LINUX SISTEMLER ICIN
GUVENLIK FRAMEWORKU

2019 - 2020

LİSANS

BİTİRME PROJESİ

BİLGİSAYAR MÜHENDİSLİĞİ

Egemen ULUSOY

LINUX SISTEMLER ICIN GUVENLIK FRAMEWORKU

Egemen ULUSOY

Karabük Üniversitesi
Fen BilimleriEnstitüsü
BilgisayarMühendisliği Anabilim
Dalında
Lisans DönemProjesi Olarak
Hazırlanmıştır.

KARABÜK
Mayıs 2020

"Bu projedeki tüm bilgilerin akademik kurallara ve etik ilkelere uygun olarak elde edildiğini ve sunulduğunu;ayrıca bu kuralların ve ilkelerin gerektirdiği şekilde, bu çalışmadan kaynaklanmayan bütün Atıfları yaptığını beyan ederim."

EgemenULUSOY

Lisans Bitirme Projesi

LINUX SISTEMLER ICIN GUVENLIK FRAMEWORKU

Egemen ULUSOY

Karabük Üniversitesi
Fen BilimleriEnstitüsü
BilgisayarMühendisliği Anabilim Dalı

Proje Danışmanı:
Dr. Ferhat Atasoy

Mayıs2020,21sayfa

ÖZET

Proje kapsamı dahilinde; günümüzde karşılaşılabilecek siber saldırı vektörlerive savunma çözümleri ele alınıp NT ve Linux işletim sistemlerinin mimarisi ve genel zaafiyetleri incelenmiştir. Temel anti-virus mekanikleri, yazılımsal firewall sistemleri ve diğer güvenlik çözümlerinin çalışma prensipleri baz alınarak linux işletim sistemi için sistem yöneticilerinin genel güvenlik çözümleri amacıyla ihtiyaç duyacağı üç ana çalışma hattına sahip güvenlik frameworkun'un tasarımsal çizimleri uygulanarak projenin tamamlanması amaçlanmaktadır.

İlk kısımda penetration testlerin süreci, potansiyel saldırıların hangi yöntemlerle gerçekleştirilebileceği ve yaygın olarak kullanılan saldırı vektörleri anlatılmıştır. İkinci kısımda proje mimarisinin tasarımına ve teknik detaylarına yer verilmiştir.

Üçüncü kısımda çalışma boyunca karşılaşılabilecek problemler, işletim sistemlerinin çekirdek sürüm farklılığında ortaya çıkabilecek potansiyel sorunlar ve donanımsal kaynaklara bağlı genel optimizasyon problemlerinin minimize edilmesi için izlenmesi gereken yönergelerden bahsedilip rapor sonlandırılmıştır.

TEŞEKKÜR

Bu tez çalışmasının planlanmasında, araştırılmasında, yürütülmesinde,oluşumundailgi ve desteğini esirgemeyen, engin bilgi ve tecrübelerinden yararlandığım sayınhocam Dr.FerhatATASOY'a, MalwareAnalist olarak görev yaptığım ZEMANA A.Ş bünyesinde 2020 Q2 planlamaları çerçevesinde projenin ticarileşmesini destekleyen Genel Müdürümüz sayın Mesut DEMİRTAŞ'a, Teknolojiden Sorumlu Müdürümüz sayın Yağızhan ATMACA'ya, çalışma arkadaşlarım Fatih ERDOĞAN ve MertcanALICI'ya teşekkürlerimi sunarım.

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

Sayfa

1. SECURITYFRAMEWORK(SF-LINUX)

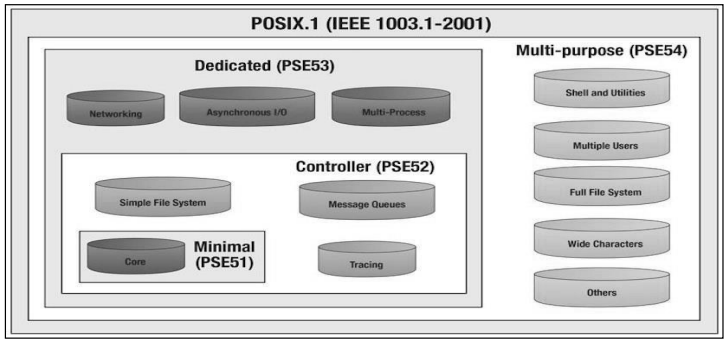
1.1. ProjeninAmacı

Yazılım,sistem ve ağ güvenliğini hedef alan siber tehditler internet teknolojilerinin yaygınlaşması ile birlikte server sistemlerin güvenliği kritik bir sorun haline gelmiştir. Sistem yöneticilerinin insan zaafiyetini baz alan sosyal mühendislik teknikleri hakkında bilgilendirilmesi, teknik yönden bir zaafiyet içermeyen sistemlerin güvenli kalması açısından önemlidir. Kullanıcı kaynaklı olmayan teknik zaafiyetler için, proje konusu olarak, linux tabanlı işletim sistemlerinin mimarisi incelenip, güvenlik zaafiyetlerinin hangi teknikler ile istismar edilebileceği belirlenip, saldırıların engellenmesi için geliştirilen üç ana hatta sahip güvenlik framework'unun mimarisi tasarlanacaktır.

Güvenlik uzmanları tarafından gerçekleştirilen penetration testler siber saldırı simülasyonudur ve ortaya çıkabilecek güvenlik zaafiyetleri hakkında kapsamlı olarak bilgi verir. Sızma testlerinde izlenecek yönergenin ilk adımı hedef sistem hakkında bilgi toplamak ve sistem kullanıcılarını araştırmak/enumerate etmektir. Ikinci adım, keşif aşamasında ortaya çıkan zaafiyetler kullanılarak hedef sisteme izinsiz giriş sağlanması hedeflenir. Üçüncü adım, sistem yöneticisinin ayrıcalıklı haklarını, sudo(superuserdo) yetkisinikazanmaktır. Dördüncü adım ise hedef sistemde saldırganın bırakabileceği tüm iz ve kayıtlar (log verileri) temizlenip zaafiyetlerin raporlanmasıdır.

