Tersine Mühendislik ve CrackMe Uygulamalarında 2025 ve Sonrası İçin En Son ve En Etkili Teknikler/Trendler

I. Giriş

Modern Siber Güvenlik Ortamında Tersine Mühendislik ve CrackMe Uygulamalarına Genel Bakış

Tersine mühendislik (RE), bir sistemin, yazılımın veya donanımın bileşenlerini, tasarımını ve işlevselliğini anlamak için sistematik olarak parçalarına ayrılması sürecidir. Siber güvenlik alanında, tersine mühendislik vazgeçilmez bir disiplin olup, birçok kritik faaliyet için hayati öneme sahiptir. Bu faaliyetler arasında kötü amaçlı yazılımların niyetini ve yeteneklerini ortaya çıkarmak için kapsamlı kötü amaçlı yazılım analizi, zayıflıkları belirlemek ve gidermek için derinlemesine güvenlik açığı araştırması, rakip ürünleri anlayarak fikri mülkiyeti koruma ve güvenlik standartlarına uyumu sağlama yer almaktadır.[1]

CrackMe uygulamaları, hevesli ve deneyimli tersine mühendisler için pratik bir eğitim alanı olarak hizmet veren, kasıtlı olarak tasarlanmış yazılım zorluklarıdır. Bu uygulamalar, çeşitli tersine mühendislik karşıtı teknikleri ve koruma mekanizmalarını içerecek şekilde hazırlanmıştır ve analistleri kod sökme, hata ayıklama ve gizlenmiş kodu açma becerilerini uygulamaya ve geliştirmeye zorlar.[3] Bu uygulamalar, siber güvenlik profesyonelleri için vazgeçilmez olan eleştirel düşünme ve problem çözme yeteneklerinin gelişimini teşvik eden hayati pedagojik araçlardır.[3]

Özellikle 2025 ve sonrası için, çağdaş siber güvenlik ortamı, teknolojik ilerlemenin hızlanan temposuyla karakterize edilmektedir. Yapay Zeka (YZ) ve Makine Öğrenimi (ML), dijital alandaki hem saldırı (ofansif) hem de savunma (defansif) stratejilerini temelden yeniden şekillendiren en önemli itici güçler olarak öne çıkmaktadır.[8] Bu rapor,

bu ilerlemelerin tersine mühendislik ile yazılım koruması arasındaki karmaşık dengeyi nasıl etkilediğini derinlemesine inceleyecektir.

Raporun Amacı ve Kapsamı: 2025 ve Sonrası İçin Temel Trendlerin Belirlenmesi

Bu rapor, 2025 ve ötesinde tersine mühendislik ve CrackMe uygulamaları alanını tanımlayacak en etkili ve en son 10 tekniği ve trendi derinlemesine, uzman düzeyinde analiz etmeyi amaçlamaktadır. Kapsam, hem tersine mühendislik metodolojilerindeki son teknoloji ilerlemeleri (saldırı ve analiz perspektiflerini temsil eden) hem de sofistike tersine mühendislik karşıtı önlemleri (savunma stratejilerini temsil eden) içermektedir. Belirlenen her trend için ayrıntılı bir açıklama sunulacak, 2025'teki tahmini etkisi analiz edilecek ve tüm iddialar, sağlanan araştırma materyallerine yapılan atıflarla desteklenerek kanıta dayalı ve yetkili bir tartışma sağlanacaktır.

II. Tersine Mühendislik ve CrackMe Teknikleri/Trendleri (2025 ve Sonrası)

A. Tersine Mühendislik ve Analizdeki Gelişmeler (Saldırı/Analiz)

1. YZ Destekli Otomatik Tersine Mühendislik ve Gizlenmiş Kodu Açma

Yapay Zeka (YZ), Makine Öğrenimi (ML), Doğal Dil İşleme (NLP) ve Büyük Dil Modellerinin (LLM) entegrasyonu, geleneksel olarak karmaşık ve zaman alıcı görevleri otomatikleştirerek tersine mühendislik alanında devrim yaratmaktadır. Bu paradigma değişimi, otomatik kod analizi, hassas kötü amaçlı yazılım sınıflandırması, sofistike ikili benzerlik analizi, gelişmiş hata ayıklama yetenekleri ve kritik olarak gelişmiş gizlenmiş kodu açma tekniklerini kapsamaktadır.[8] Örneğin, YZ destekli derleyiciler, karmaşık kod

desenlerini tanıyarak makine kodunu üst düzey programlama dillerine geri dönüştürme konusunda giderek daha yetkin hale gelmekte, sinir ağları ise eksik kaynak kodunu tahmin edip mantıksal yapıları daha yüksek doğrulukla yeniden inşa edebilmektedir.[8] Ayrıca, LLM'ler, karmaşık eserleri yoğunlaştırmak ve yorumlamak, fonksiyonları ve ikili dosyaları özetlemek ve kötü amaçlı yazılım analizinde akıllı yardım sağlamak için kapsamlı tersine mühendislik problemleri veri kümeleri üzerinde özel olarak ince ayar yapılmaktadır.[11] Radare2 disassembler'ının bir YZ uzantısı olan r2ai gibi araçlar, Python programları oluşturup yürüterek dize gizlenmiş kodunu açma ve şifre çözme zorluklarını otonom olarak çözerek, insan muhakemesini taklit eden yinelemeli, YZ destekli bir analiz iş akışını sergilemektedir.[12] Tersine mühendislik yazılımı pazarının, otomasyon, desen tanıma ve düşük seviyeli koddan daha yüksek seviyeli soyutlamalar çıkarma yeteneği için YZ/ML'nin yaygın olarak benimsenmesiyle önemli bir büyüme yaşaması beklenmektedir.[9]

Bu trend, verimlilikte yeni bir dönemi başlatmaya hazırlanmaktadır. Analiz iş akışlarında önemli bir hızlanmaya yol açacak ve daha önce ilk sınıflandırma ve karmaşık ikili dosyalara derinlemesine dalışlar için gereken manuel insan çabasını önemli ölçüde azaltacaktır.[8] YZ'nin gelişmiş yetenekleri, analistlerin giderek daha karmaşık gizleme teknikleriyle başa çıkmasını sağlayacak, daha önce çözülemeyen tersine mühendislik problemlerini daha yönetilebilir ve uygun maliyetli hale getirecektir.[8] YZ destekli tersine mühendislik araçları pazarının büyümesi, bu araçların dönüştürücü potansiyelinin altını çizmektedir.[9] Dahası, YZ destekli araçlar, gizli şifreleri otomatik olarak belirleyerek ve gizlenmiş kod mantığını basitleştirerek karmaşık CrackMe zorluklarını çözmede şimdiden oldukça etkili olduğunu kanıtlamaktadır, böylece gelişmiş tersine mühendislik yeteneklerine erişimi demokratikleştirmektedir.[14]

YZ'nin birincil etkisi, yalnızca yeni bir özellik eklemek değil, tersine mühendisliğin gerçekleştirilebileceği *ölçeği* ve *hızı* temelden değiştirmektir. Bu, analitik verimde önemli bir artışa yol açmaktadır. Eğer görevler daha hızlı ve daha hassas hale gelirse, analistler daha yüksek hacimli örnekleri işleyebilir veya en zorlu olanlara daha fazla zaman ayırabilir. Bu durum, bir tersine mühendislik ekibinin analitik kapasitesini etkili bir şekilde artırmaktadır. Bu artan verimlilik, tehdit istihbaratını doğrudan etkilemekte, yeni kötü amaçlı yazılım türlerinin ve güvenlik açıklarının daha hızlı anlaşılmasını sağlamaktadır. Bu, saldırganların fırsat penceresini potansiyel olarak azaltarak, savunucuların gelişen tehditlere daha hızlı yanıt vermesini sağlayan stratejik dengeyi değiştirmektedir.

YZ otomasyon sağlarken, insan rehberliği ve uzmanlığı vazgeçilmez olmaya devam etmektedir, özellikle yeni veya belirsiz senaryolar için. Bu, tamamen otonom bir gelecekten ziyade işbirlikçi bir geleceği işaret etmektedir. Bir YZ destekli analizde,

analiz kalitesi genel olarak YZ yardımı olmadan elde edilenden daha iyi veya eşit olsa da, YZ'nin tek başına çalışamayacağı ve deneyimli bir analist tarafından sürekli olarak yönlendirilmesi gerektiği belirtilmektedir.[12] Bu durum, tersine mühendisliğin geleceğinin insan analistleri YZ ile değiştirmekten ziyade, insan yeteneklerini artırmakla ilgili olduğunu göstermektedir. Analistin rolü, manuel, sıkıcı ayrıştırmadan, YZ araçlarını düzenleme, çıktılarının doğruluğunu onaylama ve YZ belirsizliklerle veya "halüsinasyonlarla" karşılaştığında stratejik yönlendirme sağlama gibi daha üst düzey bir fonksiyona dönüşmektedir. Bu durum, eğitim ve beceri gelişimi için önemli çıkarımlara sahiptir. Gelecekteki tersine mühendisler, güçlü temel tersine mühendislik becerilerine ek olarak, prompt mühendisliği, YZ aracı entegrasyonu ve YZ tarafından üretilen bilgilerin eleştirel değerlendirmesi konusunda uzmanlığa ihtiyaç duyacaklardır. Odak noktası, kaba kuvvet analizinden akıllı düzenlemeye kaymaktadır.

