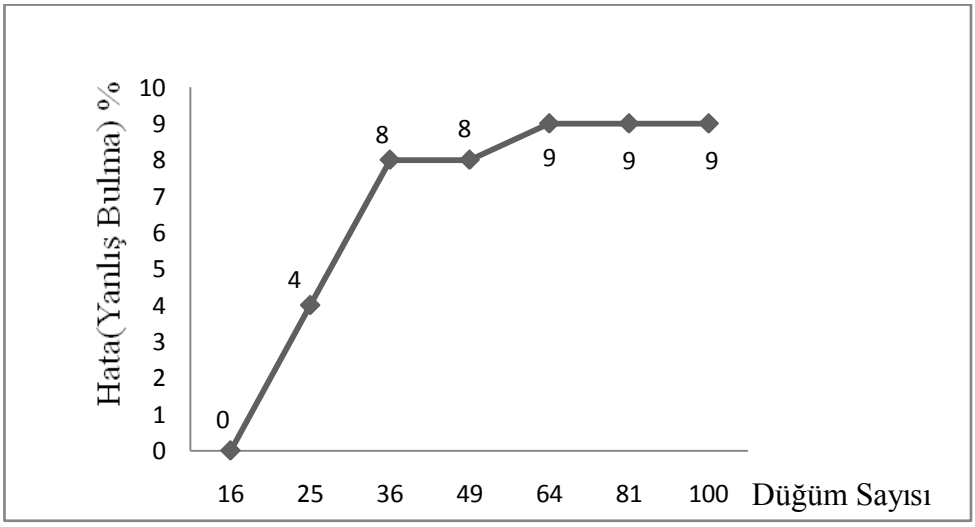
Şekil 5.8. Bir örnek senaryoda paket kaybında KBD algoritmasının etkisi

5.5. KDB Algoritmasında, Düğüm Sayısının Etkisi

KDB Algoritmasında düğüm sayısının etkisini incelemek için çeşitli senaryolar çeşitli ölçülerde (scale) hazırlanmıştır. Bu senaryoların sonucu Çizelge 5.5'te gösterilmektedir. Tüm bu senaryolarda bir tane kötücül düğüm bulunmaktadır. çizelgede gösterildiği gibi algoritmada iki çeşit *negatif* ve *pozitif* yanlışlık, bulunabilir. Eğer ağda kötücül düğüm varsa ve algoritma bulamazsa *negatif yanlışlık* ve eğer ağda her hangi bir normal düğüm kötücül düğüm tanınacak olursa *pozitif yanlışlık* adlandırılmıştır. Çizelgede gösterildiği gibi ağda bir kötücül düğüm bulunduğunda çeşitli senaryolarda pozitif yanlışlık bulunmamaktadır. Ağın boyutu arttıkça bulunmayan kötücül düğümlerin sayısı da artmaktadır. Senaryoların ortalaması negatif yanlışlık parametresini %6,7 göstermektedir. Bu yanlışlık, ağda kaybolan paketlerden kaynaklanmaktadır.

Çizelge 5.5 Algılama sonuçları (ağda 1 adet kötücül düğüm bulunduğunda)

Ağın Boyutu	Ağdaki Düğüm Sayısı	Kötücül Düğüm Sayısı	Doğru Bulma	Yanlış Bulma			
				Negatif		Pozitif	
			(%)	Sayı	(%)	Sayı	(%)
4×4	16	1	%100	0	%0	0	%0
5×5	25	1	%96	1	%4	0	%0
6×6	36	1	%92	3	%8	0	%0
7×7	49	1	%92	4	%8	0	%0
8×8	64	1	%91	6	%9	0	%0
9×9	81	1	%91	7	%9	0	%0
10×10	100	1	%91	9	%9	0	%0
Ortalama	***	***	**%93,3**	***	**%6.7**	***	**%0**



Şekil 5.9. KBD algoritmasında düğüm sayısının etkisi

Şekil 5.9'da KDB algoritmasında düğüm sayısının etkisi gösterilmektedir. Bu şekilde sadece ağda bir tane kötücül düğüm olduğu farz edilmiştir. KDB algoritmasında kötücül düğüm sayısının etkisi Bölüm 5.6'da incelenmektedir.

5.6. KDB Algoritmasında, Kötücül Düğüm Sayısının Etkisi

Bölüm 5.5'te, KBD algoritmasında bir tane kötücül düğümün etkisi incelenmiştir. Bu bölümde kötücül düğüm sayısının etkisini, ölçmek için çeşitli senaryolar hazırlanmıştır. Bu senaryoların sonucu Çizelge 5.6'da gösterilmektedir.

Çizelge 5.6. KBD algoritmasında kötücül düğüm sayısının etkisi

Kötücül Düğümler Sayısı	Doğru Bulma **(%)**	Yanlış Bulma	
		Negatif **(%)**	Pozitif **(%)**
1	%93.3	%6.7	%0.0
2	%93.1	%6.9	%0.0
3	%92.2	%7.8	%0.0
4	%91.6	%8.4	%0.0
5	%90.5	%9.5	%0.0
6	%90.1	%9.9	%0.0
7	%85.2	%10.2	%4.6
8	%82.8	%10.8	%6.4
9	%79.7	%12.5	%7.8
10	%75.7	%13.8	%10.5
Ortalama	**%87.42**	**%9.65**	**%2.93**

Çizelgede gösterildiği gibi kötücül düğümlerin sayısı arttıkça algoritmanın bulma hassasiyeti (Detection Precision) azalmaktadır çünkü kötücül düğümlerin sayısı arttıkça baz istasyona gelen bilgilerin doğruluk oranı da azalmaktadır. Eğer kötücül düğümlerin sayısı normal düğümlerden fazlaysa bu algoritma iyi performans göstermemektedir. Çizelgede gösterildiği gibi kötücül düğümlerin sayısı arttıkça pozitif yanlış bulma da artmaktadır. Çizelgenin ilk satırlarında kötücül düğümlerin sayısı hiçbir senaryoda normal düğümlerden fazla değildir. Bu nedenle *pozitif yanlış* bulma parametresinin değeri bu satırlarda sıfırdır. Ama son satırlarda pozitif yanlış bulma parametresinin değeri pozitiftir. Bu simülasyonların sonucunda KDB algoritması %87,42 oranında kötücül düğümü doğru bulmaktadır, %9.65 oranında ağda kötücül düğümü bulamıyor ve %2,92 oranında da normal düğümü, kötücül düğüm olarak bulmaktadır.