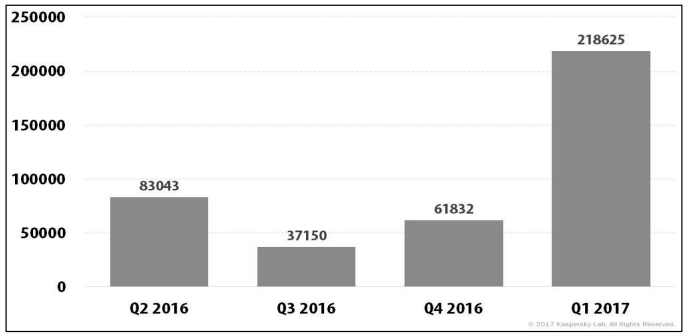
Security-Framework projesinin genel amacı; sistemin hem local hem remote hattındaki güvenliği sağlamak, framework bünyesindeki Firewall (güvenlik duvarı) yardımıyla saldırganın dışarıdan ve iç ağdan gelebilecek paket trafiğini bloke etmek, sisteme izinsiz giriş sağlanırsa işletim sisteminin kritik bölgelerine müdahalesini önlemek, sunucu için önem arz eden dosyaların ele geçirilmesini / çoğaltılmasını / görüntülenmesini önlemek, saldırganın sistemde bıraktığı iz / logları ortadan kaldırmasını önlemek, malware saldırılarını ve güvenlik yazılımını devre dışı bırakabilecek kabuk kodların yürütülmesini engellemektir. Servisin koruma seviyesini belirleyen ayarların ve kontrolün yalnızca sistem yöneticisinin insiyatifinde olması gerekmektedir.

İşletim sistemleri, donanımsal gelişmelere bağlı olarak güncelleştirilmektedir ve güncel teknolojiler ile stabil olarak çalışabilmesi için mimariler farklılaşmaktadır. Güvenlik frameworku tasarlanırken bu gelişmeler ve POSIX standartları gözönünde bulundurulmuştur. Günümüzde yaygınolarak kullanılan major server sistemler için fonksiyon testleri sağlanıp rapora aktarılmıştır. IEEE POSIX.1[2] dosya ve proses kontrol/ manipulasyonşeması.(Şekil1.1.1)



Şekil1.1.1. IEEE POSIX.1

Siber saldırılar bir çok farklı vektör tarafından gerçeleştirilebilir, günümüzde en yaygın olan üç saldırı vektörü; malware saldırıları, distributed deniel of service (ddos) saldırıları ve exploitlerdir. Mayıs 2017'de ortaya çıkan ve yaklaşık 300.000 bilgisayarı enfekte eden WannCry[1] adlı malware ile birlikte ransomware,yani sistem dosyalarına erişimi engelleyen; şifreleyen, şifrenin geri çözülmesi için kullanıcıdan veya sistem yöneticisinden fidye talep eden zararlı yazılımların sayısında ciddi bir artış görülmektedir. (Şekil1.1.2Kaspersky Ransomware İstatistik Raporu[5]) Framework mimarisi kapsamında, dosya yapısının sistem çağrıları ve inode blokları ile detaylı olarak takip edilmesi, nihai sonuç olarak ransomware tipi malwarelere tam koruma sağlanması hedeflenmektedir.



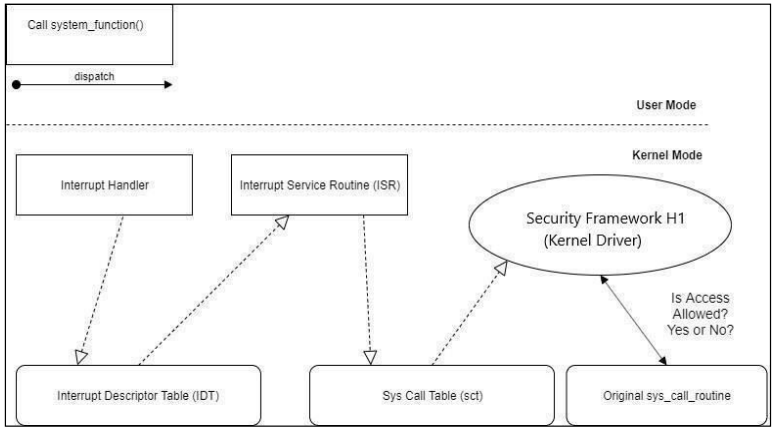
Şekil1.1.2. KasperskyRansomware İstatistik Raporu

1.2. ProjeMimarisi

1.2.1. Security-Framework(HAT1)

Projenin ilk hattında sistemçağrılarının izlenmesi, sistem aktivitelerinin ve raporlarının önemsırasına göre sistem yöneticisine aktarılması, kritik güvenlik tehditi taşıyan aktivitelerin bloke edilmesi ve güvenlik framework'unun sürekliliği için bypass kodu yürütebilecek tüm aktivitelerin engellenmesi amaçlanmaktadır. IRQL (Interrupt Request Level) tarafından maskelenmeyen, önceliği yüksek interruptlar Interrupt Handler tarafından ilgili servis rutinine yönlendirilmektedir.

Sistem çağrısının yürüteceği prosedür adresi güvenlik servisi tarafından değiştirilir, çağrı parametreleri filtrelendikten sonra orjinal rutine dalanmasına izin verilir veya istek reddedilir. Servis kimliğini ortaya çıkarmakve servisi durdurmak isteyen kabuk kodlar sistem çağrılarında farkedilip sonlandırılır. Bu noktada driverın kendini koruması(SelfProtection) sağlanmış olur.(Şekil1.2.1.1)



Şekil1.2.1.1. Hat-1TasarımŞeması

1.2.2. Security-Framework(HAT2)

1.2.3.

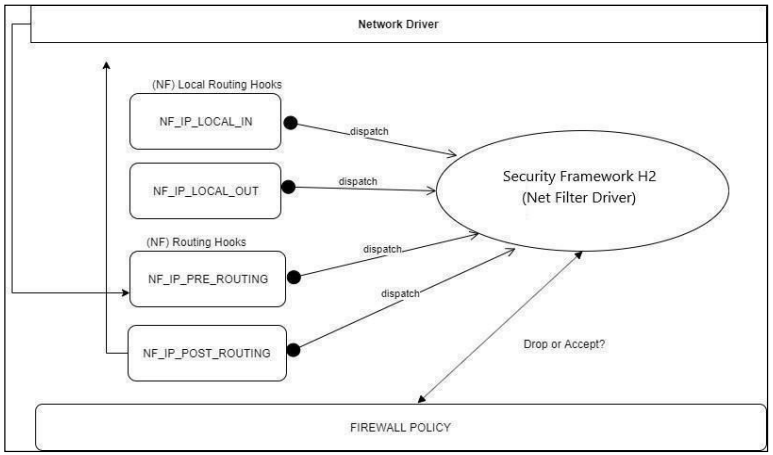
Projenin ikinci hattında ağ trafiğinin takip edilmesi, araya girilen ağ trafiğinin (m.i.t.m) şifresinin çözülüp payload hashleri ile karşılaştırılması, sistem yöneticisi tarafından belirlenen kurallar yardımıyla paket trafiğin güvende tutulması amaçlanmaktadır.