LLM'ler, statik kod üretiminin ötesine geçerek, bir analistin problem çözme sürecinin yönlerini taklit eden ve hatta otomatikleştiren dinamik, yinelemeli tersine mühendislik iş akışlarına aktif olarak katılmaktadır. LLM'ler, tersine mühendislik eserlerini "yoğunlaştırmak ve yorumlamak", "fonksiyon ve ikili özetleme" ve "LLM destekli kötü amaçlı yazılım analizi" için kullanılmaktadır.[11] r2ai'nin "OTOMATİK modu", bir analistin iş akışını insan etkileşimi olmadan gerçekleştirerek "radare2 komutlarını planlayabilir, yürütebilir ve çıktılarını analiz edebilir".[14] Bu, LLM'lerin sadece cevap sağlamakla kalmayıp, aynı zamanda

yinelemeli problem çözme yeteneğine sahip olduğunu göstermektedir: bir plan formüle etme, araçlar (Radare2 gibi) aracılığıyla komutları yürütme, sonuçları analiz etme ve yaklaşımı iyileştirme. Bu, basit sorgu-yanıttan, tersine mühendislik bağlamında daha "ajanik" bir davranışa doğru bir sıçramadır. Bu yetenek, karmaşık tersine mühendislik görevleri için giriş engelini önemli ölçüde düşürebilir, daha az deneyimli analistlerin daha önce derin uzmanlık gerektiren zorluklarla başa çıkmasını sağlayabilir veya analizin "ilk geçişini" otomatikleştirerek hızlı ilk değerlendirmelere olanak tanıyabilir.

2. Modern Dillerin (Rust ve Go) Gelişmiş Kötü Amaçlı Yazılım Analizi

Siber güvenlik ortamı, kötü amaçlı yazılım geliştirme için Rust ve Go gibi modern programlama dillerine yönelik saldırgan tercihlerinde dikkate değer bir kayma yaşamaktadır. Bu benimseme, bu dillerin doğal performans avantajları, sağlam eşzamanlılık özellikleri ve geleneksel tersine mühendislik ve analiz araçları için oluşturdukları doğal zorluklar gibi çeşitli faktörlerden kaynaklanmaktadır. Sonuç olarak,

bu dillerle derlenmiş ikili dosyaları ayrıştırmak için özel tersine mühendislik teknikleri hayati önem kazanmaktadır. 2025 için eğitim programları, Rust ve Go tersine mühendisliğinin inceliklerini öğrenmeye açıkça odaklanmaktadır.[15] Bu özel analiz, benzersiz derleme süreçlerini ve çalışma zamanı mekaniklerini anlamayı, soyulmuş ikili dosyalarla (hata ayıklama sembolleri kaldırılmış, analizi önemli ölçüde zorlaştıran yürütülebilir dosyalar) etkili bir şekilde başa çıkmayı, bu dillere özgü gelişmiş gizleme tekniklerinde gezinmeyi ve içlerinde uygulanan sofistike kötü amaçlı yazılım tekniklerini ayrıştırmayı içermektedir. Bu tür tekniklere örnek olarak Rust ikili dosyalarındaki süreç boşaltma (process hollowing), API kanca (API hooking), DLL enjeksiyonu, yükleyici enjeksiyonu ve özel paketleyicilerin kullanımı verilebilir.[16] Go ikili dosyaları için analiz, Goroutines (hafif eşzamanlılık birimleri), kanallar (Goroutineler arası iletişim için), senkronizasyon ilkelleri (örn. bekleme grupları, mutexler), bellek yönetimi paradigmaları ve yansıma yetenekleri gibi karmaşık özelliklerin anlaşılmasını kapsar; bunların hepsi programın gerçek davranışını anlamak için temeldir.[16]

Rust ve Go ikili dosyalarını etkili bir şekilde tersine mühendislik yapma yeteneği, siber güvenlik profesyonelleri için kritik bir yetenek haline gelmektedir. Bu uzmanlık, zamanında ve doğru tehdit istihbaratı geliştirmek, kötü amaçlı yazılım algılama yeteneklerini önemli ölçüde iyileştirmek ve gelişen tehditlere karşı koyabilecek sağlam kötü amaçlı yazılım karşıtı çözümler tasarlamak için esastır.[15] Bu diller hem meşru yazılım geliştirmede hem de kötü amaçlı kampanyalarda ilgi görmeye devam ettikçe, tersine mühendislik konusunda yetkin analistlere olan talep artacaktır. Bu özel bilgi, savunucuların kötü şöhretli Sandworm APT grubu tarafından kullanılan Go tabanlı BACKORDER yükleyicisi gibi yeni kötü amaçlı yazılım ailelerinin iç işleyişine ilişkin önemli bilgiler edinmesini sağlamaktadır.[17]

Saldırgan araç setlerindeki değişim, doğrudan özel tersine mühendislik becerilerine olan ihtiyacı belirlemekte ve yeni eğitim ve uzmanlık talepleri yaratmaktadır. 2025 eğitimlerinde "Saldırganlar neden kötü amaçlı yazılım geliştirmek için Rust kullanıyor?" ve "Saldırganlar neden kötü amaçlı yazılım geliştirmek için Go kullanıyor?" gibi modüllerin açıkça yer alması, bu dillerin kötü amaçlı yazılım geliştirmede artan kullanımını göstermektedir.[15] Bu durum, genellikle C/C++ veya montaj diline odaklanan geleneksel tersine mühendislik becerilerinin yetersiz kaldığı anlamına gelmektedir. Tersine mühendislerin, Rust ve Go'nun benzersiz derleme, çalışma zamanı ve gizleme özelliklerinde uzmanlaşması zorunlu hale gelmektedir. Bu, saldırganların dil seçimi inovasyonunun savunma uzmanlaşmasını zorladığı açık bir neden-sonuç ilişkisidir. Özel eğitim kurslarının ortaya çıkışı [15], tanınan bir beceri açığını göstermektedir. Bu durum, bu dilleri daha iyi desteklemek için yeni, özel tersine mühendislik araçlarının veya mevcut araçların (Ghidra/IDA gibi) geliştirmelerinin hızlanmasına yol açacaktır.

Go için "Goroutines, kanallar, senkronizasyon (bekleme grupları, mutexler)" ve "bellek yönetimi ve yansıma" gibi belirli gelişmiş konuların [16] yanı sıra Rust için "veri yapılarını (Option, Result enumları, dilimler, yapılar) anlama" ve "kontrol akışını analiz etme: Fonksiyonlar, metotlar, döngüler, koşullar" [16] gibi konuların ele alınması, bu dillerin üst düzey özelliklerinin montaj diline karmaşık eşlemeleri olduğunu göstermektedir. Sadece kodu ayrıştırmak, programın gerçek mantığını anlamak veya güvenlik açıklarını belirlemek için yeterli değildir. Tersine mühendislerin, bu dil yapılarının

semantik çıkarımlarını ve çalışma zamanında nasıl davrandıklarını kavramaları gerekmektedir. Örneğin, Goroutinelerin kanallar aracılığıyla nasıl etkileşim kurduğunu anlamak, Go kötü amaçlı yazılımlarını analiz etmek için kritiktir ve bu, sadece ham montaj kodundan çıkarılamaz. Bu durum, bu üst düzey dil kavramlarını düşük seviyeli ikili dosyalardan doğru bir şekilde soyutlayabilen daha sofistike derleyicilere ve ara gösterimlere (IR'ler) ihtiyaç duyulmasını gerektirmektedir. Bu, otomatik analiz araçlarının sınırlarını zorlayarak, sadece "CPU'nun ne yaptığı" yerine "programın ne yapmayı amaçladığı" hakkında daha anlamlı, insan tarafından okunabilir çıktı sağlamalarını sağlamaktadır.

3. Otomatik Güvenlik Açığı Keşfi (Fuzzing ve Sembolik Yürütme)

Güvenlik açığı keşfi alanı, giderek artan bir şekilde, başta fuzzing ve sembolik yürütme olmak üzere yüksek düzeyde otomatik ve akıllı teknikler tarafından domine edilmektedir. AFL, QEMU ve Frida gibi gelişmiş fuzzing araçları, çeşitli test durumları oluşturmak, yazılım güvenlik açıklarını ve çökmeleri hızla belirlemek için kullanılmaktadır.[8] Güçlü bir program analiz tekniği olan sembolik yürütme, güvenlik açıklarını belirlemek için tüm olası yürütme yollarını sistematik olarak keşfetmekte ve yeni geliştirilen dinamik sembolik yürütme (DSE) karşıtı opak predikatlar da dahil olmak üzere sofistike gizleme tekniklerine etkili bir şekilde karşı koymak için sürekli olarak geliştirilmektedir.[19] Bu otomatik yöntemler, sürekli genişleyen saldırı yüzeyi ve saldırganların kendi YZ destekli tarama yeteneklerini benimsemeleri nedeniyle endişe verici bir şekilde artan sıfır gün güvenlik açıklarını belirlemek için vazgeçilmez hale gelmektedir.[10] Ayrıca, ARM TrustZone'daki Güvenli Monitör Çağrıları (SMC) gibi kritik ve karmaşık saldırı yüzeylerine özel fuzzing teknikleri uygulanarak Güvenilir Uygulamalar (TA'lar) içindeki güvenlik açıkları ortaya çıkarılmaktadır.[22]