6. SONUÇLAR VE ÖNERİLER

Kablosuz iletişim teknolojilerinin bir çeşidi olan KAA'lar günümüzde üzerine en çok araştırma ve geliştirme yapılan konulardan biri haline gelmiştir. Bu ağlar günümüzde çok geniş uygulama alanı bulmaktadır. Bazı uygulamalarda özellikle askeri ve sağlık uygulamaları gibi güvenlik yönünden hassas veri iletiminin yapıldığı KAA'larda veri gizliliğini ve bütünlüğünü sağlayacak güvenlik mekanizmalarına fazlasıyla ihtiyaç vardır.

Bu ağlarda, algılayıcılar bulundukları sahada kötü niyetli kişiler tarafından ele geçirilebilir ve kötücül amaçlar için kullanılabilirler. Bu nedenle bu ağlara özgü birçok saldırı çeşidi bulunmaktadır. Sinkhole deliği saldırısı bu çalışmada ele alınmış ve bu atağa karşı bir savunma mekanizması önerilmiştir. Bu mekanizma ağ içindeki veri hareketlerinin analizine ve kötücül düğümlerin "oy çokluğu" metoduna göre tespitine dayanmaktadır. Önerilen çözüm iki aşamadan oluşmaktadır. Birinci adımda saldırının olup olmadığı incelenmektedir. İkinci aşamada kötücül düğüm bulunmaktadır. Önerilen savunma mekanizması NS-2 simülatörü kullanılarak gerçekleştirilmiştir.

6.1. Sonuçlar

Yönlendirme protokollerinde iki önemli parametre (gecikme ve paket boyutu) paket kaybına yol açmaktadır. Bu çalışmada bu iki parametrenin etkisi çeşitli değerlerle, çeşitli senaryolarda ölçülüp ve bu değerlerin etkisi paket kaybında incelenmiştir. Çeşitli senaryoların simülasyonu sonucunda, gecikme arttıkça kaybolan paket sayısı azalmaktadır. Buna karşın paket boyutu arttıkça kaybolan paket sayısı artmaktadır. Simülasyonlarda bu iki parametrenin değerinin göz önüne alınması gerekmektedir.

Önerilen savunma mekanizması sayesinde kaybolan paketlerin sayısının %83,6 oranında azaldığı tespit edilmiştir.

Önerilen yöntem AODV protokolü üzerinde gerçekleştirilmiştir. Yöntem AODV yönlendirme protokolüne yeni bir paket eklemeye gerek duymadan çalışmaktadır. Bu nedenle tüm standart AODV yönlendirme protokolü çalışan başka ağlarda da (örneğin ad-hoc ağlarda) uygulanabilir.

Önerilen yöntemin başka avantajlarından bazıları, düğümler üzerinde hiçbir donanım ilave edilmemesi, düğümler arasında hiçbir zaman ayarlamasına gerek duyulmaması ve normal düğümler üzerinde fazla yazılım kodu eklenmemesi söylenebilir.

6.2. Öneriler

Bu çalışmanın devamında bazı öneriler de bulunmuştur:

- Önerilen mekanizmanın başka mekanizmalarla (örneğin zaman tabanlı mekanizma) karşılaştırılması ve hangisinin ne zaman iyi sonuç vermesi önerilmektedir.

- Ad hoc ağlarda başka parametreler - örneğin mobile düğümler sayısı - kullanarak algoritmanın başka ortamlarda performansının test edilmesi tavsiye edilmektedir.

- Önerilen mekanizma AODV yönlendirme protokolü üzerinde yapılmaktadır. Başka yönlendirme protokollerinde (örneğin DSDV, DSR ve OLSR) test edilmesi ve en uygun protokolün seçilmelidir.

- Önerilen mekanizma başka mekanizmalarla (örneğin Güven Mekanizması) birleştirilmesi ve sonuçları karşılaştırılması tavsiye edilmektedir.

- İkinci önerilen çözümün gerçekleştirilmesi ve sonuçlar karşılaştırılmalıdır.

- KDB mekanizmasında, kötücül düğümün uzaklığı baz istasyonundan algoritmanın performansında etkisi ne kadar olmalıdır?

KAYNAKLAR

1. Chong, C., Y., Kumar, S., P., "Sensor Networks: Evolution, Opportunities, Challenges", ***Proceedings of The IEEE***, 91(8): 1247-1256 (2003).

2. Akyildiz, A., Su, I., F., Sankarasubramaniam, W., Cayirci, E., "A Survey on Sensor Networks", ***IEEE Communications Magazine***, 40(8): 102-114 (2002).

3. Gupta, H., Das, S., R., Gu, Q., "Connected Sensor Cover: Self-Organization of Sensor Networks for Efficient Query Execution", ***MobiHoc, USA***, 189-200 (2003).

4. Hadim, S., Mohamed, S., N., "Middleware Challenges and Approaches for Wireless Sensor Networks", ***IEEE Distributed Systems,*** 7(3):101-110 (2006).

5. Haenselmann, T, "Sensor Networks", ***FDL'ed Textbook on Sensor Network***, (2006).

6. Römer, K., Mattern, F., "The Design Space of Wireless Sensor Networks", ***IEEE Wireless Communications***, 11(6): 54-61 (2004).

7. Karl, H., Willig, A., "A Short Survey Of Wireless Sensor Networks", ***TKN Technical Reports Series***, (2003).

8. Akyildiz, I., F., Melodia, T., Chowdhury, K., R., "A survey On Wireless Multimedia Sensor Networks", ***Computer Networks Journal***, 51(4): 921- 960 (2006).

9. Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W., "Surface Reconstruction From Unorganized Points", ***ACM SIGGRAPH Computer Graphics***, 26(2): 71-78 (1992).

10. Choudhury, R., R., Yang, X., Vaidya, N., H., Ramanathan, R., "Using Directional Antennas For Medium Access Control In Ad Hoc Networks", ***Proceedings Of The 8th Annual International Conference On Mobile Computing And Networking,*** 23-28, (2002).

11. Yick, J., Mukherjee, B., Ghosal, D., "Wireless Sensor Networks Survey", ***Computer Networks Journal***, 52(12):2292-2330 (2008).

12. Howitt, I., Gutierrez, J.A., "IEEE802.15.4 Low Rate-Wireless Personal Area Network Coexistence Issues", ***Wireless Communications and Networking***, 3 (2003): 1481–1486 (2003).