Exploit ve ddos saldırılarının engellenmesi bu hatta gerçekleştirilecektir, günümüzde d-dos saldırılarının bir çoğu bot-net alanındaki zombi ağlar veya serverlar yardımıyla gerçekleştirilmektedir.

Gelensaldırı, sunucunun sahip olduğu bant genişliğinden çok daha yüksek bir seviyede ise bu tip bir saldırı ancak ISP (telekom) seviyesinde alt yapıya sahip olan cloud networklar tarafından tamponlanıp private clouda ulaşması önlenebilir.

Yazılımsal veya donanımsal olan güvenlik duvarlarının bu tip saldırılar için kabiliyetleri sınırlıdır. proje kapsamında düşük seviye ddos saldırılarının önlenmesi planlanmaktadır.
Kaspersky 2015 D-DOS istihbarat raporunda gelen saldırı vektörünün en fazla SYN-flood olduğu paylaşılmıştır. [4]

Prerouting aşamasında DNAT ile yerelağ içerisinde yönlendirme (forward)olmadan network stack'e dönen (loopback) veya mevcut ağdan bir başka ağa SNAT ile yönlendirilen paketlerin postrouting dahil olmak üzere beş farklı methoddan hook edilmesi (paketlerin işlendiği temel 3 hattın input,output ve forward) policy ile filtrelenmesi gösterilmiştir.(Şekil1.2.2.1).



Şekil1.2.2.1-(Hat-2-TasarımŞeması)

Network driver ile işletim sistemi arasında gerekli hooklar oluşturulduktan sonra socket buffera (sk_buff) erişilebilir.Mac header,network header,transport header ve paketdata (applicationlayer) olmak üzere L2 - L7 arası protocol stack'ın içeriğini filtrelenebilir. Firewal'ın temel mekanikleri OSI referans modelinin 7. Kat- manında çalışıp, eklenmesi planlanan diğer özellikler temel yapının üzerine inşa edilecektir.

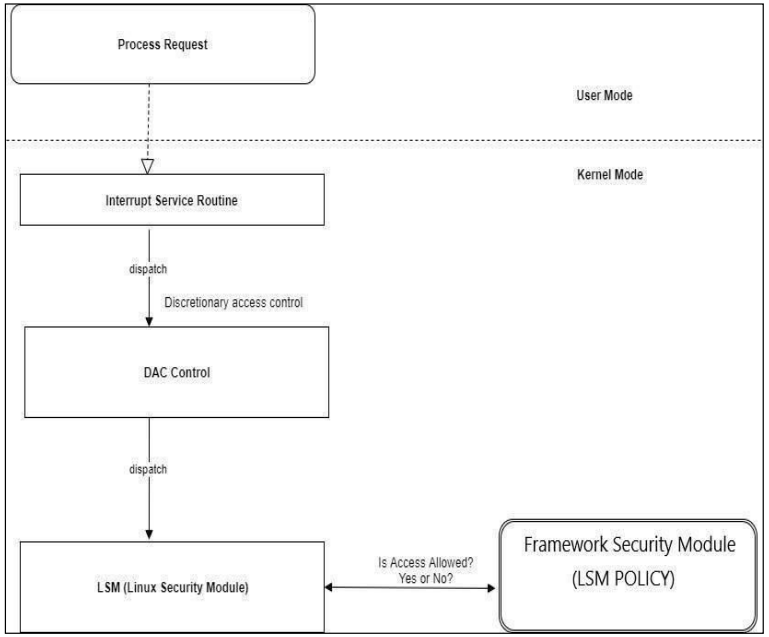
Eklenen her yeri özellik için optimizasyon sorunları göz önünde bulundurulmalıdır, örnek olarak VPN desteği için gerekli çalışmalar sağlandığında IP katmanında çalışacak olan protocol'ün (IP-Sec) sağlayacağı avantaj ve dezavantajlar göz önünde bulundurulup sistem yöneticisine opsiyonal bir tercih olarak bırakılmalıdır.

Ağ topolojisine bağlı olarak; firewall gateway olarak tutulup DMZ koruma altına alınabilir. Bindshell'ler kolayca önlenebilise de, çıkış trafiği serbest bırakıldığından reverse shell, firewall'lar için genel bir sorundur. Sorunun tamamiyle önüne geçmek mümkün olmasada Shodan[1] gibi C2s(command& controlserver) bilgilerini işaretleyen istihbarat servisleri ile entegrasyonunu sağlamak isabetli bir çözüm sağlayacaktır.

1.2.4. Security-Framework(HAT3)

Projenin üçüncü ve son hattında 22 aralık 2000 tarihinde NSA ve Red Hat tarafından geliştirilip kernele dahil edilen se-linux (NSASecurityEnhancedLinux) güvenlik frameworku baz alınarak LSM (LinuxSecurity Module) geliştirilmesi amaçlanmaktadır.

Bu hat, sistemçağrılarının kernel bölgesinde inodelara ulaştığı son katman olduğundan (Şekil1.2.3.1), unix tabanlı dosya sistemlerinin inode blokları ve metadataların policy ile kontrol edilmesi güvenlik yazılımının sisteme hakimiyeti açısından kritik bir önem taşıyacaktır.



Şekil1.2.3.1. (Hat-3TasarımŞeması)

1.1. SERVIS GEREKLILIGI

Linux security framework hat-2 hibrit güvenlik duvarı dahilinde, network address translation (nat) ve port address translation (pat) servisleri trafiğin bir uçtan diğer uca yönlendirilmesi için paket manipulasyonu ve aynı zamanda esnek yönlendirme filtreleri sayesinde sistemin lokal ve remote hattında ilgili köprüleri (proxy bridge) kurmamıza olanak sağlayacaktır.

Translation için destek sağlamak router'ın temel görevidir ve güvenlik duvarının öncelikli görevlerinden birisi değildir, örnek olarak yerli firewall üreticilerimizden coslat nat/pat desteği sunmamıştır. İds/ips çözümlerine odaklanmıştır. Ancak her firewall, teknik olarak "route" görevi üstlendiği için modern olan tüm seriler içerisinde yönlendirme ve adres/port translation özellikleri sabit olarak bulunur.

