T.C.
İSKENDERUN TEKNİK ÜNİVERSİTESİ
MÜHENDİSLİK VE FEN BİLİMLERİ ENSTİTÜSÜ

ENDÜSTRİYEL NESNELERİN İNTERNETİNDE HIZLI VE GÜVENLİ VERİ İLETİMİ

Mustafa KARA

ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

YÜKSEK LİSANS TEZİ

HATAY

OCAK-2018

T.C.
İSKENDERUN TEKNİK ÜNİVERSİTESİ
MÜHENDİSLİK VE FEN BİLİMLERİ ENSTİTÜSÜ

ENDÜSTRİYEL NESNELERİN İNTERNETİNDE HIZLI VE GÜVENLİ VERİ İLETİMİ

Mustafa KARA

ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

YÜKSEK LİSANS TEZİ

HATAY

OCAK-2018

T.C.
İSKENDERUN TEKNİK ÜNİVERSİTESİ
MÜHENDİSLİK VE FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

Tezin Adı: Endüstriyel Nesnelerin İnternetinde Hızlı ve Güvenli Veri İletimi

Öğrencinin, Adı Soyadı: Mustafa KARA

Tez Savunma Tarihi: 02.01.2018

Kod No: 76
Enstitü Onayı:

Doç. Dr. Mustafa DEMİRCİ
Enstitü Müdürü

Bu tezin Yüksek Lisans tezi olarak gerekli şartları sağladığını onaylarım

Doç.Dr. Emin ÜNAL
Enstitü ABD Başkanı

Bu tez tarafımca (tarafımızca) okunmuş, kapsamı ve niteliği açısından bir Yüksek Lisans tezi olarak kabul edilmiştir.

(Unvanı, Adı ve SOYADI)
İkinci Tez Danışmanı (varsa)

Yrd. Doç. Dr. Murat FURAT
Tez Danışmanı

Bu tez tarafımızca okunmuş, kapsam ve niteliği açısından bir Yüksek Lisans tezi olarak oy birliği/oy çokluğu ile kabul edilmiştir.

Jüri Üyeleri (Ünvanı, ADI ve SOYADI):	İmzası
Yrd. Doç. Dr. Murat FURAT	.....
Doç. Dr. Serdar YILDIRIM	.....
Yrd. Doç. Dr. Yaşar DAŞDEMİR	.....

Not : Bu tezde kullanılan özgün ve başka kaynaktan yapılan bildirişlerin, çizelge, şekil ve fotoğrafların kaynak gösterilmeden kullanımı, 5846 sayılı Fikir ve Sanat Eserleri Kanunundaki hükümlere tabidir.

02.01.2018

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını ve tez üzerinde Yükseköğretim Kurulu tarafından hiçbir değişiklik yapılamayacağı için tezin bilgisayar ekranında görüntülendiğinde asıl nüsha ile aynı olması sorumluluğunun tarafıma ait olduğunu beyan ederim.

Mustafa KARA

ÖZET

ENDÜSTRİYEL NESNELERİN İNTERNETİNDE HIZLI VE GÜVENLİ VERİ İLETİMİ

Endüstride son yıllara kadar yoğun olarak tercih edilen güvenlik tedbiri, sistemin dış dünya ile olan bağlantısının kesilerek internet üzerinden gelecek saldırıları önlemek şeklindeydi. Ancak bu teknolojiden uzak tedbir, sistemdeki sorunların hızlı bir şekilde çözülmesini engellemiş ve endüstriyel sahadaki cihazlara yetkili kişilerin anlık müdahalesini önlemiştir. Her geçen gün biraz daha gelişen ve farklı alanlara mimari yenilikler oluşturarak adapte olan Nesnelerin İnterneti teknolojisi, endüstriyel alanlarda da cihazlar üzerinden kulanılmaya başlanmıştır. Bu endüstriyel cihazlar, verilerin iletimi aşamasında teknolojinin kolaylıklarını kullanmasının yanı sıra, ağın yapısından kaynaklanan bazı güvenlik sorunlarından da etkilenmektedir.

Bu çalışmada, endüstriyel sahadaki cihazların sensörlerinden gelen anlık veri paketlerinde ortadaki adam saldırısı ile yerel alan ağına sızan saldırgan tarafından değişiklik yapılıp yapılmadığının tespitini sağlayan bir yazılım geliştirilmiştir. Bu saldırılar için yapılandırılması gereken güvenlik cihazları ve yazılımları, geliştirilen saldırı tespit yöntemi ile çok daha güvenli bir sistem haline getirilmiştir. Geliştirilen yöntem istemci-sunucu mimarisinde çalışmaktadır. Önerilen yöntem ile sensörlerden gelen verilerdeki saldırı tespitinin anlaşılması kolaylaştırılmıştır ve bu tespit anlık seviyelerde yapılabilmektedir.

Önerilen yönteminin test edilmesi amacıyla en sık kullanılan MD5, SHA-1 ve SHA-256 doğrulama yöntemleri ile hız performansı açısından karşılaştırma yapılmıştır. Bu algoritmalarla birlikte önerilen yöntem üzerinden 10, 100 ve 500 sensör verisi için doğrulama değeri üretilerek hız performansları ölçülmüştür. Ayrıca üretilen sensör verisi paketlerine ortadaki adam saldırısı yapılarak önerilen yöntem üzerinden bozulmuş veri tespiti yapılmıştır. Sensör sayısı artırılarak her paketin aslına benzerliği ölçülmüş ve böylece saldırılara karşı tespit yapabilmesi test edilmiştir.

Bu yöntem açık kaynak kodlu Java programlama dili ile geliştirilmiştir. Herhangi bir bilgisayar üzerine kurularak merkezi bir noktadan onlarca sensör veri paketinden bozulmuş veri paketinin saldırı tespitini yapabilmektedir. Geliştirilen yöntem ile sensör verisinin saldırı tespitinde anlık seviyede başarılı sonuçlar elde edilmiştir.

2018, 60 sayfa

Anahtar kelimeler: Güvenli İletim, Saldırı Tespit Sistemi, Ağ Mimarisi, Nesnelerin İnterneti, Sensör Verisi

ABSTRACT

FAST AND SECURE DATA TRANSMISSION IN INDUSTRIAL INTERNET OF THINGS

The security precaution, which has been intensively preferred in recent years, was to prevent probable attacks on the internet by cutting off the connection of the system with the outside world. However, irrelevant measures from technology have prevented the rapid resolution of system problems and the obstructed instant intervention of the authorized persons to the devices on the industrial field. The technology of Internet of Things, which is gradually evolving and adapting to the different areas by creating architectural innovations, has begun to be used also in industrial fields through the devices. These industrial devices besides using the convenience of technology during transmission of datas, they also affected by some security issues arising from the network structure.

In this study, we have developed a software that detects whether there are any changes in the data packets after the intrusion into the local area network with the man-in-the-middle (MITM) attack in the instantaneous data packets coming from the sensors of the industrial field devices. The security devices and software that need to be configured for these attacks have been made the much more secure system with the intrusion detection method we have developed. The developed method uses the client-server architecture. Along with the proposed method, the detection of the attack in the data from the sensors is facilitated. And this detection can be made at instantaneous levels.

In order to test the proposed method, the most frequently used verification methods such as MD5, SHA-1 and SHA-256 were compared in terms of speed performance. The speed performance was measured by generating verification value for 10, 100 and 500 sensor data through the proposed method with these algorithms. In addition, after performing MITM attack to the sensor data packets, the corrupted data was detected by using the proposed method. By increasing the number of sensors, it was measured whether each package is the same. And in this way, the proposed method has been tested for trying the attack detection.

This method has been developed with open source Java programming language. It can be installed on any computer to detect corrupted data packet in the dozens of sensor data packages through a central point. With the developed method, successful results have been obtained for attack detection of the sensor data in the instant levels.

2018, 60 pages

Key words: Secure Transmission, Intrusion Detection System, Network Architecture, Internet of Things, Sensor Data

TEŞEKKÜR

Yüksek Lisans tez çalışmamın her aşamasında yardımlarını esirgemeyen, değerli fikir ve katkıları doğrultusunda çalışmama yön veren danışmanım Yrd. Doç. Dr. Murat FURAT'a çok teşekkür ederim. Yine bu süreçte bana hep destek olan Doç. Dr. Yakup Hameş'e ve ayrıca değerli çalışma arkadaşım Öğr. Gör. Alper Kahriman ile her zaman yanımda olan aileme çok teşekkür ederim.

İÇİNDEKİLER

ÇİZELGELER DİZİNİ

ŞEKİLLER DİZİNİ

SİMGELER VE KISALTMALAR DİZİNİ

SİMGELER

q	: Başlangıç anahtarı değeri
IX_{key}	: İstemci doğrulama anahtarı
SX_{key}	: Sunucu doğrulama anahtarı
log	: Logaritma (10 tabanında)

KISALTMALAR

DIDS	: Dağıtık Saldırı Tespiti Sistemleri
DMZ	: Sivilleştirilmiş Alan (Demilitarized Zone)
EPL	: Eclipse Public Licence
HIDS	: Bilgisayar Tabanlı Saldırı Tespit Sistemi
IDE	: Integrated Development Environment (Tümleşik Geliştirme)
IDS	: Saldırı Tespit Sistemi (Intrusion Detection System)
IIOT	: Endüstriyel Nesnelerin İnterneti
IoT	: Nesnelerin İnterneti
MAC	: Ortak Erişim Kontrolü (Media Access Control)
MD5	: Message-Digest Algorithm 5
MITM	: Ortadaki Adam Saldırısı
NIDS	: Ağ Saldırı Tespit Sistemi
OSI	: Açık Kaynak İnisiyatifi (Open Systems Interconnection)
PLC	: Programlanabilir Mantıksal Kontrolcü
SCADA	: Denetleyici Kontrol ve Veri Toplama
SHA-1	: Secure Hash Algorithm 1
SHA-256	: Secure Hash Algorithm 2 (256 bits)
TCP	: İletişim Kontrol Protokolü (Transmission Control Protocol)

1. GİRİŞ

Son yıllarda gelişen teknoloji, sensörler ile cihazların bir ağ üzerinde anlık haberleşerek uyum içinde çalışmasını daha da fazla mümkün kılmıştır. Bunlar arasında en önemli teknolojilerden biri olan Nesnelerin İnterneti (Internet of Things, IoT) cihazların anlık olarak ağ üzerinden iletişim kurmasına olanak sağlayan ve tarım, enerji, güvenlik, ulaşım, sağlık gibi alanlarda etkili olarak kullanılan bir bağlantı teknolojisidir (Madakam ve ark., 2015). Birbiriyle bağlantılı nesnelerin bir ağ içerisinde uyumlu bir şekilde çalışması nesnelerin interneti kavramının endüstriyel alanda da yer bulmasına yol açmıştır. IoT teknolojisi ikaz sistemleri, programlanabilir mantıksal kontrolcüler (Programmable Logic Controller, PLC) ve diğer endüstriyel cihazlar üzerinden insanların gözlemleyerek anlamaya ve ölçmeye çalıştığı çevremizdeki fiziksel ortam (nem, basınç, sıcaklık vb.) değişiklerini anlık olarak algılayan cihazlar olan sensörler yardımıyla üretim sahasındaki her cihazın anlık haberleşmesini sağlar ve üretilen verilerin analiz edilmesini amaçlar. Endüstriyel saha cihazlarının IoT teknolojisi ile iletişimi Endüstriyel Nesnelerin İnterneti (Industrial Internet of Things, IIoT) kavramı ortaya çıkarmıştır (Madakam ve ark., 2015; Clerck, 2017; Gubbia ve ark., 2013).

Endüstriyel saha içerisindeki anlık veri izleme kontrol merkezleri, daha fazla parametrenin daha yakından izlenebilmesi ve sonuçta süreçlerin ve tehlikeli durumların daha karmaşık kontrolünü mümkün kılmak için IIoT teknolojisini benimser. Bu açıdan bakıldığında, IoT ve IIoT arasındaki fark teknolojinin uygulanması esnasında ortaya çıkmaktadır. IIoT, tesiste daha hassas sistemler, daha keskin sensörler ve tedarik zinciri tarafında daha fazla konum bilen teknolojiler kullanır. IIoT içinde yapılan kontroller IoT ile karşılaştırıldığında çok daha gelişmiş, yetenekli ve daha analitiktir. IIoT teknolojisi, endüstriyel alanda kalite yönetimi ve dokümantasyon için ayrıntılı veriler yakalarken sensör değerlerini izlemek ve kontrol etmek için sıklıkla kullanılır. IIoT teknolojisi içerisinde endüstriyel sahada birbirleriyle etkileşim halindeki sensörlerin bilgi transferini sağlayan bir sistem kurulur. Böylece sensörler ile en uç noktalara kadar ulaşılan bir anlık iletim ağı oluşturulur (Tayeb ve ark., 2017; Katsikeas ve ark., 2017).

Gelişen IoT teknolojisine paralel olarak üretilen IoT uyumlu akıllı sensörler mevcut bir ağ üzerinden veri iletimi yeteneğine sahiptirler. IoT sensörler adı verilen bu akıllı sensörler, nesnelerin interneti teknolojisinin yalnızca belli bir bölümünü kapmasına

rağmen onlar olmadan IoT teknolojisini kullanmak mümkün değildir. Bu akıllı anlık iletim cihazlarının amacı endüstriyel sahada iletim teknolojisine analitik çalışma mantığı kazandırmaktadır. IoT sensörler, ağ mimarisi içerisinde eyleyici (Örn: motor, pompa, ısıtıcı, v.b.) ve kontrol sistemleri ile birlikte endüstriyel sahayı izleme ve müdahale etmeyi anlık seviyeye düşürerek büyük bir avantaj sağlarlar. Endüstriyel sahada IoT sensörler ile dijital kontrolcüler beraber kullanılarak anlık süreç kontrollerinin otomatik yapılmasına olanak verirler. Bu yüzden, aktarılan verinin ölçülmesi, iletilmesi ve anlık değerlendirilmesi birçok üretim süreci için hayati önem taşır.

Temel altyapılar, akıllı şebekeler ve çeşitli endüstriyel alanlar stratejik hizmetler vermektedir. Stratejik hizmetler değerli bilgiler üretir ve sürekli büyüyen bir yapıya sahiptir. Sistemin gün geçtikçe büyümesi ve iletişim ağındaki trafiğin artması güvenlik sorunlarını da beraberinde getirmektedir. IoT cihazlarının endüstriyel sistemlerin ağlarına katılması sonucu cihazlar arası iletişim anlık mertebesine düşerken, kurulan bu iletişim ağı ile beraber güvenli iletişim üzerine tehdit riskleri de artmaktadır (Bonomi ve ark., 2012; Yun ve Yuxin, 2010; Tellez ve ark., 2016).

Bilgi güvenliği, bilginin bir kavram olarak zarar görmemesi, uygun teknolojinin uygun amaçla ve gerekli ihtiyaç halinde kullanılarak bilgiye her türlü sistemde saldırgan kişiler tarafından erişilmesini engellemek olarak açıklanabilir (Canbek ve Sağıroğlu, 2006). Güvenli iletişim kavramı elektronik ortamdaki bilginin korunması ve doğru bilginin, doğru zamanda, doğru kişiye ulaştırılmasıyla ilgilenir. Verilerin güvenliği insan faktörü, doğal felaketler ve kötü amaçlı yazılımlar gibi sebeplerle her zaman risk altındadır. Her tür cihazın birbiriyle iletişimde olduğu bir ağ ortamında, sürekli gelişen, daha karmaşık ve karışık hale gelen kötü amaçlı yazılımlar sistemlerin veri güvenliğini daha fazla tehdit etmekte; bunların sonucunda daha esnek, duyarlı ve bütünleşik bir güvenlik çemberine ihtiyaç duyulmaktadır.

Güvenli iletişime en çok ihtiyaç duyan alanlardan biri olan endüstriyel sistemler için birçok bilim adamı ve mühendis tarafından hızla geliştirilen protokoller ve algoritmalar ile çeşitli sistem mimarileri kullanılmaktadır. Önerilen güvenlik mimarilerinde gizlilik, bütünlük, süreklilik, kimlik denetimi ve izlenebilirlik hedeflerine ulaşılması ortak amaçlar arasında yer almaktadır (AbdAllah ve ark., 2015).

Güvenlik mimarisi içindeki iletişimde kullanılan şifreleme protokolleri iletişimde gizliliği amaçlar. Veri içeriğini gizler. MD5 (Message-Digest algorithm 5), SHA-1

(Secure Hash Algorithm 1) ve SHA-256 (Secure Hash Algorithm 2, 256 bits) benzeri şifreleme yöntemleri veri içeriğini gizlemek amaçlıdır (Daemen ve Rijmen, 2010; Kim ve ark., 2016). Ancak veri içeriğinden ziyade veri bütünlüğünü önemseyen iki tip yöntem vardır. Bunlar sırasıyla hata tespiti ve hata düzeltme yöntemleridir. Hata düzelten kodlar olası hatayı bulur ve düzeltir. Veriler aktarılırken bozulan verinin tekrar aktarılması talep edilir. Diğer yöntem olan hata tespiti aktarım hızı yüksek olan ve anlık olarak sensör verileri ile gözlem yapılan endüstriyel saha benzeri sistemlerde kullanılmaktadır. Veri kontrolü gönderilen veri paketinin içerisine eklenen bir doğrulama anahtarı ile hata tespiti yapmaktadır. Hata tespiti yöntemleri sadece verinin düzgün iletilip iletilmediğini denetlemek için kullanılır (Huang ve Cohen, 1988; Cohen, 1987).