Bu otomatik tekniklerin yaygın olarak benimsenmesi, daha önce bilinmeyen güvenlik açıklarının (sıfır gün) keşfini önemli ölçüde hızlandıracak, böylece proaktif yazılım

denetimini artıracak ve güvenlik testi metodolojilerini güçlendirecektir.[8] Bu, güvenlik araştırmacılarının ve savunucularının kritik kusurları kötü niyetli aktörler tarafından istismar edilmeden önce belirlemesini ve gidermesini sağlayarak daha sağlam ve dirençli yazılım sistemlerinin geliştirilmesine yol açmaktadır. Güvenilir Yürütme Ortamlarındaki (TEE'ler) SMC'ler gibi karmaşık, düşük seviyeli arayüzleri etkili bir şekilde fuzzing yapma yeteneği, yüksek düzeyde hassas bileşenleri ve verileri güvence altına almak için kritik yeni yollar açmaktadır.[22]

Saldırganların güvenlik açığı keşfi için YZ ve otomasyonu giderek daha fazla kullanması, güvenlik araştırmacıları ve savunucularından eşit derecede otomatik ve sofistike bir savunma yanıtını doğrudan gerektirmektedir. Saldırganların "güvenlik ekipleri yamalamadan önce bu çatlakları keşfetmek için otomasyon ve YZ destekli taramadan yararlandığı" açıkça belirtilmektedir.[10] Aynı zamanda, YZ'nin "güvenlik açığı değerlendirmesi" ve "YZ tabanlı fuzzing araçları"ndaki rolü vurgulanırken [8], AFL ve QEMU gibi araçların "hata avcılığı" için kullanıldığı da belirtilmektedir.[18] Sembolik yürütmedeki gelişmeler, yeni gizleme tekniklerine

karşı koymayı amaçlamaktadır [19], bu da gizlenmiş kodda güvenlik açıkları bulmak için kullanıldığını ima etmektedir. Bu, saldırganların otomasyonu güvenlik açığı bulmak için kullandığını ve savunucuların da aynı amaçla otomasyonu kullandığını göstermektedir. Bu, açık bir neden-sonuç ilişkisidir. Saldırganların otomasyonu (YZ destekli tarama) kullanması, savunma tarafının benzer otomatik teknikleri (fuzzing, sembolik yürütme) benimsemesini

zorlamaktadır ki bu da rekabet avantajını korumak ve güvenlik açığı penceresini azaltmak için gereklidir. Bu, keşif hızı ve ölçeğinin hayati önem taşıdığı bir silahlanma yarışıdır. Kuruluşlar için bu, sıfır gün saldırılarının hacmi ve hızının geleneksel yöntemleri alt edeceği için otomatik güvenlik açığı keşif platformlarına ve bunları işletme uzmanlığına önemli ölçüde yatırım yapmaları gerektiği anlamına gelmektedir.

Sembolik yürütme, sadece hata bulma aracı olmakla kalmayıp, aynı zamanda karmaşık kodu, özellikle yeni, dirençli gizleme yöntemlerine karşı gizlenmiş kodu açmak için kritik bir teknik haline gelmektedir. "Kodu Gizleme Gelişmeleri: Dinamik Sembolik Yürütmeye Karşı Yeni Opak Predikat Teknikleri" başlıklı belgeler [19], yeni gizlemenin amacının belirli bir analiz tekniğine (DSE) "karşı koymak" olduğunu açıkça belirtmektedir. Sanallaştırma tabanlı koruyucuları gizlemek için "sanal talimat çıkarma" için sembolik yürütme kullanımı da tartışılmaktadır.[21] Bu, gizleme ile sembolik yürütme arasında daha derin, birbirine bağımlı bir ilişkiyi ortaya koymaktadır. Sembolik yürütme, belirli gizleme tekniklerini "kırabilen" güçlü bir analitik araçtır. Buna karşılık, gizleme tasarımcıları, sembolik yürütmeyi engellemek için özel olarak tasarlanmış yeni yöntemler (anti-DSE

opak predikatlar gibi) geliştirmektedir. Bu, bir alandaki gelişmelerin diğerindeki inovasyonu doğrudan tetiklediği sürekli bir döngü oluşturmakta, sembolik yürütmeyi tersine mühendislik ve tersine mühendislik karşıtlığı çatışmasında önemli bir savaş alanı haline getirmektedir. Bu durum, geleneksel olarak güvenlik açığı araştırması için kullanılan araçlar ve kötü amaçlı yazılım analizi/gizlenmiş kodu açma için kullanılan araçların yakınlaşmasını önermektedir. Gelecekteki tersine mühendislik platformları, sadece hata avcılığı için değil, aynı zamanda gizlenmiş kodu açma motorlarının temel bir bileşeni olarak gelişmiş sembolik yürütme yeteneklerini entegre edecektir.

4. Bellek Analizi ve Çalışma Zamanı Manipülasyonu

Yazılım giderek dinamik davranışları benimsedikçe ve sofistike analiz karşıtı teknikler kullandıkça, doğrudan bellek analizi ve çalışma zamanı manipülasyonu, etkili tersine mühendislik için vazgeçilmez hale gelmektedir. Bu, program davranışını bellekte yürütülürken anlamak, gizli veya dinamik olarak oluşturulan kod yollarını belirlemek ve daha derin bir anlayış veya hatta istismar için çalışma zamanı yürütmesini manipüle etmek için bir dizi tekniği içermektedir.[5] Temel yönler arasında güvenli enklavların (Sanallaştırma Tabanlı Güvenlik, VBS'de kullanılanlar gibi) belleğe nasıl eşlendiğini ve yüklendiğini analiz etmek ve kötü amaçlı iş parçacıklarının bu enklavlar içinde uyku halindeyken gizli ve korunmuş kaldığı "uyku maskeleme" gibi gelişmiş kaçınma taktiklerini anlamak yer almaktadır.[22] Ayrıca, gelişmiş bellek analizi, ayırt edici değerler veya bilinen işaretçiler olmasa bile, nesnelerin tanımlanabilir yapısal özelliklerine göre konumlandırılmasına olanak tanımakta, çalışma zamanında karmaşık veri yapılarını analiz etmek için yeni bir bakış açısı sağlamaktadır.[23] Geleneksel olarak oyun hileleri için kullanılan Cheat Engine gibi araçlar da, tam açma gerektirmeden bellek hileleri ve paketlenmiş programları yamalama için kullanılmaktadır, bu da çalışma zamanı manipülasyonunun çok yönlülüğünü göstermektedir.[5] Bu yaklaşım, genellikle gizleme ve beyaz kutu kriptografisine dayanan Dijital Haklar Yönetimi (DRM) uygulamalarını atlamak için özellikle önemlidir; çalışma zamanı bellek analizi, temel kriptografik sırları elde etmek için kritik bir vektör olabilir.[23]

Bu dinamik yaklaşım, yalnızca yürütme sırasında gerçek işlevselliğini ortaya çıkarabilecek veya statik analizin tek başına üstesinden gelemeyeceği çalışma zamanı korumalarını atlamak için yüksek düzeyde dinamik veya analiz karşıtı kötü amaçlı yazılımları analiz etmek için vazgeçilmez hale gelmektedir. Özellikle donanım destekli güvenlik özellikleri veya yoğun şekilde gizlenmiş kod içeren senaryolarda, karmaşık yazılım etkileşimleri hakkında daha derin, daha doğru bir anlayış sağlamaktadır.[22] İkili

analizin bellek keşfi ile entegrasyonu, tersine mühendislik tekniklerinin geleceği için umut verici bir yön olarak açıkça tanımlanmaktadır, daha kapsamlı ve etkili analiz sağlamaktadır.[23]

Statik gizleme ve sanallaştırma tekniklerinin artan karmaşıklığı, tersine mühendisliği, program davranışını anlamanın en etkili yolu olarak dinamik, çalışma zamanı analizine yöneltmektedir. Kod yoğun bir şekilde gizlenmiş veya sanallaştırılmışsa, geleneksel statik analiz gerçek mantığını ortaya çıkarmakta zorlanır.[7] Bellek analizi ve ikili analizin bellek keşfi ile entegrasyonu "umut verici bir yön" olarak tartışılmaktadır.[23] Cheat Engine'in "bellek hackleme" ve "paketlenmiş programları açmadan yamalama" için kullanılması da belirtilmektedir.[5] Bu durum, statik analiz yetersiz kaldığında, bellekteki dinamik analizin kritik hale geldiğini göstermektedir. Statik anti-RE teknikleri ne kadar etkili olursa, bellek analizi gibi dinamik RE tekniklerinin o kadar gerekli ve sofistike olması gerektiği arasında doğrudan bir neden-sonuç bağlantısı vardır. Programın gerçek mantığı, yürütme sırasında bellekte kendini göstermelidir, bu da çalışma zamanı gözlemini statik korumalar için nihai bir atlama yolu haline getirmektedir. Bu, gelişmiş hata ayıklayıcılar, bellek adli araçları ve gizlice çalışabilen ve zengin çalışma zamanı bilgileri sağlayabilen dinamik enstrümantasyon çerçeveleri dahil olmak üzere gelişmiş dinamik analiz araçlarına sürekli yatırım yapılmasını gerektirmektedir.