1.2. YONTEMLER

Donanımsal firewall'lar işletim sistemine bağımlı olmadıkları için yönlenen trafiği gateway olarak karşılayıp sorunsuzca işleyebilirler, yazılımsal firewall modelinde ise işletim sisteminin mimarisine bağımlıdır.
Translation ve maskeleme işleminin gerçekleşebilmesi için farklı yöntemler vardır; packet mangling, packet duplicate, application level socket proxy, netfilter level nat table (xtables) kullanabileceğimiz dört yöntemdir.

1.2.1 Packet Mangling

Dört yöntem arasından en sağlıklı yoldur, herhangi bir yönlendirme tablosu kullanılmadan pre_routing (inbound traffic) ve post_routing (outgoing traffic) aşamalarında tcp header ve ip header üzerinden source/destination bilgileri maskelenerek trafik dinamik olarak yönlendirilir. dnat (destination nat) ve snat (source nat) tanımlanabilir. Masklenen bilgiler ile checksum yeniden hesaplanarak üçlü el sıkışmanın sekteye uğraması engellenip trafik devam ettirilir.

1.2.2 Packet Duplicate

Bilindik bir yöntem değildir, kernel tarafında uygulandığı yönünde herhangi bir bilgiye ulaşamadım ancak teorik olarak çalışması mümkün. Yönlendirilecek paketin socket buffer'ı kopyalanarak ilgili protokolün düzenine göre (tcp udp icmp) paket yeniden oluşturulup protocol stack'e girilir, önceki paket ise drop edilir. İşletim sisteminin kerneli tarafından paketin mangle edildiği anlaşılamaz ancak buffer'ın kopyalanması ilk maliyet, paketin yeniden oluşturulması ise ikinci maliyettir. Ttl (time to live) süresinde ciddi bir gecikmeye yol açabilir.

1.2.3 Nat Table

Yönlendirme bilgileri range tablosuna yazılarak nat setup gerçekleştirilir. Mangling gibi dinamik bir yönlendirme değildir, statik olarak tabloya girilir. dnat ve snat tanımlanabilir.

1.2.4 Application Level Socket Proxy

Olağan dışı bir yöntemdir, netfilterin desteklediği bir teknik değildir. Sysguard'ın syscall hooklarından faydalanılarak socket çağrıları hijack edilir. Sockaddr yapısı üzerinden port ve/veya ip bilgileri maskelenip yönlendirme yapılır. Sadece dnat için çalışır kural kümesinden önce açılmış bir socket iletişime devam ediyorsa yönlendirmeden etkilenmez. Son olarak tercih edilmesi gereken yöntemdir.

1.2. SSL STRIP

Kullanılabilecek Teknikler:

1.3.1 klasik mitm

Saldırganlar bu tekniği arp cache poisoning veya daha mantıklı olan adıyla arp poison routing ile birlikte kullanırlar. Firewall aynı şekilde http trafikte araya girerek plain trafiği okuyabilir. Ancak günümüzde hsts (http strict transport security) bir standart haline gelmiştir. Modern işletim sistemlerinde ve browser içerisinde direk olarak ssl protokolü devreye girdiği için çalışması mümkün değildir.

1.3.2 setsockopt hijacking:

Sysguard'ın native hookları yardımıyla socket api olan setsockopt parametrelerine erişilerek sertifika ve private key ele geçirilebilir ancak socket apiler ile netfilter arasında senkronizasyon olmayacaktır. Bu sebeple ele geçirilen key'in hangi pakete ait olduğu bilinemez. Bruteforce ile key denemesi trafiği yavaşlatır.

1.3.2 local proxy:

Tüm teknikler arasında en verimli ve geçerli olan çözümdür. Teorik olarak bu yöntem de bir mitm'dir. Lokalde olma avantajımızı kullanarak https trafiği yine lokaldeki https dinleyiciye yönlendirip yine aynı şekilde lokal dinleyiciyi gateway üzerinden internete yönlendirdiğimizde SSL Strip kolaylıkla sağlanmış olacaktır.

1.3. KARŞILAŞILAN SORUNLAR

Firewall'ın son modülü olan nat/pat servisleri için kullanabileceğimiz en isabetli yöntem packet mangling olacaktır ancak tcp header değiştirildiğinde protocol stack paketin kendi stackından yönlenmediğini düşünerek üçlü el sıkışmanın son aşamasında RST (reset) flagını set ederek trafiği terminate ediyor. -RST paketi drop edilse bile kuyruktaki paketler de otomatikmen drop edildiğinden trafik devam ettirilemiyor, RST flagı ACK (acknowledgement) ile değiştirilmesi mümkün ancak üçlü el sıkışma tamamlansa bile trafik stacke bağlı ve tcp retransmission döngüsünde takılı olarak kalıyor.

1.1. SYS GUARD ÖN TEST SONUÇLARI

ZSF Linux projesinin ilk hattı olan sys guard'ın güncel dağıtımlar üzerinde testleri sağlanıp, testlerin gerçekleştiği dağıtım ve kernel bilgileri aşağıda listelenmiştir.

CentOS-8-x86_64-1905
Kernel: 4.18.0-80.11.2.el8_0.x86_64

Pardus x86_64-19.1
Kernel: 4.19.0-6-amd64

Debian-9.11.0-amd64
Kernel: 4.9.0-11-amd64

OpenSUSE-Leap-15.1-x86_64
Kernel: 4.12.14-lp151.28.36-default

Ubuntu-19.10-desktop-amd64
Kernel: 5.3.13-4-amd64

Implementasyon için öncelikli hedefimiz olan Pardus, debian tabanlı hibrit bir işletim sistemidir ve orjinal linux kernelini kullanmaktadır, dolayısıyla debian için çıkaracağımız her güncelleme/eklenti pardus için de sorunsuz olarak çalışması beklenmektedir.