Anlık değerlendirmedeki ağ topolojisi endüstriyel sahanın yönetimini kolaylaştırıp verimini arttırsa da aktarılan bilgiyi saldırıya açık hale getirir. Anlık izleme süreci, sahada akıllı denetleyiciler, iletim ağı ve saha cihazlarını bir araya getirerek ortaya sürekli bir bilgi akışı çıkarır. Gerçekleşen bu süreç yüksek hassasiyet ile korunmalıdır. Çünkü bilginin aktarımı dijital olarak ağ üzerinden iletilirken, IoT sensör verilerine herhangi bir şekilde saldırı olabilir. Ortadaki adam saldırısı (Man-in-the-middle attack, MITM) benzeri saldırılar veri iletimi esnasında geçen hedef ile ağ unsurları (sunucu, yönlendirici ya da anahtar cihazı) arasında geçen trafiği dinlemek veya değiştirmek olarak tanımlanan saldırıdır. OSI (Open Systems Interconnection) modeli içinde Veri İletim Katmanı yani 2. Katman içerisinde gerçekleştiği için ağ güvenliği konusunda koruma önlemi en az alınan saldırı tipidir.

Otomasyon sistemlerinde saldırılar genellikle iletişim kanalı üzerinden yapılır. Endüstride yoğun olarak tercih edilen güvenlik tedbiri, sistemin dış dünya ile olan bağlantısının kesilerek internet üzerinden gelecek saldırıları önlemek niteliğindedir. Ancak alınan bu tedbir, günümüz teknolojisi düşünüldüğünde ve her gün daha da gelişen nesnelerin interneti teknolojisine ters düşmektedir.

Endüstriyel ortamda önemsiz veri olamaz düşüncesi ile yola çıkıldığında akıllı üretim tesislerinde en uç noktalarda bile sensörler yardımıyla veri toplayabilen bir sistemdeki yapı anlık olarak saldırı tespiti yapılarak korunmalıdır. Saldırıları tespit etmek ve sistemi korumak hayati önem taşımaktadır. Bir bilgisayar ağı üzerindeki Saldırı Tespit Sistemi (Intrusion Detection System, IDS), sistemin kötü amaçla kullanılmasını önlemenin ilk adımıdır (Daya, 2013). Çeşitli sensörlerden gelen veriler analiz etmek için

sunuculara (server) göndermek gerekir. Bu gönderilen veriler özellikle endüstriyel alanlar gibi anlık bilgi değişikliklerinin takip edilmesi gereken sahalarda çok büyük zararlara sebep olabilecek sistemlerin takibini yapan gözlem merkezleri için hayati öneme taşımaktadır. Bu nedenle, saldırı tespit sistemleri sürekli olarak geliştirilmektedir (Sadotra ve Sharma, 2016) .

Bu çalışmanın geri kalanı genel hatlarıyla şu şekildedir: Güvenlik duvarı ve saldırı tespit sistemleri gibi endüstriyel ağ güvenliğini sağlamak için mevcut yöntemler ve olabilecek ortadaki adam saldırısının anlatıldığı ikinci bölümün ardından üçüncü bölümde, konu ile ilgili literatür çalışmaları ele alınmıştır. Tezin oluşturulmasında kullanılan araçların tanıtıldığı Materyal ile geliştirilen sensör verisi kontrol algoritmasının ayrıntılı açıklamasına yer verildiği Yöntem dördüncü bölümde yer almaktadır. Yapılan testler ile çeşitli değerlendirmeler grafik, tablo ve görüntü olarak Araştırma Bulguları ve Tartışma bölümünde yer almaktadır. Son bölümde ise, bu çalışmadan elde edilen sonuçlar değerlendirilmiştir. Ayrıca tezde önerilen yöntem için hazırlanan bilgisayar yazılımının kaynak kodu ekler bölümünde sunulmuştur.

2. AĞ GÜVENLİĞİ

Ağ güvenliğinde temel amaç bilgi güvenliğini sağlamaktır. Anlık bilgiyi üreten akıllı sensörler ile saha sürekli gözlem altındadır. Sahadaki sensörlerden gelen veriler merkezi ağ yönetim birimi ile takip edilir. Dolayısıyla, endüstriyel ağlarda saha güvenliği son derece önemlidir. En kritik üretim sahalarının sensörlerden gelen verilerin tutulduğu sunuculara yetkisiz olarak erişimini engelleyen fiziksel bir güvenlik sunulmalıdır. Bu güvenlik sistemine, fiziksel olarak güvenlikli ekipman raf odaları, korunaklı mühendislik çalışma merkezleri veya operasyonel kontrol merkezlerine sınırlı erişimi de dahildir. Bir endüstriyel sahanın yerel ağına yapılabilecek siber saldırılar, ağ üzerinde diğer cihaz bağlantıları vasıtasıyla yapılır. Bunu önlemek amacıyla kritik noktalara kurulan güvenlik duvarı yazılımlarının temel işlevi yetkilendirme ve kısıtlama yapmaktır. Ancak veri aktarımında paket içeriğinin doğruluğunun denetimini yapmamaktadır.

Bu bilgi güvenliği 4 temel unsurdan oluşmaktadır.

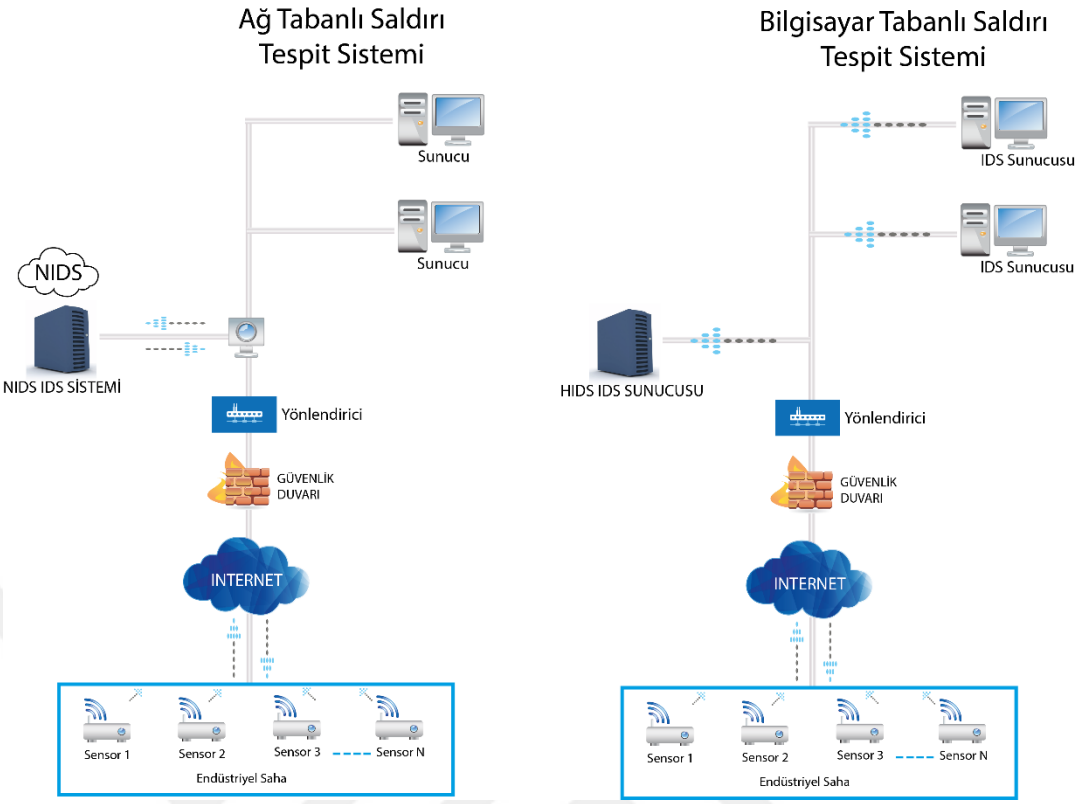
- **Gizlilik:** Gizliliğin sağlanması için alınan tedbirler, hassas bilgilerin yetkisiz kişilere ulaşmasını engellemek için tasarlanırken aynı zamanda yetkili kişilerin rahatlıkla erişmesini sağlar. Gönderilecek ve alınacak bilgilerin, sadece gönderici ve belirlenen alıcı tarafından paketin içeriğini anlayabilmesidir.
- **Mesaj Doğruluğu:** Verinin tüm yaşam döngüsü boyunca tutarlılığını, doğruluğunu ve güvenilirliğini korumayı içerir. Gönderici ve alıcı birbirlerinin kimlik denetimini yapsalar bile, iletişimlerinin içeriğinin, iletim halindeyken dinleyen ve değiştirmek isteyenlerden dolayı değiştirilmemiş olduğundan emin olmaktır. Verinin ağlar arasında iletimi esnasında değiştirilmemesini amaçlar. Anlık verilerin herhangi bir değişikliğini algılamak için bazı araçlar ve yazılımlar mevcuttur.
- **Uç Nokta Kimlik Denetimi:** Herhangi bir paket gönderiminde gönderici ve iletişime geçen diğer tarafın birbirlerinin kimliklerinin doğrulamasıdır.
- **Kullanılabilirlik:** Yetkili kişilerce bilgiye güvenilir erişimin garantisidir. Tüm donanımın titizlikle korunması, gerekli olduğunda donanım onarımlarının gerçekleştirilmesi ve yazılım çakışmalarının olmadığı doğru bir şekilde çalışan bir işletim sistemi ortamının sağlanmasını amaçlar.

Programlanabilir saha cihazları, örneğin PLC, akıllı sensörler, saldırı tespit sistemleri, güvenlik duvarı yazılımları, veri kontrol ya da şifreleme yöntemleri ile bir bütün olarak doğru bir şekilde yapılandırıldığı takdirde yukarıda sayılan unsurlar sağlanabilir.

2.1. Saldırı Tespit Sistemi

Saldırı tespiti, bir sisteme yapılacak izinsiz bir müdahale gerçekleşmeden önce, anlık olarak veya saldırı sonrası sunucu kayıtları ile bilgisayar ağı trafiğinden alınan verilerden yola çıkılarak çeşitli metotların yardımı ile gerçekleştirilen bir analizdir (Can, 2007).

Bilgisayar Tabanlı Saldırı Tespit Sistemi (Host-based Intrusion Detection System, HIDS) yapısındaki sistemlerin yanında bilgisayarlar ile çeşitli cihazların bir ağ üzerinden haberleşmesi gibi farklı ihtiyaçların oluşması ile Ağ Saldırı Tespit Sistemi (Network Intrusion Detection System, NIDS) teknolojisi geliştirilmiştir. HIDS sistemi yapısı gereği NIDS sisteminden farklıdır. HIDS sadece üzerinde çalıştığı sistemdeki saldırıları tespit eder. NIDS ise saldırıyı tespit etmek için ağ üzerindeki tüm trafiği izler. Şekil 2.1.'de de gösterildiği gibi, saldırı tespiti amacıyla HIDS ve NIDS sistemlerinin hibrit olarak kullanıldığı Dağıtık Saldırı Tespiti Sistemleri (Distributed Intrusion Detection System, DIDS) de mevcuttur.



Şekil 2.1. NIDS ve HIDS saldırı tespit sistemlerinin karşılaştırılması.

2.1.1. Bilgisayar Tabanlı Saldırı Tespit Sistemi

Bilgisayar Tabanlı Saldırı Tespit Sistemi, (Host-Based Intrusion Detection System, HIDS), iletim sistemlerinde kritik olayların görülebilmesini sağlar. Ağ üzerinde keşfedilen kötü niyetli saldırıları veya olağandışı hareketleri tespit edebilir ve bunlara tepki verebilir. HIDS, ağdaki cihazlar açısından her cihaz için ayrıca çalışır. Bir bilgisayar tabanlı saldırı tespit sistemi, gelen ve giden paketleri yalnızca üzerinde çalıştığı bilgisayar için izler ve şüpheli durum algılanırsa alarm vererek ilgili kontrolcüyü veya sistem yöneticisini uyarır. Sistemde bulunan dosyaların anlık görüntüsünü alır ve onu önceki anlık görüntü ile eşleştirir. Kritik ağ paketleri değiştirilmiş veya bozulmuşsa, kontrol edilmesi için yöneticiye bir uyarı gönderir (Kamel ve ark., 2005).

2.1.2. Ağ Tabanlı Saldırı Tespit Sistemi

Ağ Tabanlı Saldırı Tespit Sistemi, (Network-Based Intrusion Detection System, NIDS), trafik akışını kontrol etmek ve bilinen saldırı türlerini bir kayıt sistemi ile

karşılaştırma yapmak amacıyla ağ sistem altyapısında (güvenlik duvarı dışında, özellikle de Sivilleştirilmiş Alan {Demilitarized Zone, DMZ} gibi alanlarda) kullanılır (Varunkumar ve ark., 2014, Jadidoleslamy, 2012). NIDS, ağdaki bütün trafiği izlemek için ağ içindeki önemli noktalara yerleştirilir. Ağdaki tüm trafiği analiz eder, veri paketlerini kontrol eder ve daha önceki saldırıların kayıtlarını tutarak yeni saldırılarla karşılaştırır ve analiz eder. Saldırı tespiti durumunda veya olağan dışı bir durum tespit edilir edilmez ilgili birime uyarı verir. Daimi olarak gelen giden bütün trafiği tarar.

2.1.3.Dağıtık Saldırı Tespit Sistemi

Dağıtık Saldırı Tespiti Sistemleri (Distributed Intrusion Detection System, DIDS), genellikle tüm cihazların birbirleriyle veya gelişmiş ağ izleme, olay analizi ve anlık saldırı verilerini basitleştiren merkezi bir sunucu ile iletişim kurduğu geniş bir ağ için kullanılır (Robbins, 2002; Einwechter, 2017). Çizelge 2.1.'de saldırı tespit sistemlerinin yerleştirme konumu, veri kaynağı ve kontrol alanı açısından karşılaştırılması yapılmıştır (Jadidoleslamy, 2012).

Çizelge 2.1. Farklı IDS'lerin mimarilerine göre karşılaştırılması.

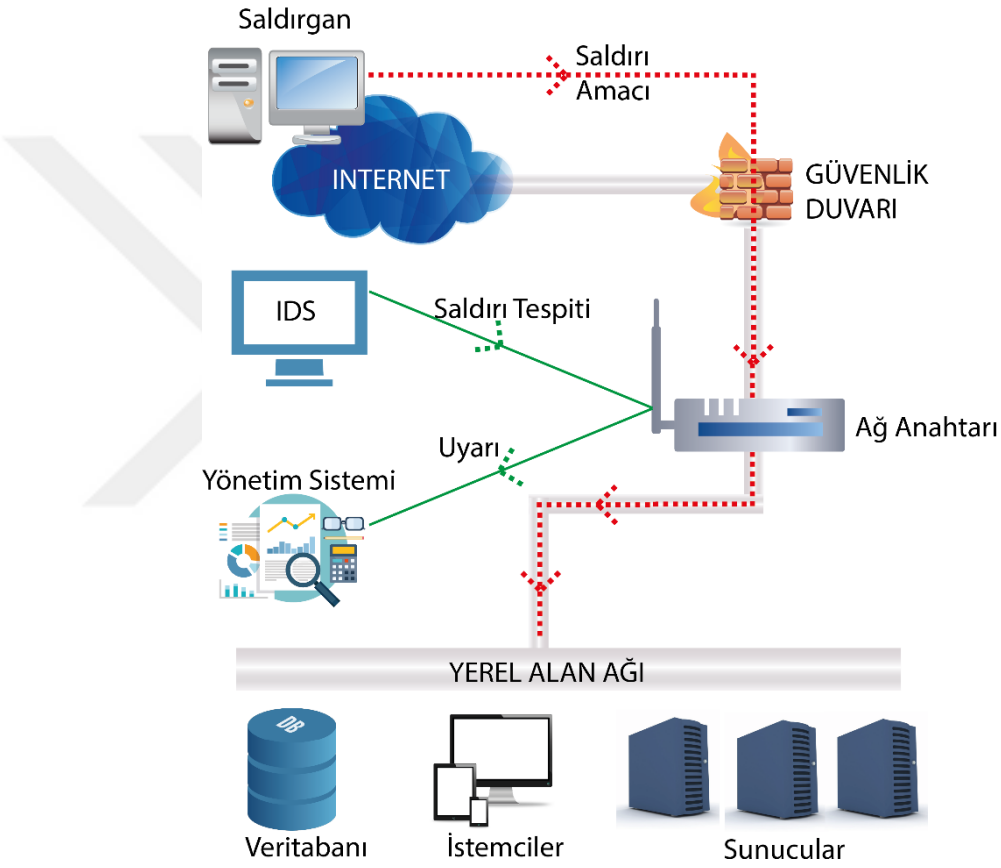
IDS Türü	Yerleştirme Konumu	Veri Kaynağı	Kontrol Alanı
HIDS	Sistem Kontrolü Altında, Yazılım Süreci	Yerel Ağ Trafiği (İşletim Sistemi düzeyinde) ve Günlük Dosya Kayıtları	Yerel Sunucu Sistemi
NIDS	İzole Edilmiş Bir Ağ Trafik Rotasında, Yazılım Süreci	Ağ Trafiği (Ağdaki Ham Veri Paketleri)	Yerel Segment veya Tüm Ağ
DIDS	Dağıtık ve Heterojen (Ev Sahibi, Ağ ve Merkezi Yönetim Sistemi)	Ana Trafik ve Ağ Trafiği	Ağ Genişliği (Tüm Ana Bilgisayarlar ve Farklı Ağ kesimleri)

2.2. Güvenlik Duvarı

Bir kurallar dizini içerisinde ağa gelen ve giden paket trafiğini sürekli kontrol eden yazılım ve donanım tabanlı ağ güvenlik sistemidir. Güvenlik duvarları, belli bir noktadan erişim denetimi yaparak yerel ağımızdaki bilgisayar sistemlerine internet ağından erişen kullanıcıların kullanımlarını kurallar ile sınırlandırarak güvenlik açıklarının en aza

indirilmesini sağlar (Wool, 2004; Cheswick ve ark., 2003). Ağ güvenliğinde sağlanması istenen gizlilik, erişilebilirlik ve kimlik denetimi gibi ana unsurları tehdit eden saldırı girişimlerine karşı güvenlik duvarı bazı önlemler almaktadır (Ranathunga ve ark., 2016). Fakat tek başına güvenlik duvarı bu güvenliği sağlamada yeterli değildir (Borhade ve Kahate, 2016).