13. Simon, G., Maroti, M., Ledeczi, A., Balogh, G., Kusy, B., Nadas, A., Pap, G., Sallai, J., Frampton, K, "Sensor Network-Based Counter Sniper System", ***Proceedings of the Second International Conference on Embedded Networked Sensor Systems***, 1-12 (2004).

14. Carman, D.W., Krus, P.S., Matt, B.,J., "Constraints And Approaches For Distributed Sensor Network Security", ***Technical Report 00-010, NAI Labs, Network Associates, Inc., Glenwood, MD*** (2000).

15. Chan, H., Perrig, A., "Security and Privacy in Sensor Networks", ***IEEE Computer Magazine***, 36(10): 103–105 (2003).

16. Perrig, A., Szewczyk, R., Tygar, J., D., Wen, V., Culler, D., E., "Spins: Security Protocols For Sensor Networks", ***Wireless Networking***, 8(5): 521–534 (2002).

17. Liu, D., Ning, P., "Efficient Distribution of Key Chain Commitments For Broadcast Authentication In Distributed Sensor Networks", ***10th Annual Network and Distributed System Security Symposium***, 263–276 (2003).

18. Liu, D., Ning, P.,"Multi level μTESLA: Broadcast Authentication For Distributed Sensor Networks", ***Transaction on Embedded Computing Systems***, 3(4): 800–836 (2004).

19. Ko, Y., Vaidya, N., H, "Location-Aided Routing (LAR) In Mobile Ad Hoc Networks", ***Wireless Networks***, 6(1):321-324 (2000).

20. Karp, B., Kung, H., T., "Greedy Perimeter Stateless Routing for Wireless Sensor Networks", ***MobiCom'00, USA,*** 243-254 (2000).

21. Mauve, M., Widmer, J., Hartenstein, H., "A Survey on Position-Based Routing in Mobile Ad Hoc Networks", ***IEEE Network Magazine***, 15(6): 30-39 (2001).

22. Yan, T., He, T., Stankovic, J., A., "Differentiated Surveillance Service for Sensor Networks", ***First ACM Conference on Embedded Networked Sensor Systems (SenSys '03),*** USA**,** 51-62 (2003).

23. Karlof, C., Wagner, D, "Secure Routing In Wireless Sensor Networks: Attacks And Countermeasures", ***Elsevier's Ad Hoc Network Journal, Special Issue on Sensor Network Applications and Protocols,*** 1(2): 293-315 (2003).

24. Wood, A., D., Stankovic, J., A., "Denial Of Service In Sensor Networks", ***IEEE Computer***, 35(10): 54–62 (2002).

25. Wood, A., D., Stankovic, J., A., Son, S., H., "JAM: A Jammed-Area Mapping Service for Sensor Networks", ***24th IEEE Real-Time Systems Symposium***, 286-297 (2003).

26. Tseng, H., C., Jack Culpepper, B."Sinkhole Intrusion In Mobile Ad Hoc Networks: The Problem And Some Detection Indicators", ***Computer and Security***, 24(7): 561-570 (2005).

27. Newsome, J., Shi, E., Song, D., Perrig, A., "The Sybil Attack In Sensor Networks: Analysis & Defenses", ***Proc. of the Third International Symposium On Information Processing In Sensor Networks, ACM,*** 259 – 268 (2004).

28. Wang, X., Gu, W., Chellappan, S., Xuan, D., Laii., T., H., "Search-Based Physical Attacks In Sensor Networks: Modeling And Defense", ***Technical report, Dept. of Computer Science and Engineering, The Ohio-State University,*** (2005).

29. Wang, X., Gu, W., Schosek, K., Schosek, Chellappan, S., Xuan, D., "Sensor Network Configuration Under Physical Attacks", ***International Journal of Ad Hoc and Ubiquitous Computing***, 4(3): 174-182 (2004).

30. Karakehayov, Z., "Using REWARD to Detect Team Black-Hole Attacks In Wireless Sensor Networks", ***Workshop on Real-World Wireless Sensor Networks REALWSN'05***, 1-11 (2005).

31. Patcha, A., Mishra, A., "Collaborative Security Architecture for Black Hole Attack Prevention in Mobile Ad Hoc Networks", ***Proc. of RAWCON '03***, 75-78 (2003).

32. Ramaswamy, S., Fu, H., Sreekantaradhya, M., Dixon, J., Nygard, K., "Prevention of Cooperative Black Hole Attack in Wireless Ad Hoc Networks", ***Proceedings of the 2003 International Conference on Wireless Networks (ICWN'03),*** USA, 1-11 (2003).

33. Yin, J., Madrıa, S., K., "A Hierarchical Secure Routing Protocol against Black Hole Attacks in Sensor Networks", ***Sensor Networks, Ubiquitous, and Trustworthy Computing, IEEE International Conference***, 1(5): 376 – 383 (2006).

34. Wang, W., Bhargava, B., "Visualization of Wormholes in Sensor Networks", ***Proceedings of the ACM Workshop on Wireless Security***, 51–60 (2004).

35. Hu, Y., C., Perrig, A., Johnson, D., B., "Packet Leashes: A Defense against Wormhole Attacks In Wireless Networks", ***INFOCOM 2003, Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies***, 3:1976 – 1986 (2003).

36. Hsiao, Y., K., Hwang, R.J., "An Efficient Secure Data Dissemination Scheme For Grid Structure Wireless Sensor Networks", ***International Journal of Security and Networks***, 5(1): 26-34, (2010).

37. Yun, J., H., Kim I., H., Lim J., H., Seo, S., W, "WODEM: Wormhole Attack Defense Mechanism in Wireless Sensor Networks", ***Book chapter in Lecture Notes in Computer Science,*** 4412: 200-209 (2007).

38. Song, N., Qian, L., Li, X., "Wormhole Attacks Detection In Wireless Ad Hoc Networks: A Statistical Analysis Approach", ***Proceeding of 19th IEEE International Parallel and Distributed Processing Symposium***, 8-16 (2005).

39. Hu, L., Evans, D., "Using Directional Antennas to Prevent Wormhole Attacks", ***The 11th Annual Network and Distributed System Security Symposium***, 22-32 (2004).

40. Cagalj, M., Capkun, S., Hubaux, J., P., "Wormhole-based Anti-Jamming Techniques in Sensor Networks", ***IEEE Transactions on Mobile Computing***, 6(1): 100-114 (2000).