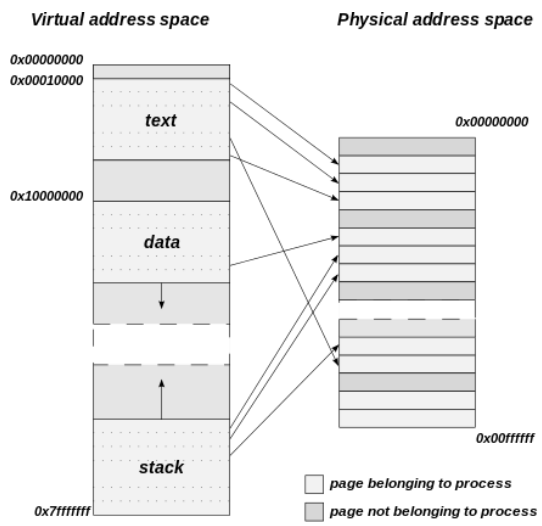
Ön test aşamasında temel fonksiyon testlerinin çalışması amaçlanmıştır, stres testi ve uzun kullanıma tabi tutulması bir sonraki aşama olacaktır. Test kiti hazırlanıp; azami 48 saat sürecek stres ve kararlılık testi sonucunda herhangi bir kırılma gözlenmez, hafıza taşması ortaya çıkmaz ve güvenlik yazılımının işletim sistemine bindireceği yük göz önünde tutularak CPU için tolere edilebilir bir performans kaybı gözlemlenirse (~%1-5) kavramsal tasarıma ve üst segmentlerin planlamasına geçilebilir.

. 2. Hat olan Netfilter (firewall) hattı için ön test gerekli değildir, işletim sisteminin kurallarına aykırı bir çalışma mantığı olmadığından kernel sürümlerinin hepsine uyum sağlaması beklenmektedir. Stres testi netfilter için de önem taşımaktadır, yoğun trafik ve yük altında tampon bölgelerinde leak olmamalı ve performans kaybı tolere edilebilir ölçüde tutulmalıdır.

1.1.1 Kontrolcü İle Haberleşme

Süreçler arası iletişim IPC (Inter Process Communication) farklı teknikler ile gerçekleşebilir, kullanabileceğimiz yöntemler aşağıda listelenmiştir.

1) Pipes
2) Semaphores
3) Shared Memory
4) Sockets
5) Syscall Hijacking



Şekil 1User/Kernel Hafiza Adreslenmesi

Şekil (1)'de işletim sisteminin sayfalama mekanizması gösterilmiştir, kullanıcı uzayındaki hafıza adresi kernel içerisinde farklı bir sayfanın adresinde yer alabilir. Ring3'te CR3 biti modifiye edilemez ve segment tanımlayıcı kontrol edilemez, dolayısıyla, shared memory bizim için en hızlı yol olsa da çalışması mümkün değildir. Ring0'da socket açılarak kontrolcü ile full-duplex iletişim sağlanabilir ancak bu güvenli bir yol değildir, iletişim sağlandığı port filtrelenerek iletişim durdurabilir.

Yöntemler arasında alternatif olarak en hızlı ve güvenli yol, sistem çağrılarının hijack edilmesidir. Kontrolcü ile driver arasında bütünlük bozulmadan güvenli ve yarı kapalı (half-duplex) iletişim sağlanacaktır.

Tüm testler tamamlandığında, detaylı teknik rapor hazırlanacaktır. Rapor dahilinde; interrupt kontrolcüleri (APIC) DMA yolu, I/Omem(/proc/iomem) bellek haritalandırmaları ve debugger çıktıları ile kernelin çalışacağı sürece etki eden donanımsal ve yazılımsal tüm süreçler açıklanacaktır, şimdilik kontrolcü ile haberleşmenin en iyi alternatif olarak hangi yöntem ile sağlanacağına kısaca yer verilmiştir.

1.1.2 Karşılaşılan Sorunlar/Çözümleri

Sys guardın testleri esnasında üç farklı sorun ile karşılaşıldı, sorunun kaynağı ve çözümleri aşağıda özetlenmiştir.

x64 >= v4.17 kernel için 80. İnterrupt (syscall) çağrılarının tablosuna erişmek amacıyla bruteforce bir arama gerçekleştirilerek kernel sayfaları enumerate edilerek ilgili rutinin pointer adresine ulaşılmaya çalışıldı. Bu durum, yeni nesil 64 bit kerneller'de NULL pointerları işaret etmek kernelin kırılmasına sebep açıyor.
bruteforce yerine /proc/kallsyms üzerinden ya da alternatif olarak IDT üzerinden dolaylı olarak tablo adresine ulaşılarak sorun çözüldü.

x64 kerneller için prosedürlere iletilen parametreler x86'lardaki gibi doğrudan akumülatörden başlamıyor, farklı kaydediciler (esi, edi ...) ile aktarım tercih edilmiş. İlgili kaydedicilerden parametreler okunarak sorun çözüldü.

x64 >= v5.1 kernel için yazım/okuma izni (RW) olmayan kernel tablolarını patchlemek amacıyla CR0 bitini kullanıyoruz. Exploitleri ve rootkitleri önlemek amacıyla CR0 registerı kontrol edilerek / yazma izni verilmeyerek bu durum önlenmiş.
https://www.phoronix.com/scan.php?page=news_item&px=Linux-5.1-Pin-CR0-CR4-Bits
yazım yapılacak sayfanın adresi başka bir bellek bölmesine map edilince sorun çözüldü, bir nevi bypass edildi şu an için herhangi bir sorunumuz yok ancak gelecekte daha sıkı güvenlik önlemleri ile karşı karşıya kalabiliriz.

1.1.3 Planlanan süreçler ve stres testleri

Sys-guard:

+device controller
+self-protection
+file and process protection
+anti-rootkit
+anti-malware

Netfilter:

+Layer-7 firewall functionality
+Mail Gateway
+Shodan service plugin
+Bruteforce / basic d-dos protection.
+Exploit protection (linked with exploit databases)

Sysguard'ın test süresi boyunca, cpu'nun aktif tüm çekirdekleri asenkron olarak interrupt edilerek ~%90 ve üzeri yük altında 24 saat çalıştırılıp, akabinde, Netfilter; test süresince, google cloud alt yapısında 100-250 Mbit inbound/outbound trafikte 24 saat boyunca tüm kural tablosu dolu halde firewall tarafından filtrelenecektir. (48 saatte iki hattın testi tamamlanmış olacak)

Karşılaşılabilecek en kötü seneryoyu bu testin sonucunda gözlemleyeceğiz ve sonuç olarak, iyileştirmenin gerekli olduğu modülleri bir sonraki testte başarılı olana kadar yeniden tasarlayıp son kullanıcıya kararlı ve stabil çalışan bir servis sunmuş olacağız.

2. NTMİMARİSİVESİSTEMGÜVENLİĞİ

2.1. SavunmaVektörleri

NT bünyesinde yer alan güvenlik yazılımlarının mimarisinin geliştirilmesi, güncel zaafiyetlerinin ortaya çıkarılması ve isabetli tarama sonuçlarına ulaşılması amacıyla son kullanıcıların beklentilerinin değerlendirilmesi önem taşımaktadır. Rapor dahilinde; günümüzde Anti-Malware altyapısına entegre edilen, Makine Öğrenmesini baz alarak sınıflandırma yapan ileri jenerasyon tarama tipi ve false positiveler ele alınacaktır.