İkisi de ağ güvenliği ile ilgili olmasına rağmen saldırı tespit sistemi güvenlik duvarından farklıdır. Şekil 2.2.'de bu görev farklılığı gösterilmektedir.



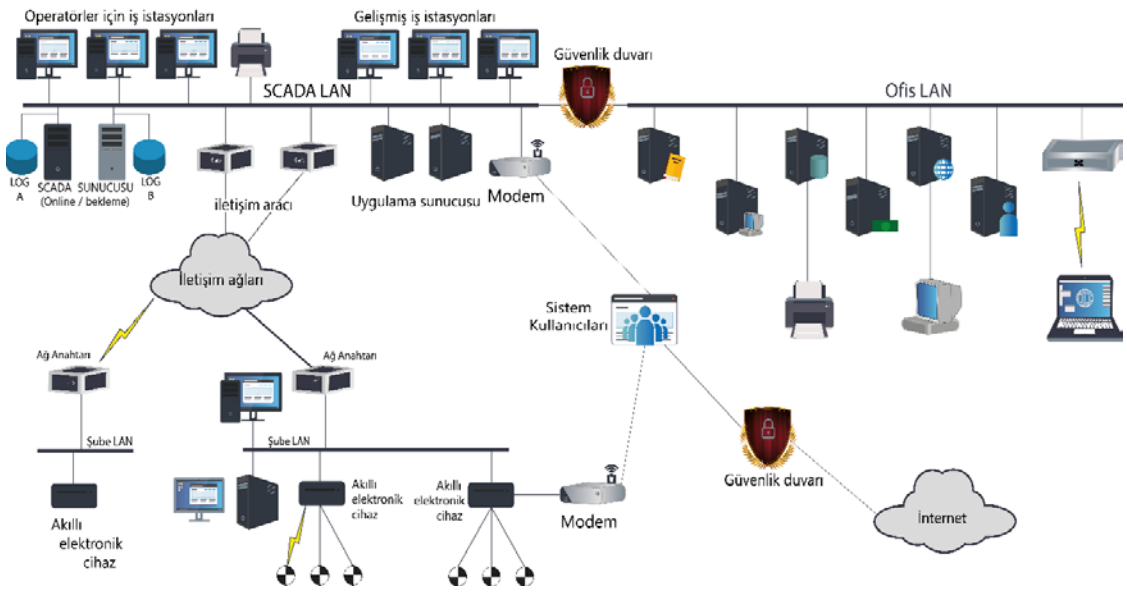
Şekil 2.2. Güvenlik duvarı ve saldırı tespit sisteminin birlikte çalışması.

Bir güvenlik duvarı saldırıları durdurmak için yerel ağ dışından gelen paketlere paket içerisindeki değerler açısından bakmaz. Güvenilir paket ya da değil olarak tespit yapar. Paketin içerisindeki değişiklerle ilgilenmez (Sadotra ve Sharma, 2017). Bu açıdan paket analizinde sıkıntı yaşanır. Güvenlik duvarları, saldırıları önlemek için ağlar arasındaki erişimi sınırlar ve ağın iç tarafından gelen bir saldırı sinyali vermez. Saldırı Tespit Sistemi, şüphelenilen bir saldırı durumu olduğu anda değerlendirir ve alarm uyarısı

verir. Bağlantıları kesen sistem, saldırı engelleme sistemi olarak adlandırılır ve uygulama katmanlı güvenlik duvarının başka bir formudur.

2.3. Ağ Topolojisi

Topoloji kavramı terim olarak, bir ağın fiziksel ve mantıksal yapısının bir arada olma durumunu açıklar. Ağ yapısını meydana getiren kavramların birbirleri ile olan bağlantı tipi, kablolama şekli, ağ yapısında kullanılan cihazlar ve ağ iletişim protokollerinin ağa uygulanış şekli topoloji kavramı içerisinde tanımlanır. Doğru yapılandırılmış ağ topolojisi güvenlik için ilk önceliktir. Şekil 2.3.'te endüstriyel saha için güvenli bir topoloji örneği gösterilmektedir.



Şekil 2.3. Endüstriyel saha için gösterilen örnek bir ağ topolojisi.

IIoT ile bütün cihazların internet ağına bağlandığı sistemlerde ve ağ üzerinden uzaktan erişim ile çalışan mimarilerde, güvenlik açısından iyi tasarlanmamış bir ağ topolojisi mevcut ise endüstriyel sistemlere yapılacak bir saldırı çok kötü sonuçlar doğurabilir.

2.4. Tanımlama, Kimlik Doğrulama ve Yetkilendirme

Tanımlama, kimlik doğrulama ve yetkilendirme benzer güvenlik kavramlarıdır. Ancak temelde bazı farklılıklar içerirler. Tanımlama, ağ içerisindeki cihaz kullanım yetkisini ilgili sisteme bildirmektir. Daha açık bir şekilde ifadeyle kullanıcı adını bildirmektir. Şifre girmek gibi bir işlem değildir. Şifre girmek tanımlamada kullanılan doğrulama yöntemidir. Kimlik doğrulama bir kullanıcının sisteme ya da kaynağa erişimde kimliğinin doğrulanması işlemidir. Kimlik doğrulama tanımlama sonrası hedef sistem ile istemci (client) arasındaki ortak anahtar bilgisini içerir (Kim ve Lee, 2017). Karşılıklı şifre benzeri bir anahtar kullanırlar ve kimlik doğrulamasını gerçekleştirirler. Kimlik doğrulama yetkilendirme işleminden önce yapılır. Tanımlama ve kimlik doğrulama işlemleri sonrası hedef sistemde iki işlem gerçekleştirilmiş olur. Kimlik bilgisi verilmiş ve o kimliğin asıl sahibi olduğu kanıtlanmış olur. Bu süreçten sonrası kimliği doğrulanan istemcinin erişmek istediği hedef sistem içerisinde yetkileri belirlenir ve kurallar dizini tespit edilir. Diğer bir ifade ile yetkilendirme gerçekleşmiş olur (Metz, 1999).

2.5. Veri Doğrulama

Veri Doğrulama (Data Validation), verileri kullanmadan önce kontrol etmek veya işlemeden önce kaynak verilerin doğruluğunu kontrol etmek anlamına gelir. Üretilen sensör verilerinin ağ sisteminde veri paketleri kurallarına göre uygun şeklide veri iletimi yapıldığını kontrol eder ve yine bu kurallara uygun olmayan veri iletimini önlememizi ya da kontrol etmemizi sağlar. Hedef kısıtlamalarına veya amaçlarına bağlı olarak verilere farklı doğrulama türleri uygulanabilir. Verilerin tutarlılığı, aralığı, tipi ve uzunluğu gibi kriterler üzerinden doğrulanması yapılabilmektedir (Xie ve ark., 2017).

2.6. Veri Bütünlüğü

Veri bütünlüğü kontrolü, verinin sahada üretiminden sunucuda işlem aşamasına kadar tüm yaşam döngüsü boyunca tutarlılığının kontrol edilmesi durumudur. Veri bütünlüğü veri paketinin bozulması kavramının tamamen aksi yönünde bir kavramdır.

Veri bütünlük kontrolü bazı algoritmik yöntemler ile yapılabilmektedir (Cohen, 1987; Huang ve Cohen,1988; Tyushev ve ark. 2016). Bunlar sırasıyla:

- Sağlama Toplamı (Checksum Algoritması)
- Mesaj Özütü Fonksiyonları (Hash Algoritması)

2.6.1. Sağlama Toplamı

Sağlama Toplamı bir veri paketinin gerçeği ile aynı olup olmadığını doğrulamak için kullanılan bir yöntemdir (Huang ve Cohen, 1988). Bu yöntem ile veri paketinin içeriğinden yararlanarak üretilen doğrulama değeri üzerinden giden ve gelen paketlerde bozulma olup olmadığı kontrol edilir (Cohen, 1987).

Gelişmiş ağ mimarilerinde kullanılmak üzere ve paket anahtarlamalı bilgisayar iletim hattında kayıpsız veri gönderimi gerçekleştirmek için iletişim kontrol protokolü (Transmission Control Protocol, TCP) geliştirilmiştir. TCP protokolü OSI referans modelinin iletim katmanında görev yapar. TCP her paketi oluştururken, veri doğrulama amacıyla bir de sağlama toplamı hesap eder ve başlığa ekler. TCP bu sayıyı kontrol ederek transfer sırasında pakette herhangi bir hata meydana gelip gelmediğini kontrol eder. İletim esnasında önerilen sağlama toplamı ve TCP protokolü ile veri bütünlüğü sağlanmış olur (Balan ve ark., 2002). Verideki hatayı sezen ve verinin düzgün iletilip iletilmediğini kontrol eden yöntemlerde tercih edilir.

2.6.2. Mesaj Özütü Fonksiyonları

Mesaj Özütü Fonksiyonlarının (Hash Algorithm), çalışma şekli uzun bir girdiyi alarak daha kısa bir alanda göstermektir. Amaç giren veride bir değişiklik olduğunda bunun çıkan veriye de yansımasıdır. Buna göre mesaj özütü fonksiyonları ya veri güvenliğinde, verinin farklı olup olmadığını kontrol etmeye yarar ya da verileri sınıflandırmak için kullanılır (Wang ve ark., 2017).

Hash algoritması, herhangi bir verinin, içeriğinin çeşitli şifreleme yöntemleriyle şifrelenerek anlaşılamaz hale getirilmesi işlemine verilen isimdir. Mesaj özütü fonksiyonları her zaman bir algoritmaya bağlı olarak çalışır ve günümüzde mesaj özütü fonksiyonunu kullanmak için MD2, MD5, SHA-1 ve SHA-256 gibi çeşitli şifreleme

algoritmalarından faydalanılır (Daemen ve Rijmen, 2010; Kim ve ark., 2016). Hash algoritmaları ile verileri şifrelemenin asıl amacı verilerin güvenli tutulmasıdır. Bu yöntemlerde veri güvenliği içerik doğrulamasından daha önceliklidir.

2.7. Ağ Güvenliğine Yapılan Ortadaki Adam Saldırısı

Ortadaki Adam Saldırısı (Man-in-the-Middle Attack, MITM), araya girenin, tarafların haberi olmadan iki taraf arasındaki mesajları okuyabilmesi ve değiştirebilmesi olarak tanımlanabilir (Guha ve ark., 2007). Ağ üzerinde veri paketleri serbestçe dolaşır. Hedef adrese gönderilen paketler, ağ ortamında bir bilgisayarın hedef sunucuya göndereceği ya da sunucudan alacağı herhangi bir bilgi bilgisayardan çıktıktan sonra gideceği hedef bilgisayarı bulmaya çalışır. Temel olarak hedefinde kendi IP adresi olmayan bir paketi alan makineler, bu paketlerle ilgili herhangi bir işlem yapmamaları gerekir. Ancak istenirse bu paketlere müdahale edebilir ya da içeriğini öğrenebilirler. Ortadaki adam saldırısı ağ üzerindeki paketleri yakalayarak içeriğinde değişiklik yapan bir tehdittir (Nam ve ark., 2010; Callegati ve ark., 2009). Sensör verileri istemci ve sunucu arasında doğrudan iletilmek yerine, saldırgan tarafından değişime uğratılarak gönderilir. Saldırı tespit sistemi olmaması durumunda iletilen veride yapılan değişiklik her iki bilgisayar tarafından da tespit edilemez.

Örnekle açıklamak gerekirse, ağa gönderilmek üzere hazırlanan paket ethernet kartına (ağ kartı) teslim edilir. Bağlı bulunduğunuz ağın yapısına göre paketlerin istenen hedef bilgisayara gönderilmesinin doğruluğunu kontrol eden protokoller vardır. Basit olarak çoğu fiziksel ağ, ethernet protokollerini kullanır. İstemci bilgisayarından çıkan bir paketin sunucu bilgisayarına gitmesi özetle şu şekilde gerçekleşir: İstemci bilgisayarı bulunduğu ağa sunucu bilgisayarının Ortak erişim kontrolü adresini (Media Access Control, MAC) ARP REQUEST paketleriyle sorar. Ağ içinde bu paketi alan bilgisayarlar adresi biliyorsa bildirim yapar. Adresi bilmeyen cihazlar tepkisiz kalır. Hedef sunucu paketi aldığında ağa ARP REPLY paketleri göndererek MAC adresini bildirir ve her bilgisayar bu kaydı saklar. Artık tekrar gerektiğinde sunucu adresi kayıt altına alınmıştır. Paketin gideceği sunucu bilgisayarının MAC adresini bulan istemci paket üretildikçe bu adrese gönderir. Paket trafiği bu şekilde gerçekleşmelidir. Ancak, saldırgan bilgisayar ARP REQUEST yani MAC adresi sorgulaması yapılmadan ARP REPLY paketleri

göndererek ve sunucu bilgisayarının MAC adresini kullanarak kendi MAC adresinin sunucu MAC adresi ile aynı olduğunu bildirir. Paketi alan bilgisayarlar artık sunucu bilgisayarının yerine göndermek istedikleri paketleri saldırgan bilgisayarına yollamak zorunda kalırlar. MAC adresini taklit etmek ARP SPOOF saldırısı olarak adlandırılır. Bu saldırı, ortadaki adam saldırısı için kullanılan en yaygın saldırı biçimidir.

Şekil 2.4. Ortadaki adam saldırısı.

Ortadaki adam saldırısının saptanması, herhangi bir kimlik doğrulama protokolünün temel amacıdır. Bu saldırıyı önlemek ya da tespit etmek için olası iki çözüm yolu aşağıda verilmiştir:

- Sunucu sertifikası doğrulaması ya da anahtar paylaşımı gibi şifreleme teknolojileri kullanarak çözülebilir. Ancak bu yöntemle şifrenin çözülmesi durumunda veri saldırgan tarafından görülür ve bu saldırının tespiti yapılamaz.
- Sunucuya gelen mesaj doğruluğu kontrol edilerek çözülebilir. Sunucu ve istemci üzerinde doğrulama anahtarı oluşturularak üretilen anahtar değerleri karşılaştırılmalıdır. Bu yöntem bozulan verinin tespitini yapabilir ancak veri içeriğinin gizliliğinin önem taşımadığı sistemlerde tercih edilmelidir.

Endüstriyel otomasyon sistemlerinde sahada algılama ile gözlem ve kontrol yapmak için uzun yıllardır kablolu algılayıcı ağlardan yararlanılmaktadır. Fakat kablolu algılayıcıların dağıtımı ve konumlandırılması yüksek maliyet gerektirmektedir. Bunun yanında algılayıcılar endüstriyel sahada uygun şekilde konumlanması durumunda bile algılayıcıların yenileme yapılmasının ve güncellenmesinin maliyeti ciddi boyutlara ulaşmaktadır. Algılama cihazlarının yanında belirli noktalardan insan kullanarak el

analizörleri ile kontrol yapan endüstriyel sahalar da bulunmaktadır. Algılayıcı yerleştirmenin yüksek maliyeti ve insan müdahalesiyle yapılan hassasiyeti düşük kontroller ağ üzerinden kablosuz olarak anlık veri iletiminin gerekliliğini ortaya koymuştur (Oğuz, 2012). Ancak kablosuz ağ üzerinden yapılan veri iletimi ortadaki adam saldırısı gibi tehditlere açık duruma gelir. Özellikle, geniş endüstriyel sahalardaki sensörlerin sağladığı büyük veri, saldırı tespiti amacıyla anlık olarak kontrol edilmelidir.

Veri iletiminin sık gerçekleştiği ve ortaya büyük verilerin çıktığı sistemler için şifreleme benzeri yöntemlerin çeşitli olumsuz yönleri bulunmaktadır. Bu durumun iki önemli sebebi vardır:

- Şifrelemenin Boyutu: Simetrik şifrelemeden ziyade özellikle asimetrik şifreleme veri boyutunu artırır. AES ve SHA Hash algoritması ile şifrelenen bir veri nomal boyutundan daha fazla alanda diskte yer kaplayabilir ve ayrıca ağın bant genişliğini daraltır. Bunun sonucu olarak sensör verileri kontrol noktasına gönderilirken ağ trafiğinde oluşan daralmadan dolayı gecikmeler meydana gelir. Veri boyutu artıkça veri iletim zamanında da yavaşlama bakımından doğrusal olarak bir artış gerçekleşir.
- Performans: Şifreleme desteğine sahip gelişmiş bir bilgisayarda, şifrelenmemiş bir veriyi işlemek için harcanan enerjiden daha fazlası şifreleme yapmak veya şifrelenmiş bir verinin şifresini çözmek için harcanır. Sensörden gelen verilerin büyüklüğü ve sıklığı arttıkça bu durum ciddi bir sorun olarak ortaya çıkar. Dolayısıyla, gerçek verinin elde edilip işlenmesi geciktiği gibi donanımın harcadığı enerjide de artış meydana getirir. Güvenlik amacıyla uygulanan şifreleme algoritması karmaşıklaştıkça bu oran daha da artar.

3. ÖNCEKİ ÇALIŞMALAR

Bilgisayar ağ sistemleri üzerine yapılan çalışmalarda en önemli konu güvenlik açıklarının tespiti ve güvenliğin sağlanmasına yöneliktir. Ağ içerisinde veri iletim teknolojisinde en kritik konu bilgi güvenliğidir. Veri iletimi esnasında cihazlar ağa bağlıdır. Her türlü cihazın ağa bağlı olduğu sistemlerde güvenli veri iletiminin sağlanması için doğru güvenlik algoritmalarının tasarlanması son derece önemlidir. Bilgisayar sistemlerindeki güvensiz veri iletim ortamı IoT teknolojisine geçişte ivmeyi yavaşlatmaktadır. Hassas verilerin anlık takip edilmesi ve saldırı anında müdahale edilmesi kurulan ağ mimarisi ile bu mimaride çalışması için seçilen iletişim protokollerine bağlıdır (Livshits ve Lam, 2005; Klaus, 1999; Suresh ve Prasad, 2012).