41. Pirzada, A., McDonald, C., "Circumventing Sinkholes and Wormholes in Wireless Sensor Networks", ***International Workshop on Wireless Ad-hoc Networks,*** 1-9 (2005).

42. Ngai, E., C., H., Liu, J., Lyu, M. R., "On the Intruder Detection for Sinkhole Attack in Wireless Sensor Networks", ***IEEE Conference On Communication,*** 8(1): 3383 – 3389 (2007).

43. P., Agrawal, R., K., Ghosh, S., K., Das, "Cooperative black And Gray Hole Attacks In Mobile Ad Hoc Networks", P**roceedings of the 2th International Conference on Ubiquitous Information Management and Communication (ICUIMC'08)**, USA, 310-314, (2008).

44. Deng, H., Li, W., Agrawal, D., P., "Routing Security in Wireless Ad Hoc Networks", ***IEEE Communications Magazine***, 40(10):70-75 (2002).

45. Zhu, S., Setia, S., Jajodia S., "LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks", ***In Proceedings of 10th ACM Conference on Computer and Communications Security,*** 62-72 (2003).

46. Capkun, S., Buttyan, L., Hubaux J., "Sector: Secure Tracking Of Node Encounters In Multi-Hop Wireless Networks", ***Proceedings of the ACM Workshop on Security of Ad Hoc and Sensor Networks***, 1-11 (2003).

47. Deng H., Zeng Q., Agrawal, D., "SVM-Based Intrusion Detection System For Wireless Ad Hoc Networks", ***Proceedings of the IEEE Vehicular Technology Conference***, 2147-2151 (2003).

48. Dijkstra E., W., "A Note on Two Problems in Connection With Graphs", ***Numerische Mathematik,*** 1: 269-271 (1959).

49. Stafrace, S., K., Antonopoulos, N., "Military Tactics in Agent-Based Sinkhole Attack Detection for Wireless Ad Hoc Networks", ***Computer Communications Journal***, 33(5): 619–638 (2010).

50. Zhu, S., Setia, S., Jajodia, S., "LEAP: efficient security mechanisms for large-scale distributed sensor Networks***", Proceedings of CCS '03,*** 62–72 (2003).

51. Kurkowski, S., Camp, N., Colagrosso, M., "A Visualization and Analysis Tool for NS-2 Wireless Simulations: INSpect", ***Proceedings of the IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS),*** 503-506 (2005).

52. Dokurer, S., "Simulation of black-hole attack in wireless ad-hoc networks", Master's thesis***, Atılım University***, 27-32, (2006).

53. Internet: C., Perkins, "(RFC) Request for Comments – 3561", Category: Experimental, Network, Working Group, http://www.ietf.org/rfc/rfc3561.txt (2003).

54. Meghdadi**,** M.**,** Özdemir, S., Güler, İ., "Security in Wireless Sensor Networks: Problems and Solutions", ***Journal of Informatics Technologies,*** Turkey**,** 1(1): 35-42 (2008).

55. Ceken, C., "An Energy Efficient and Delay Sensitive Centralized MAC Protocol for Wireless Sensor Networks", ***Computer Standards & Interfaces***, 30(2) :20-31 (2008).

56. Meghdadi**,** M.**,** Özdemir, S., Güler, İ., "An Algorithm for Defending Black-Hole Attacks in Wireless Sensor Networks", (Manuscript in Turkish), ***in Proc. of HABTEKUS 08,*** Turkey, 71-76 (2008).

57. Phuong, T., V., Hung, L., X., Lee, Y., K., Lee, S., Lee, H., "TTM: An Efficient Mechanism to Detect Wormhole Attacks in Wireless Ad-hoc Networks", ***4th IEEE Consumer Communications and Networking Conference,*** 593-598 (2007).

58. Phuong, T., V., Canh, N., T., Lee, Y., K., Lee, S., Lee, H., "Transmission Time-Based Mechanism to Detect Wormhole Attacks", **Proceedings of the 2nd IEEE Asia-Pacific Service Computing Conference**, 172-178 (2007).

59. Ganeriwal, S., Srivastava, M., B., "Reputation-based framework for high integrity sensor networks**¨,** ***Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks***, 66–77 (2004).

EKLER

EK-1. Önemli Kablosuz Ağ Simülatörlerin Avantajları ve Dezavantajları

Simülatör	Avantaj	Dezavantaj
AdventNet	• Kullanımı kolay bir simülatördür. • Bir güçlü grafiksel arayüze sahiptir. • Ölçeklenirlik açısından zengin sayılmakt	• Bu simülatör ticari amaçla tasarlanmıştır. • Bu simülatöre yeni bir protokol çok zor eklenmektedir. • Bazı yaygın protokoller (örneğin SNMP) desteklenmemektedirler. • Bir bit tabanı simülatör olmasına rağmen ağlar arasında iletişim desteklenmemektedir
Avrora	• KAA'ları desteklemektedir. • Atmel ve Mica2 düğümleri ile çalışabilmektedir. • Enerji ve (stack) analiz etme araçlarına sahiptir.	• Bu simülatör ticari amaçla tasarlanmıştır. • Internet'te Beta Sürümü bulunmaktadır. • 2005 yılından beri desteklenmemektedir.
Atemu	• Bu simülatör XATBD isminde bir grafiksel arayüze (interface) sahiptir. Bu ara yüz sayesinde kodlar çalıştırabilmektedir, kesme noktası (Break Point) kullanılabilmektedir ve kodlar adım-adım çalıştırabilmektedir. • Bu simülatörde düğümler üzerinde farklı kodlar çalıştırabilmek ihtimali vardır. • Ölçeklenirlik açısından zengin sayılmaktadır.	• Bu simülatör ticari amaçla tasarlanmıştır. • Düğümler sayısı 120'den fazla olduğunda doğru düzgün çalışamaz. • Sadece AVR işlemcilerinin üzerinde çalışmaktadır ama MICA2 düğümlerinin bazı yan birimlerini (örneğin radyoyu) desteklemektedir. • 2004 yılından beri desteklenmemektedir.
CNET	• Kullanıcılar açısından bu simülatörü kullanmak çok kolay.	• Simülatör çok zor kurulmaktadır. • Bu simülatörde gerçek dünya ile iletişim kurulmamaktadır • Kütüphanesinde pek çok fazla protokol bulunmamaktadır.
EmStar	• Simülatör/Emulator sayılmaktadır. • KAA'ları desteklemektedir.	• Gerçek dünya ile iletişim kurulabilmektedir • Kütüphanesinde pek çok fazla protokol bulunmaktadır. • Konfigürasyon dosyasına çok zor düzen vermektedir
GloMoSim	• Bu simülatör KAA'ları desteklemektedir. • GloMoSim geliştirilebilir bir simülatördür. • Ayrıca kütüphanesinde pek cok protokol içermektedir	• 2000 yılından güncelleştirilmemektedir.
JNS	• Bu simülatöre yeni protokoller kolayca eklenebilmektedir.	• Kütüphanesinde pek fazla protokol bulunmamaktadır. • En son elde edilebilen sürümü 1.7 ve 2000 yılından sonra güncelleştirilmemektedir.