Kullanıcı güvenliğini ve güvenlik yazılımının bütünlüğünü tehdit eden tüm aktivitelerin araştırılması, ortaya çıkarılması ve engellenmesi için gereken yönergelerin hazırlanması Malware Analistlerin sorumluluğun- dadır.

Güvenlik yazılımının savunma vektörleri dahilinde, işlemcide anahtarlanan ve işlemciye girme potansiyeli olan tüm processlerin disk, bellek ve network aktivitelerinin gerçek zamanlı takip edilmesi, şüpheli sistem aktivitelerinin durdurulması, browser kaynaklarının güvende tutulması ve kullanıcı insiyatifinde olmayan işlemlerin sürdürülememesi amaçlanmaktadır.

2.2. TaramaTeknikleri

Güvenlik yazılımı, sistem kaynaklarına yerleşen malwareleri işaretleyebilmek için hem statik hem dinamik taramaya ihtiyaç duyar, statik taramanın gerçekleşebilmesi için disk blokları üzerinden çalıştırılabilir uygulamanın sectionları veya bütününü kapsayan imza örnekleri alınır ve imza veritabanındaki örnek hashler ile karşılaştırılır. Eşleşmeye rastlanmaz ise, dosya formatının hafiza haritalanmasına bakılır. Çalıştırılması planlanan sistem fonksiyonları ve parametrelerinin sıralaması izlenip benzerlik aranır.

Obfuscation (koruma, gizlenme, paketleme) içeren veya polymorphic olan malwarelerin statik analizde tespit edilmesi mümkün değildir. Çalışma zamanında bellek datalarını decrypt ederek zararlı kodları dinamik olarak yürütür, yürütme tamamlandığında geri şifreleyerek anlamsız veri haline getirirler. Koruma içeren, dolayısıyla anti-malware taramalarını bypass etmeye çalışan kod parçaları sık olarak karşılaştığımız bir durumdur ve genel olarak NT mimarisi hedef alınır.

Dinamik taramaların gerçekleşebilmesi için, bellek üzerinde adreslenen process kademeli olarak (simüle edilerek) çalıştırılır, sistem çağrıları izlenir, şifresi çözülen hafiza bloklarındaki API isteklerinin zararlı kod parçalarını içeren veritabanı ile eşleşmesi ve sonuç olarak kompleks koruma bulunmayan veya rootkit aktiviteleri görülmeyen malware karakteristiğinin tespit edilmesi beklenir.

Karakteristiği diğer malware tiplerinden farklı olan rootkitler, hem kullanıcı hem kernel uzayında faaliyet gösterir, çok yönlü koruma ve gizlenme teknikleri içerir. Analiz edilmesi ve sistemden silinmesi kolay değildir. Nadiren karşımıza çıkan bu tip malwarelerin davranışları kernel uzayında takip edilmeli, eğer güvenlik yazılımının güncel mekanikleri yetersiz kalırsa -ki çoğu zaman yetersiz kalabilir, MalwareAnalist tarafından tersine mühendislik teknikleri ile incelenmeli ve sistemi enfekte aşamalarına özel kademeli güvenlik methodlarının oluşturulması gerekmektedir.

2.3. FalsePositiveler

Hatalı sınıflandırma güvenlik yazılımları için genel bir sorundur. sorunun önüne geçmek tamamiyle mümkün olmasa da kullanıcışikayetinin azaltılması için tümçalışmalar yapılmalıdır.

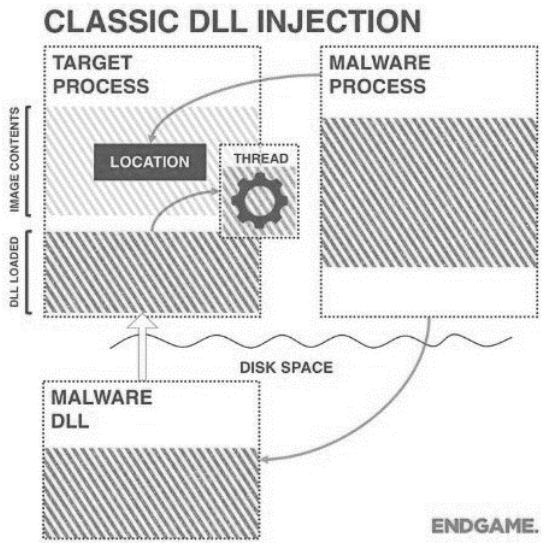
UPX ile sıkıştırılan zararsız uygulamanın 15 farklı false positive aldığıgösterilmiştir (Şekil2.3.1).

Şekil2.3.1. False Positive Örnekleri

NT mimarisi, threadler konusunda belirsizlikler olsa da posix uyumlu bir işletim sistemidir ve c2- seviye güvenlik standartlarını karşılar. Güvenlik yazılımlarının mimarisi için belli standartlar bulunmamaktadır,

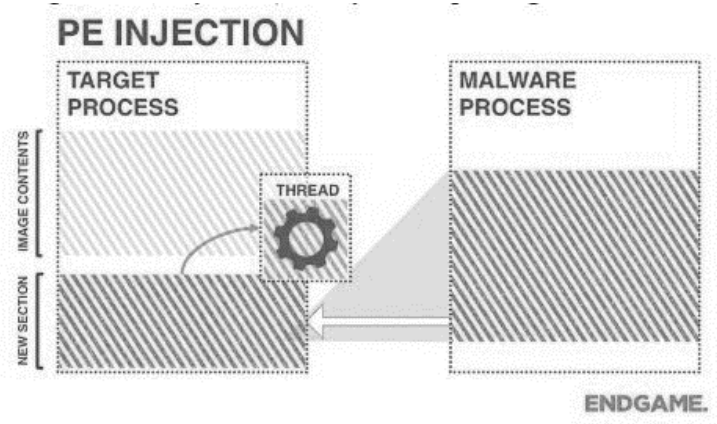
1.1 Injection Teknikleri

DLL Injection
Yaygın olarak karşılaştığımız tekniklerden birisidir, process açılıp handle üzerinden heap alanında yer açılır. Enjekte edilecek DLL'in disk konumu RVA'ya yazıldıktan sonra RemoteThread veya APC rutini loader apiye set edilip parametre ile çalıştırılır, ring3'te fazla sayıda api kullanması sebebiyle tespiti kolaydır, davranış modelinden kolayca tespit edilebilir.