Endüstriyel alanda sensörlerden alınan veriyi iletmek için gerekli güvenlik kavramları birkaç başlık altında toplanabilir. Bunlar sırasıyla, bilgisayar güvenliği, ağ güvenliği, endüstriyel otomasyon güvenliği, nesnelerin interneti güvenliği ile veri iletim ve sensör güvenliği endüstriyel saha sistemlerinde kompakt bir güvenlik sağlar. Bu açıdan ilk önce en temel güvenlik seviyesi olan bilgisayar güvenliği ele alınır. Bu konuda Bishop (2003; 2005) ve Dieter (2010) tarafından bilgisayar güvenliği konusunda yapılan önemli çalışmalarda, güvenlik ihtiyaçları, politikaları, mekanizması, güvencesi ve bileşenleri üzerine detaylı incelemeler mevcuttur. Bilgisayarların bir araya gelmesiyle oluşan ağ sistemi ağ güvenliğini ihtiyacını ortaya çıkarmıştır. Bilgisayar güvenliği sağlandıktan sonra bir araya gelen bilgisayarların tasarım olarak güvenliği ikinci aşama olarak ele alınabilir. Ağ tasarımı, Açık Sistem Arayüzü modeline dayanan gelişmiş bir süreçtir. OSI, modülerlik, esneklik ve protokollerin uyumluluğunu sunar. Bu uyumluluk ağ güvenliği konusunda tasarımı sağlamlaştırır. Çünkü ağ tasarımı içerisinde güvenlik bir bütün olarak sağlanmalıdır. Ağı korumak hayati önem taşır. Pawar ve Anuradha (2015) ağ güvenliğini ve saldırı türlerini tanımlamışlardır. Ayrıca, ağı platformalara ayırarak güvenlik yönetimini kolaylaştırma üzerine Tan ve ark. (2016) tarafından entegre bir güvenlik platformu sunularak önemli ölçüde katkı sağlamıştır. Puthal ve ark. (2017) da ağlardaki siber saldırı türlerini tanımlayarak bu konuda detaylı bilgi vermişlerdir.

Ağ güvenliğini sağlandıktan sonra endüstriyel otomasyon güvenliği ele alınarak tehditler değerlendirilir. Büyük güvenlik problemlerinden kaçınmak için endüstriyel kontrol sistemlerinin güvenliğinin sağlanması hayati öneme sahiptir. Sahadaki

sensörlerden anlık bilgi alan Denetleyici Kontrol ve Veri Toplama (Supervisory Control and Data Acquisition, SCADA) ve Programlanabilir Mantıksal Denetleyici, PLC, sistemlerini hedefleyen siber saldırılar büyük hasar meydana getirebilir. Son yıllarda SCADA ve PLC sistemlerini hedef alan saldırılar önemli ölçüde artmıştır ve bununla ilgili çeşitli tespit ve analizler Schuett (2014) tarafından yapılmıştır.

Gelen ve giden ağ trafiğini kontrol etmek açısından Ranathunga ve ark. (2016) tarafında kritik sensörler ile veri toplayan sunucular üzerinden güvenlik duvarı yapılandırmalarıyla daha sağlam bir sistem elde etmiştir. Üst düzey güvenlik politikası geliştirmek için endüstri tarafından en iyi uygulamalar önerilmiştir. SCADA ve PLC'lerin bazı kritik hizmetler için devamlılığı önemlidir. Bu sistemlerin iletişim cihazları, bilgisayarlar, sensörler ile doğru iletişim kurması sağlanmalıdır. Endüstriyel otomasyon sistemlerinde kullanılan iletişim protokollerinin çoğunun güvenlik mekanizmalarının olmaması nedeniyle, her biri internete bağlı cihazlar vasıtasıyla siber saldırı riski altındadır. Chandia ve ark. (2008) tarafından yapılan çalışmada, ağa bağlı cihazlar üzerinden saldırıya açık hale gelen endüstriyel otomasyon sistemlerinin ağ merkezli saldırıların odağında olduğu bildirilmiştir. Endüstriyel sahadaki siber tehditler sürekli artmaktadır ve bu tehditlerin tespiti anlamında Ding ve ark. (2017), Igure ve ark. (2006), Ralston ve ark. (2007), Fovino ve ark. (2009) ve Nicholson ve ark. (2012) çalışmalar yaparak endüstriyel alandaki tehditleri önceden belirlemiş ve olası tehlikelere karşı önlem alınmasını sağlamışlardır. Miyachi ve Yamada. (2014) tarafından siber saldırı konuları ele alınarak endüstriyel otomasyon sistemleri için gizli saldırılar üzerine bir olay yönetimi algoritması sunmuşlardır. Hazırlık, tanımlama, kapsama, yok etme, kurtarma ve paketleme aşamalarını belirleyerek olay yönetim modellemesini ortaya çıkarmıştır.

Ağa bağlı endüstriyel otomasyon sistemleri gelişen teknolojilere uyumlu olarak üretilmektedir. Çok yoğun olmamakla birlikte IoT teknolojisi endüstriyel sahada yerini almaya başlamıştır. Geniş alanda algısal özellikleri ve insan müdahalesi olmaksızın doğrudan birbirleriyle iletişim kurma yeteneği bakımından IoT, endüstriyel sistemlerde uygulaması çoğu alanda yeni gelişmekte olsa da endüstri, enerji sistemleri, ev otomasyonu, lojistik, sağlık, tarım gibi alanlarda gittikçe artmaktadır. Alaba (2017) ve Weber (2010) nesnelerin interneti üzerine detaylı bir sınıflandırma ve inceleme yapmışlardır. Bu incelemeler neticesinde nesnelerin internetinin güvenlik endişesi taşıdığına dikkat çekilmiştir. Aynı zamanda bu çalışmalarda IoT güvenliği alanında

mevcut eserlerin kapsamlı bir incelemesi yapılmış, en gelişmiş IoT güvenlik tehditleri ve güvenlik açıklarına odaklanılmış, alınabilecek tedbirler belirtilmiştir. Güvenlik tehditleri karşılaştırılarak saldırıların analizi yapılmıştır. Yaqoob ark. (2017) 21. Yüzyılın en tehlikeli sanal tehditlerinden biri olan Ransomware veri saldırı türünü değerlendirmiş ve IoT ile ortaya çıkan güvenlik tehditlerini ele almıştır.

Zhang ve ark. (2011) güvenli IoT teknolojileri sunmak için güvelik mimarisi tasarlamışlardır. IoT mimarisini uygulama, ağ ve algı katmanları olarak üçe ayırmışlardır. Bu tasarım üzerine kurulu olan IoT güvenlik mimarilerini 4 katmandan oluşturmuşlardır. İç içe geçmiş güvenlik ve bilgi zinciri mantığıyla kurulan güvenlik mimarisi uygulama katmanı, çekirdek katmanı, erişim katmanı ve algılama katmanı olarak tasarlanmıştır. Olası tehditleri engelleme amacı güdülmüştür. Shi ve Perrig (2004) sensör ağlarına dıştan ve içten gelen saldırıları tanımlayarak tehdit ve güven modellemeleri geliştirmiştir.

Wu ve ark. (2016) bir sensör veri seti hazırlayarak değerlendirmek üzere sanallaştırma teknolojilerinden yararlanarak kaynak tüketimi ve saldırı tespit oranını değerlendirmek için bir simülasyon ortamı oluşturmuştur. Önerdikleri hiyerarşik güvenlik çerçevesi içerisinde bir saldırı tespit mekanizması geliştirmişlerdir. Genelde gözetimsiz ortamlarda kullanılan ve kısıtlı kaynaklarla çalışan algılayıcılardan oluşan kablosuz algılayıcı ağları içerden ve dışarıdan gelebilecek birçok güvenlik atağına karşı savunmasız olduğu da ortaya konulmuştur.

Lee ve ark. (2010) hassas veri iletiminin yapıldığı kablosuz sensör teknolojisinde veri gizliliğini ve bütünlüğünü sağlayacak güvenlik mekanizmaları açısından üç başlık altında incelemiştir. Bunlar sırasıyla şifreleme algoritması, blok şifrelerin çalışma şekilleri ve mesaj doğrulama algoritmalarıdır. Prathima ve ark. (2016) iletim maliyeti açısından çoklu sorgularda güvenli veri toplama metodu önermektedir. Sensör düğümlerinde, çoklu ihtiyaç duyulan taleplere ait verilerin tek bir pakete toplanmasıyla yanıt vermektedir. Çünkü veri toplayıp toplama işlemi gerçekleştirirken güvenlik sağlamak kablosuz algılayıcı ağlarda endişe kaynağıdır. Önerdiği yöntem ile her sensör düğümünün şifrelenmiş verileri ileterek ve küme kafalarının şifrelenmiş veriler üzerinde matematiksel bir işlem gerçekleştirerek sensörlerden güvenli veri iletimini sağlamıştır.

IoT teknolojisinde temel amaç sensörler ile aynı ağ üzerinde bağlı olan cihazların haberleşmesini sağlayıp insan müdahalesi olmadan sistemin kendi kendine çalışmasını sürdürmektir. Ancak, sensörün veri ilettiği mikroişlemci üzerinde işlem yapmak için

ihtiyaç duyulan işlemci, bellek ve güç kaynaklarındaki kısıtlı donanım ve cihazlar arası ağ üzerinden kurulan iletişim esnasında ağa yapılan saldırılar ve engellemeler nedeniyle sensörler aracılığıyla iletişim kuran cihazlar için bazı zorluklar teşkil etmektedir.

Bu çalışmada, endüstriyel alanda IoT cihazlarının sensörler üzerinden iletilen verilerinin doğrulunu tespit etmek amacıyla bir karşılaştırma ve kontrol algoritması önerilmiştir. Ağ üzerindeki paketleri yakalayarak içeriğini değiştiren ortadaki adam saldırı tipi hedef alınıp buna karşı bozulmuş veri kontrol yöntemi geliştirilmiş ve anlık veri takip sistemi yapan bir yöntem önerilmiştir. Ayrıca önerilen yöntemin aynı amaç için kullanılan benzer yöntemlere olan çeşitli üstünlükleri ortaya konmuştur.

4. MATERYAL VE YÖNTEM

Bu çalışmada, sensörlerden gelen verinin saldırıya uğraması durumunda değiştirilme tehlikesine karşı veri kontrol yöntemi önerilmiştir. Bu yöntem sensörlerden gelen verilerin bozulması durumunda endüstriyel sahadaki IoT cihazlarının anlık kontrol problemine uygulanmıştır. Anlık sensör verisi kontrolü yapılarak IoT cihazlarının endüstriyel sahada kullanımı güvenlik hassasiyeti ve tespiti açısından önerilen yöntem incelenmiştir.

Yöntemin geliştirilmesinde kullanılan araçlar Materyal başlığı altında yer almaktadır. Önerilen yöntem kullanılarak sensörlerden gönderilecek veri paketlerinin hazırlanması ile sunucu tarafında gelen veri paketlerinin açılması, kontrol yöntemi ve hata tespitiyle yapılan anlık gözlem hakkında ayrıntılı bilgi Yöntem başlığı altında verilmiştir.

4.1. Materyal

Önerilen yöntem ile veri iletimi için bir yazılım ortamı kullanılmıştır. Bu amaçla örnek sensör verileri üretilmiş ve ağ ortamında veri iletimi yapılırken ortadaki adam saldırısı simülasyonu yapılarak yöntemin başarısı ölçülmüştür. Bu bölümde sırasıyla, kullanılan yazılım ortamı, donanım cihazları ve sensör verisi üretimi anlatılmaktadır.

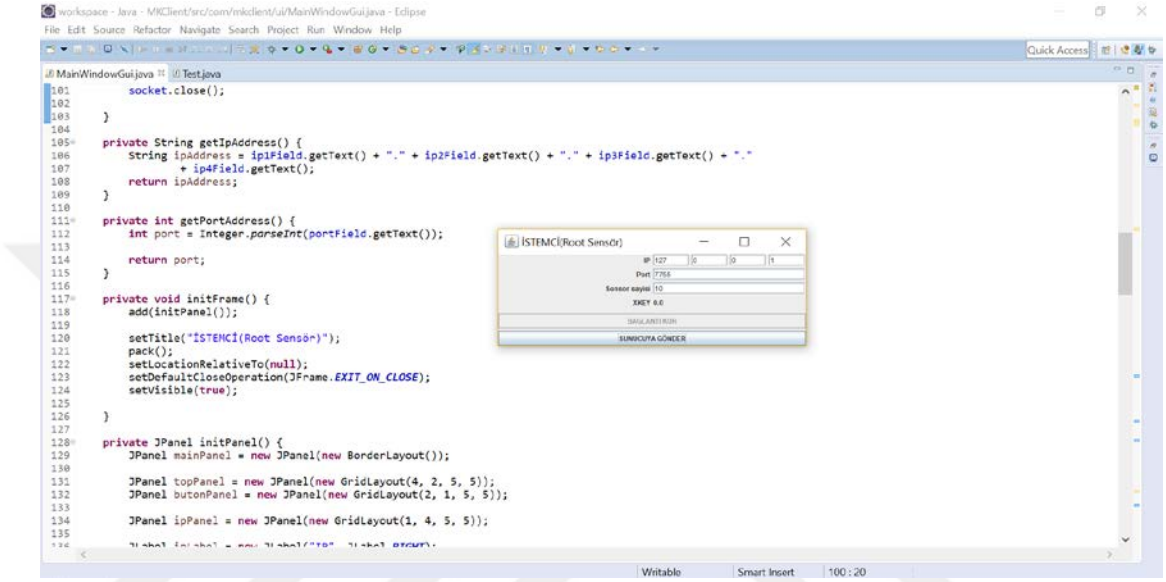
4.1.1. Yazılım Ortamı

Bu çalışmada yapılan endüstriyel saha senaryosu gereği belli sayıda sensörden alınan veri önerilen yöntem ile bir ağ paketi haline getirilip sunucu tarafına aktarılacaktır. Bu aktarım aşamasında tez konusu olan ortadaki adam saldırısına maruz kalan ağda verinin doğruluğu ya da bozulmuş verinin tespiti kullanıcıya bildirilecektir. Bu amaç çerçevesinde istemci sunucu mimarisi içerisinde bilgisayar yazılımları hazırlanmıştır. Bu yazılımlar için JAVA programlama dili kullanılmıştır. JAVA'nın diğer programlama dillerinden öne çıkan özellikleri açık kodlu, nesneye yönelik, zeminden bağımsız, yüksek verimli, çok işlevli, yüksek seviye, adım adım işletilen bir yapıya sahip olmasıdır.

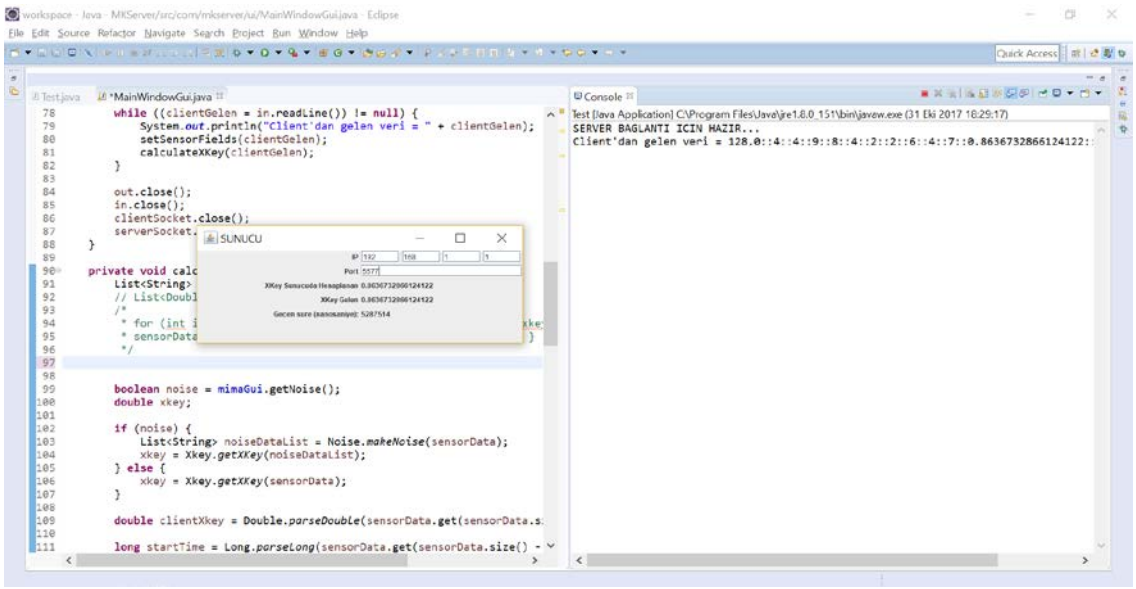
Yazılımların JAVA ile hazırlanmasında Eclipse ortamı kullanılmıştır. Eclipse, açık kaynak kodlu ve özgür bir tümleşik geliştirme ortamıdır (Integrated

Development Environment, IDE). Çoğunlukla üzerinde kullanılan yazılım dili Java ve Java ile ilişkili teknolojilerdir. Diğer programlama dilleri olan C, C++ ve benzeri dillerde kullanılabilmektedir. Genel kullanıma açık Eclipse lisansı (Eclipse Public Licence, EPL), Eclipse Vakfı tarafından kendi yazılımları için kullandığı açık kaynaklı lisans yazılımıdır.

Şekil 4.1 v.e 4.2.'de sırasıyla istemci ve sunucu tarafı için güvenli veri iletimi gerçekleştirmek amacıyla önerilen yöntemin hazırlandığı Eclipse ortamı görülmektedir.