EK-1 (Devam)

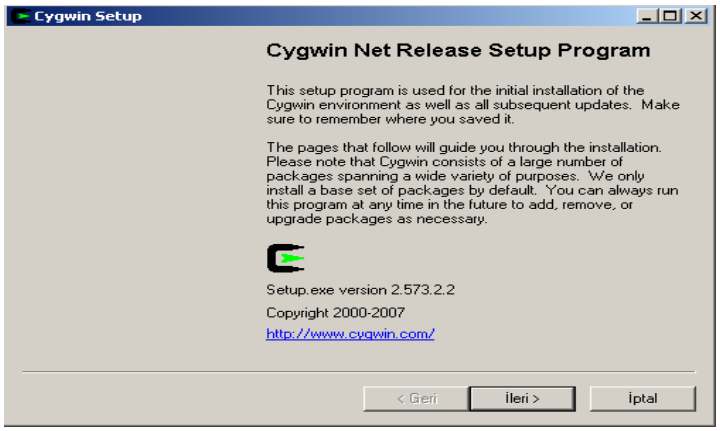
Simülatör	Avantaj	Dezavantaj
J-Sim	• Genişletilebilir bir simülatördür (simülatöre yeni modeller veya yeni protokoller kolayca eklenebilmektedir). • Nesneye yönelik (object oriented) yerine bileşen yönelik (component oriented) mimariye sahiptir. • Gerçek dünya ile iletişim kurulabilmektedir. • KAA'ları desteklemektedir.	• KAA'ları desteklemesine rağmen KAA'lar uygulamalarında iyi performans sağlamamaktadır.
NS-2	• Genişletilebilir bir simülatördür ve bu simülatöre yeni modeller veya yeni protokoller kolayca eklenebilmektedir. • Açık kaynak bir simülatördür. • Grafiksek araca sahiptir. • Eski bir simülatör olduktan dolayı pek çok bilimsel çalışmalar bunun üzerinde yapılmıştır. Bu nedenle araştırmanın sonuçlarını karşılaştırmak daha kolay gelmektedir.	• Bazı dokümanlar güncelleştirilmemiştir. • NS-2 simülatöründe kablosuz kanal çok basit tasarlanmış ve kolayca değiştirmek kabiliyetine sahip değildir. • Ölçeklenirlik açısından zengin sayılmamaktadır. Normalde düğüm sayısı 400 civarında olduğunda iyi sonuç vermektedir. • İki programlama dili kullanma nedeniyle, araştırmacılar iki programlama dili öğrenme zorundadırlar. • KAA'ların tüm uygulamalarını desteklememektedir.
OMNeT++	• Güçlü bir grafiksek ara-yüze sahiptir. • Genişletilebilir bir simülatördür. • Tüm kütüphanedeki protokoller değiştirebilmektedir. • Hız açısından NS-2'dan hızlı çalışmaktadır.	• KAA'ları desteklemesine rağmen bazı modeller desteklenmemektedir. Örneğin enerji verimliliği (energy efficiency) modeli desteklenmemektedir. • KAA'lara uygun tüm protokoller tasarlanmamıştır. • Ölçeklenirlik açısından zengin sayılmaktadır. • İyi bir dokümana sahip değildir.
OpNet	• Grafiksel editörü yardımıyla çeşitli ağ modelleri desteklenmektedir	• Bu simülatör ticari amaçla tasarlanmıştır. • Kütüphanesinde bulunan protokoller sayısı çok fazla değilmiştir.
SENS	• KAA'ları desteklemektedir. • Genişletilebilir bir simülatördür. • Gerçek algılayıcılar ile iletişim kurabilmek için imkan sağlamaktadır.	• Kullanıcıların istekleri çok kolay izin verilmemektedir. • Sadece algılayıcılarda ses algılama desteklenmektedir.
SENSE	• Genişletilebilir bir simülatördür. • Ölçeklenirlik açısından zengin sayılmaktadır.	• Simülatörün elemanlarının arasında iletişim iyi bir performansa sahip değildir.

EK-1 (Devam)

Simülatör	Avantaj	Dezavantaj
SensorSim	• KAA'ları desteklemektedir.	• İyi bir dokümana sahip değildir. • Ticari amaçla tasarlanmıştır.
Sidh	• Bu simülatörde modüllerin değiştirilmesi çok kolaydır. • Ölçeklenirlik açısından zengin sayılmamaktadır.	• Çok fazla kullanılmamaktadır. • Yani bir simülatör olduğundan dolayı iyi bir dokümana sahip değildir. • Kütüphanesinde desteklenen protokol sayısı pek fazla değildir
Tossim	• KAA'ları için tasarlanmıştır. • Test ve analiz etme aracına sahiptir. • Simülatör/Emulator sayılmaktadır. • Grafiksel TinyViz aracı bu simülatörle kullanılabilir.	• Bit düzeyinde bir simülatör sayılmaktadır. • Ölçeklenirlik açısından zengin sayılmamaktadır. Düğümler sayısı 100'den fazla olduğu halde iyi performansa sahip değildir.