Bellek etkileşimlerini ve güvenli enklavların çalışma zamanı davranışını anlamak, donanım destekli güvenlik özellikleriyle korunan sistemlere saldırmak ve tersine mühendislik yapmak için kritik bir vektör haline gelmektedir. Enklavların belleğe nasıl eşlendiği ve yüklendiği ile "uyku maskeleme" gibi teknikler [22] ve "ARM TrustZone" kullanan platformlardaki "DRM uygulamaları" üzerinden "donanım DRM anahtarlarını elde etmenin beklenenden daha kolay olabileceği" tartışılmaktadır.[23] Bu durum, donanım enklavlarının güçlü izolasyon sağlamasına rağmen, sistemin geri kalanıyla etkileşimlerinin (örn. bellek eşlemesi, veri transferi, iş parçacığı durumları) güvenlik açıkları yaratabileceğini vurgulamaktadır. Bellek analizi, tersine mühendislerin bu çalışma zamanı etkileşimlerini gözlemlemesine ve potansiyel olarak manipüle etmesine olanak tanır, hatta enklavın dahili kodu opak olsa bile. Bu, "güvenli" sınırın, donanımı kırmakla değil, bellekteki yazılım arayüzünü istismar ederek aşılabileceği anlamına gelmektedir. Bu durum, güvenliğin sadece donanımın kriptografik veya izolasyon özelliklerini değil, aynı zamanda potansiyel saldırı yüzeylerini ortaya çıkaran yazılım ve bellek etkileşimlerini de dikkate alan bütünsel bir güvenlik yaklaşımına ihtiyaç duyulduğunu vurgulamaktadır. Savunucular için bu, güvenli enklavlar içindeki uygulamaların çalışma zamanı davranışını dikkatle incelemek anlamına gelmektedir.

5. Donanım Destekli Tersine Mühendislik ve İstismar

Bu yükselen trend, hem tersine mühendislik hem de doğrudan saldırılar için düşük seviyeli donanım özelliklerinden ve doğal güvenlik açıklarından yararlanmayı veya bunları istismar etmeyi içermektedir. Önemli bir örnek, Intel CPU'larında (1. nesil Intel Core işlemcilerden itibaren) tanımlanan "SPOILER" spekülatif yürütme kusurudur. Bu kusur, fiziksel sayfa eşlemeleri hakkında kullanıcı alanı süreçlerine kritik bilgiler sağlamaktadır. Bu mikro mimari sızıntı, bir web tarayıcısı içindeki kötü amaçlı JavaScript, kötü amaçlı yazılım veya yasa dışı oturum açmış kullanıcılar tarafından istismar edilebilir. Bu, sanal makineler ve sanal alanlı ortamlar içinde bile Rowhammer ve önbellek saldırılarının verimliliğini önemli ölçüde artırmaktadır.[25] Ayrıca, Intel SGX (Software Guard Extensions) ve ARM TrustZone gibi donanım destekli güvenlik özelliklerindeki güvenlik açıkları aktif olarak araştırılmakta ve istismar edilmektedir. Örneğin, ARM TrustZone içindeki Güvenilir Uygulamalara (TA'lar) Güvenli Monitör Çağrıları (SMC) fuzzing yoluyla veya Sanallaştırma Tabanlı Güvenlik (VBS) enklavlarındaki bellek işlemeyi istismar ederek yapılan saldırılar, kritik saldırı vektörleri olarak ortaya çıkmaktadır.[22] Yazılımın ötesinde, donanım tersine mühendisliği de ilerlemekte, YZ çip düzenlerini analiz etmek, mikroskobik görüntülerden devre şemalarını yeniden oluşturmak ve sahte bileşenleri tespit etmek gibi görevleri otomatikleştirmek için uygulanmaktadır.[8] Gömülü sistemlerdeki (EV şarj cihazları gibi) bellenimi tersine mühendislik yaparak istismarları keşfetme ve güncellemeler boyunca kalıcılık sağlama yeteneği, tersine mühendisliğin geleneksel yazılım ikili dosyalarının ötesine genişleyen kapsamını vurgulamaktadır.[23]

Bu trend, özellikle ayrıcalıklı erişim elde etmek veya hassas verileri sızdırmak için, daha önce güvenli kabul edilen ortamlarda bile tamamen yeni ve yüksek etkili saldırı yüzeyleri açmaktadır.[25] Donanım destekli güvenlik varsayımlarını temelden sorgulamakta, hem saldırgan hem de savunma siber güvenlik stratejileri için donanım-yazılım ortak tasarımının daha derin, entegre bir şekilde anlaşılmasını gerektirmektedir. Bellenimi tersine mühendislik yapma ve çok çeşitli IoT cihazlarındaki güvenlik açıklarını istismar etme yeteneği [23], tersine mühendislik alanının geleneksel bilgi işlem sistemlerinin ötesine genişlediğini vurgulamaktadır. YZ'nin donanım tersine mühendisliği ile yakınlaşması [8], fiziksel analizin karmaşık yönlerini otomatikleştirmeyi, daha erişilebilir ve verimli hale getirmeyi vaat etmektedir.

Yazılım savunmaları olgunlaştıkça, saldırganlar giderek artan bir şekilde odaklarını temel donanım ve bellenim katmanlarına kaydırmaktadır, bu da yüksek ayrıcalıklı erişim ve kalıcı dayanak noktaları sunmaktadır. Intel CPU'larındaki "SPOILER" kusuru, sanal

alanlı ortamlardan bile saldırılara izin vermektedir.[25] ARM TrustZone ve VBS enklavlarındaki güvenlik açıkları ile IoT/otomotiv sistemleri için bellenim tersine mühendisliği de tartışılmaktadır.[22] Ecovacs Robotlarının hacklenmesi de belirtilmektedir.[27] Bu, saldırı ortamında stratejik bir kaymayı göstermektedir. Yazılım düzeyindeki savunmalar sağlamlaştığında, sofistike saldırganlar için mantıklı bir sonraki adım, temel katmanları hedeflemektir. Donanım kusurlarını veya bellenim güvenlik açıklarını istismar etmek genellikle yazılım katmanından tespit edilmesi ve düzeltilmesi daha zor olan daha derin kontrol, kalıcılık ve kaçınma yetenekleri sağlamaktadır. Bu durum, tersine mühendislik uzmanının mikro mimari, gömülü sistemler, düşük seviyeli donanım-yazılım etkileşimi ve tedarik zinciri güvenliği konularında beceri setini önemli ölçüde genişletmesini gerektirmektedir. Ayrıca, donanım ve bellenim tedarik zincirinin tamamını güvence altına almanın kritik önemini vurgulamaktadır, çünkü bu aşamalarda ortaya çıkan güvenlik açıkları domino etkisi yaratabilir.

Güvenlik özelliklerinin kendisi, korumayı artırmak için tasarlanmış olmasına rağmen, uygulamalarında kusurlar içeriyorsa veya yan kanal saldırılarına karşı savunmasızsa, paradoksal olarak yeni, yüksek değerli saldırı yüzeyleri haline gelebilir. Intel SGX ve TSX'in *kötü amaçlı yazılımları gizleyebileceği* ancak aynı zamanda tüm Intel nesillerini etkileyen "SPOILER" kusurunun ayrıntılarını da içerdiği belirtilmektedir.[25] ARM TrustZone ve VBS enklavlarındaki güvenlik açıkları da tartışılmaktadır.[22] Bu, kritik bir paradoks yaratmaktadır. "Güven kaynağı" veya güvenli yürütme ortamı sağlamayı amaçlayan mekanizmalar, tasarım veya uygulamada kusurluysa, çekici hedefler haline gelebilir. Saldırganlar, bu kusurları, en güçlü savunma katmanı olarak algılananı atlamak için kullanmaktadır. Bu, donanıma duyulan

güvenin kötüye kullanılabileceği anlamına gelmektedir. Bu durum, güvenlik profesyonellerinin donanım güvenlik iddialarını doğrulamak için daha şüpheci ve titiz bir yaklaşım benimsemeleri gerektiğini ima etmektedir. Güvenli donanım-yazılım ortak tasarımına ve mikro mimari saldırılar ile yan kanallara yönelik sürekli araştırmalara duyulan ihtiyacı vurgulamaktadır, çünkü bunlar genellikle donanım destekli güvenliği zayıflatmak için kullanılan vektörlerdir.

B. Tersine Mühendislik Karşıtlığı ve Yazılım Koruması (Savunma)

6. Gelişmiş Kod Gizleme (Anti-DSE Opak Predikatlar ve Kontrol Akışı Düzleştirme)

Kod gizleme, yazılımı tersine mühendisliğe ve yetkisiz analize karşı korumak için temel ve gelişen bir teknik olmaya devam etmektedir. 2025 için beklenen önemli bir gelişme, Dinamik Sembolik Yürütme (DSE) saldırılarına direnmek için özel olarak tasarlanmış yeni opak predikatların geliştirilmesi ve yaygın olarak benimsenmesidir.[19] Bu "anti-DSE opak predikatlar", sembolik yürütme motorlarının çözmesi için hesaplama açısından zor veya pratik olarak imkansız olan kısıtlamalar oluşturmak için tek yönlü fonksiyonlardan (örn. kriptografik hash fonksiyonları, logaritmik hesaplamalar) yararlanarak dirençlerini sağlamaktadır.[19] Ek olarak, "yol patlaması opak predikatları", kasıtlı olarak aşırı sayıda yürütme yolu oluşturarak otomatik analiz araçlarını bunaltmak ve felç etmek için tasarlanmıştır.[20] İyi bilinen bir gizleme tekniği olan kontrol akışı düzleştirme, yapılandırılmış kontrol akışı yapılarını karmaşık, durum makinesi benzeri uygulamalarla değiştirmeye devam etmektedir, genellikle karmaşık switch ifadeleri kullanarak. Bu dönüşüm, orijinal yürütme sırasını etkili bir şekilde gizler ve kod blokları arasında karmaşık karşılıklı bağımlılıklar oluşturarak statik analizi son derece zorlaştırır.[24] Bu gelişmiş teknikler, program yürütme performansları üzerindeki etkilerini en aza indirirken kontrol akışı karmaşıklığını artırmak için titizlikle tasarlanmıştır.[19]