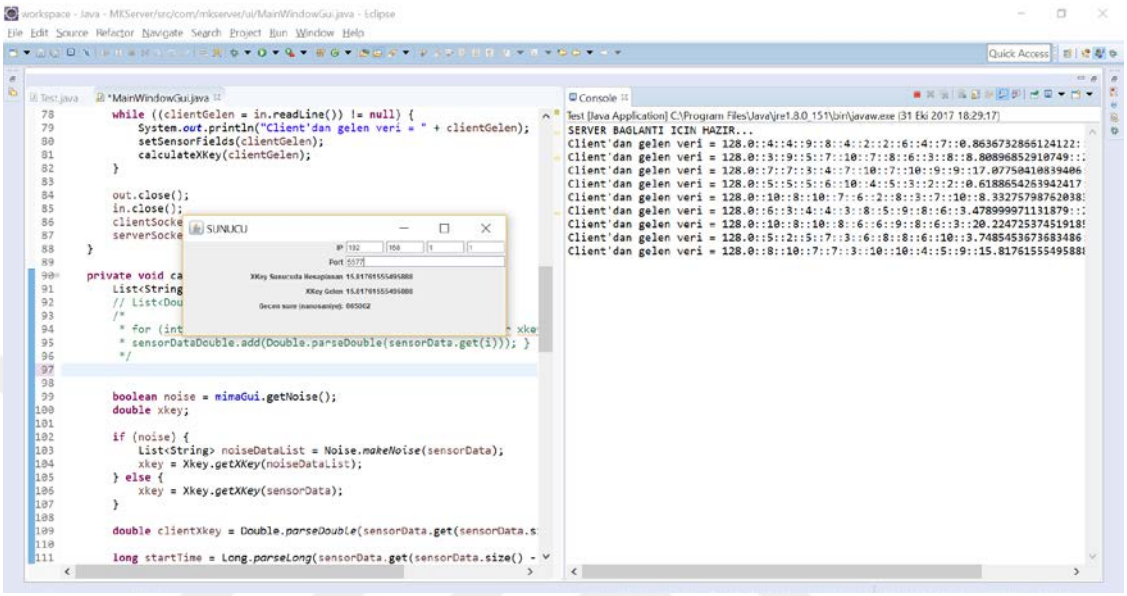


Şekil 4.1. İstemci (Root Sensör) tarafı Eclipse ortamı



Şekil 4.2. Sunucu tarafı Eclipse ortamı.

Sunucu tarafında veriler geldikçe yukardan aşağıya sürekli olarak konsolda sensör verileri görülmektedir. Şekil 4.3.'te konsolun veri paketleme sıralaması Eclipse ortamı görülmektedir.



Şekil 4.3. Konsol ekranı verileri eclipse ortamı.

4.1.2. Donanım Cihazları

Yazılımın hazırlandığı ve testlerin yapıldığı bilgisayar, Mac mini 1.4GHz dual-core Intel Core i5 (2.7 GHz'e kadar Turbo Boost) işlemcili, üzerinde 4 GB 1600 MHz LPDDR3 RAM, 500 GB (5400 rpm) sabit sürücülü ve Intel HD Graphics 5000 ekran kartı bulunan bir masaüstü bilgisayarıdır. Bu bilgisayar aynı zamanda sunucu olarak kullanılmıştır.

İstemci ve sunucudan oluşan bir mimari kullanılarak kurulan iletim ortamında, ikinci bilgisayar istemci olarak kullanılmıştır. Toshiba Kirabook 8 Intel Core i7 5500U 2.4GHz işlemcili, Intel HD Graphics, 8GB bellek 256GB SSD sabit sürücülü taşınabilir bir bilgisayardır.

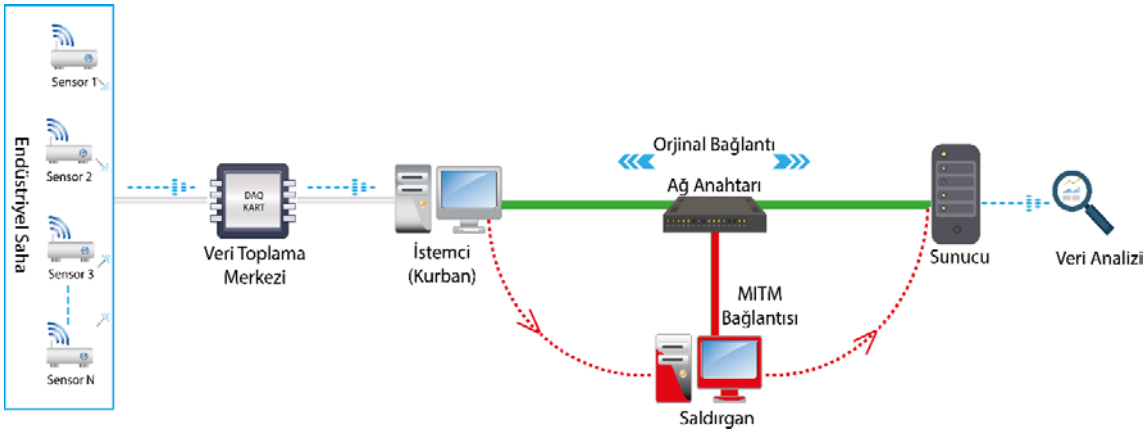
İstemci ve sunucu bilgisayarların veri iletimi amacıyla iletişim kurmak için AIRTIES AIR-0224 24 PORT 10/100/1000 24P-GIGABIT RACKMOUNT ağ anahtarı kullanılmıştır.

4.1.3. Sensör Verisi Üretimi

Testler için sık aralıklarla anlık sensör verisi üretilmiştir. Java dilinde rastgele sayı üretimi için ya `Math.random()` yöntemi ya da `java.util` paketinin altındaki `Random` sınıfı kullanılır. Bu çalışmada `java.util` paketinin altındaki `Random` sınıfını kullanılmıştır. Testler için üretilen tüm sensör verileri gerçek endüstriyel sahalarda görülebilecek tam sayı değerlerden oluşmaktadır. Şekil 4.4'te `Random` sınıfının kullanımı gösterilmiştir. Sahada ortalama 1-10 derece aralığında değerler üreten bir sistem örneklemiştir. Değerler aynı aralıkta her seferinde değişmektedir. Bu değerler iletim verisi olarak önerilen algoritma ile bir ağ içerisinde gönderilecektir.

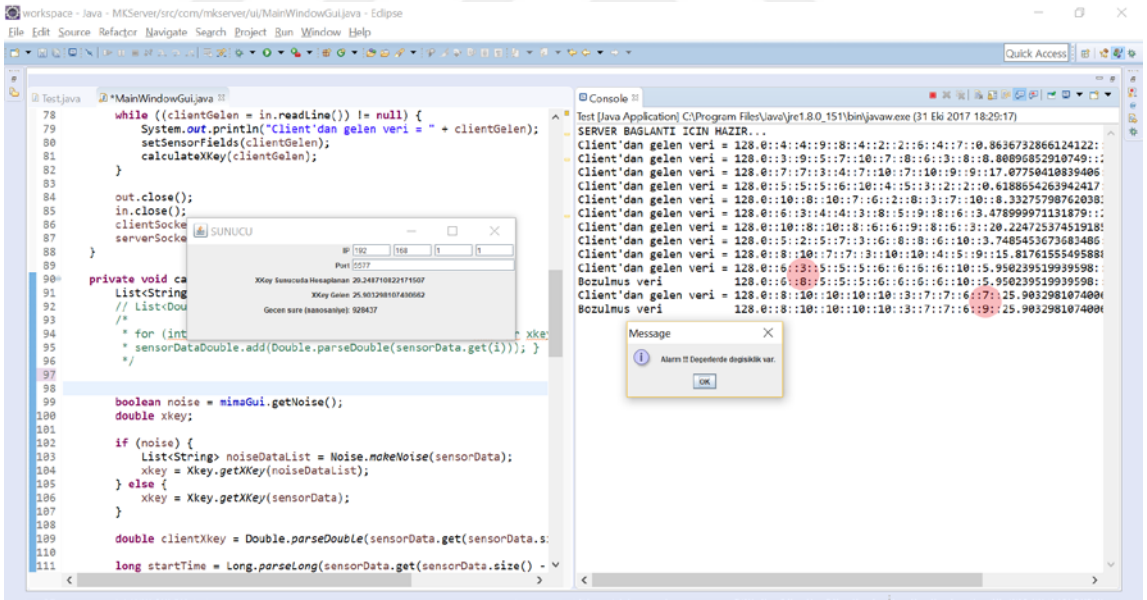
Şekil 4.4. Testler için sensör değeri üreten JAVA programlama dili random sınıfı.

Öngörülen endüstriyel saha senaryosunda üretilen sensör verilerine yapılan saldırı ortadaki adam saldırısıdır. Yöntemimiz doğrultusunda bu senaryo için bir ortam hazırlanmıştır. Şekil 4.5.'te saldırı anında yapılan ortadaki adam saldırı senaryosu gösterilmektedir. İstemci ve sunucu birbirleriyle doğrudan iletişim kurarken ortadaki adam saldırısı ile araya giren saldırgan veri paketlerini bozarak sunucuya benzer fakat bozulmuş veri paketleri göndermektedir.



Şekil 4.5. Oluşturulan ortam ve ortadaki adam saldırısı (MITM) senaryosu

Saldırı şeklinin Eclipse ortamında alarm ile tespit edilişi Şekil 4.6.'da gösterilmektedir. Sensör verileri iletim anında araya giren saldırgan bozulmuş veri paketlerini sunucuya göndermektedir. Önerilen yöntem alarm vererek bildirim yapar. Gönderilen veri paketi ile bozulmuş veri paketini açarak değiştirilen değeri gösterir.



Şekil 4.6. Ortadaki adam saldırısı esnasında alarm ile saldırı tespiti Eclipse ortamı.

4.2. Yöntem

Bu çalışma ile endüstriyel sahadaki sensörlerden anlık olarak aktarılan veriyi kontrol eden ve ortadaki adam saldırısı sonucu bozulmuş veriyi tespit eden bir yöntem önerilmiştir. Öncelikle sahadaki sensör verileri toplu halde yönlendirici (root sensör) görevi yapan istemci bilgisayarda toplanmaktadır. Bu bilgisayar gelen verileri paket haline getirir ve sunuculara gönderir. Önerilen yöntemde, sensör verilerinin gönderilen sırada ve değerde olduğundan emin olmak için kullanıcıya bağlı bir algoritmaya dayanan bir doğrulama anahtarı sensör verileri ile beraber gönderilecek pakette yer alır.

4.2.1. İstemci Tarafında Paket Oluşturma

Ortadaki adam saldırısının tespit edilmesi amacıyla istemci tarafında sensör verilerinden oluşan bir veri paketi hazırlanacaktır. Önerilen yöntem ile hazırlanan veri paketi ağ üzerinden sunucuya gönderilecektir. Bu amaçla sensör verilerinin oluşturulması, IP adresini kullanarak başlangıç anahtarının üretilmesi, doğrulama anahtarı oluşturulması ve son olarak veri paketinin hazırlanıp gönderilme süreçleri gerçekleştirilir. Bu çalışmada, doğrulama anahtarını oluşturmanın ilk adımı olarak istemci IP adresi kullanılmıştır. Şekil 4.7.'de önerilen algoritma adımları gösterilmiştir:

İstemci Tarafında Paket Oluşturma

1. Adım: Sensör verilerini al ve numaralandırarak sırala.

2. Adım: İstemci IP adresinden başlangıç anahtarını üret.

3. Adım: Sensör verileri ve anahtar sayıyı kullanarak doğrulama anahtarını oluştur.

4. Adım: Tüm veriler ile doğrulama anahtarından bir TCP paketi oluştur ve sunucuya gönder.

Şekil 4.7. Önerilen yöntemin istemci tarafındaki algoritması.

4.2.1.1. Sensör Verilerini Alma

Sensör, fiziksel bir niceliği, örneğin sıcaklık, basınç, kuvvet, hız, ivme veya aydınlık şiddeti, tespit eden ve ölçüp değerini elektrik sinyali olarak veren cihazdır. Sensör tarafından ölçülen ilgili nicelik daha sonra istemciye bağlı bir veri toplama kartı tarafından, Şekil 4.8.'de görüldüğü gibi, sayısal bilgi haline getirilir.

Bu çalışmada kullanılacak sensör verileri Eclipse ortamında, tanımlı aralıklar içinde, üretilen sayı dizileridir. Rastgele sayı üretimi için JAVA'nın `random` sınıfını kullanılarak bir uygulama hazırlanmıştır. İstemci bilgisayardaki uygulama ekranına gelen değerler tanımlı aralıkta rasgele tamsayı üretimi gerçekleştirmektedir. Bu amaçla hazırlanan JAVA kodu Şekil 4.8.'de verilmiştir.

```
import java.util.Random;
Random rastGeleVeriUret=new Random();
int sensorDegeri=rastGeleVeriUret.nextInt(n);
```

Şekil 4.8. JAVA ile rastgele değer üretme.

Burada parantez içerisine verilen `n` rakamı ile Random fonksiyonu tarafından 1'den `n`'e kadar (`n` hariç) tam sayı değerler üretilir. Yukardaki kod ile üretilen sayısal değerlerin bir örneği `n=5` için Çizelge 4.1.'de gösterilmiştir. Gösterilen şekilde üretilen sensör verileri işleme alınmış ve sensörlerden geldiği kabul edilerek yöntemimizde kullanılmıştır.

Çizelge 4.1. Rastgele üretilen sensör verileri.

Sensör 1	Sensör 2	Sensör 3	Sensör 4	Sensör 5
4	2	1	1	3

Önerilen yöntemde doğrulama anahtarının oluşturulması amacıyla gelen sensör verileri logaritmik bir fonksiyonla işleme girmektedir. Ancak gelen sensör verileri arasında negatif veya 0 değer bulunabilir. Ancak negatif sayılar ile 0 değerinin logaritması alınamaz. Dolayısıyla, sahadan gelen sensör verileri arasında negatif ya da 0 bulunması problemine karşı yöntem şu şekilde geliştirilmiştir: Gelen sensör verilerinin önce mutlak

değeri alınarak sistemde negatif sensör değerlerinin önerilen yöntemdeki olumsuz etkisi giderilmiştir. Ayrıca logaritma 10 tabanında 0 durumunu çözmek için her bir sensör verisi doğrulama anahtarını oluşturmak için kullanılmadan önce kendi sıra numarası nispetinde değeri arttırılarak yeni bir değer üretilir. Örnek bir sensör değerinin olası olumsuz etkisinin yöntemimiz tarafından önlenmesi Şekil 4.9.'da yer alan program kodunda gösterilmektedir.

```
import java.lang.*;

for (int i = 1; i < SensörSayisi; i++)
{
double x= sensordizisi[i];

x= Math.abs(x); //sensor degerinin mutlak degeri alınması

x=x+i; //sensor degerine dizi numarasının eklenmesi

Math(log10(x));
}
```

Şekil 4.9. Sensör verisinin doğrulama anahtarı için uygun hale getirilmesi

4.2.1.2. Başlangıç Anahtarının Oluşturulması

Önerilen yöntemde istemci, iletişimin başladığını bildiren taraftır. Sunucu tarafı ise pasif olarak bekleyen ve istemciden gelen verileri karşılayan taraftır. İstemci sunucu ile iletişim kurmak için sunucuda çalışan programın port ve IP adreslerini bilmek zorundadır. Yazılım gereği karşılıklı olarak istemci tarafına sunucunun, sunucu tarafına istemcinin IP adresi ve port bilgileri bildirilir. Böylece karşılıklı haberleşme başlar.

Önerilen yöntemde doğrulama anahtarını oluşturmak için ilk adım başlangıç anahtarının üretilmesidir. Bu değer için istemci IP adresi seçilmiştir. İstemci bilgisayarın IP adresinin numaralarını toplayarak başlangıç anahtarının değeri `(q)` elde edilir. Örnek bir IP adresi ile rastgele değer oluşturulması aşağıda görülmektedir:

```
İstemci IP Adresi: 10.1.10.39
IP Adresi Sayılar Toplamı = 10+1+10+39 = 60
Başlangıç anahtarı: q = 60
```

İlk oluşturulan başlangıç anahtarı önerilen algoritma ile her seferinde tekrarlı bir döngüye girerek en son doğrulama anahtarı elde edilir.

4.2.1.3. Doğrulama Anahtarı Oluşturma

Önerilen yöntemde verilerinin sunucu tarafına gönderildiğinde doğrulanması amacıyla, beraberinde gönderilecek doğrulama anahtarı (IX_{key}) öncelikle istemci tarafında oluşturulur. Doğrulama anahtarı, başlangıç anahtarı olan q değerinin işleme alarak başlar. Yöntemin geri kalanındaki doğrulama anahtarı oluşturmak için kullanılacak algoritma tamamen kullanıcıya özgü seçilebilmektedir. Bu çalışmada, logaritmik bir fonksiyon kullanılarak sensörden gelen veriler başlangıç anahtarı ile bir döngüye girmektedir. Başlangıç anahtarı ile ilk sensör verisinin logaritmasının çarpılması ile elde edilen sonuç bir sonraki döngüde kullanılacak başlangıç anahtarını oluşturur. Her döngü sonucunda elde edilen bu değer bir sonraki döngüye son sensör verisine kadar aktarılır. Şekli 4.10.'da gösterildiği gibi son döngünün sonucu doğrulama anahtarı olarak belirlenir.

```
int i; double q;
q = Sum(istemci IP adres);
for ( i=1; i<=sensör sayisi; i++)
{
     q= q * log S(i);
}
print q;
```

Şekil 4.10. Doğrulama anahtarının oluşturulması.

Buna örnek olarak Çizelge 4.2.'de verilen sensör değerleri 127.0.0.1 IP numaraları istemciden gönderilmek üzere düşünüldüğünde elde edilecek doğrulama anahtarı Şekil 4.11.'deki gibi olacaktır.

Çizelge 4.2. Üretilen sensör verileri.