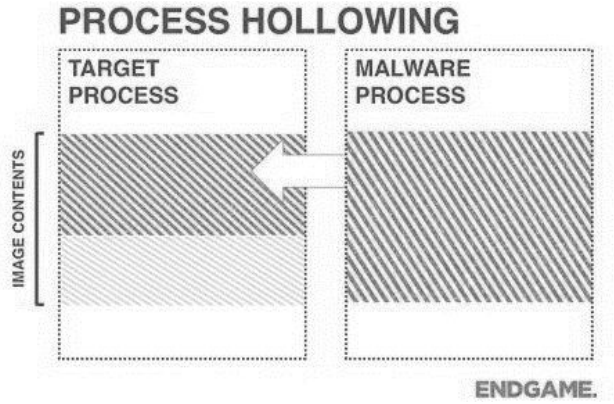
PE Injection
AV bypass tekniklerinde sıkça karşımıza çıkar, DLL Injectiondan farkı harici bir dll drop edilmez, yeni bir section allocate edilip RVA section içerisine shell code veya dll imageı yazılıp remote thread ile çalıştırılır.

EPROCESS -> Virtual Address Descriptors yapısı üzerinden RVA'da map edilen (page_execute) bellek bölgelerini izleyebilir, bu sayede tespiti sağlanabilir. (VAD track)



Process Hallowing
Custom packer içeren malware tiplerinde sıkça karşılaştığımız bir tekniktir, process suspend edilir yani uykuya geçirilir, ardından section unmap edilir (flap) hafızadan yeni alan allocate edilip üzerine yazılır ve son bir dokunuş ile RVA entry point shell code adresine ayarlanır.

POSIX mimarisine göre multi thread OS'ler için scheduling algoritamaları uygulanır, işlemci her thread'i eşit mesafede anahtarlayabilir ya da öncelik sırasına göre anahtarlar, her process işlemcide anahtarlanmak için birbiri ile yarışır, bu yarış esnasında process/threadler switch edilinceye kadar uykuya alınır ve context bilgileri kaydedilir, thread / process id'si, eip ve temel register bilgileri kaydedilen bilgiler arasındadır, bilgiler değiştirilerek bir sonraki anahtarlanma aşamasında araya girilebilir hijack edilebilir. ntoskrnl export edildiyse (x86) NtGetThreadContext/*NtSetThreadContext* apilerinin izlenmesi bu noktada önemlidir.



Mimari-Tasarımları
Mimari tasarımları özgün olup genel sonucu belirleyecek olan kullanıcı tecrübeleri ve malware setleri üzerindeki sınıflandırma kabiliyetidir. Entegre edilen makine öğrenmesi;setler ile doğru eğitildiği sürece, statik tarama amacıyla hızlı bir çözüm kazandıracaktır.

Yapay zekanın eğitim süreci, güvenlik yazılımının teknik fonksiyonlarıyla ilgili olmayan matematiksel süreçler içerir ve nöronların düşünme biçimlerini taklit etmeyi amaçlayan teknolojidir. Malware setlerinin işlendiği süreç yapay zeka ekibinin sorumluluğundadır ve Malware Analistlerin kontrolü dahilinde değildir. Geliştirilmekte olan malware istihbarat projeleriile(vt-intelligence[3]) güncel malware örneklerine ulaşarak yapay zekanın kaliteli malware setleri ile eğitilmesi amaçlanmaktadır.

False positiveler beklenen oranda tutulması için, sınıflandırma yapay zekanın insiyatifine bırakılmamalı. Anti-malware yazılımının temel mekanikleri ile desteklenmeli, Sertifikaya sahip olan yazılımlar ve güvenilir kaynaklardan indirilen uygulamalar yapay zekanın yorumuna bakılmadan zararsız olarak işaretlenmelidir. Zararlı ve güvenilir indirme kaynaklarının tespiti için internet trafiğinin filtrelenmesi gerekir. Netfilter entegrasyonu bu noktada gerekli olmaktadır.

Günümüzde, kullanıcı browserını hedef alan ortadaki adam saldırıları, man in the browser (m.i.t.b) sık karşılaşılan saldırı vektörleri arasındadır. Browserın karşılaşabileceği tüm saldırı tiplerine karşı güvende tutulması için kernele entegrasyonu önerilen mimari çizilmiştir. Miniport ile NDIS (Network Driver Interface Specification) hedef alınarak gerçekleştirilen bu saldırı tipinde, arama sonuçlarının farklılaşması durumunda kernel uyarılarak şüpheli driverın sonlandırılması, browserın ilgili dosyalarının disk ve bellek manipulasyonunun önlenmesi amaçlanmaktadır.(Şekil2.3.2).
Bir diğer farklı zararlı tipi olan bootkitler, MBR(masterbootrecord) ve VBR(volumebootrecord) enfekte ederek sisteme yerleşirler. Gerçek mod süresince biosun13h ve 15h interruptlarını manipule ederek dosya sisteminin gizli bölmesine yerleştirilen imzasız driverın çalıştırılması amaçlanır, bu işlemler korumalı moda geçiş öncesi yapıldığından işletim sistemi savunmasızdır. Son olarak orjinal MBR geri yüklenip boot-man- ager(bootmgr) süreci başlatılır.

Rootkit ve Bootkitlerin analizi, Anti-Debug/Anti-Dumping tekniklerinin incelenmesi, NT internaller ile birlikte ileri seviye tersine mühendislik teknikleri içeren derin bir araştırma konusudur. Güvenlik yazılımları ile Rootkitlerin çalışma mantığı neredeyse aynıdır. Her ikiside; sisteme tamamiyle hakimiyet kurmak, ku- rallar neticesinde hafıza akışını kontrol etmek, gizlenme teknikleri i le kendi korumalarını sağlamayı amaçlar.