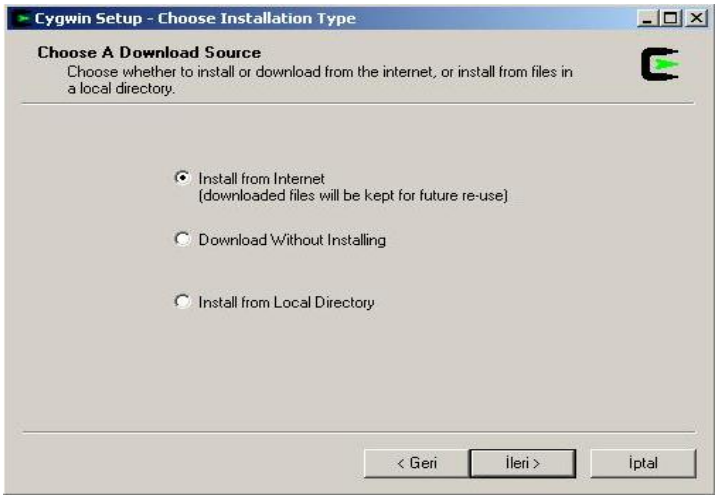
EK-2. NS-2 Simülatörünün Kurulması

NS-2 simülatörünü Windows XP işletim sistemi üzerinde kurabilmek için önce Cygwin.exe programını http://www.cygwin.com/ adresinden indirilmesi ve daha sonra bu programı bilgisayar üzerinde çalıştırılması gerekmektedir. Bu programın kurulma sayfası Şekil EK2.1. de gösterilmiştir.



Şekil EK2.1. Cygwin programının kurulma sayfası.

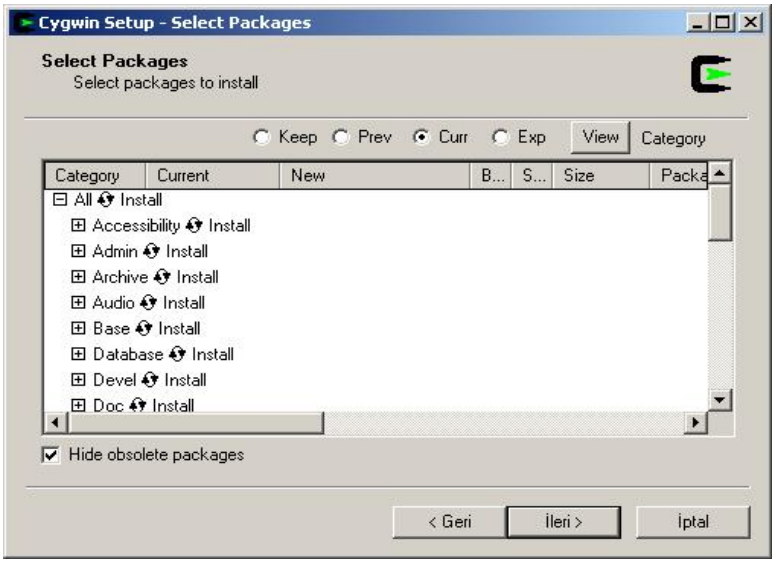
Bu program ilk kurulduğunda gerekli dosyaların internet üzerinden yüklenmesi gerekir. Bu nedenle ileri tuşuna bastıktan sonra açılan pencerede birinci seçeneğin seçilmesi gerekir (Şekil EK2.2.).



Şekil EK2.2. Cygwin programı ilk kurulduğunda İnternet üzerinden gerekli dosyaların indirilmesi gerekir.

EK-2 (Devam)

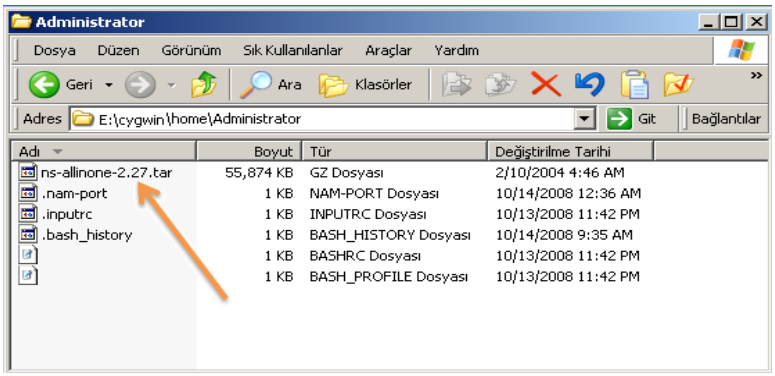
Bu aşamadan sonra Cygwin programının kurulduğu yer belirtilmelidir. Bu program kurulduğunda gerekli paketlerin seçilmesi gerekir. Önerilen bu paketlerin hepsini seçmek ve bu paketleri eksiksiz kurmaktır (Şekil EK2.3.).



Şekil EK2.3. Cygwin programı kurulduğunda paketlerin hepsini seçilmesi önerilmektedir.

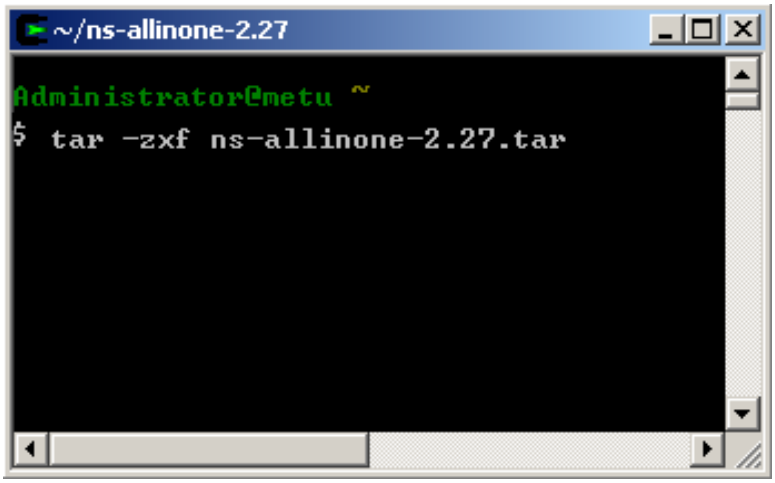
Cygwin programı Linux ortamını Windows işletim sistemi üzerinde taklit etmektedir. Cygwin programı kurdurulduktan sonra NS-2 simülatörü kurulabilir. NS-2 programı 'ns-allinone-[surum].tar' paketi isminde http://www.isi.edu/nsnam/dist/ adresinde bulunmaktadır. Bu paketi indirdikten sonra Cygwin programının kurdurulduğu yerde '\home\Administratr' klasörüne kopyalanması gerekir. Örneğin eğer Cygwin programı 'E:\cygwin' adresinde kurulmuşsa, 'ns-allinone-2.27.tar' paketini 'E:\cygwin\home\Administrator' adresine kopyalanması gerekir (Şekil EK2.4). Bu çalışmada NS-2 sürüm 2.27 kurulmuştur.