Sensör 1	Sensör 2	Sensör 3	Sensör 4	Sensör 5
4	7	9	10	6

```
İstemci IP adresi: 127.0.0.1
q = 127 + 0 + 0 + 1 = 128 (Başlangıç anahtarı)
1. Adım: 128*log(4)=77.06367888997917
2. Adım: 77.06367888997917*log(7)=65.12636398620945
3. Adım: 65.12636398620945*log(9)=62.146345000859384
4. Adım: 62.146345000859384*log(10)=62.146345000859384
5. Adım: 62.146345000859384*log(6)=48.35925606919202
```

Şekil 4.11. Örnek doğrulama anahtarının oluşturulması.

127.0.0.1 IP adresli istemciden gelen Çizelge 4.2.'deki sensör verileri ile birlikte gönderilecek olan doğrulama anahtarı IX_{key} = 48.35925606919202 olarak elde edilmiştir.

4.2.1.4. Paket Oluşturma ve Gönderme

Sunucuya gönderilmek üzere hazırlanan veri paketi içerisinde başlangıç anahtarının değeri, sensör verileri, istemci tarafında üretilen doğrulama anahtarı, $IX_{key,}$ değeri bulunmaktadır. Çizelge 4.3.'te örnek değerler sırasıyla gösterilmektedir.

Çizelge 4.3. İstemciden sunucuya gönderilen örnek paketler.

Başlangıç Anahtarı	Sensör 1	Sensör 2	Sensör 3	Sensör 4	Sensör 5	Doğrulama Anahtarı (IX_{key})
128	4	7	9	10	6	48.35925606919202
128	2	6	10	7	9	24.17962803459602
128	8	9	4	2	3	9.53845555606189
128	6	5	4	3	10	19.998667505961542
128	8	2	4	5	7	12.375312381285116

4.2.2. Sunucu Tarafında Veri Doğrulama

Sunucu tarafında yapılan doğrulama, önerilen yöntem doğrultusunda iletilen ağ paketinin kontrolü ve bozulmuş veri paketinin tespiti amaçlıdır. Bu süreçte sırasıyla gelen veri paketi açılacak, paketten başlangıç anahtarının değeri alınacak, gelen sensör verileri

ile sunucu doğrulama anahtarı oluşturulacak, oluşturulan sunucu anahtarı ile istemciden gelen paket içindeki doğrulama anahtarı karşılaştırılacak ve durum tespiti yapılacaktır. Eğer karşılaştırılan anahtarlar eşit ise gelen sensör verileri işleme alınacaktır. Doğrulama anahtarları faklı ise alarm verilecektir. Şekil 4.12.'de önerilen yöntemin sunucu tarafındaki algoritması görülmektedir.

```
Sunucu Tarafında Veri Doğrulama

1.Adım: Gelen veri paketini aç.
2.Adım: İstemci tarafından gelen başlangıç anahtarını ve
        sensör verilerini sırasıyla al.
3.Adım: Doğrulama anahtarını oluştur.
4.Adım: Sunucu tarafında oluşturulan doğrulama anahtarı ile
        istemciden gelen paket içindeki doğrulama anahtarını
        karşılaştır.
5.Adım: Doğrulama anahtarları aynı ise gelen sensör
        verilerini işleme al farklı ise alarm ver.
```

Şekil 4.12. Önerilen yöntemin sunucu tarafındaki algoritması.

4.2.2.1. Gelen Veri Paketi

Sunucu tarafındaki ilk işlem gelen veri paketini açarak değerlendirmesidir. Paket içinde sırasıyla istemci tarafında üretilen rastgele değer, sahadaki sensör noktalarına göre dizilmiş sensör veri değerleri ve istemci tarafında oluşturulan doğrulama anahtarı bulunmaktadır. Gelen veri paketlerini incelemek amacıyla örnek değerler Çizelge 4.4.'te gösterilmektedir.

Çizelge 4.4. Sunucuya gelen veri paketi.

Başlangıç Anahtarı	Sensör 1	Sensör 2	Sensör 3	Sensör 4	Sensör 5	Doğrulama Anahtarı ($IX_{key)}$
128	10	2	4	6	4	10.86834203787514
128	9	9	7	5	8	62.17619615501513
128	9	6	9	9	2	26.053138021196855
128	6	8	3	9	10	40.953642963104805
128	5	8	6	4	9	36.12119275563002

4.2.2.2. Başlangıç Anahtarının Alınması

Önerilen yöntemin sunucu tarafı, iletişimin son ve doğrulanma noktasıdır. Aynı zamanda, pasif olarak bekleyen ve istemciden gelen verileri karşılayan taraftır. Ancak gelen veri endüstriyel saha senaryosu gereği değerlendirilir. Bu değerlendirme süreci istemciden gelen paketteki başlangıç anahtarı ile başlar. Başlangıç anahtarı istemci tarafındaki bir değişkenden, IP adresinden, üretilen bir sayıdır. Önerilen yöntem ile doğrulama anahtarını oluşturmak için paketin başında yer alan başlangıç anahtarı ile geri kalan sensör verileri sırasıyla işleme alınır.

4.2.2.3. Doğrulama Anahtarının Oluşturulması

Sensör sayısı arttıkça, sensörlerden üretilen ve bir noktada toplanarak gönderilen verilerin anlık olarak doğrulanması donanım açısından ciddi performans gerektiren bir iş yüküdür. Güvenlik seviyesi yüksek ağ mimarileri içerisinde anlık olarak gelen veri paketleri doğrulama yapılmadan kullanılamaz. Dolayısıyla doğrulama işleminin en az iş yükü ile yapılması gelen verinin hızlı bir şekilde işleme alınması açısından oldukça önemlidir. Önerilen yöntemde, istemci tarafındaki doğrulama anahtarının algoritması sunucu tarafında da çalıştırılarak gelen verilerden doğrulama anahtarı (SX_{key}) tekrar üretilir. Çizelge 4.5.'de örnek bir veri paketi açılmıştır. Sensör verileri ile yöntem doğrultusunda işlemler gerçekleştirilmiştir.

Çizelge 4.5. Gelen veri paketi.

Başlangıç Anahtarı	Sensör 1	Sensör 2	Sensör 3	Sensör 4	Sensör 5	Doğrulama Anahtarı (IX_{key})
215	6	3	2	10	4	10.112052659453719

4.2.2.4. Doğrulama Anahtarlarının Karşılaştırılması

Önerilen yöntemde karşılıklı iki doğrulama anahtarı karşılaştırılması yapılır. İstemci tarafından gönderilen paketteki doğrulama anahtarı, IX_{key}, ile sunucu tarafında, gelen verilerden oluşturulan sunucu tarafındaki doğrulama anahtarı, SX_{key}, ile karşılaştırma yapılarak doğrulama işlemi gerçekleştirilir. Bu bakımdan, doğrulama anahtarının oluşturan algoritmanın sonucunun gerçel bir sayı olması, kendisini oluşturan verideki en ufak değişikliğin sonuca yansımasına neden olacaktır. Bunun sonucu olarak gelen verinin doğrulanmasındaki hassasiyet yüksek seviyede tutulmuş olur.

Doğrulama sürecini uygulamak için yapılan işlem bu noktaya kadar adım adım istemci tarafında yapılan işlemle aynıdır. En son üretilen doğrulama anahtarı istemciden gelen paketin doğrulama anahtarı ile karşılaştırılır. Doğrulama anahtarı eşleşirse gelen sensör verileri işleme alınır. Ancak doğrulama anahtarlarında uyuşmazlık varsa alarm vererek uyarıda bulunur.

Çizelge 4.6.'da bu amaçla oluşturulmuş senaryo gösterilmektedir. Bu senaryoda istemci tarafından gelen orijinal ve bozulmuş iki veri paketi gösterilmiştir. İlk satırda verilen orijinal sensör değerlerinden elde edilen doğrulama anahtarı ile paket içinden çıkan doğrulama ile eşleştiğinden bu pakette doğrulama gerçekleştirilmiş olup işleme konulur.

Çizelge 4.6. Gelen veri paketinin bozuk veri paketi ile karşılaştırması.

Paket Türü	Başlangıç anahtarı	X_1	X_2	X_3	X_4	X_5	Doğrulama Anahtarı (X_{key})
Gelen Veri	215	6	3	2	5	4	10.112052659453719
Bozuk Veri	215	6	3	2	10	4	10.112052659453719

İkinci satırda ise saldırıya uğramış bir veri paketi görülmektedir. Dördüncü sensör değeri 5 değerine sahipken 10 olarak değiştirilmiştir. Netice itibariyle, yeni verilerden

elde edilecek sunucu tarafındaki doğrulama anahtarı paketten çıkan doğrulama anahtarı ile aynı olmayacaktır. Bu durumda konsola alarm olarak yansıtılır ve bu paket işleme alınmaz. Ancak her ne kadar pakette değişiklik algılansa da paketin hangi verisinin saldırıya uğradığı tespit edilmez.

Sistemdeki sunucuyu aldatmak için doğrulama anahtarı aynen gönderilmiştir. Ancak sunucu kendi doğrulama anahtarını oluşturarak gelen veri paketinin bozulmuş olduğunu tespit eder. Kontrol veri bazında adım adım yapılır. Yukardaki tabloya göre istemci tarafındaki işlemler Şekil 4.13.'de gösterilmiştir.

```
İstemci IP adresi: 10.1.65.139
q = 10 + 1 + 65 + 139 = 215 (Başlangıç anahtarı)
1. Adım: 215 *log(6)=167.30251883248334
2. Adım: 167.30251883248334*log(3)=79.82358770311438
3. Adım: 79.823587703114380*log(2)=24.029294260151943
4. Adım: 24.029294260151943*log(5)=16.795755913209874
5. Adım: 16.795755913209874*log(4)=10.112052659453719
```

Şekil 4.13. Sunucu tarafı doğrulama anahtarının oluşturulması (ilk paket)

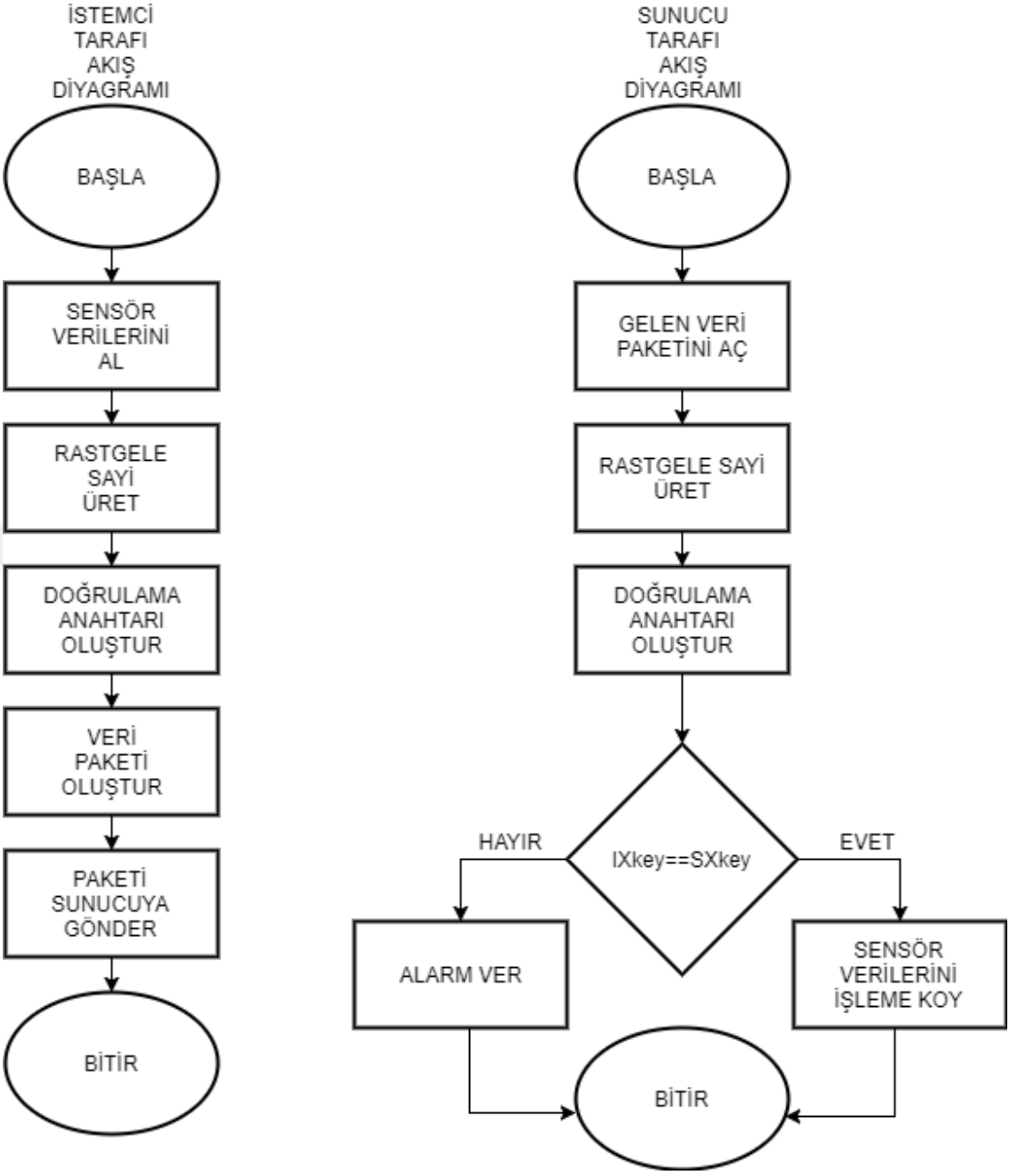
Sunucu tarafında elde edilen doğrulama anahtarı, SX_{key} = 10.112052659453719 istemci tarafından gönderilen IX_{key} ile aynı olduğundan Şekil 4.14.'deki ilk paketin doğru olduğu tespit edilmiştir. Aynı işlemler ikinci paket için yürütüldüğünde doğrulama anahtarı aşağıdaki gibi elde edilir.

```
İstemci IP adresi: 10.1.65.139
q = 10 + 1 + 65 + 139 = 215 (Başlangıç anahtarı)
1. Adım: 215 *log(6)=167.30251883248334
2. Adım: 167.30251883248334*log(3)=79.82358770311438
3. Adım: 79.823587703114380*log(2)=24.029294260151943
4. Adım: 24.029294260151943*log(10)=24.029294260151943
5. Adım: 24.029294260151943*log(4)=14.467076693884133
```

Şekil 4.14. Sunucu tarafı doğrulama anahtarının oluşturulması (ikinci paket)

Saldırıya uğrayan paketin sunucu tarafında incelenmesi sonucu elde edilen doğrulama anahtarı, SX_{key}= 14.467076693884133 istemciden gelen pakette yer alan doğrulama anahtarı ile IX_{key} = 10.112052659453719 değeri aynı olmadığından bu paketin saldırıya uğradığı anlaşılmaktadır.

Bozulmuş veri tespiti, sunucu tarafında oluşturulan doğrulama anahtarı ile istemci tarafından gelen doğrulama anahtarının karşılaştırılması sonucu yapılır. Eğer sunucu tarafı doğrulama anahtarı, SX_{key}, istemci tarafındaki doğrulama anahtarından, IX_{key}, farklı ise alarm verilir. Böylece gelen veriler arasında bozulmuş bir veri olduğu anlaşılmaktadır. Doğrulama anahtarı değerleri eşit ise istemciden gelen sensör verileri işleme alınmasında sakınca yoktur. Şekil 4.15.'de önerilen yöntemin istemci ve sunucu tarafındaki akış diyagramları gösterilmektedir.



Şekil 4.15. İstemci ve sunucu tarafındaki önerilen yöntemin akış diyagramları.

5. ARAŞTIRMA BULGULARI VE TARTIŞMA

Endüstriyel sahada sensörlerdeki veriyi bir ağ üzerinden iletirken iletilen veri ile alınan verinin bozulmamış olduğu doğrulanmalıdır. Veriler iletim hattında bozunuma uğrarsa bunun fark edilmesi ve alarm verilerek uyarı yapılması gerekir. Önerilen yöntem, özellikle iletişim kanallarında sıkça rastlanan ortadaki adam saldırısına yöneliktir. Yöntem saldırı sonucu olası veri bozulmasına karşı doğrulama sağlamak amacıyla tasarlanmıştır ve test edilmiştir. Ayrıca bu yöntemde teslim edilen mesajların tutarlılığının hızlı ve güvenli biçimde gerçekleşmesi, doğrulamanın anlık seviyede yapılması açısından önem taşımaktadır. Önerilen yöntem iletilen verinin içeriğinin gizli olmadığı varsayarak bütünlüğünü hızlı bir şeklide doğrulamayı amaçlamaktadır. Bu amaç çerçevesinde, gözlem verileri gibi içeriğinin görülmesinin sakıncası olmayan verilerin doğrulanmasında rahatlıkla tercih edilebilir. Diğer yandan, anlık seviyede aktarılan verinin hız performansı da önem taşımaktadır. Bu yüzden, işlem süresi açısından da değerlendirilip milisaniye seviyesinde tespitler yapılmıştır.

Doğrulama anahtarı oluşturmanın esnek yapısı sebebiyle kullanıcı tarafından farklı fonksiyonlar bu döngüsel yapı içerisinde kullanılabilir. Geliştirilen yöntemde doğrulama algoritmasında tercih edilen logaritmik fonksiyon döngüsel kodun temelini oluşturur. Veri paketine doğrulama amaçlı bir doğrulama anahtarı eklenerek mesajların doğruluğu kontrol edilmiştir.

Önerilen yöntem ile birlikte veri iletiminin hızlılığı ve veri içeriğinin bütünlüğünün bir arada test edilmiştir. Bu amaçla günümüzde kullanılan mesaj özütü fonksiyonlarından MD5, SHA-1 ve SHA-256 veri üretim süreleri önerilen yöntem ile karşılaştırılmıştır.

5.1. Hız Performans Değerlendirmesi

Önerilen yöntem ve karşılaştırılan diğer yöntemler Eclipse ortamında JAVA programlama dili ile kullanılarak yapılmıştır. Veri iletimi için algoritma işlemleri adım adım uygulanarak ortaya çıkan doğrulama değerlerinin hız performansı ortaya konulmuştur ve hız performansı açısından karşılaştırmalar yapılmıştır.