Bu gelişmiş gizleme tekniklerinin yaygınlaşması, tersine mühendisler ve kötü amaçlı yazılım analistleri için otomatik analizle ilişkili zorluğu, zamanı ve maliyeti önemli ölçüde artıracak, özel gizlenmiş kodu açma araçlarına ve kritik olarak insan uzmanlığına daha fazla bağımlılık gerektirecektir.[12] Birincil amaç, fikri mülkiyeti koruyarak, rekabetçi analizi engelleyerek ve kötü amaçlı yazılımların analizini geciktirerek tersine mühendislik çabalarını yavaşlatmak veya tamamen caydırmaktır. Gizleme mekanizmalarının sürekli evrimi, "CrackMe" uygulamalarının bu yeni, daha dirençli yöntemleri hızla içereceği ve hevesli tersine mühendisler için giderek daha zorlu ve gerçekçi eğitim senaryoları sağlayacağı anlamına gelmektedir.[4]

Gelişmiş gizlemenin temel amacı, tersine mühendisliği imkansız hale getirmek değil, saldırgan için ekonomik olarak uygulanamaz veya aşırı zaman alıcı hale getirmektir. Gizleme, "analiz karmaşıklığını artırır" ve "kötü amaçlı yazılımın işlevselliğini anlamak için gereken zamanı ve kaynakları artırır".[24] "Yol patlaması opak predikatları" gibi teknikler, "sembolik yürütme motorlarını bunaltmak" için tasarlanmıştır.[19] Bu, gelişmiş gizlemenin arkasındaki stratejinin genellikle ekonomik olduğunu göstermektedir. "Saldırı maliyetini" (zaman, bilgi işlem kaynakları ve insan uzmanlığı açısından) önemli ölçüde artırarak, savunucular en kararlı ve iyi kaynaklara sahip düşmanlar dışındaki herkesi caydırmayı hedeflemektedir. Bu, mutlak bir engelden ziyade bir yıpratma savaşıdır. Kuruluşlar için bu, yazılım korumalarının etkinliğinin, saldırganlara uyguladıkları "maliyet" ile doğrudan ilişkili olduğu anlamına gelmektedir. Gelişmiş

gizlemeye yatırım yapmak, varlıklarını daha az çekici hedefler haline getirmek için gereken çabayı artırarak stratejik bir karardır.

Yeni gizlenmiş kodu açma teknikleri (YZ destekli DSE gibi) hemen yeni, hedefe yönelik gizleme karşı önlemlerinin geliştirilmesini tetiklemekte, bu da hızlı ve reaktif bir silahlanma yarışını göstermektedir. "Kodu Gizleme Gelişmeleri: Dinamik Sembolik Yürütmeye Karşı Yeni Opak Predikat Teknikleri" [19] başlığı, yeni gizlemenin amacının belirli bir analiz tekniğine (DSE) "karşı koymak" olduğunu açıkça belirtmektedir. Bu, tersine mühendislik silahlanma yarışında açık, hızlı ve reaktif bir döngüyü göstermektedir. Tersine mühendisler daha etkili araçlar (örn. gelişmiş DSE, YZ destekli gizlenmiş kodu açma) geliştirdikçe, gizleme tasarımcıları hemen bu yeni araçları engellemek için özel olarak tasarlanmış yeni teknikler geliştirerek yanıt vermektedir. Bu, yazılım anlayışı savaşının dinamik ve sürekli gelişen kalmasını sağlamaktadır. Bu durum, hem gizleme hem de gizlenmiş kodu açma alanında sürekli araştırma ve geliştirme ihtiyacını vurgulamaktadır. Her iki taraf da başarılarıyla yetinemez, çünkü bir tarafın herhangi bir önemli atılımı, diğer taraftan hızla karşı önlemlerle karşılanacaktır. Bu aynı zamanda CrackMe uygulamalarının en son savunma tekniklerini yansıtacak şekilde sürekli güncellenmesi gerektiği anlamına gelmektedir.[7]

7. Kurcalama Önleme ve Tespit Kaçınma Teknikleri

Saldırganlar, güvenlik ürünlerini, özellikle de genellikle ele geçirilmiş bir sistemdeki son savunma hattı olan Uç Nokta Tespit ve Yanıt (EDR) ve antivirüs (AV) çözümlerini devre dışı bırakmak veya atlatmak için sofistike teknikleri durmaksızın yenilemektedir. Temel yöntemler arasında, saldırganların belirli olaylar üzerine güvenlik yazılımına gönderilen sistem bildirimlerini (geri aramalar) ustaca atladığı **Geri Çağırma Kaçınması** yer almaktadır.[28] Bu kritik kancaları devre dışı bırakarak veya kaldırarak, saldırganlar faaliyetlerini maskeleyebilir ve uzun süreler boyunca tespit edilmeden çalışabilir. Bir diğer giderek yaygınlaşan ve tehlikeli taktik ise, ayrıcalıklı erişim elde etmek ve ardından güvenlik bileşenlerini devre dışı bırakmak veya manipüle etmek için meşru, dijital olarak imzalanmış sürücülerdeki bilinen güvenlik açıklarını istismar eden

"Kendi Güvenlik Açığı Olan Sürücünüzü Getirin" (BYOVD) tekniğidir.[17] Bu teknik, imzalı sürücülere duyulan doğal güveni kullanmaktadır.

ntdll.dll içindeki EtwEventWrite fonksiyonunu yamalamak gibi **Windows için Olay İzleme (ETW) kurcalaması**, günlük akışlarını bozmak, güvenlik sistemlerini ve EDR

çözümlerini kötü amaçlı faaliyetlere karşı etkili bir şekilde kör etmek için kullanılmaktadır.[28] Bu gelişen tehditlere doğrudan yanıt olarak, Bitdefender gibi önde gelen güvenlik satıcıları gelişmiş kurcalama önleme özellikleri uygulamaktadır. Bunlar arasında, güvenlik bileşenlerinin yetkisiz değiştirilmesini veya sonlandırılmasını önlemek için dosya, kayıt defteri ve süreç düzeylerinde çalışan "Kendi Kendini Koruma" işlevselliği yer almaktadır.[28] Ayrıca, Geri Çağırma Kaçınma Tespiti (CBE) gibi sofistike tespit mekanizmaları, şüpheli faaliyetleri aktif olarak izlemekte ve kritik geri aramaları devre dışı bırakma veya manipüle etme girişimlerini uyarmaktadır.[28] BYOVD ile mücadele, bilinen güvenlik açığı olan sürücülerin sürekli veritabanı güncellemelerini ve çok katmanlı bir savunma yaklaşımını içermektedir.[28] ETW kurcalama tespiti de gelişmiş tehdit kontrol sistemlerine entegre edilmektedir.[28]

Saldırganlar ve savunucular arasındaki bu tırmanan kedi-fare oyunu yoğunlaşacak, kuruluşların sağlam ve kurcalanmamış uç nokta güvenliğini sürdürmesini giderek daha zorlu hale getirecektir.[17] Saldırganlar, kendilerini durdurmak için tasarlanmış güvenlik kontrollerini zayıflatarak gizlilik, kalıcılık ve ayrıcalık yükseltme elde etmek için yeni yollar aramaya devam edecektir. Savunucular, tehditleri proaktif olarak azaltmak ve güvenlik çözümlerinin bütünlüğünü ve operasyonel sürekliliğini sağlamak için yüksek düzeyde gelişmiş, YZ tabanlı kendini koruma yetenekleri, otomatik düzeltme ve sıfır güven güvenlik modelleri dağıtmak zorunda kalacaklardır.[29] AV-Comparatives gibi bağımsız kuruluşlar tarafından yürütülen düzenli kurcalama önleme sertifikasyon testleri, bu sağlam savunmaların kalıcı kaçınma girişimleri karşısındaki kritik önemini ve devam eden ihtiyacını vurgulamaktadır.[30]

Saldırganlar, güvenlik yazılımının kendisini devre dışı bırakmaya veya alt etmeye giderek daha fazla odaklanmaktadır, bu da kurcalama önleme özelliğini kritik, vazgeçilmez bir temel savunma haline getirmektedir. "Saldırganlar, araçları devre dışı bırakmaya veya değiştirmeye ve uç nokta güvenlik ürünlerinin ana yeteneklerinden kurtulmaya çalışmaktadır".[30] Geri Çağırma Kaçınması ve BYOVD gibi "Doğrudan Devre Dışı Bırakma Teknikleri" ve "Gelişmiş Kaçınma Teknikleri" açıkça güvenlik yazılımını hedef almaktadır.[28] Bu, saldırganlar için stratejik bir kaymayı temsil etmektedir. Sadece tespiti atlatmaya çalışmak yerine, tespit mekanizmasını tamamen

kaldırmaya çalışmaktadırlar. Bu, güvenlik ürünlerinin etkili olabilmesi için öncelikle kendilerini koruyabilmeleri gerektiği anlamına gelmektedir. Kurcalama önleme artık ikincil bir özellik değil, birincil, temel bir güvenlik gereksinimidir. Bu durum, güvenlik çözümlerinin kendini iyileştirme yetenekleri, güçlü bütünlük kontrolleri ve doğrudan saldırılara karşı dirençli olmalarını sağlayan işletim sistemi düzeyinde korumalar (çekirdek düzeyinde çalışan minifiltre sürücüleri gibi [28]) içermesi ihtiyacını artırmaktadır. Ayrıca, bu kendini koruma yeteneklerini doğrulamak için bağımsız test ve

sertifikasyonun [30] önemini vurgulamaktadır.