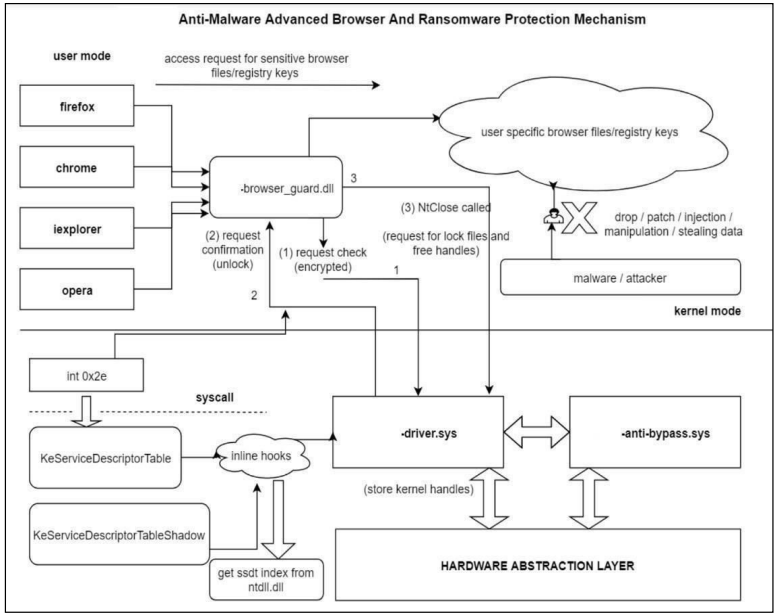


Figure1Şekil2.3.2

Tasarlanan mimari geliştirilerek, kullanıcı için önem arz eden dosyalara ransomware tipi malwarelerin erişimininengellenmesimümkündür. İlgiligüvenlik hattıyürürlüğe girdiğinde, yüksek koruma içermesine karşın bize yaratabileceğien büyük sorunoptimizasyon ve false positiveler olacaktır.

Kullanıcı browserına erişim talep eden zararsız istekler şüpheli olarak nitelendirilip engellenecek, tarayıcının kendi kaynaklarına erişimi için her seferinde kernele erişim açma talebi göndermesi sistem kaynaklarının gereğinden fazla kullanılmasına sebebiyet verecektir.Bu tür yüksek koruma içeren vektörlerin implementasyonu sağlanıp,aktif hale getirilmesi kullanıcının tercihine bırakılmalıdır.

2.4. GenelDeğerlendirme

Mimariye entegre edilen her yeni özelliğin ortaya çıkardığı avantaj ve dezavantajlar vardır. Koruma seviyesinden ödün vermeden false positivelerin tamamiyle önüne geçmek güvenlik yazılımları için olanaksız

Bir sorundur. Yapay zekanın güncel ve kaliteli malware setleri ile eğitilmesi, anti-malware'in temel mekanikleri ile desteklenmesi, güvenilir ve güvenilir olmayan indirme kaynaklarının firewall entegrasyonu ile filtrelenmesi ve iş gücü açısından maliyetli bir süreç olan sandbox alt yapısının tasarlanmasıı le dinamik ana- lizlerin otomatize edilmesi, hatalı sınıflandırmalar ile karşılaşma oranını büyük ölçüde azaltacaktır.

3. PROJENİNSÜREKLİLİĞİ

3.1. Performans

Yoğun network operasyonu gerçekleştiren büyük ölçekli sunucu sistemler için framework servislerinin optimizasyonu sağlanması gerekmektedir. Net-filter servisinin açık trafiği görüntüleyebilmesi için paketlerin şifresini çözmesi gerekmektedir. Kullanıcı trafiğinin yüksek olduğu saatlerde sistemyöneticisi tarafından bu özelliğin devre dışı bırakılması sağlanılabilir. Donanımsal firewall ve load balancer'lar için genellikle böyle bir handikap söz konusu değildir, optimize edilmiş donanım sayesinde sunucuların şifreleme yükü tek bir noktadan sağlanmaktadır.(SSLOffload)

Aynı şekilde, yoğun disk operasyonu gerçekleştiren, database'e giriş/çıkış yükünün fazla olduğu sunucular için tüm sistem aktivitelerinin raporlanması disk üzerinde darboğaz oluşturabilir.

Sorunların önüne geçmek adına sistem yöneticisi framework servislerinin performans ayarları hakkında detaylı olarak bilgilendirilmelidir.

3.2. Uyumluluk

Projenin iki hattının temel fonksiyon testleri ubuntu-server16.04 LTS kernel4.4.0-142-generic ve Cent OS 7 kernel3.10.0-1062.el7 üzerinde sağlandı. Frameworkun 1. ve 2. Hattının diğer kernel versiyonları içinde uyum sağlaması beklenmektedir. 3. Hat olan SE-Linux diğer hatlara göre daha kompleks bir yapı içeriyor. Kernel3.x üzerinde se-linux'un temelyapıları olsa da, kernel4.x için kullanabileceğimiz bu yapılar ortadan kaldırılmıştır.Linux'un kernel headerları ile implementasyon sağlanabilir.

3.3. Sürekdilik

Mimari tasarlanıp tüm fonksiyon testleri tamamlandığında, kullanabilecek yüksek seviyeli fonksiyon tanımlarıkullanıcıuzayına aktarılmalı ve kernel ile ilişkilendirilmelidir. (SDK)

SDK'yı kontrol eden web paneli tasarlanıp, sistem yöneticisinin oturumu ve servisleri yönetebileceği panel fonksiyonları oluşturulmalı. Projenin sürekliliği için, kernel tarafındaki gelişmeler shell arayüzünden güncellenmelidir.

4. Sonuç

Tüm savunma vektörlerini kapsayan yerli bir güvenlik framework'unun tasarımı, sıfırdan tasarlanan bir otomobil modeline benzetilebilir, kernel mimarisi otomobilin motor kısmıdır, kontrol edilebilir ve en temel mekanikleri yerine getirir. Shell ise otomobilin tasarım ve kontrol bölümüdür. Motor ne kadar kaliteli olursa olsun, alıcıya hitap eden öncelikle tasarımdır.

Sistem yöneticilerinin ihtiyaç duyacakları panel; hızlı, sade ve yönetilebilirliği kolay olmalıdır. İhtiyaçlar doğrultusunda, projenin sağlıklı olarak hayata geçebilmesi için, pazar payı büyük majör linux dağıtımları üzerinde geliştirilmeli ve siber güvenlik ihtiyaçlarının kesintiye uğramaması amacıyla sürekli olarak güncelleştirilmesi gerekmektedir.

INTERNET KAYNAKLARI

[1] WannaCry,
https://en.wikipedia.org/wiki/WannaCry_ransomware_attack

[2] IEEE.POSIX.1,
https://fossbytes.com/posix-what-is-the-portable-operating-system-interface/

[3] VT-Intelligence,
https://support.virustotal.com/hc/en-us/articles/360001387057-VirusTotal-Intelligence-Introduction

[4] Kaspersky 2015 D-DOS intelligence report, https://securelist.com/kaspersky-ddos-intelligence-report-for-q4-2015/73414/

[5] KasperskyQ12017Statistics,https://securelist.com/it-threat-evolution-q1-2017-statistics/78475/