Algoritmalar aynı boyuttaki sensör verileri ile anlık olarak iletiminde harcanan süreler ile ortalama süre ölçülmüştür. Bu amaçla, aynı veri türünde ve hafızada aynı kapasitede yer kaplayan çeşitli sayılardaki sensör verilerinin istemci-sunucu mimarisinde aynı donanım kullanılarak iletimi gerçekleştirilmiştir.

Algoritmalarının doğrulama değerlerinin oluşturulma süresinin karşılaştırılması hız performans göstergesi açısından önem taşımaktadır. Dolayısıyla, istemci tarafında doğrulama anahtarının da yer aldığı TCP paketinin oluşturulması için her bir algoritmanın hız performansı incelenmiştir. Bu amaç doğrultusunda, 10, 100 ve 500 sensör verisi için sırasıyla karşılaştırma yapılarak algoritmalar denenmiş ve hızları milisaniye (ms) seviyesinde ölçülmüştür. Çizelge 5.1., 5.2. ve 5.3.'te sırasıyla 10, 100, 500 sensör verisi için algoritmaların çalışma süreleri 5 farklı zamanda ölçülmüş ve ortalama değerleri verilmiştir.

Çizelge 5.1. 10 sensör verisi için hız performans değerlendirmesi.

Algoritmalar	#1.(ms)	#2.(ms)	#3.(ms)	#4.(ms)	#5.(ms)	10 sensör verisi için ortalama süre (ms)
MD5	11,82	10,17	12,64	12,52	11,84	11,80
SHA-1	11,39	15,00	13,71	12,83	13,20	13,22
SHA-256	12,22	14,57	12,91	17,41	13,01	14,02
Önerilen Yöntem	1,73	1,91	1,98	1,69	2,21	1,90

Yapılan ilk çalışmada 10 sensör verisi istemci bilgisayar üzerinde uygulanmıştır. Herbir algoritmayla hazırlanan TCP paketi için harcanan süre milisaniye seviyesindedir. Ölçülen en düşük süre önerilen yöntem kullanılarak 1,73 ms ile elde edilirken diğer algoritmalarda 10,17 ile 17,41 ms arasında değişen değerler arasında değerler elde edilmiştir.

Çizelge 5.2. 100 sensör verisi için hız performans değerlendirmesi.

Algoritmalar	#1.(ms)	#2.(ms)	#3.(ms)	#4.(ms)	#5.(ms)	100 sensör verisi için ortalama süre (ms)
MD5	12,39	12,75	13,93	12,58	13,73	13,07
SHA-1	13,37	13,78	15,78	14,60	15,70	14,64
SHA-256	14,92	17,71	16,47	16,39	16,21	16,34
Önerilen Yöntem	2,18	2,22	2,73	2,16	1,91	2,24

Çizelge 5.1.'deki değerlerle karşılaştırıldığında, Çizelge 5.2.'deki değerlerde sensör veri sayısına göre artışlar görülmektedir. Ancak SHA-256'in ortalama değerindeki artış diğerlerine göre daha fazladır. Ancak önerilen yöntemde artış görülmesine rağmen bu süre diğerlerine göre çok daha düşük seviyelerde ölçülmüştür.

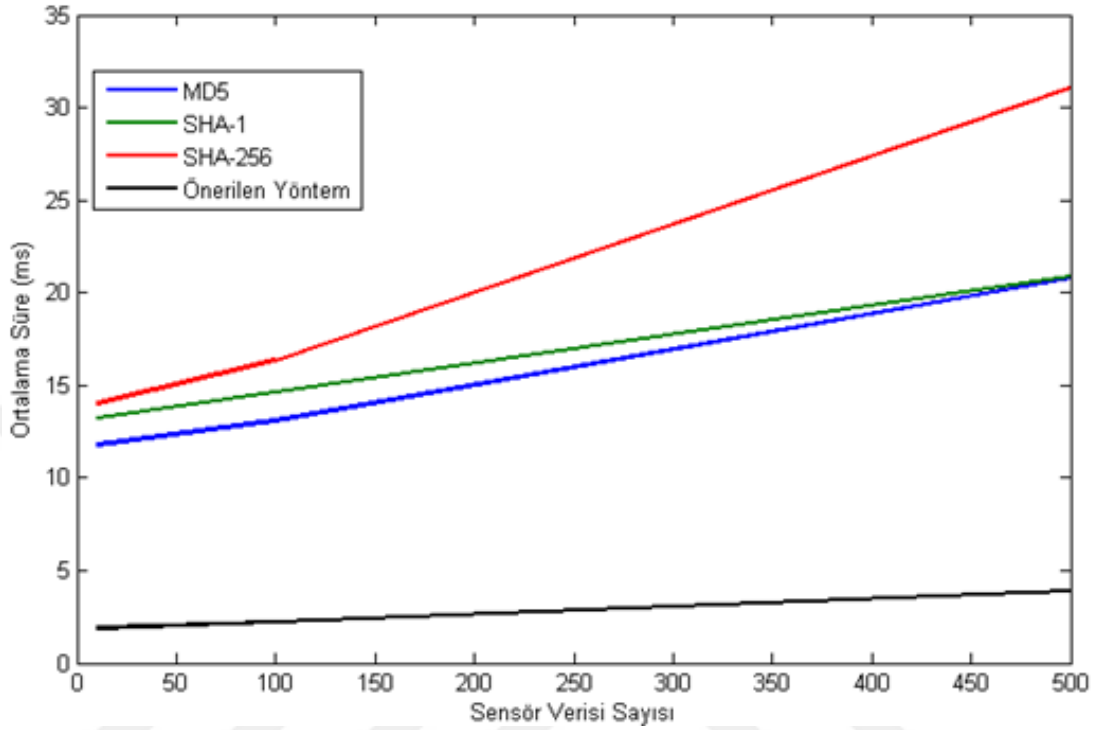
Çizelge 5.3. 500 sensör verisi için hız performans değerlendirmesi.

Algoritmalar	#1.(ms)	#2.(ms)	#3.(ms)	#4.(ms)	#5.(ms)	500 sensör verisi için ortalama süre (ms)
MD5	22,01	22,28	18,12	18,60	19,69	20,14
SHA-1	20,11	20,28	21,76	22,64	19,71	20,90
SHA-256	29,24	32,95	33,16	28,77	31,18	31,06
Önerilen Yöntem	3,34	4,95	3,08	3,44	4,77	3,92

Yapılan çalışmada sensör veri sayısı 500'e yükseltildiğinde değişik oranlarda artış göstermiştir. MD5 algoritması yaklaşık %54 artış gösterirken, SHA-1 algoritması yaklaşık %42 artış göstermiştir. SHA-256 algoritmasında ise yaklaşık %90 olan artış diğer algoritmalardan oldukça fazla gerçekleşmiştir. Önerilen yöntemde ise yaklaşık %64 gibi bir artış oranı gerçekleşse de diğer yöntemlere göre harcadığı süre çok daha düşüktür.

10, 100 ve 500 sensör verisi için yapılan er çalışmanın sonucu elde edilen süreler bir grafik altında toplanıp Şekil 5.1.'de gösterilmiştir. TCP paketi için işleme alınan

sensör verisinin sayısının artması önerilen yöntemin hız bakımından iyi olan performansını düşürmediğini göstermiştir.



Şekil 5.1. Algoritmalarda sensör verisi sayılarının hız performansına etkisi.

5.2. İstemci Sunucu Mimarisinde Bozuk Paket Tespiti

İstemciden sunucuya yapılan anlık iletim sonucu gelen paketler arasındaki bozulmuş bir paketin tespiti anlık olarak yapılmış ve alarm verilerek uyarı yapılmıştır. Bu konuda yapılan çalışma sunucu tarafındaki doğrulama anahtarlarının karşılaştırılması sonucu negatif çıktığı takdirde Şekil 4.6.'da görüldüğü gibi alarm verilmektedir.

Logaritma tabanlı işlemlerle tasarlanan önerilen yöntemde, istemci tarafından alınan bir değişkenle başlayan ve her bir sensör verisinin sırayla işleme alınmasıyla elde edilen gerçel sayı tipindeki doğrulama anahtarının en büyük üstünlüğü, gelen paket verisindeki bir değişikliğin kolaylıkla tespit edilebilmesidir. Dolayısıyla, gelen doğrulama anahtarı ile sunucu tarafındaki fark çok küçük oranlarda olsa bile tespit edilebilir niteliktedir.

6. SONUÇ VE ÖNERİLER

Nesnelerin interneti teknolojilerinin endüstriyel ortamda aktif bir şekilde kullanılmasıyla birlikte gelen en büyük sorunlardan biri veri iletiminde bütünlüğün hızlı bir şekilde gerçekleştirilmesidir. Bu tezde ele alınan problem ise çok büyük veri paketleri dahil olmak üzere, özellikle endüstriyel ortamda karşılaşılabilecek ortadaki adam saldırısına karşı, veri paketlerinin veri bütünlüğünü sağlayarak en hızlı şekilde tespit edilmesidir. Önerilen yöntem bir istemci, bir sunucu ve aralarındaki iletimi sağlamak açısından bir ağ anahtarı ile gerçekleştirilmiştir. Anlık olarak gelen veri paketinin doğrulama anahtarı ile sunucu tarafında elde edilen doğrulama anahtarı karşılaştırılmış ve varsa bozulmuş veri paketleri tespit edilmiştir.

Sensör verisinin hızlı iletilmesi ve anlık olarak kontrol edilmesi için ağ güvenlik sistemi bir bütün olarak tasarlanır. Bundan dolayı ağ sistemlerinin güvenliğini sağlamak için saldırı tespit sistemi, güvenlik duvarı, veri doğrulama işlemleri ve veri bütünlük kontrolünü sağlayan protokoller uygun mimari içerisinde kullanılmaktadır. Bu teknolojilerin esas kullanım amacı dışardan gelebilecek saldırılara yönelik tehditlerdir. İnternet üzerinden gelebilecek ortadaki adam saldırısı benzeri tehditler, nesnelerin interneti teknolojisinde sensörler üzerinden iletişim gerçekleştiren sistemleri büyük zarara uğratabilir. Önerilen yöntem uygun ağ mimarisi içerisinde saldırı tespit sisteminin bir parçası olarak, ortadaki adam saldırısının sensör veri iletimindeki paketleri yakalayıp bozması sonucu yanlış tespite yol açmasını engellemektedir.

Endüstriyel sahada kullanılacak nesnelerin interneti teknolojisi sensör ağı oluşturarak diğer cihazlarla bir iletişim mekanizması sağlamaktadır. İletişim ağ cihazları üzerinden OSI referans modelinin 4. katmanı olan taşıma katmanında TCP protokolü üzerinden gerçekleşir. Hazırlanan veri paketleri anlık olarak gözlem yapılan merkezi sunuculara iletilir. Gözlem merkezinde bozulmuş bir veri paketinin dikkate alınmaması endüstriyel sahada tehlikeli bir durum ortaya çıkarır.

Önerilen yöntemde doğrulama anahtarı oluşturarak istemci tarafında sensörlerden alınan verinin TCP paket haline getirilip iletilmesi gerçekleştirilmiştir. TCP protokolü, kayıpsız veri gönderimi sağlayabilmek için kullanılan protokoldür. Her veri paketinin adım adım değerlendirildiği ve geliştirildiği yöntemde, iletilen paketler veri iletim anında

değişikliğe uğraması durumunda anlık olarak tespit edilerek alarm verilmiştir. Önerilen yöntem ile ortadaki adam saldırısına yönelik bir çözüm geliştirilmiştir.

Veriler bir noktadan başka bir noktaya iletilirken saldırıya açık hale gelebilmektedir. OSI referans modeli içerisinde 2. katman olan veri bağlantı katmanının kullandığı cihaz olan ağ anahtarının saldırıya uğrayarak yanlış yönlendirilmesi verinin saldırgan tarafından bozulmasına sebep olmaktadır. Bu durumu geliştirilen algoritmalar ya da yöntemler ile önlemek gerekmektedir.

Güvenlik duvarları yerel alan ağımız ile dış ağ olan internet arasında güvenlik sağlamaktadır. Bu iki ağ arasında tüm trafik güvenlik duvarı tarafından incelenir. Ancak güvenlik duvarları tek başına yerel alan ağımızı korumada yetersiz kalır. Güvenlik duvarları iç ağa yani yerel alan ağına veri paketi girişi sağladıktan sonra paket seviyesinde zararsız görünen veri paketlerini kabul eder ve bu paketlere karşı tespit yapamaz. Güvenlik duvarı teknolojisi paket içeriği seviyesinde tespit yapamamaktadır. Önerilen yöntem ile bu duruma bir çözüm elde edilmiştir. Bu tür bozulmuş veri paketi durumlarına karşı doğrulama anahtarları üzerinden doğrulama yapılır. Doğrulama anahtarları karşılaştırma sonucu farklılık varsa bozulmuş veri paketi tespit edilerek sistemde uyarı verilir.

İnternete bağlı cihaz sayısının artışını özellikle son yıllarda günden güne artmaktadır. Endüstri 4.0 ile sensörlerin tamamının internete bağlanması amaçlanmaktadır. Günlük yaşantıda kullanılan çeşitli cihazların internete bağlanarak iletişim kurmasıyla başlayan değişim, gelişen bu teknolojinin endüstriyel alanda da kullanılmasına yol açmıştır. Bu açıdan nesnelerin interneti kavramı endüstriyel sahada farklı çözümler ortaya koymaktadır.

Ancak bu kolaylık bazı problemleri de beraberinde getirmiştir. İletişim güvenliği ile bant genişliği problemleri bu teknolojinin önündeki engellerden bazılarıdır. Internete bağlanan cihazların artmasıyla orantılı olarak saldırıya maruz kalacak veri trafiğin de artmaktadır. Dolayısıyla, güvenlik tedbirlerinin de gelişen bu yapıya uygun olarak alınması gerekmektedir. Ayrıca, sensör sayısının artması ağ bant genişliğini daraltabilir. Böylece, IoT teknolojisi kullanan cihazlar erişilemez hale gelebilir.

Bu tespitler çerçevesinde önerilen yöntem üzerinden sensör veri iletimi geliştirilen yöntemler ile daha hızlı ve daha güvenli iletilebilmektedir. İletim anlık olarak kontrol edilmektedir. Önerilen yöntem karmaşık ağ sistemleri içerisinde endüstriyel saha

cihazlarında sensör veri iletimi için kullanılabilir. Güvenlik protokollerinin geliştirilmesi, güvenlik duvarı içerisinde kullanılan yöntemler, saldırı tespit sistemleri içerinde kullanılan özellikler önerilen yöntemin uygulanabileceği alanlara örnek olarak verilebilir.

Geliştirilen yöntem ile veri bütünlüğünün hızlı iletimi önemsenmiştir. Veri iletimi için önerilen yöntem uygulanarak ortaya çıkan doğrulama değerlerinin hız performansı ortaya konulmuştur ve hız performansı açısından mesaj özütü fonksiyonları ile karşılaştırmalar yapılmıştır. Karşılaştırmalar sensör sayısı üzerinde her algoritma için ayrı ayrı değerlendirilmiştir. Sensör sayısındaki artış hız performansını doğal olarak doğrudan etkilemiştir. Önerilen yöntem içinde sensör sayısı artışı sonucu hız performansında yavaşlama olmasına rağmen diğer algoritmalara göre yavaşlama performansı oldukça azdır. Ayrıca hız performansı olarak oldukça hızlı olduğunda dolayı sensör sayısına bağlı olarak ortaya çıkan bu yavaşlama veri iletim sistemini en az seviyede etkilemektedir.

KAYNAKÇA

AbdAllah, E. G., Hassanein, H. S. ve Zulkernine, M. 2015. A survey of security attacks in information-centric networking. **IEEE Communications Surveys & Tutorials**, 17(3), 1441-1454.

Alaba, F. A., Othman, M., Hashem, I. A. T. ve Alotaibi, F., 2017. Internet of things Security: A Survey. **Journal of Network and Computer Applications**, 88, 10–28.

Balan, R. K., Lee, B. P., Kumar, K. R., Jacob, L., Seah, W. K. G. ve Ananda, A. L., 2002. TCP HACK: A mechanism to improve performance over lossy links**. Computer Networks**, 39(4), 347-361.

Bishop, M. A., 2005. Introduction to computer security. **Pearson Education**, USA.

Bishop, M. A., 2003. What is computer security. **IEEE Security & Privacy**, 67-69.

Bonomi, F., Milito, R., Zhu, J. ve Addepalli, S., 2012. Fog computing and its role in the internet of things. **In Proceedings of the first edition of the MCC workshop on Mobile cloud computing,** 13-16.

Borhade, S. R. ve Kahate, S. A., 2016. Intrusion Detection System Based On Hashing Technique. **Global Journal Of Engineering Science And Researches,** 3(6), 31-34.

Callegati, F., Cerroni, W. ve Ramilli, M., 2009. Man-in-the-Middle Attack to the HTTPS Protocol. **IEEE Security & Privacy**,7(1): 78-81.

Can, E., 2007. Gerçek zamanlı veriler yardımı ile karar veren bir bilgisayar ağı saldırı tespit sisteminin tasarlanması ve gerçeklenmesi. **Doktora Tezi**, YTÜ Fen Bilimleri Enstitüsü.

Canbek, G. ve Sağıroğlu, Ş., 2006. Bilgi, Bilgi Güvenliği ve Süreçleri Üzerine Bir İnceleme. **Politeknik Dergisi**, 9(3):69-72.

Chandia, R., Gonzalez, J., Kilpatrick, T., Papa, M. ve Shenoi, S., 2008. Security strategies for SCADA networks, **Critical Infrastructure Protection**, 117-131.

Clerck, J-P. D., 2017. The Internet of Things for beginners: IoT beginners guide 2017. https://www.i-scoop.eu/internet-of-things-guide/internet-things-beginners. 14.11.2017.