Saldırganlar, güvenlik kontrollerini atlatmak için meşru sistem bileşenlerine (örn. imzalı sürücüler, sistem geri aramaları) duyulan doğal güveni giderek daha fazla istismar etmektedir. BYOVD [17], "meşru sürücüler tarafından oluşturulan güveni" kullanmaktadır. Geri Çağırma Kaçınması [28], "sistem tarafından belirli olaylar üzerine güvenlik yazılımına gönderilen bildirimleri" hedef almaktadır. Bu, meşru sistem bileşenlerinin saldırganlar tarafından kötüye kullanıldığını göstermektedir. Saldırganlar, işletim sisteminin kendisinde yeni güvenlik açıkları bulmak yerine, güvenilir, imzalı sürücülerdeki güvenlik açıklarını

istismar etmekte veya güvenlik yazılımının dayandığı meşru arayüzleri (geri aramalar) manipüle etmektedir. Bu, sistemin kendi mekanizmalarını savunmalarına karşı kötüye kullanan sofistike bir saldırı biçimidir. Bu durum, savunma stratejilerinde sadece imza tabanlı tespitten, meşru bileşenlerin kötü niyetli *kötüye kullanımını* tespit edebilen gelişmiş davranışsal izlemeye geçişi gerektirmektedir. Ayrıca, sürücüler ve diğer güvenilir bileşenler için yazılım tedarik zincirinin güvence altına alınmasının önemini de vurgulamaktadır, çünkü tehlikeye atılmış meşru bir sürücü güçlü bir silah haline gelebilir.

8. Sanallaştırma Tabanlı Gizleme

Sanallaştırma tabanlı gizleme, yerel bir uygulamanın kodunu, korunan yazılımın içine doğrudan gömülü özel, genellikle tescilli bir sanal makine (VM) için bayt koduna dönüştürmeyi içeren oldukça etkili bir tersine mühendislik karşıtı tekniktir.[24] Bu dönüşüm, tersine mühendislerin programın gerçek işlevselliğini ve orijinal mantığını anlamasını son derece zorlaştıran ek, karmaşık bir soyutlama katmanı oluşturur. Bu özel VM'nin talimat seti tipik olarak her korunan uygulamaya özgüdür ve bu sanal talimatlar için dağıtım mantığı daha da karmaşık ve gizlenmiş olabilir, birden fazla karmaşıklık katmanı ekler.[21] Bu tür korunan yazılımları gizlenmiş kodunu açmada kritik bir ilk adım, bu sanal talimatların çıkarılmasıdır. Bu zorluğun üstesinden gelmek için, VMProtect gibi ticari gizleyicilerin birden fazla sürümünü destekleyen ve sanal dal atlamalarını doğru bir şekilde çözümleyen dinamik ikili enstrümantasyon ve sembolik yürütmenin bir kombinasyonunu kullanan "Devmp" gibi yeni yöntemler ortaya çıkmaktadır.[21]

Sanallaştırma tabanlı gizleme, hem statik hem de dinamik analiz için önemli zorluklar

oluşturmaya devam edecek, orijinal program mantığını yeniden yapılandırmak için yüksek düzeyde özel araçlar ve son derece yetenekli analistler gerektirecektir.[21] Gerçek işlevselliği gizlemedeki kanıtlanmış etkinliği, otomatik analize, fikri mülkiyet hırsızlığına ve kötü amaçlı yazılım analizine karşı sağlam bir savunma sağlamaktadır. Devmp gibi gelişmiş sanal talimat çıkarma yöntemlerinin devam eden gelişimi, gizlenmiş kodu açma araçlarının gelişen sanallaştırma tekniklerine ayak uydurmak için giderek daha sofistike ve uyarlanabilir olması gerekeceği tırmanan bir silahlanma yarışını işaret etmektedir.[21]

Sanallaştırma, orijinal mantığı gizleyen bir "kara kutu" yaratır ve tersine mühendisliği dinamik davranışa odaklanmaya zorlar. Bu, statik analizi neredeyse imkansız hale getirir ve savaş alanını çalışma zamanı analizi ve sanal talimat çıkarmaya kaydırır. Sanallaştırma tabanlı gizleme, "yerel kodu, kötü amaçlı yazılımın içine gömülü özel bir sanal makine için bayt koduna dönüştürür" ve "ek bir soyutlama katmanı oluşturur".[24] Bu, statik analizin işlevselliği anlamasını "çok zor" hale getirir.[24] Bu durum, sanallaştırmanın, tersine mühendislik için bir "kara kutu" etkisi yarattığını göstermektedir. Kodun statik olarak incelenmesi, gerçek işlevselliği ortaya çıkarmada büyük ölçüde etkisiz hale gelir, çünkü yürütülebilir dosya artık doğrudan orijinal program mantığını temsil etmemektedir. Bu, tersine mühendislik çabalarını, programın çalışma zamanı davranışına ve sanal makine içindeki sanal talimatların çıkarılmasına odaklanmaya yönlendirmektedir. Bu, geleneksel statik analiz araçlarının ve metodolojilerinin sınırlamalarını vurgulamaktadır ve dinamik analiz ile sanal makine mimarisi anlayışına daha fazla vurgu yapılmasını gerektirmektedir.

Gizlenmiş kodu açma araçlarında (Devmp gibi) sürekli inovasyon, gelişen sanallaştırmaya ayak uydurmak için gereklidir. Bu, bu karmaşık katmanları parçalamak için dinamik ikili enstrümantasyon ve sembolik yürütme ihtiyacını vurgular, gizleme ve gizlenmiş kodu açma arasındaki devam eden, yüksek riskli rekabeti vurgular. "Devmp" gibi yeni yöntemler, "dinamik ikili enstrümantasyon ve sembolik yürütmenin bir kombinasyonunu" kullanarak "sanal talimat çıkarmayı" amaçlamaktadır.[21] Bu, gizleme ve gizlenmiş kodu açma arasındaki sürekli bir silahlanma yarışını açıkça göstermektedir. Bir taraf yeni bir savunma tekniği (sanallaştırma) geliştirdiğinde, diğer taraf bunu analiz etmek ve atlatmak için yeni bir saldırı tekniği (Devmp gibi sanal talimat çıkarma) geliştirmek zorundadır. Bu, dinamik ikili enstrümantasyon ve sembolik yürütme gibi gelişmiş tekniklerin, gizlenmiş kodu açma araçlarının temel bileşenleri haline geldiğini göstermektedir. Bu, tersine mühendislik alanının sürekli olarak yeni zorluklara uyum sağlaması ve bunlara yanıt vermesi gerektiğini, aksi takdirde gizleme tekniklerinin analiz yeteneklerini geride bırakacağını vurgulamaktadır.

9. Mobil Uygulama Güçlendirme ve Anti-Tersine Mühendislik Teknikleri (Savunma)

Hassas verileri işleyen mobil uygulamalar, giderek daha fazla hedef haline gelmekte ve bu da kendini koruma ve tersine mühendislik karşıtı (anti-RE) önlemlere güçlü bir vurgu yapılmasına yol açmaktadır. Mobil Uygulama Güvenlik Doğrulama Standardı (MASVS), kurcalamayı, veri sızıntısını önlemek ve jailbreak veya rootlamayı tespit etmek için çeşitli güçlendirme teknikleri önermektedir.[31] Bu teknikler arasında kod gizleme, bütünlük kontrolleri, root/jailbreak tespiti ve güvenli veri depolama mekanizmaları bulunmaktadır.[7] Bu önerilere rağmen, özellikle iOS'ta önemli sayıda uygulama hala kapsamlı kendini koruma uygulamalarında yetersiz kalmaktadır.[31] Ancak, MASVS tarafından tanımlanan çok çeşitli zayıflıkları ve derinlemesine savunma tekniklerini içeren MAS Referans Uygulaması gibi referans uygulamaların geliştirilmesi, güvenlik araçlarını doğrulamak ve çeşitli kod gizleme ve root tespiti atlatma teknikleri için gerçekçi eğitim senaryoları sağlamak açısından kritik öneme sahiptir.[7] Bu uygulamalar, geleneksel, sınırlı CrackMe'lerin ötesine geçerek gerçek dünya güvenlik açıklarını ve anti-RE zorluklarını simüle etmek için tasarlanmıştır.[7]

Mobil uygulama güvenliğinin sürekli evrimi, hızla genişleyen mobil ekosistemde hassas verileri ve fikri mülkiyeti korumak için daha sağlam ve uyarlanabilir anti-RE önlemlerini gerektirecektir. Tersine mühendisler, güçlendirme mekanizmalarını atlamak ve gizlenmiş kodu anlamak için gelişmiş dinamik enstrümantasyon ve analiz teknikleri gerektiren giderek daha sofistike mobil uygulama korumalarıyla karşılaşacaklardır.[7] Gerçek dünya mobil uygulama güvenlik açıklarını ve savunmalarını yansıtan özel CrackMe uygulamalarının geliştirilmesi, yeni nesil mobil güvenlik analistlerini eğitmek için hayati önem taşıyacaktır.[7]

Mobil platformlar (iOS/Android), sanal alanları, işletim sistemi düzeyindeki korumaları ve çeşitli cihaz ekosistemleri nedeniyle anti-RE için benzersiz zorluklar ve fırsatlar sunmaktadır. Bu durum, özel güçlendirme tekniklerini tetiklemekte ve tersine mühendislerin araç setlerini ve metodolojilerini uyarlamalarını gerektirmektedir. Mobil uygulamalar, sistemin izolasyon ve sanal alan mekanizmalarını atlayan saldırı tehdidiyle karşı karşıyadır.[31] Bu durum, mobil platformların kendine özgü güvenlik mimarileri nedeniyle anti-RE için ayrı bir "sınır" oluşturduğunu göstermektedir. Mobil işletim sistemleri, uygulamaları ve verileri korumak için belirli sanal alan ve izin modelleri uygular. Bu, geleneksel masaüstü tersine mühendislik tekniklerinin doğrudan uygulanamayacağı anlamına gelir ve tersine mühendislerin mobil platformlara özgü yeni teknikler ve araçlar geliştirmesini gerektirir. Bu durum, mobil uygulamalar için özel

güçlendirme tekniklerinin (örn. MASVS önerileri) geliştirilmesine yol açar ve tersine mühendislerin bu özel savunmaları anlamak ve atlatmak için uzmanlaşmasını zorunlu kılar.