EK-2 (Devam)



Şekil EK2.4. NS-2 programını kurma yeri.

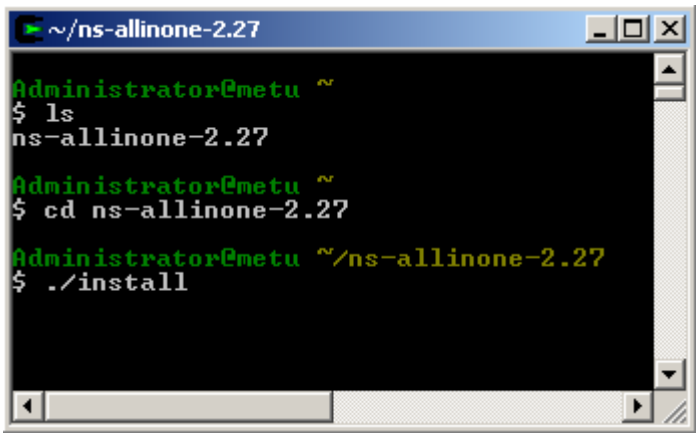
Kopyalama işleminden sonra Cygwin programını çalıştırıp ve **'tar'** komutu ile sıkıştırılmış dosya açılabilir. (Şekil EK2.5.).



Şekil EK2.5 'tar' komutunun kullanımı

Bu komutu çalıştırdıktan sonra bir klasör 'ns-allınone-2.27' isminde oluşturulur. Tüm NS-2 simülatörünün dosyaları bu klasörde bulunmaktadır. NS-2 simülatörünü kurabilmek için bu dosyanın içinde 'install' komutunun çalıştırılması yeterlidir. (Şekil EK2.6.).

EK-2 (Devam)



Şekil EK2.6. 'install' komutu

'*install'* komutu bittikten sonra NS-2 similatörü bu klasörün içinde kurulmuş bulunmaktadır. Cygwin programı Unix işletim sistemini Windows işletim sistemi üzerinde taklit etmektedir. Cygwin programını çalıştırmak için '*cygwin'* klasöründen '*cygwin.bat'* dosyayı çalıştırmak yeterlidir. Bu aşamadan sonra Unix komutlarını kullanarak NS-2 simülatöründeki _.*TCL* ve _.*CC* dosyalarını değiştirip yeni dosyalar yaparak istenen protokoller çalıştırılabilir.

EK-3. AODV Protokolünde Değiştirilmiş Değişkenler ve Parametreler

(AODV)	(newAODV)	(new)
__aodv_packet_h__	"Agent/newAODV"	newBroadcastID
AODV_MAX_ERRORS	<newaodv/newaodv.h>	newBroadcastTimer
AODVTYPE_HELLO	newAODV	newHelloTimer
AODVTYPE_RREQ	newAODV()	neweighborTimer
AODVTYPE_RREP	newAODV::	newLocalRepairTimer
AODVTYPE_RERR	newAODVHeaderClass	newRouteCacheTimer
AODVTYPE_RREP_ACK	newAODVclass	
hdr_aodv	newAODV_LINK_LAYER_DETECTION	
hdr_aodv_request	newAODV_LOCAL_REPAIR	
hdr_aodv_reply	newAODV_Neighbor	
hdr_aodv_error	newaodv_rt_entry	
hdr_aodv_rrep_ack	newaodv_rt_failed_callback	
HDR_AODV(p)	class_rtProtonewAODV	
HDR_AODV_REQUEST()	class_rtProtonewAODV_hdr	
HDR_AODV_REPLY()	PacketHeader/newAODV	
HDR_AODV_ERROR()	recvnewAODV	
HDR_AODV_RREP		
hdr_all_aodv		
<aodv/aodv_packet.h>		
PT_AODV		
AODVTYPE_RREP		
AODVTYPE_RREQ		
AODVTYPE_RERR		
AODVTYPE_HELLO		

EK-4. Örnek "Senaryo.Tcl" Dosyası

```
# Seçeneklerin tanımlaması

set val(chan)           Channel/WirelessChannel  ;# Kanal tipi
set val(prop)           Propagation/TwoRayGround ;# Radyo-propagation
                                                 ;# modeli
set val(netif)          Phy/WirelessPhy          ;# Şebeke arayüz tipi
set val(mac)            Mac/802_11               ;# MAC tipi
set val(ifq)            Queue/DropTail           ;# Arayüz kuyruk tipi
set val(ll)             LL                       ;# bağlantı katmanı tipi
set val(ant)            Antenna/OmniAntenna      ;# anten modeli
set val(ifqlen)         50                       ;# kuyrukta en fazla
                                                 ;# paket sayısı
set val(nnaodv)         8                        ;# normal Düğümlerin
                                                 ;# sayısı
set val(nn)             1                        ;# kötücül Düğümlerin
                                                 ;# sayısı
set val(rp1)            AODV                     ;# yönlendirme
                                                 ;# protokolü
set val(rp2)            newAODV                  ;# yeni yönlendirme
                                                 ;# protokolü
set val(x)              600                      ;# topografının X boyutu
set val(y)              600                      ;# topografının Y boyutu
set val(stop)         150                        ;# Simülasyonun son zamanı

set val(cstop) 145                               ;# bağlantıların süresi

#   ./setdesti   komutunu   kullanmak   için   yaratılan   bağlantı   desenini   eklemek,
parametreler, aşağıda gösterdi

# ./setdest -n 9 -p 1.0 -M 1.0 -t 150 -x 600 -y 600 > senaryoAODV-n9-
t500-x750-y750

set val(cp) "scenarios/sce-n9-t150-x600-y600"    ;#Bağlantıların modeli

set val(cc) "scenarios/cbr"                       ;#Trafik Seçenekleri

# Genel değişkenleri sıfırlama

set ns_ [new Simulatsor]
$ns_ use-newtrace
set tracefd [open out.tr w]
$ns_ trace-all $tracefd
set namtrace [open out.nam w]
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)

# Topografya nesnesiyi kur

set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)

# Düğümlerin yapısını oluşturmak
```