Cohen, F., 1987. A cryptographic checksum for integrity protection. **Computers & Security**, 6(6), 505-510.

Daemen, J., ve Rijmen, V., 2010. The first 10 years of advanced encryption. **IEEE Security & Privacy**, 8.6: 72-74.

Daya, B., 2013. Network security: History, importance, and future. **University of Florida Department of Electrical and Computer Engineering.**

Dieter, G., Computer security. **Wiley Interdisciplinary Reviews: Computational Statistics**, 2(5): 544-554.

Ding, D., Han, Q. L., Xiang, Y., Ge, X. ve Zhang, X. M., 2017. A Survey on Security Control and Attack Detection for Industrial Cyber-Physical Systems. **Neurocomputing**, 1-10.

Einwechter, N., 2017. An Introduction To Distributed Intrusion Detection Systems. http://online.securityfocus.com/infocus/1532, 16.11.2017.

Fovino, I. N., Carcano, A., Masera, M. ve Trombetta, A., 2009. An experimental investigation of malware attacks on SCADA systems. **International Journal of Critical Infrastructure Protection**, 2(4): 139-145.

Gubbia, J., Buyyab, R., Marusic, S. ve Palaniswamil, M., 2013. IoT: A vision, architectural elements, and future directions. **Future. Gener. Comput. System**, vol. 29,no. 7, 1645-1660.

Guha, R. K., Furqan, Z. ve Muhammad, S., 2007. Discovering man-in-the-middle attacks in authentication protocols. **In Military Communications Conference, MILCOM 2007**, **IEEE**, 1-7.

Huang, Y. J. ve Cohen, F., 1988. Some weak points of one fast cryptographic checksum algorithm and its improvement. **Computers & Security**, 7.5: 503-505.

Igure, V. M., Laughter, S. A. ve Williams, R. D., 2006. Security issues in SCADA networks. **Computers & Security,** 25(7): 498-506.

Jadidoleslamy, H., 2012. Weaknesses, Vulnerabilities and Elusion Strategies Against Intrusion Detection Systems. **International Journal of Computer Science and Engineering Survey,** 3(4): 15.

Kamel, N., Hamdy, N. ve Ahmed, S. H., 2005. A Proposed Intrusion Detection System for Encrypted Computer Networks. **Third International Conference on Informatics and Systems**, 19-22.

Katsikeas, S., Fysarakis, K., Miaoudakis, A., Van Bemten, A., Askoxylakis, I., Papaefstathiou, I. ve Plemenos, A., 2017. Lightweight & Secure industrial IoT communications via the MQ telemetry transport protocol. **IEEE Computers and Communications (ISCC) Symposium**, 1193-1200.

Kim, B. K., Kang, J. Y. ve Lee, D. H., 2016. A new hash algorithm exploiting triple-state bucket directory for flash storage devices. **IEEE Transactions on Consumer Electronics**, 62.4: 398-404.

Kim, H. ve Lee, E. A., 2017. Authentication and Authorization for the Internet of Things. **IT Professional**, 19(5): 27-33.

Klaus, C. W., 1999. Method and apparatus for detecting and identifying security vulnerabilities in an open network computer communication system. **U.S. Patent,** No.5:892-903.

Lee, J., Kapitanova, K., Son, S. H., 2010. The price of security in wireless sensor networks. **Computer Networks**, 54(17): 2967-2978.

Livshits, V. B. ve Lam, M. S., 2005. Finding Security Vulnerabilities in Java Applications with Static Analysis. **In USENIX Security Symposium**, Vol. 14, 18-18.

Madakam, S., Ramaswamy, R. ve Tripathi, S., 2015. Internet of Things (IoT): A literature review. **Journal of Computer and Communications**, 3: 164-173.

Metz, C., 1999. AAA protocols: authentication, authorization, and accounting for the Internet. **IEEE Internet Computing**, 3(6): 75-79.

Miyachi, T. ve Yamada, T., 2014. Current issues and challenges on cyber security for industrial automation and control systems. **IEEE in SICE Annual Conference (SICE)**, 821-826.

Nam, S. Y., Kim, D. ve Kim, J., 2010. Enhanced ARP: preventing ARP poisoning-based man-in-the-middle attacks. **IEEE communications letters**, 14(2):187-189.

Nicholson, A., Webber, S., Dyer, S., Patel, T. ve Janicke, H., 2012. SCADA security in the light of Cyber-Warfare. **Computers & Security,** 31(4): 418-436.

Oğuz, A., 2012. Kablosuz Duyarga Ağlarda Azami Veri Güvenliğini Sağlamak için Mimari Tasarım, **Doktora Tezi**, Trakya Üniversitesi.

Pawar, M. V. ve Anuradha, J., 2015. Network security and types of attacks in network. **Procedia ComputerScience**, 48: 503-506.

Prathima, E. G., Prakash, T. S., Venugopal, K. R., Iyengar, S. S., ve Patnaik, L. M., 2016. SADA: Secure approximate data aggregation in wireless sensor networks. **IEEE Data Science and Engineering (ICDSE)**, 1-6.

Puthal, D., Mohanty, S. P., Nanda, P. ve Choppali, U., 2017. Building Security Perimeters to Protect Network SystemsAgainst Cyber Threats [Future Directions]. **IEEE Consumer Electronics Magazine**, 6(4):24-27.

Ralston, P. A., Graham, J. H. ve Hieb, J. L., 2007. Cyber security risk assessment for SCADA and DCS networks. **ISA transactions**, 46(4): 583-594.

Ranathunga, D., Roughan, M., Nguyen, H., Kernick, P. ve Falkner, N., 2016. Case Studies of SCADA Firewall Configurations and the Implications for Best Practices. **IEEE Transactions on Network and Service Management,** 13(4): 871-884.

Royce, R., 2002. Distributed intrusion detection systems: An introduction and review. **SANS Reading Room**, GSEC Practical Assignment.

Sadotra, P. ve Sharma, C., 2016. A Survey: Intelligent Intrusion Detection System in Computer Security. **International Journal of Computer Applications**, 151(3):18-22.

Sadotra, P. ve Sharma, C., 2017. Intrusion Detection in Networks Security: A New Proposed Min-Min Algorithm. **International Journal of Advanced Research in Computer Science,** vol.8(3).

Schuett, C. D., 2014. Programmable logic controller modification attacks for use in detection analysis. **MSc Thesis**, Air Force Institute of Technology, Air University.

Shi, E. ve Perrig, A., 2004. Designing secure sensor networks. **IEEE Wireless Communications**, 11(6): 38-43.

Suresh, K. S. ve Prasad, K. V. 2012. Security issues and security algorithms in cloud computing. **International Journal of Advanced Research in Computer Science and Software Engineering,** 2(10), 110-114.

Tan, S., Li, X. ve Dong, Q., 2016. TrustR: An integrated router security framework for protecting computer networks**. IEEE Communications Letters**, 20(2): 376-379.

Tayeb, S., Latifi, S. ve Kim, Y., 2017. A survey on IoT communication and computation frameworks: An industrial perspective. **IEEE Computing and Communication Workshop and Conference (CCWC)**.

Tellez, M., El-Tawab, S. ve Heydari, H. M., 2016. Improving the security of wireless sensor networks in an IoT environmental monitoring system. **IEEE In Systems and Information Engineering Design Symposium (SIEDS**), 72-77.

Tyushev, K., Amelin, K. ve Andrievsky, B., 2016. The method of saving data integrity for decentralized network of group of UAV using quantized gossip algorithms. **IFAC-PapersOnLine**, 49.13: 259-264.

Varunkumar, K. A., Prabakaran, M., Kaurav, A., Chakkaravarthy, S. S., Thiyagarajan, S. ve Venkatesh, P., 2014. Various Database Attacks and its Prevention Techniques. **International Journal of Engineering Trends and Technology (IJETT)**, vol.9, no.11.

Wang, D., Jiang, Y., Song, H., He, F., Gu, M. ve Sun, J., 2017. Verification of implementations of cryptographic hash functions. **IEEE Access**, 7816 - 7825.

Weber, R. H., 2010. Internet of Things- New security and privacy challenges. **Computer Law&Security Review**, 26(1): 23-30.

Wool, A., 2004. A quantitative study of firewall configuration errors. **Computer,** 37(6): 62-67.

Cheswick, W. R., Bellovin, S. M. ve Rubin, A. D., 2003. Firewalls and Internet security: repelling the wily hacker. **Addison-Wesley Longman Publishing Co., Inc**..

Wu, J., Ota, K., Dong, M. ve Li, C., 2016. A hierarchical security framework for defending against sophisticated attacks on wireless sensor networks in smart cities. **IEEE Access**, 4: 416-424.

Xie, C., Gao, J. ve Tao, C., 2017. Big Data Validation Case Study. **IEEE Third International Conference on Big Data Computing Service and Applications**, 281-286.

Yaqoob, I., Ahmed, E., ur Rehman, M. H., Ahmed, A. I. A., Al-garadi, M. A., Imran, M. ve Guizani, M., 2017. The rise of ransomware and emerging security challenges in the Internet of Things. **Computer Networks**, 2(9): 1-15.

Yun, M., ve Yuxin, B., 2010. Research on the architecture and key technology of Internet of Things (IoT) applied on smart grid, 2010. Research on the architecture and key

technology of Internet of Things (IoT) applied on smart grid. **IEEE Advances in Energy Engineering (ICAEE) International Conference**, 69-72.

Zhang, B., Ma, X. X. ve Qin, Z. G., 2011. Security architecture on the trusting internet of things. **Journal of Electronic Science and Technology**, 9(4): 364-367.

ÖZGEÇMİŞ

Mustafa Kara (1989), 2013 yılında Beykent Üniversitesi Bilgisayar Mühendisliği bölümünden mezun oldu. Çalışma hayatına bazı özel şirketlerde, bilgisayar ve sistem güvenliği ile ilgili yazılım alanında 3 yıl kadar çalışarak devam etti. 2016 yılında Mustafa Kemal Üniversitesi'nde Öğretim Görevlisi olarak göreve başladı ve yine 2016 yılında yüksek lisansa İskenderun Teknik Üniversitesi'nde başladı. Araştırma konuları; bilgisayar ve sistem güvenliği, yazılım mühendisliği, endüstriyel robotlar, elektronik ve mobil elektronik imza ve halka açık anahtar altyapısı, kişisel ve kurumsal bilgi güvenliği ve ilgili alanlar.

EKLER

EK A

Önerilen Yöntemin İstemci Tarafı Kaynak Kodları

A.1. Kullanılan Kütüphaneler

```
import java.awt.BorderLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.Socket;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextField;
import com.mkclient.xkey.Xkey;
```

A.2. Sunucuyla Bağlantı Kurulması Metodu

```
private void initClient() throws IOException {
try {
socket = new Socket(getIpAddress(), getPortAddress());
}
catch (Exception e) {
System.out.println("Port Hatasi!");
JOptionPane.showMessageDialog(null,  "Sunucu  bilgilerinin
dogru oldugunda emin olunuz");
connectButton.setEnabled(true);
}
out = new PrintWriter (socket.getOutputStream(), true);
in=new                BufferedReader                (new
InputStreamReader(socket.getInputStream()));
```

A.3. Sunucuya Veri Gönderme Metodu

```
private void sendDataToServerFromClient(String data)
{
out.println(data);
}
private void closeClient() throws IOException
 {
out.close();
in.close();
socket.close();
}
```

A.4. İstemci Tarafı TCP Paketinin Hazırlanması Metodu

```
public void actionPerformed(ActionEvent arg0) {
Double xKey = Double.parseDouble(ip1Field.getText()) +
Double.parseDouble(ip2Field.getText())+
Double.parseDouble(ip3Field.getText()) +
Double.parseDouble(ip4Field.getText());
StringBuilder builder = new StringBuilder();
builder.append(Double.toString(xKey) + "::");
for (int i = 0; i < Integer.parseInt(s1Field.getText());
i++) {
int temp = rand. nextInt(10) + 1;
temp= Math.abs(temp);
temp=temp+i;
xKey = xKey * (Math.log10(temp));
builder. append(Integer.toString(temp) + "::");}
builder.append(Double.toString(xKey) + "::");
builder.append(TimeCalculations.giveCurrentNanoTime());
xKeyCalcLabel.setText(Double.toString(xKey));
sendDataToServerFromClient(builder.toString());
}
} );
public class Xkey {
public static double getXKey(List<Double> data) {
double xkey = 0;
for (int i = 0; i < data.size(); i++) {
xkey += data.get(i);
}
return xkey;
} }
```

EK B

Önerilen Yöntemin Sunucu Tarafı Kaynak Kodları

B.1. Kullanılan Kütüphaneler

```
import java.awt.BorderLayout;
import java.awt.GridLayout;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.List;
import java.util.Random;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextField;
import com.mkserver.util.Decoder;
import com.mkserver.util.Noise;
import com.mkserver.xkey.Xkey;
```

B.2. İstemci Tarafı İle Bağlantı Kurulması

```
private void initServer() throws IOException {
String clientGelen;
ServerSocket serverSocket = null;
Socket clientSocket = null;
double sayi;
try {

serverSocket = new ServerSocket(7755);
} catch (Exception e) {
System.out.println("Port Hatasï¿½!");
}
System.out.println("SERVER BAGLANTI ICIN HAZIR...");
clientSocket = serverSocket.accept();
PrintWriter out = new
PrintWriter(clientSocket.getOutputStream(), true);
BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
while ((clientGelen = in.readLine()) != null) {
System.out.println("Client'dan gelen veri = " +
clientGelen);
setSensorFields(clientGelen);
calculateXKey(clientGelen);
}
out.close();
in.close();
clientSocket.close();
serverSocket.close();
}
```

B.3. İstemciden Gelen Sensör Verilerinin Alınması Metodu

```
private void setSensorFields(String data) {
List<String> sensorData = Decoder.decode(data);
xKeyComingValueLabel.setText(sensorData.get(sensorData.size
() - 2));
```

B.4. Gelen Veri Paketinin Açılması Sınıfı

```
public class Decoder {
public static List<String> decode(String data) {
String[] words = data.split("::");
List<String> list = Arrays.asList(words);
return list;
}
}
```

B.5. Doğrulama Anahtarı Oluşturulma Sınıfı

```
public class Xkey {
public static Double logRoot = 128.0;
public static Random rand = new Random();
public static double getXKey(List<String> data) {
double xkey = Double.parseDouble(data.get(0));
for (int i = 1; i < data.size() - 2; i++) {
Double temp = Double.parseDouble(data.get(i));
xkey = xkey * (Math.log10(temp));
}
return xkey;
}
}
```

B.6. Doğrulama Anahtarı Karşılaştırma Sınıfı

```
private void calculateXKey(String data) {
List<String> sensorData = Decoder.decode(data);
double xkey;
double                    clientXkey                    =
Double.parseDouble(sensorData.get(sensorData.size() - 2));
xKeyCalcvalueLabel.setText(Double.toString(xkey));
if (xkey != clientXkey) {
JOptionPane.showMessageDialog(null, "Alarm !!! Degerlerde
degisiklik var.");
}}
```

EK C

Algoritmalar ile Önerilen Yöntemin Karşılaştırılması

C.1. Kullanılan Kütüphaneler

```
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.Random;
```

C.2. Değerlerin Rastgele Üretilmesi Metodu

```
public static byte randomFill()
{
Random rand=new Random();
byte randomNum=(byte) rand.nextInt();
return randomNum;
}
```

C.3. Üretilen Sensör Verilerinin Sensör Toplama Noktasına Aktarılması Metodu

```
public class Logaritmik {
private static byte [] anArray;
public static byte[] list() {

anArray = new byte [100];
for (int i=1;i<anArray.length;i++)
{
anArray[i]=randomFill();
}
return anArray;
}
```

C.4. Üretilen Sensör Verilerinin Ekranda Gösterilmesi

```
public static void degerGor()
{
For (byte n:anArray)
{
System.out.println(n+" ");
} }
```

C.5. MD5 Algoritmasının Doğrulama Değeri Oluşturma Metodu

```
public static void md5print() throws
NoSuchAlgorithmException
{
byte md5Digest[];
MessageDigest md5;
md5 = MessageDigest.getInstance("MD5");
for(int i=1;i<anArray.length;i++)
{
md5Digest = md5.digest(anArray);
} }
```

C.6. SHA-1 Algoritmasının Doğrulama Değeri Oluşturma Metodu

```
public static void Sha1print() throws
NoSuchAlgorithmException
{ byte sha1Digest[];
MessageDigest sha1;
sha1 = MessageDigest.getInstance("SHA-1");
for(int i=1;i<anArray.length;i++)
{ sha1Digest = sha1.digest(anArray);
} }
```

C.7. SHA-256 Algoritmasının Doğrulama Değeri Oluşturma Metodu

```
public static void Sha256print() throws
NoSuchAlgorithmException
{
byte sha256Digest[];
MessageDigest sha256;
sha256 = MessageDigest.getInstance("SHA-256");
for(int i=1;i<anArray.length;i++)
{
sha256Digest = sha256.digest(anArray);
}
}
```

C.8. Önerilen Yöntemin Doğrulama Değeri Oluşturma Metodu

```
public static void print() throws Exception
{
double logaritmik=1;
for (int i=1;i<anArray.length;i++)
{
double x = (double)(anArray[i]);
x= Math.abs(x);
x=x+i;
logaritmik = logaritmik * Math.log10(x);
}
System.out.println("Deger:"+logaritmik);
}
```

C.9. Hız Performans Karşılaştırması

```
public static void main(String [] args) throws Exception {
long startTime=System.nanoTime();
{
list();
//Test edilecek yöntemin başındaki yorum işareti kaldırılır.
md5print();
//Sha1print();
//Sha256print();
//print();
}
long estimatedTime=System.nanoTime()-startTime;
System.out.println(" Sure Nanosaniye:"+ estimatedTime);
System.out.println(" Sure Microsaniye:"+
(estimatedTime/1000));
System.out.println("Sure Milisaniye:"+
(estimatedTime/1000000));
degerGor(); } }
```