MAS Referans Uygulaması gibi modern CrackMe'ler, statik bulmacaların ötesine geçerek dinamik, gerçek dünya simülasyonlarına dönüşmektedir. Bu değişim, mobil güvenlik profesyonelleri için daha ilgili eğitim sağlamakta, teorik bilgi ile sürekli değişen tehdit ortamındaki pratik uygulama arasındaki boşluğu doldurmaktadır. Mevcut "Crackme" uygulamaları güvenlik profesyonelleri için değerli eğitim sağlarken, her zaman gerçek dünya güvenlik açıklarıyla uyumlu değildir.[7] Bu durum, modern CrackMe'lerin sadece bir dizi soyut bilmece olmaktan çıkıp, kök tespiti atlatma veya dinamik enstrümantasyon üzerine giriş dersleri gibi belirli zorlukları uygulamaya odaklanan, gerçek kötü amaçlı yazılımlarda kullanılan seçilmiş hileleri içeren alıştırmalar haline geldiğini göstermektedir.[3] Bu, eğitim materyallerinin ve meydan okumaların, gerçek dünya tehdit ortamındaki gelişmeleri yansıtacak şekilde sürekli olarak güncellenmesi gerektiğini vurgulamaktadır. Bu, mobil güvenlik profesyonellerinin, sadece teorik bilgiye değil, aynı zamanda hızla değişen mobil tehdit ortamında karşılaşacakları pratik zorluklara da hazırlıklı olmalarını sağlamaktadır.

10. Kuantum Bilgisayar Tehditleri ve Kriptografik Tersine Mühendislik (Saldırı/Analiz ve Savunma)

Büyük ölçekli kuantum bilgisayarların mevcut kriptografik standartları kırabilecek kapasitede olması 2025'e kadar yaygınlaşmasa da, oluşturdukları tehdit siber güvenlik stratejilerini şimdiden etkilemektedir. Önemli bir endişe, kötü niyetli aktörlerin, özellikle ulus devletlerin, kuantum bilişim olgunlaştığında (2027 ile 2032 arasında tahmin edilmektedir) şifresini çözmek amacıyla şifrelenmiş verileri bugün topladığı "şimdi topla, sonra şifresini çöz" senaryosudur.[10] Bu yaklaşan tehdit, modern iletişim ve veri depolamayı güvence altına alan temel kriptografik ilkeleri temelden sorgulamaktadır. Sonuç olarak, tersine mühendislik alanı, Kuantum Sonrası Kriptografi (PQC) algoritmalarını analiz etmeye ve geliştirmeye giderek daha fazla odaklanacaktır. Bu, potansiyel güvenlik açıklarını yaygın olarak dağıtılmadan önce belirlemek ve tersine, gelecekteki kuantum etkin saldırılara karşı sağlam bir şekilde uygulanmalarını sağlamak için bu yeni kriptografik şemaların matematiksel temellerini ve uygulama ayrıntılarını anlamayı içermektedir. Tersine mühendislik, mevcut sistemleri analiz ederek kuantum algoritmaları tarafından istismar edilebilecek kriptografik zayıflıkları belirlemek için de

kritik olacaktır, bu sistemler PQC uyumlu olmasa bile.

Yaklaşan kuantum tehdidi, kuantum dirençli algoritmalar üzerindeki araştırma ve geliştirmeyi hızlandıracak, kriptografik tersine mühendisliği hem saldırgan hem de savunma amaçları için kritik bir disiplin haline getirecektir. Saldırı tarafında, PQC uygulamalarındaki yan kanal güvenlik açıklarını veya mantıksal kusurları analiz etmeyi içerecektir. Savunma tarafında ise, yeni PQC kütüphanelerinin güvenliğini doğrulamaya ve yazılıma doğru entegrasyonlarını sağlamaya, ayrıca eski sistemlerdeki "kuantum savunmasız" bileşenleri belirlemeye ve azaltmaya odaklanacaktır. Bu, kriptografi ve kuantum mekaniği konusunda derin uzmanlığa sahip tersine mühendisler için talep yaratacak ve alanda yeni bir alt disiplini teşvik edecektir.

Kuantum tehdidi, güvenliğe proaktif bir yaklaşım gerektirmektedir. Kuruluşlar sadece mevcut tehditlere karşı savunma yapmakla kalmıyor, aynı zamanda gelecekteki kriptografik ihlallere de hazırlanıyorlar. Kuantum bilgisayarların 2027 ile 2032 arasında ortaya çıkabileceği tahmin edilmektedir ve siber suçluların "şimdi toplama, sonra şifre çözme" stratejisiyle şifrelenmiş verileri topladığı belirtilmektedir.[10] Bu durum, kuruluşların sadece mevcut tehditlere karşı savunma yapmakla kalmayıp, aynı zamanda gelecekteki kriptografik ihlallere de hazırlanması gereken proaktif bir güvenlik yaklaşımının zorunlu olduğunu göstermektedir. Bu, tersine mühendislik çabalarının, doğası gereği anlaşılması ve doğru bir şekilde uygulanması daha zor olan yeni, karmaşık PQC algoritmalarını analiz etmeye ve doğrulamaya yöneleceği anlamına gelmektedir. Bu, güvenlik stratejilerinin, gelecekteki yetenekleri ve tehditleri tahmin etme ve bunlara karşı koyma yeteneğini içerecek şekilde genişlemesi gerektiğini vurgulamaktadır.

PQC'ye odaklanan tersine mühendislik içinde yüksek düzeyde uzmanlaşmış bir alanın ortaya çıkması beklenmektedir. Bu, sadece geleneksel tersine mühendislik araçlarında değil, aynı zamanda ileri matematik, kuantum mekaniği ve bu yeni kriptografik ilkellere özgü yan kanal analizinde de uzmanlık gerektirecektir. Bu derin uzmanlaşma, hem sağlam PQC geliştirmek hem de uygulamasındaki potansiyel zayıflıkları belirlemek için kritik olacaktır. Kuantum bilişimin "kriptografik standartları temelden sorguladığı" ve "kriptografik tersine mühendisliğin kritik bir disiplin haline geleceği" belirtilmektedir.[10] Bu, kuantum tehdidinin, tersine mühendislik alanında yeni ve oldukça uzmanlaşmış bir alt disiplinin ortaya çıkışını tetiklediğini göstermektedir. Geleneksel tersine mühendisler, bu yeni kriptografik algoritmaların karmaşık matematiksel temellerini ve kuantum fiziği ilkelerini anlamak için gerekli bilgiye sahip olmayabilirler. Bu, bu yeni alanda uzmanlaşmış tersine mühendisler için artan bir talep yaratacaktır. Bu uzmanlar, PQC uygulamalarını hem teorik olarak hem de yan kanal saldırıları gibi pratik uygulamalar aracılığıyla analiz edebileceklerdir. Bu durum, gelecekteki siber güvenlik iş

gücü için yeni beceri setleri ve eğitim yolları geliştirme ihtiyacını vurgulamaktadır.

III. Sonuç

2025 ve sonrası için tersine mühendislik ve CrackMe uygulamaları alanı, hem saldırgan hem de savunma stratejilerinde hızlı ve dönüştürücü değişikliklerle karakterize edilen dinamik bir ortam sunmaktadır. Yapay Zeka ve Makine Öğrenimi, bu evrimin temel itici güçleri olarak öne çıkmakta, otomatik analiz, gizlenmiş kodu açma ve güvenlik açığı keşfinde benzeri görülmemiş verimlilik ve yetenekler sağlamaktadır. YZ destekli araçlar, analiz süreçlerini hızlandırarak ve daha önce çözülemeyen sorunları ele alarak tersine mühendislerin analitik kapasitesini artırmaktadır. Ancak, bu ilerlemeler YZ'nin insan uzmanlığını tamamen değiştirmek yerine, onu artırdığı bir işbirliği paradigmasını da beraberinde getirmektedir; bu da analistlerin YZ çıktılarını yönlendirme ve doğrulama konusunda yeni beceriler geliştirmesini gerektirmektedir.

Aynı zamanda, Rust ve Go gibi modern programlama dillerinin kötü amaçlı yazılım geliştirmede benimsenmesi, tersine mühendislik uzmanlığında önemli bir değişimi zorunlu kılmaktadır. Bu dillerin benzersiz derleme ve çalışma zamanı özelliklerini anlamak, etkili kötü amaçlı yazılım analizi ve karşı önlemler için kritik hale gelmiştir. Otomatik güvenlik açığı keşfi, fuzzing ve sembolik yürütme gibi tekniklerle, sıfır gün güvenlik açıklarının bulunmasını hızlandırmakta ve saldırganların otomasyon kullanımına karşı koymak için temel bir savunma mekanizması olarak hizmet etmektedir. Bellek analizi ve çalışma zamanı manipülasyonu, statik analizden kaçınan veya donanım destekli güvenlik özelliklerini istismar eden sofistike kötü amaçlı yazılımlara karşı vazgeçilmez bir yaklaşım sunmaktadır. Donanım destekli tersine mühendislik ve istismar, CPU kusurlarını ve güvenli enklavları hedef alarak yeni, yüksek etkili saldırı yüzeyleri açmakta, donanım-yazılım ortak tasarımının entegre bir şekilde anlaşılmasını zorunlu kılmaktadır.