EK-3 (Devam)

```
$ns_ node-config -adhocRouting $val(rp) \
   -llType $val(ll)                    \ ;# ll     = bağlantı katmanı
   -macType $val(mac)                  \ ;# mac    = MAC katmanı
   -ifqType $val(ifq)                  \ ;# ifq    = Arayüz kuyruk tipi
   -ifqLen $val(ifqlen)                \ ;# ifqlen = Kuyruktaki paket sayısı
   -antType $val(ant)                  \ ;# ant    = Anten Modeli
   -propType $val(prop)                \ ;# prop   = Radyo-propagation
   -phyType $val(netif)                \ ;# netif  = Şebeke arayüz tipi
   -topoInstance $topo                 \ ;# topo   = Topografya
   -agentTrace ON                      \ ;# Agent Trace etkınleştir
   -routerTrace ON                     \ ;# Router Trace etkınleştir
   -macTrace ON                        \ ;# MAC izlemeyi etkınleştir
   -movementTrace ON                   \ ;# Harekat izlemeyi etkınleştir
   -channel $chan_1_                     ;# kanal türü = kanal 1



# Normal düğümlerin oluşturulması...

$ns_ node-config -adhocRouting AODV
for {set i 0} {$i < $val(nnaodv)} {incr i} {
   set node_($i) [$ns_ node]
   $node_($i) random-motion 0 ;# Rastgele Hareketleri Etkisizleştirmek
}


# Kötücül düğümlerin oluşturulması...

$ns_ node-config -adhocRouting newAODV
for {set i $val(nnnewaodv)} {$i < $val(nn)} {incr i} {
   set node_($i) [$ns_ node]
   $node_($i) random-motion 0 ;# Rastgele Hareketleri Etkisizleştirmek
   $ns_ at 0.01 "$node_($i) label \"kötücül Düğüm\""
}


set god_ [God instance]
source $val(cp)

source $val(cc)

# düğümlerin ilk konumunu tanımlama
for {set i 0} {$i < $val(nn) } {incr i} {
   $ns_ initial_node_pos $node_($i) 30
}

for {set i 0} {$i < 9 } {incr i} {
   $ns_ at $val(cstop) "$cbr_($i) stop"
}

# simülasyonun tamamlama zamanını belirtme

for {set i 0} {$i < $val(nn) } {incr i} {
   $ns_ at $val(stop).000000001 "$node_($i) reset";
}
```

EK-3 (Devam)

```
# NAM programını bitirme zamanı

$ns_ at $val(stop) "finish"
$ns_ at $val(stop).0 "$ns_ trace-annotate \"Simülasyon bitti...\""
$ns_ at $val(stop).00000001 "puts \"NS-2'den Çıkış...\" ; $ns_ halt"


proc finish {} {
   global ns_ tracefd namtrace
   $ns_ flush-trace
   close $tracefd
   close $namtrace
   exec nam senaryo.nam &
exit 0
}
puts "Starting Simulation..."
$ns_ run
```

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı : MEGHDADI,Majid

Uyruğu : İRAN.

Doğum tarihi ve yeri : 25.05.1965 Zanjan

Medeni hali : Evli

Telefon : 0098 (241) 515 2613

Faks : 0098 (241) 425 9202

e-mail : majid_m1344@yahoo.com , meghdadi@znu.ac.ir

Eğitim

Derece	Eğitim Birimi	Mezuniyet tarihi
Yüksek lisans	Şarif Teknik Üniversitesi/ Bilgisayar Mühendisliği Bölümü	1995
Lisans	Tahran Üniversitesi / Matematik Bölümü	1990
Lise	Amirkabir Lisesi	1983

İş Deneyimi

Yıl	Yer	Görev
1995-…	Zanjan Üniversitesi	Araştırma Görevlisi

Yabancı Dil

İngilizce, Arapça, Türkçe

Yayınlar

1. Salahi, A., Meghdadi, M., "Design and Implementation of TUP in System Signaling no.7", (Manuscript in Farsi), Third Iranian Conference on Electrical Engineering – Iran University of Science& Technology, 1995.

2. Meghdadi, M., Sharifi, M., " Study of Systems Described by Coupled Differential or Integra Differential Equation via Finite Element Method", Second Join on Applied Mathematics, Zanjan University & Baku State University,1995.

3. Meghdadi, M. , "The Analysis of Computer Problems in 2000" (Manuscript in Farsi), Omide-Zanjan Newspaper, Zanjan, Iran, 1999.

4. Meghdadi, M., Jalilzade, S., "The Use of Biometric Identification Methods in Fingerprint Recognition systems", International Symposium on Health Informatics and Bioinformatics (HIBIT`05) Belek, Antalya, Turkey 2005.

5. Meghdadi, M., Jalilzade, S., "Validity and Acceptability of Results in Fingerprint Scanners" 7`th WSEAS International Conference on Mathematical Methods and Computational Techniques in Electrical Engineering (MMACTEE`05) – Sofia, Bulgaria 2005.

6. Bayani, A. M., Meghdadi, M., " Nesimi`de Yerel İmajlar ve Tanık Sözcüler", (Manuscript in Turkish), 1.Uluslararası SEYYİT NESİMİ Semposyumu Bildirisi, Ankara, Turkey, 2005.

7. Meghdadi, M., Jalilzade, S., "Fingerprint Scanner's Ability to Distinguish Between Live Fingerprint and Artificial Clones", WSEAS Transaction on Signal Processing Journal, Issue 1, Volume 1, October 2005 - Sofia, Bulgaria, 2005.

8. Meghdadi, M., Guler, I., "A New Approach for Off-Line HSV Using the Optimal DTW Algorithm", AIMB38, Zanjan University, Zanjan, Iran , 2007.

9. Meghdadi, M**.,** Ozdemir, S., Guler, I., "Security in Wireless Sensor Networks: Problems and Solutions" (Manuscript in Turkish), Journal of Information Technologies, Gazi University, Ankara, Turkey, 1(1): 35-42, 2008.

10. Guler, I., Meghdadi, M., "A Different Approach to Off-Line Handwritten Signature Verification Using the Optimal Dynamic Time Warping Algorithm", Digital Signal Processing Journal, 18(6): 940-950, 2008.

11. Meghdadi, M., Ozdemir, S., Guler, I., “Black Hole Attacks Detection in Wireless Sensor Networks”, (Manuscript in Turkish), HABTEKUS 08, :71-76, , Istanbul, Turkey, 2008.

12. Meghdadi, M.**,** Ozdemir, S., Guler, I., "A Survey of Wormhole-Based Attacks and Their Countermeasures in Wireless Sensor Networks” Accepted to appear in IETE Technical Review Journal, 2010.

Hobiler

Tenis, Basketbol ve Yürümek.