Savunma tarafında, gelişmiş kod gizleme teknikleri, özellikle Dinamik Sembolik Yürütmeye direnmek için tasarlanmış yeni opak predikatlar ve kontrol akışı düzleştirme, tersine mühendislik çabalarının maliyetini ve karmaşıklığını artırmayı hedeflemektedir. Bu, yazılım korumasının bir yıpratma savaşı haline geldiğini, en kararlı saldırganlar dışındaki herkesi caydırmayı amaçladığını göstermektedir. Kurcalama önleme ve tespit kaçınma teknikleri, güvenlik ürünlerinin kendisini hedef alarak, güvenlik çözümlerinin bütünlüğünü sağlamak için kendine kendini koruma ve davranışsal izleme gibi sağlam

mekanizmaların gerekliliğini vurgulamaktadır. Sanallaştırma tabanlı gizleme, program mantığını gizleyen bir "kara kutu" etkisi yaratmakta, gizlenmiş kodu açma araçlarında sürekli yeniliği ve dinamik analiz tekniklerine bağımlılığı tetiklemektedir. Son olarak, kuantum bilişimin yükselişi, kriptografik tersine mühendislik alanında yeni bir uzmanlaşmayı zorunlu kılmakta, gelecekteki kuantum saldırılarına karşı koymak için kuantum sonrası kriptografi algoritmalarının analizini ve doğrulanmasını gerektirmektedir.

Genel olarak, tersine mühendislik ve CrackMe uygulamaları alanındaki bu trendler, sürekli bir silahlanma yarışını ve siber güvenlik profesyonelleri için sürekli uyum sağlama ihtiyacını yansıtmaktadır. Bu ortamda başarılı olmak için, kuruluşların ve bireylerin gelişmiş araçlara, özel eğitimlere ve hem saldırgan hem de savunma tekniklerindeki en son gelişmeleri anlamaya sürekli yatırım yapmaları gerekmektedir. Uzmanlık, sadece mevcut tehditlere yanıt vermekle kalmayıp, aynı zamanda gelecekteki zorlukları tahmin etmek ve hazırlıklı olmak için de hayati önem taşımaktadır.

Alıntılanan çalışmalar

1. CSCI 7250 - Advanced Reverse Engineering - Modern Campus Catalog™ - University of North Georgia, erişim tarihi Haziran 12, 2025, https://catalog.ung.edu/preview_course_nopop.php?coid=93784&catoid=41&print
2. Unraveling the Art of Reverse Engineering - Infosec, erişim tarihi Haziran 12, 2025, https://www.infosecinstitute.com/resources/reverse-engineering/hacking-tools-reverse-engineering/
3. Malwarebytes CrackMe 2: try another challenge, erişim tarihi Haziran 12, 2025, https://www.malwarebytes.com/blog/news/2018/04/malwarebytes-crackme-2-another-challenge
4. Crackme - Wikipedia, erişim tarihi Haziran 12, 2025, https://en.wikipedia.org/wiki/Crackme
5. Top Reverse Engineering Courses Online - Updated [June 2025] - Udemy, erişim tarihi Haziran 12, 2025, https://www.udemy.com/topic/reverse-engineering/?p=2
6. Reversing for mortals: Solving Yoire crackme average challenge - Fluid Attacks, erişim tarihi Haziran 12, 2025, https://fluidattacks.com/blog/reversing-mortals
7. MAS Reference App - Implementing Mobile App Vulnerabilities and Defenses, erişim tarihi Haziran 12, 2025, https://www.redguard.ch/blog/2025/03/06/owasp-mas-reference-app/
8. AI in Reverse Engineering | How Artificial Intelligence is Transforming Cybersecurity, Malware Analysis, and Software Auditing - Web Asha Technologies, erişim tarihi Haziran 12, 2025, https://www.webasha.com/blog/ai-in-reverse-engineering-how-artificial-intelligence-is-transforming-cybersecurity-malware-analysis-and-software-auditing
9. Reverse Engineering Software Market Size | Growth Forecast To 2033, erişim tarihi Haziran 12, 2025,

https://www.businessresearchinsights.com/market-reports/reverse-engineering-software-market-112729
10. Top 7 Cybersecurity Trends to Watch in 2025 - Cogent Infotech, erişim tarihi Haziran 12, 2025, https://www.cogentinfo.com/resources/top-7-cybersecurity-trends-to-watch-in-2025
11. Recon Training - Automating Reverse Engineering with ... - Recon.cx, erişim tarihi Haziran 12, 2025, https://www.recon.cx/2025/trainingAutomatingReverseEngineeringwithMachineLearning.html
12. Malware analysis assisted by AI with R2AI - arXiv, erişim tarihi Haziran 12, 2025, https://arxiv.org/html/2504.07574v2
13. Extracting Hidden Malware Payloads with AI-Powered LLMs - HackerNoon, erişim tarihi Haziran 12, 2025, https://hackernoon.com/extracting-hidden-malware-payloads-with-ai-powered-llms
14. Decompiling Apps With AI Language Models - NowSecure, erişim tarihi Haziran 12, 2025, https://www.nowsecure.com/blog/2025/01/29/decompiling-apps-with-ai-language-models/
15. Malware Reverse Engineering & Detection Engineering 2025 - Cyber Security Asia 2025, erişim tarihi Haziran 12, 2025, https://cybersecurityasia.tech/malware-reverse-engineering-detection-engineering-2025/
16. Reversing Modern Binaries: Practical Rust & Go Analysis | Fuzzing ..., erişim tarihi Haziran 12, 2025, https://fuzzinglabs.com/reversing-modern-binaries/
17. What cybersecurity experts are talking about in 2025 - Virus Bulletin, erişim tarihi Haziran 12, 2025, https://www.virusbulletin.com/blog/2025/06/what-cybersecurity-experts-are-talking-about-2025/
18. Black Hat USA 2024 | Trainings Schedule, erişim tarihi Haziran 12, 2025, https://www.blackhat.com/us-24/training/schedule/
19. Advancing Code Obfuscation: Novel Opaque Predicate Techniques to Counter Dynamic Symbolic Execution - Tech Science Press, erişim tarihi Haziran 12, 2025, https://www.techscience.com/cmc/online/detail/23242/pdf
20. Advancing Code Obfuscation: Novel Opaque Predicate Techniques to Counter Dynamic Symbolic Execution - ResearchGate, erişim tarihi Haziran 12, 2025, https://www.researchgate.net/publication/391699381_Advancing_Code_Obfuscation_Novel_Opaque_Predicate_Techniques_to_Counter_Dynamic_Symbolic_Execution
21. Devmp: A Virtual Instruction Extraction Method for Commercial Code Virtualization Obfuscators (Internetware 2025 - Research Track) - Researchr, erişim tarihi Haziran 12, 2025, https://conf.researchr.org/details/internetware-2025/internetware-2025-research-track/12/Devmp-A-Virtual-Instruction-Extraction-Method-for-Commercial-Cod

e-Virtualization-Obf
22. A look back at Insomni'hack 2025 - eShard, erişim tarihi Haziran 12, 2025, https://eshard.com/posts/a-look-back-at-insomnihack-2025
23. Recon 2025 :: pretalx, erişim tarihi Haziran 12, 2025, https://cfp.recon.cx/recon-2025/featured/
24. Malware Obfuscation Techniques: Advanced Detection & Prevention Strategies - VMRay, erişim tarihi Haziran 12, 2025, https://www.vmray.com/malware-obfuscation-techniques/
25. Researchers discover Spectre like new speculative flaw, “SPOILER” in Intel CPU's - Packt, erişim tarihi Haziran 12, 2025, https://www.packtpub.com/de-ch/learning/tech-news/researchers-discover-spectre-like-new-speculative-flaw-spoiler-in-intel-cpus
26. What Is The Future of Reverse Engineering? - Metrology News, erişim tarihi Haziran 12, 2025, https://metrology.news/what-is-the-future-of-reverse-engineering/
27. DEF CON 32 - Reverse Engineering And Hacking Ecovacs Robots - Security Boulevard, erişim tarihi Haziran 12, 2025, https://securityboulevard.com/2025/01/def-con-32-reverse-engineering-and-hacking-ecovacs-robots/
28. Anti-Tampering and Detection Evasion – Bitdefender TechZone, erişim tarihi Haziran 12, 2025, https://techzone.bitdefender.com/en/gravityzone-platform/anti-tampering-and-detection-evasion.html
29. 9 Endpoint Security Software For 2025 - SentinelOne, erişim tarihi Haziran 12, 2025, https://www.sentinelone.com/cybersecurity-101/endpoint-security/endpoint-security-software/
30. Anti-Tampering Certification Test 2025 - AV-Comparatives, erişim tarihi Haziran 12, 2025, https://www.av-comparatives.org/anti-tampering-certification-test-2025/
31. SoK: Hardening Techniques in the Mobile Ecosystem — Are We There Yet? - vusec, erişim tarihi Haziran 12, 2025, https://download.vusec.net/papers/haly_eurosp25.pdf